2013 IEEE International Conference on High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing

Profit-Oriented Scheduling Optimization for Workflow in Clouds

Haoran Ji, Weidong Bao and Xiaomin Zhu
Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology
Changsha 4100773, P.R. China
Email: jimmy_nudt@hotmail.com {wdbao,xmzhu}nudt.edu.cn

Shu Yin
School of Information Science and Engineering, Hunan University
Changsha 410073, P.R. China
Email: shuyin@hnu.edu.cn

***Abstract*—Clouds have become a new paradigm by enabling on-demand provisioning of applications, platforms or computing resources for clients. Workflow scheduling is one of the most challenging problems in Clouds. Getting more profits is one of the most important objectives in workflow scheduling. Conventional workflow scheduling strategies developed on the kind of systems mainly focus on the workflow. In this paper, we take the communication into account and develop a novel scheduling algorithm based on the topology characters of degree and path length named Music Chair Algorithm (MCA). The algorithm gives a good performance on searching for the optimum schedule in getting the most profit. Also, we find that there exists a certain resource amount, which gets the most profit to help us get more enthusiasm for further developing the Clouds. Experimental results demonstrate that the analysis of the strategies for most profits are reasonable, and the MCA is available to efficiently get the optimum schedule with low computing complexity.**

I. Introduction

Cloud computing [1] is the latest emerging trend in distributed computing that delivers hardware infrastructure and software applications as services [8]. It is a good measure to meet the challenges of growing consumption of computing resources. The users can enjoy the computing service on a service level agreement, which defines their required Quality of Service (QoS)[2]. At present, specialization and modularity have become an important way to reduce costs for enterprises. Cloud optimizes the utilization of computing resource and decreases the risk of building a private computing center for client users. More and more enterprises do their tasks or reseraches on Clouds to minimize the execution costs. Although, Cloud operators built large computing centers with thousands of computers and servers to satisfy the users' need for computing. However, to meet the challenges of the the explosive growth of data in macro level and fluctuations in demand for services in short period, building larger computing centers is an very inefficient way. It is more efficient to take better scheduling algorithm in service level.

Cloud computing is one kind of distributed computing. And grid computing is also an important concept prior to Cloud computing in distributed computing. Although, there is some inherent relevance between them. But Clouds also present some new features. The most important difference is the"pay-as-you-go" pricing model of current commercial Clouds that charges users based on the number of the time intervals and the amount of resource utility. So we can see that profit is the basic motivation of Clouds.

Modern collaborative scientific researches involve a lot of works, such as structural biology, high-energy physics and astronomy, and there exist data interactions between correlative work steps. Usually we use workflow to present the correlations. A workflow is usually described by a Directed Acyclic Graph (DAG), in which a node denotes a task and a directed edge denotes the data or control dependency between tasks. Due to the complexity of the big time, workflows are very important in the researches of scheduling. The workflow scheduling in Clouds is difficult to solve for its multi-tasks nature. The pricing model of charging the clients by the number of time intervals and resource amount they have used, which used in most current commercial Clouds, also contributes to the complication.

The main contributions of this paper are as follows:

- We review the researches on workflow scheduling in Clouds.
- We give a developed description on the scheduling problem in Clouds.
- We propose an optimization algorithm (MCA) for workflow scheduling based on topology characteristics.
- We construct several workflows and test our algorithm on these workflows. By the experiments, we prove the efficiency of the proposed algorithm.

The remainder of the paper is organized as follows. The next section reviews the related work and points out the shortness of them for Cloud application. Section III formally models the scheduling problem for workflows. Section IV then analyzes the search process base on simple instance and gets the rules for fast searching for the optimum schedule. Following the analysis Section V proposes the MCA to execute the workflow scheduling. The performance evaluation experiments and results analysis are given in Section VI. Section VII concludes the paper with a summary and presents the further works we would do.

II. Related Work

Many heuristic methods have been proposed for homogeneous and heterogeneous distributed systems. Topcuoglu presented two novels scheduling algorithms for a bounded number of heterogeneous processors with an objective to simultaneously meet high performance and fast scheduling time, which were called the Heterogeneous Earliest-Finish-Time ($HEFT$) algorithm and the Critical-Path-on-a-Processor ($CPOP$) algorithm [3]. Bajaj et al. introduced a task duplication-based scheduling algorithm for network of heterogeneous systems ($TANH$), with complexity $O(V^2)$, which provides optimal results for applications represented by directed acyclic graphs ($DAGs$). They also provided a simple set of conditions on task computation and network communication time that could be satisfied [4]. Rahman et al. proposed a dynamic critical path (DCP) based workflow scheduling algorithm that determines efficient mapping of tasks by calculating the critical path in the workflow task graph at every step [5]. Gu constructed analytical cost models and formulate the workflow mapping as optimization problems [6]. Further he developed a workflow mapping algorithm based on a recursive critical path optimization procedure to minimize the latency and conducted a rigorous workflow stability analysis. Besides he developed a layer-oriented dynamic programming solution based on topological sorting to identify and minimize the global bottleneck time. Abrishami et.al proposes a new QoS-based workflow-scheduling algorithm named partial critical paths (PCP), which makes efforts on minimizing the cost of workflow execution while meeting deadline constraints [2].

For workflow scheduling in Clouds, Xu introduced a multiple QoS constrained scheduling strategy of multi-workflows ($MQMW$) to meet the challenges of Cloud service characters [7]. Based on previous work in [2], Abrishami extended the PCP algorithm and proposed two workflow scheduling algorithms, which called Infrastructure as a Service (laaS) Cloud Partial Paths ($IC-PCP$), and laaS Cloud Partial Critical Paths with Deadline Distribution ($IC-PCPD2$) [8]. On the analysis of Clouds, Kllapi et al. considered that the optimization criterion is at least two-dimensional and presented an optimization framework, then they incorporated the devise framework into a prototype system [9]. Pandey et.al presented a particle swarm optimization (PSO) based heuristic to schedule workflows account of the computation cost and data transmission cost [10]. Wu et al. proposed a market-oriented hierarchical scheduling strategy in Cloud workflow systems and described the Cloud workflow scheduling with four layers, i.e. application layer, platform layer, unified resource layer and fabric layer [11]. Yu and Buyya proposed a budget constraint based scheduling [12]. The algorithm minimizes execution time while meeting a specified budget for delivering results. And then they proposed a new type of genetic algorithm to solve the optimization problem.

In these work, although Pandey in [10] takes communication into account, the computing complexity of the proposed algorithm is high as it is hard to satisfy the challenges of big size workflows. Other works do not take communication time into account which is very important for workflows scheduling in Clouds [4]. In these works, we find it is effectual and practical to guid the serching for optimal schedule by using topology parameters [4], [6], [7], [12].

III. PROBLEM FORMULATION

In this paper, we formulate the workflow scheduling problem in Clouds with four basic objects: workflow model, service model, cost model and objective.

A. Workflow Model

A set $T = \{t_1, t_2, \cdots t_n\}$ of tasks are submitted for execution. A set $E = \{parent, child\}$ represents a precedence constraints that indicate the relationships of tasks. We model a workflow with a graph of $G = \{T, E\}$. $E = \{e_{ij}, i, j \in [1 \cdots n]\}$, as in Fig.1, is usually the data dependence for a workflow in Clouds. For multi-workflows, we could add an entry node and an exit node with execuation time of 0, and then the workflows are aggregated to one workflow.

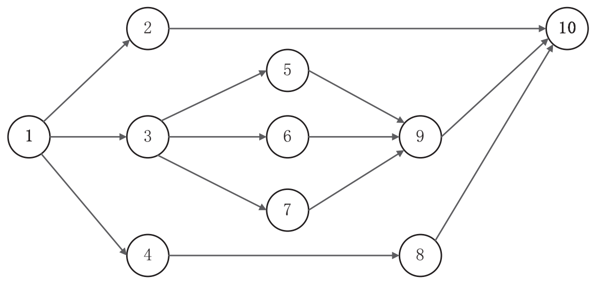


Fig. 1. A workflow sample

In a workflow G, $t_i = parent(t_j)$ or $t_j = child(t_i)$ denotes that t_j depends on t_i. We call t_i the parent task of t_j and t_j the child task of t_i. Each $t_i \in T$ has a execution time et_i, an start time a_i and a finish time f_i, $et_i = f_i - a_i$. There are two states for a task: unprepared and prepared. One task would be prepared if all of its parent tasks are scheduled. Each e_{ij} has a transmission time, denoted as tt_{ij}. So if $t_i = parent(t_j)$, there has $f_i + tt_{ij} < a_j$.

B. Service Model

Clouds consists of the resources, denoted by a set of $R = \{r_1, r_2 \cdots r_n\}$. There are mainly four kinds of resources, which are the computing resources, denoted by a set of $CR = \{cr_1, cr_2 \cdots cr_n\}$, and storage resources, denoted by a set of $SR = \{sr_1, sr_2 \cdots sr_n\}$, the network resources, denoted by a set of $NR = \{nr_1, nr_2 \cdots nr_n\}$. The NR are very important in workflow scheduling, which are always neglected in many researches. In this paper, we focus on the task-service level [14] of Cloud and take the network into account.

Clouds mainly provides two kind of services: resource services and application services [15]. Resource services provide resource as a service to remote clients. Application service allows remote clients to use their specialized applications, unlike resource services, which are capable of providing estimated service times based on the metadata of users' service requests. Normally, Cloud provides services to users through offering accesses of resources and their combination. A service s_{ij} is the mapping relationship between task t_i and resource r_j. Computing service (CS) provide applications to meet the computing needs. Storage service (StS) offers a space in the Cyberspace which is brought forward to describe the space consisted of computers, data, networks and so on, to storage the data used in Clouds. Computing service would always go with temporary storage and Clouds also provide the storage service which could not be considered separately with the network services (NS). Network services ensure the communications between users and the data transmissions between tasks by constructing the links. The service is an access to the corresponding resource. As pay for the service, you get the right to use the corresponding resources. Service is the key feature of Clouds. The Clouds service model is described in Fig.2.

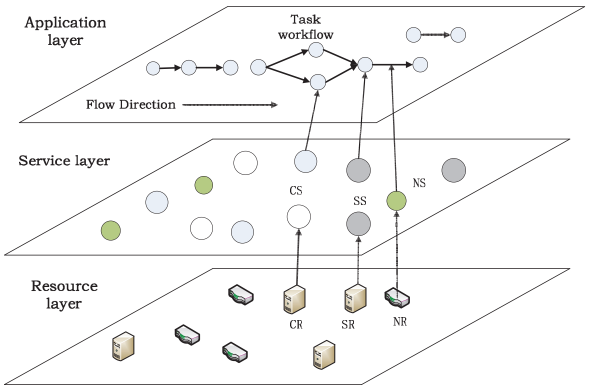


Fig. 2. Cloud service for workflow

Considering the importance of exaction time in workflow scheduling, a developed directed acyclic graph (DDAG) is used to describe the mapping relationships of tasks and the resources as in Fig.3. A time axis is used to describe the arrival time a_i and y axis to part the different resources. In the $DDAG$, we could clearly get the knowledge of the service s_{ij}, which assigns resource r_j to task t_i. We can also get the data transition time between parent-child tasks in the same resource and the communication time for the tasks with data dependence on different resources. Time is an important element for scheduling.

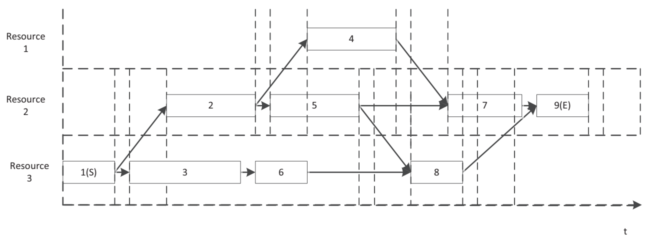


Fig. 3. DDAG model of workflow scheduling problem

C. Cost Model

Most Clouds, for example the Amazon Elastic Compute Cloud, charge by the service time based on users' choice of the instance modes [17]. A set $EC = \{ec_1, ec_2 \cdots ec_n\}$ of execution cost bijective maps to the workflow $T = \{t_1, t_2 \cdots t_n\}$. Each $ec_i, i \in [1, n]$ is the execution cost of corresponding task t_i. it is the servicing cost of CR. $StC = \{stc_1, stc_2 \cdots stc_n\}$ of the storage cost is associated with the storage amount and time intervals. In actual Clouds, the expense is directly related with EC and StC. A set $NC = \{nc_1, nc_2 \cdots nc_m\}$ of network cost arises with the communications between resources, which would use the NR. For $t_j = child(t_i)$, if t_i and t_j are exacuted on the same resource, there would be no NC. t_j can easily get the data produced by t_i. As the transmission time of data flowing from t_i to t_j is so small that it could be nearly ignored. Furthermore, the communications between tasks executed on different resources contribute to SC. SC is mainly consisted by these three factors. So SC can be described as

$$SC = EC + StC + NC. \tag{1}$$

D. Objective

As the inherent features of Clouds, profit is the main object for the problem. We describe the profit, denoted as p, of a Cloud service through two parameters: the service price (SP) and the service cost (SC).

$$p \propto \frac{SP}{SC} = \frac{EC}{SC + EEC} \tag{2}$$

And also quantity of service (QoS) is an important factor on the service price. QoS is mainly decided by the execution time. Extra waiting would decrease the QoS. To describe the enfluence, we introduce extra execution cost (EEC). Reducing QoS means generating EEC. Providing new resources would also take extra cost. However users would only be pleasure to pay for CS. Clients would not pay for uncertain charges. Therefore, minimizing the extra cost would directly contributes to minimize the SC. For a certain workflow, it contributes to the maximizing of the profits. With equation (1) and (2), we get

$$\max p \propto max \frac{\sum_G EC}{\sum_{G'} EC + EEC + StC + NC}. \tag{3}$$

The G' denotes the whole R used in scheduling. The idle resources would also make costs. So if the CR are fully used, the objective can also be described as:

$$\max p \propto min \sum_G \frac{StC + NC + EEC}{EC}. \tag{4}$$

To get higher p, for certain workflow, it is available to make efforts for better scheduling algorithms by minimize the NC and EEC while the EC and StC are relatively fixed. Taking better scheduling algorithm in task-service level is a good measure.

IV. Workflow Scheduling Optimization

The workflow scheduling optimization problem in heterogenous enviroment is a well-known NP-hard problem. The scheduling optimization would always face up to the problem of computing complexity. It is meaningful to find a optimization approach of getting a satisfied schedule with low computing complexity.

A. Theory Analysis

In former section, we suppose that topogoly characters besides path length would facilite the searching of optimal schedule. In this paper, we mainly consider three topology parameters: out-degree, in-degree and path length [15].

Definition 1. *Degree: the number of edges connecting to the vertex, with loops counted twice, in a directed graph.*

Definition 2. *Out-degree: the number of edges coming from the vertex in a directed graph.*

Definition 3. *In-degree: the number of edges coming to the vertex in a directed graph.*

Definition 4. *Path length: the longest distance an object travels in the direction of the edge to the end vertex in a directed graph.*

In $DAGs$, a node with bigger out-degree is more important as it influences more nodes. As lateness on it would lead to the lateness of its children nodes, it influences the entire performance directly. For the nodes with big in-degree, their parent nodes are critical. There is more risk just as any one of its parent nodes' falling down would lead to its hanging up. The influence would make effects indirectly, however sometimes make no effects at all. We can see that the former are more crucial for the scheduling . And also the path length from the node to the end node would affect the scheduling. Smaller path length leads to better fault tolerance. It would make effects only when there are too many tasks later. Parallel execution is also helpful to reduce the EEC, especially for the task with big out-degree. So giving higher priority to these tasks would reduce the EEC.

B. Case Analysis

We make the analysis on the assumption as follows. The resource amount would not change during the execution of the workflow. The et_i of t_i is one time unite. For a $DDAG$, we can split the task t_i to several serial tasks, whose $et = 1$, with the amount as the value of et_i. So the task, whose $et = n$, could be tranformed to n tasks whose $et = 1$, just as in Fig.4. We can find that the assumptions make no differences.

We generate several schedules for the sample workflow in Fig.1 based on the assumptionsas as in Fig.5. By scanning all the possible schedules, we find that schedule 2 is optimum, which has the maximum p.

Comparing the schedules and their topology characters as in table 1, we can find some relationships between the optimum schedule and the topology characters. The difference between

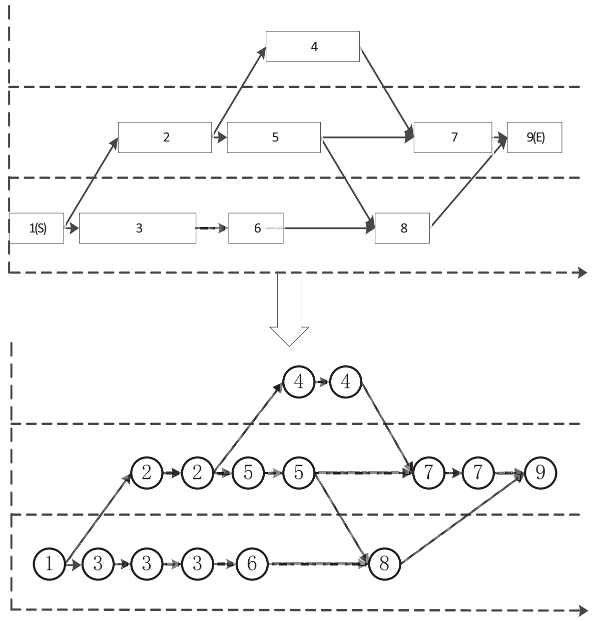
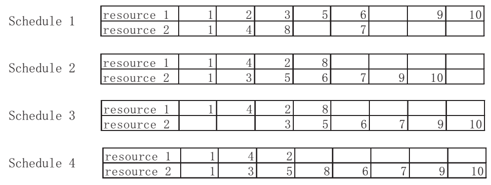


Fig. 4. DDAG transformation



Schedule 1	resource 1	1	2	3	5	6		9	10
	resource 2	1	4	8		7			
Schedule 2	resource 1	1	4	2	8				
	resource 2	1	3	5	6	7	9	10	
Schedule 3	resource 1	1	4	2	8				
	resource 2			3	5	6	7	9	10
Schedule 4	resource 1	1	4	2					
	resource 2	1	3	5	8	6	7	9	10

Fig. 5. Possible schedules for the sample workflow

schedule 1 and schedule 2 is the task choice in the second step. Schedule 1 chooses task 2 while schedule 2 takes 3. The difference between schedule 2 and 3 is that schedule 2 executes task 1 on two parallel resources. In schedule 4, we could improve it for better performance but the CC is bigger than that in schedule 2. In table 1, task 3 has bigger out-degree

TABLE I
Topology parameters for the sample workflow

$task$	$in-degree$	$out-degree$	$pathlength$
1	0	3	4
2	1	1	1
3	1	3	3
4	1	1	2
5	1	1	2
6	1	1	2
7	1	1	2
8	1	1	1
9	3	1	1
10	3	0	0

than task 2. The child node of task $5, 6, 7$ has bigger input than task 8. To confirm our surmise, we also make analysis on many other workflows and it proves there exist the relation between the topology parameters and optimal schedule.

C. Scheduling Strategies

Above all we give the rules of workflow scheduling as follows:

- Rule 1: Execute the task node with bigger out-degree .
- Rule 2: Execute the parent nodes of node with bigger input degree.
- Rule 3: The node with longer path length has priority.
- Rule 4: Execute the task which arrives earlier.
- Rule 5: Take parallel execution of big out-degree task on idle resource.

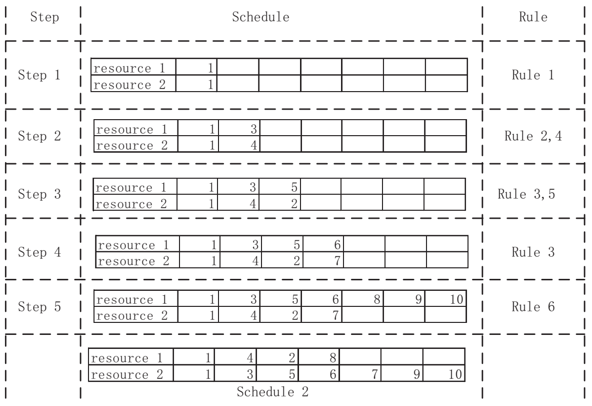


Fig. 6. The scheduling process

The rule with small number has higher priority. A task under more rules has higher priority. Since the in-degree and out-degree are both partial topology characters, it is suitable for complex workflows. If a task is scheduled, we would drop it and renew the workflow topology information. To give a clear description, we take an instance on the sample workflow as in Fig.6. We schedule the sample workflow with proposed rules and get the same p, which is the smallest for all possible schedules, with schedule 2.

V. Music Chair Algorithm

Based on the analysis above, we propose the Music Chair Algorithm.

Algorithm 1 I/O of MCA

```
Input:
  Workflow matrix $m$: description of the topology;
  Resource amount $N$: the resource amount provided by the Cloud;
  Costs information of the workflow tasks;
  Priority weight $p_i$ of each rule;
Output:
  scheduling matrix $sm$: the schedule;
```

An adjacent matrix is used to describe the workflow. For a workflow G with n tasks, an n-matrix m , the dimensions of which correspond to the tasks, is used. If there is an dependent relation from t_i to t_j, we set $m_{ij} = 1$, else we set it 0. With the matrix we could easily find the topology characters of the graph and operations on the workflow would be more flexible. The out-degree of t_i equals to the number of 1 in the i row of m. And in-degree of t_i equals to that in the i column of m. Also, we can get the path length just by scaning m. Another $m \times n$ matrix sm is used to describe the schedule. m is the number of resources and n is the maximum time span of scheduling. If we assign resource r_i to task t_j at time k, there has $sm_{ik} = t_j$. If there is no sm_{ik}, set it 0. Algorithm 1 describes the I/O of MCA.

Algorithm 2 *tasks states scaning*

```
1: $import$ function indegree, outdegree and pathlength;
2: $chair \leftarrow sm_{ik}$;
3: for i=1:n do
4:     $t_j$=parent($t_i$);
5:     if indegree($t_i$)==0&&$f_j + tt_{ji} < k$ then
6:         STATE($t_i$)=prepared;
7:     else
8:         STATE($t_i$)=unprepared;
9:         $PT \leftarrow t_i$;
10:    end if
11: end for
```

The MCA could be divided into three parts: tasks states scaning, competition and end detection. The first part aims at finding the prepared tasks. For the inherent constraints of workflow, only the prepared tasks could be scheduled. Normally, the costs of CR are much bigger than the NC. So the MCA focus on the fully use of CR. For each CR, all prapared tasks (prepared to exacution on the resource) would compete for the resource. Under the proposed rules, the task with higher priority would take the resource, just as the music chair game in which more powerful one would get the chair. That is the priority competition. Every sm_{ij} or chair would be competed until all the tasks are exacuted. That is what the end detection does.

Algorithm 3 *competition*

```
1: $import$ function indegree, outdegree and pathlength;
2: tasks states scaning;
3: $CPT$=children($PT$);
4: if isempty($PT$) then
5:     chair=$NULL$;
6: else
7:     $PT.rule_1 \leftarrow$ outdegree($PT$);
8:     $PT.rule_2 \leftarrow$ indegree($CPT$);
9:     $PT.rule_3 \leftarrow$ pathlength($PT$);
10:    $PT.rule_4 \leftarrow PT.a_i$;
11:    $PT.priority$=$\sum_{PT} p_i \times PT.rule_i$;
12:    chair=max($PT.priority$);
13: end if
```

The functions of indegree, outdegree and pathlength devote to get the proposed parameters.PT is the set of prepared tasks. For a certain chair, the task, who has the right to compete for the chair, must meet the following constraints, in line 5. First, its parents task must be scheduled. Second, its start time must be earlier than the chair competition's start time.

The rules are to measure the priority of each task. And the final priority is decided by the whole performance as in line 11. The competition is the process of comparing priority of tasks to decide who take the resource.

Algorithm 4 $MC\ Algorithm$

```
1:  $import$ function indegree, outdegree and pathlength;
2:  for i=1:N do
3:      for j=1:M do
4:          chair=$sm_{ij}$;
5:          $competition(chair)$;
6:          $T$=$max(PT.priority)$;
7:          STATE($T$)=scheduled;
8:          tasks states scaning;
9:      end for
10:     scan $sm_{ij}$;
11:     if any($sm_{ij}$)==$NULL$ then
12:         $sm_{ik}$=find($sm_{ij}$==$NULL$);
13:         $sm_{ik}$=max(outdegree($T$))
14:     end if
15: end for
```

Powerful competitor, who is the prepared task, would be able to occupy the chair under the given rules. Each resource with time property is a chair. And we play the music chair games until all the tasks be scheduled.

VI. Performance Evaluation

To evaluate the performance, we make the following works. First, we design a workflow generator which generates three classic scientific workflows. Secondly, we make efforts to prove the rationality of the idea that there exist a resource amount taking most profit. Baesd on these efforts, we can evaluate the MCA. We use the critical path algorithm (CPA) which is mainly focus on the path length as contrast. Former work in [2] proves that the algorithm is suitable to schedule the workflows. So rule 3 is proved to be effectable. We name the algorithm using rule 2 and rule 3 to guid the scheduling as in-degree and critical path algorithm ($ID-CPA$). And we compare the MCA with the $ID-CPA$ and CPA to evaluate its performance. Finally, we make an analysis on the computing complexity of MCA. A 3.3 GHz computer is used to test the algorithm.

A. Workflow generation

Scientific workflow is important in workflow scheduling and also in Cloud. We take three classic scientific workflows [16], which are Montage, Cybershake and LIGO (in Fig.7), and one instance workflow in Fig.1 to evaluate the performance of the proposed algorithm. The parameters of scientific workflows used in this section are given in table 2 [16].

B. Existion of the best resource amount

In this section, we fix the scheduling algorithm as MCA and change the resource amount to test if there exist a best resource amount. We make settings as: a_i of t_i equaling to

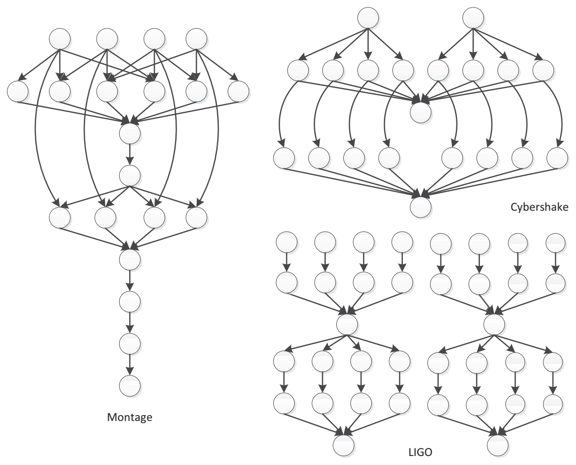


Fig. 7. Three classic scientific workflows

the path length to the start node; the expected et, denoted as eet, is 1; function of et and EEC as $EEC = et - eet$.

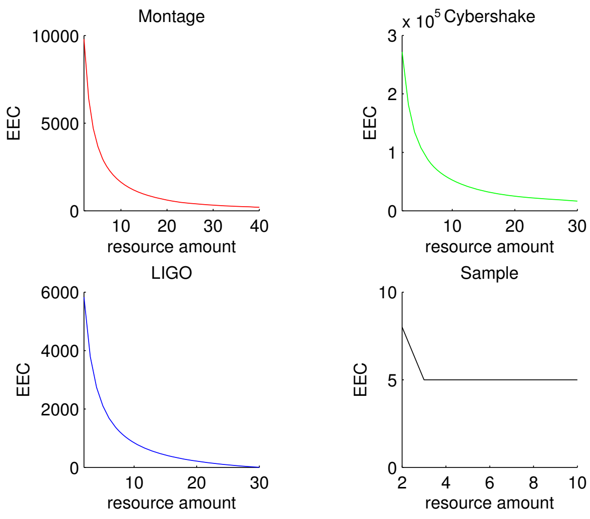


Fig. 8. EEC with different resource provision

We make experiments on the test workflows with the MCA and get results as in Fig.8. The EEC would decrease with the growing of resource amount by a negative exponential distribution for Montage, Cybershake and LIGO. Smaller EEC leads to bigger p. Meaningwhile, providing extra resource would take extra cost in the process of resource allocation, data communication and resource preparing, which would leads to smaller p. We set a linear function to describe the relationship between the cost and the resource amount. So, there exist a fix value getting the most profit.

C. Performance for Different Workflows

For different workflows, a good algorithm should present stability, which is considered important, for different workflow

TABLE II
PARAMETERS SETTINGS OF THE TEST WORKFLOWS [16]

Workflow	$Montage$	$Cybershake$	$Lingo$	$Sample$
Parameters	Count:203	Count:1053	Count:166	Count:10
	$mProject$:45	$Extract_sgt$:19	$Tmpltbank$:34	$e\tau$:1
	$mDiffFit$:107	$Seismogram_synthesis$:516	$Inspiral$:76	nc:1
	$degree$:9	$PeakValCalc_Okaya$:516	$Thinca$:34	
	$e\tau$:1	$e\tau$:1	$e\tau$:1	
	nc:1	nc:1	nc:1	

structures. We change the Montage workflow parameter of degree to generate different workflows to test the proposed algorithm. And we change the number of $mDiffFit$ to generate different workflows to test the algorithm as in Fig.10. We take the same assumption as in $partB$ and make comparison on the algorithm of MCA, ID-CP and CP. Also we compare the results of these algorithms to evaluate the performance.

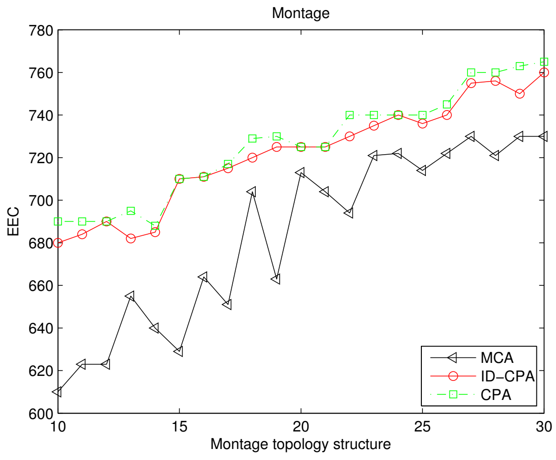


Fig. 9. EEC with different topology structure

In Fig.9, the structure of Montage changes with the degree. We can see that the MCA acts better. The $ID-CPA$ makes little optimization as the in-degree would only make effects when there are many offspring nodes. So we can see that rule 1 takes more optimizing than other rules. More rules would make help on getting the optimal schedule. Finally, the algorithm gives a good performance .

In Fig.10, we increase the number of Montage mProject. We can see that the EEC increases with the number by a certain linear coefficient. We can see that the MCA also gives a perfect performance for the changes of workflow size.

On the whole, we find that the changing of workflow topologic structure enfluences the performance of MCA more than the workflow size. We speculate that these three parameters could not describe the topologic structure completely. So if we can give all the parameters to describe the topologic structure, it is a hopeful approach to get the optimal schedule.

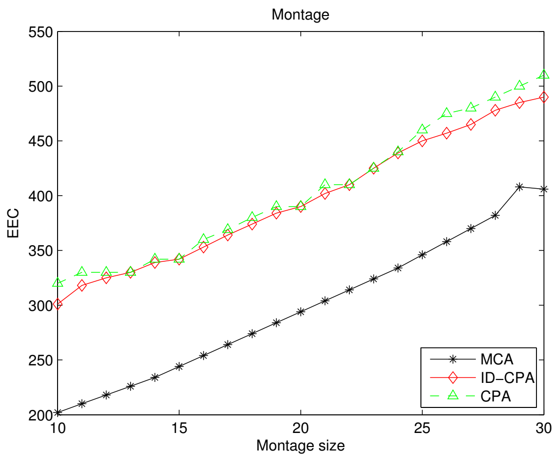


Fig. 10. EEC with different workflow size

D. Performance of Computing Time

The computing time is also an important measure to evaluate a scheduling algorithm. With low computing complexity of scheduling algorithm, more works would be done. And high computing complexity is a disaster for scheduling algorithm. It makes the algorithm lose the practical values.We analyze the computing time of experiment in part B and C.

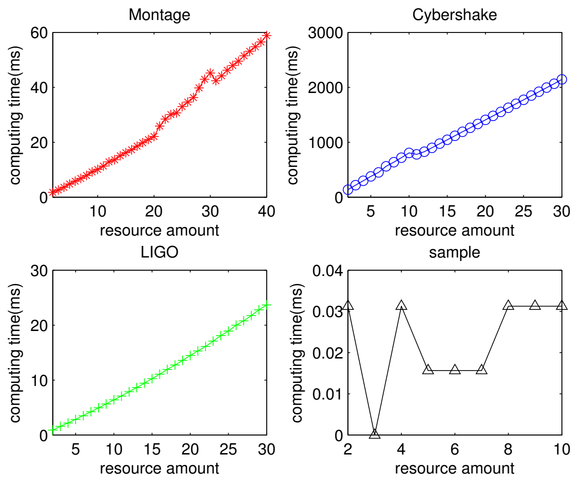


Fig. 11. Computing time of experiment B

In Fig.11 we can see that the computing time is almost linear and, for small resource amount, the computing time is very short. It indicates that the computing complexity of MCA increases linearly by the resource amount. If the resource amount is bigger, there are more possible schedules. The scanning space increases exponentially. So the MCA presents a good performance for the changing resource amount.

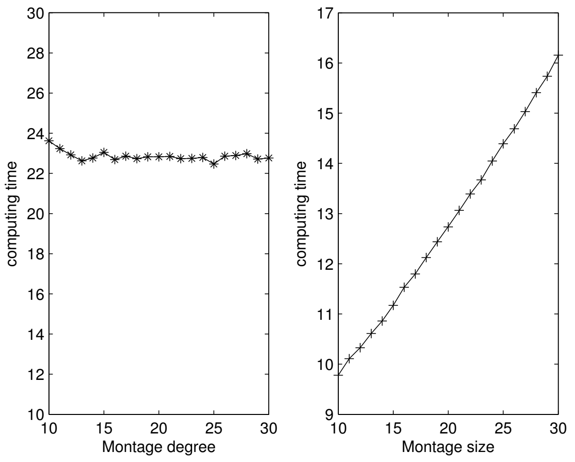


Fig. 12. Computing time of experiment C

In Fig.12 we can see that, for different topologics of the same size workflows, the computing time is steady. The computing time changes linearly by the workflow size. For large size workflows, it presents challenges to complete the scheduling in linear time. The results indicate that the MCA can adapt to big size workflow well.

On the final, the MCA gives a good performance on workflow scheduling in Clouds. With these works, we can see that the background of MCA is reasonable and the algorithm make effects on the searching for optimal schedule of workflow scheduling in Clouds especially for large size workflows.

VII. Conclusion and future work

Cloud computing would be an important part of our lives. This paper describes the workflows scheduling problem in Cloud. Give a detailed description of the Clouds enviroment. The Clouds mainly consist of three kind of resources and corresponding services. Then the paper gives an instance and all its schedules. By analyzing the instance, the relationships between optimum schedule and workflow topology are found which could be used to improve the scheduling efficiency. We propose the Music Chair Algorithm to execute the workflow scheduling. The algorithm uses the partial characters of workflow topology to optimize the scheduling. The paper proposes the MCA which efficiently reduces the searching complexity, especially for big size workflows, and successfully gets the optimized schedule. We conduct extensive experimental results to evaluate its performance changing the workflows and analyzing the computing time. It proves that the MCA could increase the searching efficiency for optimum schedule with low computing complexity.

Further works would be done for uncertain task execution time, task emerging, multi-objectives and fuzzy topology information. For the growing amount of workflows, the proposed algorithm would be developed to meet the challenges. We plan to research the influence of other topology characters on the optimum schedule searching such as clustering coefficient and betweenness and to extend the MCA considering the data uncertainty [17].

References

[1] R.Buyya, J.Broberg, I.Brandic, "Cloud computing and emerging it platforms: vison, hype and reality for delivering computing as the 5th utility," *Further Generation of Conputing System*. 25(6) (2009) p599-616.

[2] J. Saeid Abrishami, Mahmoud Naghibzadeh, Dick H.J. Epema, "Cost-driven scheduling of Grid workflows using Partical Critical Paths," *IEEE Transactions on Parallel and Distributed System*. 23 (2011) p1400-1414.

[3] H. Topcuoglu. "Performance-effective and low-complexity task scheduling for heterogeneous computing," in *IEEE Transactions on Parallel and Distributed Systems*. Mar 2002. p260-274.

[4] Rashmi Bajaj,"Improving scheduling of tasks in a heterogeneous environment," in *IEEE Transactions on Parallel and Distributed Systems*. Feb 2004. p107-118.

[5] M. Rahman, "A Dynamic Critical Path Algorithm for Scheduling Scientific Workflow Applications on Global Grids," *IEEE International Conference on e-Science and Grid Computing*. 10-13 Dec. 2007. p35-42.

[6] Gu, Y, "Performance analysis and optimization of distributed workflows in heterogeneous network environments" in *IEEE Transactions on Computers*. Volume: PP, Issue: 99 .

[7] Meng Xu, "A multiple QoS constrained scheduling strategy of multiple workflows for Cloud computing," *2009 IEEE International Symposium on Parallel and Distributed Processing with Applications*. p629-643.

[8] Saeid Abrishami, Mahmoud Naghibzadeh, Dick H.J. Epema, "Deadline-constrained workflow scheduling algorithms for infrastructure as a Service Clouds," in *Future Generation Computer Systems*. Volume 29, Issue 1, January 2013, p158169.

[9] Herald Kllapi, Eva Sitaridi, Manolis M. Tsangaris. "Schedule optimization for data processing flows on the Cloud" in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. p289-300.

[10] Suraj Pandey, Linlin Wu, Siddeswara Mayura Guru, R. Buyya. "A particle swarm optimization-based heuristic for Scheduling Workflow Applications in Cloud computing environments," in *2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*. 20-23 April 2010.

[11] J. Zhangjun Wu, Xiao Liu, Zhiwei Ni, Dong Yuan, "A market-oriented hierarchical scheduling strategy in Cloud workflow systems," *The Journal of Supercomputing*. January 2013, Volume 63, Issue 1, p256-293.

[12] Jia Yu, Rajkumar Buyya, "A budget constrained scheduling of workflow applications on utility grids using genetic algorithms," *WORKS '06. Workshop on Workflows in Support of Large-Scale Science*. 19-23 June 2006.

[13] S. Benkner et al., "GEMSS: Grid-infrastructure for medical service provision," in *In HealthGrid 2004 conference*. 29th-30th Jan. 2004, Clermont-Ferrand, France.

[14] "http://zh.wikipedia.org/wiki/Amazon_EC2."

[15] M.E.J. Newman et al.,"Network: An intruduction," in*Oxford University Press*. 2011.

[16] Shishir Bharathi, Ann Chervenak, etc., "Characterization of scientific workflows," in *Third Workshop on Workflows in Support of Large-Scale Science*. p1-10, Nov.2008.

[17] Luiz F. Bittencourt, Rizos Sakellariou, Edmundo R.M. Madeira. "Using relative costs in workflow scheduling to cope with input data uncertainty," *MGC2012*, December 3, 2012, Montreal, Quebec, Canada.