IEEE Access
The journal for rapid open access publishing

Received October 28, 2015, accepted December 2, 2015, date of publication December 17, 2015, date of current version December 23, 2015.

Digital Object Identifier 10.1109/ACCESS.2015.2508940

A Multi-Objective Optimization Scheduling Method Based on the Ant Colony Algorithm in Cloud Computing

LIYUN ZUO[1,2], LEI SHU[1], (Member, IEEE), SHOUBIN DONG[2], CHUNSHENG ZHU[3], (Student Member, IEEE), AND TAKAHIRO HARA[4], (Senior Member, IEEE)

[1]Guangdong Provincial Key Laboratory of Petrochemical Equipment Fault Diagnosis, Guangdong University of Petrochemical Technology, Maoming 525000, China
[2]School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
[3]Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC V6T 1Z4, Canada
[4]Department of Multimedia Engineering, Osaka University, Suita 565-0871, Japan

Corresponding authors: L. Shu (lei-shu@outlook.com) and S. Dong (sbdong@scut.edu.cn).

This work was supported in part by the Educational Commission of Guangdong Province, China, under Grant 2013KJCX0131, in part by the Guangdong University of Petrochemical Technology's Internal Project under Grant 2012RC0106, in part by the Natural Science Foundation of Guangdong Province, China, under Grant 2014A030313729, in part by the 2013 Top Level Talents Project Sailing Plan of Guangdong Province, in part by the 2014 Guangdong Province Outstanding Young Professor Project, in part by the Science and Technology Key Project of Guangdong under Grant 2014B010112006, and in part by the Natural Science Fund of Guangdong under Grant 2015A030308017.

ABSTRACT For task-scheduling problems in cloud computing, a multi-objective optimization method is proposed here. First, with an aim toward the biodiversity of resources and tasks in cloud computing, we propose a resource cost model that defines the demand of tasks on resources with more details. This model reflects the relationship between the user's resource costs and the budget costs. A multi-objective optimization scheduling method has been proposed based on this resource cost model. This method considers the makespan and the user's budget costs as constraints of the optimization problem, achieving multi-objective optimization of both performance and cost. An improved ant colony algorithm has been proposed to solve this problem. Two constraint functions were used to evaluate and provide feedback regarding the performance and budget cost. These two constraint functions made the algorithm adjust the quality of the solution in a timely manner based on feedback in order to achieve the optimal solution. Some simulation experiments were designed to evaluate this method's performance using four metrics: 1) the makespan; 2) cost; 3) deadline violation rate; and 4) resource utilization. Experimental results show that based on these four metrics, a multi-objective optimization method is better than other similar methods, especially as it increased 56.6% in the best case scenario.

INDEX TERMS Cloud computing, ant colony, task scheduling, deadline, cost constraint.

I. INTRODUCTION

Task scheduling is important in cloud computing [1] because it directly affects a systems load and performance. An effective task scheduling method requires not only meeting the users needs but also improving the efficiency of the whole system. Task scheduling problems are a typical NP-hard problem. At present, many researchers have solved this problem using the ant colony algorithm. The ant colony algorithm is a probabilistic and uncertain global optimization algorithm; therefore, it is easy to obtain a global optimal solution. Additionally, it is robust and does not rely on the strict mathematical optimization and structural features of the problem itself.

The ant colony algorithm has been used for all kinds of scheduling problems, achieving promising results [2]–[5]. However, the existing studies did not provide a detailed definition of the demands of these tasks for various resources. In cloud computing, resources and tasks are all diverse. For example, some tasks have a high demand for the CPU, while some require more storage. The costs of different resources vary. Accordingly, the costs of tasks are also different. Therefore, it is helpful to reflect task costs if we consider

the demand difference of the tasks for the resources in detail. In order to solve this problem, we propose a resource cost model. This model can reflect the demands of the tasks for the resources in detail. In addition, this paper proposes a multi-objective optimization-scheduling model that takes into account the two constraints of performance and budget cost. Lastly, it solves the multi-objective optimization-scheduling problem using the ant colony algorithm, which has a great advantage in addressing this combinatorial optimization problem. However, when using the ant colony algorithm, it is easy to fall into a local optimum. Therefore, this paper proposes an improved ant colony algorithm that can evaluate and adjust the quality of the solution in order to avoid falling into that local optimum. The primary targets of this scheduling method are performance and budget cost; because it is based on the ant colony optimization algorithm, it is named PBACO.

The contributions of this paper are summarized as follows.

- First, we propose a resource cost model that defines the demands of the task for resources in detail, and reflects the relationship between resource costs and user budget costs more clearly.
- Secondly, this paper proposes a multi-objective optimization-scheduling model. This model regards scheduling performance and cost as the budget constraints of the optimization problem. This model achieves the multi-objective optimization for the optimal span, deadline, resource utilization and user costs.
- Finally, this paper proposes an improved ant colony optimization algorithm to solve the multi-objective optimization-scheduling problem. The improved ant colony optimization algorithm can evaluate and adjust the quality of the solution in a timely manner by using two functions of performance and budget cost constraints. Thus the improved ant colony algorithm solves the problem that the original ant colony algorithm always falls into the local optimum.

The rest of the paper is organized as follows. The related work is introduced in Section II. The problem description, the definitions of tasks and resources are shown in Section III. The resource cost model and scheduling optimization model are presented in Section IV. The advanced ant colony optimization method is performed in Section V. Experiments are shown in Section VI, and the paper is concluded in Section VII.

II. RELATED WORK

Related research on task scheduling is usually related to the following three aspects: First, research focuses on scheduling performance, including response time, the best span and completion time. Secondly, there are many multi-objective optimization scheduling researches, which include the completion time, the economic cost, QoS, and energy consumption and so on. Thirdly, some scheduling methods use the ant colony algorithm.

A. FOCUS ON SCHEDULING PERFORMANCE

This type of research regards scheduling time as the main target. Researchers primarily use heuristics, or intelligent optimization algorithms to optimize task scheduling at the algorithm level. The focus is on reducing the time associated with scheduling performance, such as the response time, optimal span and completion time [6]–[9]. For example, reference [6] proposes a super heuristic algorithm the main target of which is the optimum span. Reference [7] proposes an adaptive scheduling algorithm on the Hadoop platform to reduce the completion time. However, [8] proposes two scheduling algorithms for three artificial neural networks (ANN) and an RBF neural network combined with a neural network. These two algorithms are based on the direct search algorithm (DSO), which has been proposed in their research early days. The primary target is to achieve the optimum span.

B. THE MULTI-OBJECTIVE OPTIMIZATION SCHEDULING METHODS

More and more researches focus on multi-objective optimization. For example, the multi-objective optimization includes the completion time, the constraints of QoS [10]–[15], energy consumption [16]–[18], [18], [20], economic cost [10], [21], [22], and the system performance [23]–[27].

The researches [14], [15] take into account the deadline guarantees. Where the paper [14] proposes a task scheduling to guarantee deadline by improving resource utilization, and the paper [21] proposes a method by dividing resources and budget to minimize the completion time of tasks and improve resource utilization. This method takes into account the status of those resources. Many other studies only optimize the task without considering the status of the resource [19], also focusing on energy by evaluating the resources. The paper [20] consolidates VMs to save energy by the ant colony system. A multi-input multi-output feedback control of dynamic resource scheduling algorithm was proposed [27] to guarantee optimal effectiveness under time constraints. This algorithm considers the task execution time, cost, and utilization of resources (CPU, Memory).

Additionally, some researchers consider all of the above targets [10]–[13], [28]–[30], where [28] researches task scheduling through a successive cooperative game approach in a hybrid cloud. The paper [28] proposes a multi-objective optimization algorithm of communications and storage-aware whose main targets are execution time, economic costs and system performance, while at the same time meeting the network bandwidth and storage constraints. The work in [30] proposes a multi-objective task scheduling method by minimizing makespan and costs in a heterogeneous multi-cloud environment. This work also considers the utilization with the exception of makespan and cost. However, it does not consider the deadline. Van den Bossche *et al.* [31] propose a model to minimize cost from a cloud users perspective. The research [32] is very similar to PBACO. It proposes a muti-objective optimization method to maximize the profit

of IaaS providers. It also considers the runtime, deadline and resource utilization. There are some experiments to compare PBACO with the research [32].

C. THE SCHEDULING METHODS BASED ON ACO ALGORITHM

There are many studies on scheduling tasks using the ant colony algorithm in cloud computing. These studies have been divided into three categories according to the focus of their targets.

First, there is a focus on scheduling efficiency, such as the completion time. For example, the paper [33] schedules tasks combined with the ant colony and bee colony algorithms. It takes SLA (Service-Level Agreement) into account and sorts tasks according to the priority of SLA. The paper [34] focuses on optimizing the total execution time of tasks (that is, the makespan), reducing the execution time by combining a multi-stage decision process and an ant colony optimization algorithm using the approximation method. The paper [35] proposes a scheduling strategy based on ant colony optimization and a two-way mechanism. That work sets a pheromone threshold to avoid premature convergence and avoids the local optimum through the two-tier search strategy and by using the pre-execution time. The work in [36] optimizes the scheduling using the ant colony algorithm and reducing the amount of candidate resources.

Secondly, some researches consider the system performance, such as load balancing. For example, the work in [37] proposes a scheduling method based on the ant colony algorithm. This work not only minimizes makespan, but also achieves load balancing by minimizing the idle time of virtual machines in order to balance the load of the virtual machines. The work done in [38] proposes an improved ant colony algorithm based on the shortest delay time of tasks, taking into consideration both equity and efficiency. The paper [39] proposes an enhanced ant colony algorithm. This work regards one assignment of the task to a virtual machine as the searching object of the ants, taking into account the shortest completion time and load balancing.

Thirdly, the research methods consider the cost. For example the work in [3] proposes an ant colony optimization algorithm to solve the problem of big workflow scheduling and multiple QoS parameters. This research allows users to specify their own preferences for service and QoS threshold. It also takes into consideration budget price constraints. However, this method is used for workflow scheduling in grid computing. The work in [40] proposes a scheduling algorithm based on QoS constraints combined with GA and integration, taking into account both time and cost. However, time and cost are regarded as fitness functions instead of optimization constraints.

There are several problems in the existing research work using the above analysis. First, when considering the cost of the tasks, these researches do not define the demands of tasks for resources in detail. Costs are different for different resources because of the diversity in those resources and tasks in cloud computing. Correspondingly, task costs are also different. Therefore, it is necessary to consider the differences in the demand of tasks for resources, which can reflect the task costs in detail. Secondly, many scheduling methods do not consider the quality of the solution or evaluate feedback. It will cause local optima because of the shortcoming of ant algorithm itself. This paper will evaluate the quality of the solution and the feedback from both the performance and cost, and then adjust itself according to feedback results.

III. THE DEFINITION AND DESCRIPTION OF PROBLEM

This section describes the definitions of tasks and resources, and the system framework in this paper. The primary parameters and their meanings are listed in Table 1.

TABLE 1. Main notation definitions.

Symbol	Definitions
T_i	The task i,$1 \leq j \leq K$
U_j	The resource j,$1 \leq i \leq N$
N, K	The amount of resources and tasks
C_i, M_i	CPU and memory of T_i
D_i	The deadline of task T_i
B_i	The budget cost of task T_i
C_j, M_j	CPU and memory of R_j
$C_{cost}(j), M_{cost}(j)$	The costs of CPU and memory of R_j
C_{base}	The base cost of CPU under the lowest usage
M_{base}	The base cost of memory under 1GB memory
t_{ij}	The duration time of task T_i in resource R_j
C_{Trans}, M_{Trans}	The transmission cost associated with CPU and memory
α, β	The weight factors of the heuristic information and pheromone
γ, δ	The weight factors of the performance and cost
ρ	The pheromone evaporation factor
D_v	The deadline violation rate
n_d	The number violating the deadline time in K tasks

A. THE DEFINITION OF TASKS AND RESOURCES

First, it is assumed that there are K tasks $T = \{T_1, T_2, \ldots, T_i, \ldots, T_K\}$ and N resources $R = \{R_1, R_2, \ldots, R_j, \ldots, R_N\}$ in the current system of cloud computing. Here, cloud resources refer to the virtual resources.

Definition 1 (Tasks): $T_i = (C_i, M_i, D_i, L_i)$. The first two parameters are CPU usage and memory usage, which the user has applied. D_i is the deadline of the task, and B_i is the budget cost of the user. These parameters come from the task manager and are submitted by users.

Definition 2 (Resources): Each virtual resource cloud is defined by the main parameters of its CPU and memory. That is to say, $R_j = (C_j, M_j)$. These two parameters are representative of CPU utilization and memory usage.

Assumption 1: In order for the research to proceed, it is necessary to provide an assumption related to the above definitions. We have assumed that the information submitted by the user is trusted. In other words, the information of resource demand, submitted by the user, is accurate.

In cloud computing, virtualization technology can monitor resource usage. If the user-accessed resources, such as CPU and memory, exceed the number of users applying during

the process of actual implementation, the system will cut off task performance, in which case the task fails. Therefore, assumption 1 is reasonable.

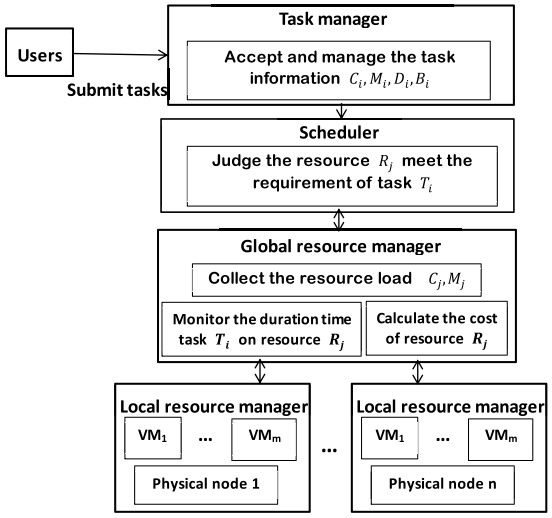


FIGURE 1. The system framework model.

B. THE SYSTEM MODEL

In this paper, the system framework model is shown in Fig. 1; here, the task manager accepts and manages task requests that the user submits, and then submits this information to the scheduler.

The local resource manager monitors and manages local resource nodes, periodically monitoring local virtual resources to obtain their CPU and memory load information, and then to submit the information to the global resource manager. The global resource manager periodically collects and updates information from local resource managers. Additionally, the local resource manager monitors the duration time of tasks running on resources and then submits that time to the global manager. The global manager calculates the resource cost using this information from the resource model.

The scheduler is the core component and is responsible for allocating tasks to resources using the optimization scheduling method, which this paper proposes, and schedules tasks to resources according to this information. First, it collects this task and resource information from the task manager and the global resource manager. Second, it judges whether the resource R_j meets the requirement of the task T_i. The requirements include deadline, cost and include the actual requirements.

Finally, the scheduler allocates the resource R_j to the task T_i.

IV. MODELS OF RESOURCE COST AND TASK SCHEDULING

This section first presents a resource cost model to reflect the relationship between resource costs and user budget. It also proposes a multi-objective optimization scheduling model based on the resource cost model to achieve multi-objective optimization scheduling in cloud computing.

A. THE RESOURCE COST MODEL

In cloud computing, resources and tasks are diverse. For example, some tasks demand many CPU resources, while others require more storage. Furthermore, the costs for different resources are different; correspondingly, task costs are also different. Therefore, if we consider the differences in the task demand for resources, it will be conducive to reflect the costs of those tasks in more detail and reflect the relationship between resource costs and user budget costs. To address this problem, this paper presents a resource cost model, which divides the resource cost into two parts of CPU and memory.

The resource cost includes two parts of the CPU and memory according to the definition of resources. The cost of CPU is defined as formula 1.

$$C_{cost}(j) = C_{base} \times C_j \times t_{ij} + C_{Trans} \tag{1}$$

Here, C_{base} is the base cost when a resource is used by the lowest utilization. t_{ij} is the duration time the task T_i runs in resource R_j. C_{Trans} is the cost associated with the CPU transmission.

The cost of memory is defined as formula 2.

$$M_{cost}(j) = M_{base} \times M_j \times t_{ij} + M_{Trans} \tag{2}$$

Similarly, M_{base} is the base cost when memory is 1 GB. t_{ij} is the duration time of the task T_i running in resource R_j. M_{Trans} is the cost associated with the memory transmission.

The cost functions can be obtained as formulas 3 and 4 based on the above cost models of CPU and memory.

$$C(j) = \sum_{j=1}^{N} C_{cost}(j) \tag{3}$$

$$M(j) = \sum_{j=1}^{N} M_{cost}(j). \tag{4}$$

B. THE SCHEDULING OPTIMIZATION MODEL BASED ON PERFORMANCE AND BUDGET CONSTRAINTS

In cloud computing, scheduling efficiency is decided by not only scheduling performance but also by the cost of the user budget. Therefore, in this paper, a scheduling optimization model is presented based on the resource cost model and the definition of tasks and resources.

First, we have assumed that there are K tasks $T = \{T_1, T_2, \ldots, T_i, \ldots, T_K\}$ and N resources $R = \{R_1, R_2, \ldots, R_j, \ldots, R_N\}$ in the cloud computing system. Here, the scheduling problem is an optimization problem: the matter of scheduling these K tasks to N resources and achieve optimal span. At the same time, it is necessary to meet constraints of deadlines and budget costs. This is a multi-objective optimization problem and is described as

formulas 5-8.

$$Minimize \sum_{x} H(x) = F(x), B(x) \tag{5}$$

$$s.t.\ B(x) = C(x) + M(x) \tag{6}$$

$$B(x) \leq \sum_{i=1}^{K} B_i \tag{7}$$

$$F(x) \leq \sum_{i=1}^{K} D_i \tag{8}$$

Here x is a feasible solution. $F(x)$ is a function of the performance objectives that refer to makespan. $B(x)$ is the objective function of the user budget costs which consist of the task demand costs for the CPU and memory that correspond to the cost functions in the resource cost model.

This is a multi-objective optimization problem that is difficult to solve; in particular, it is difficult to obtain the optimal solution. Consequently, this paper uses the ant colony algorithm to solve this problem.

V. THE MULTI-OBJECTIVE OPTIMIZATION SCHEDULING METHOD BASED ON ANT COLONY ALGORITHM

This optimization-scheduling problem includes several targets of performance and cost. It also has two additional constraints of deadline and budget cost. It is difficult to solve this type of combinatorial optimization problem, and it is especially difficult to obtain the optimal solution. The ant colony algorithm has an advantage when it comes to solving a combinatorial optimization problem. The ant colony algorithm has been used for a variety of scheduling problems and has achieved good results. However, the ant colony algorithm has a shortcoming that it is easy to fall into local optimal solution. Therefore, this paper presents an improved ant colony optimization algorithm. It evaluates the quality of the solution and provides feedback for the evaluation results using a suitable function. It can thus avoid the local optima.

A. THE ANT COLONY OPTIMIZATION SCHEDULING METHOD

The ant colony optimization algorithm is a distributed algorithm that is used to solve combinatorial optimization problems. The algorithm completes the scheduling process by simulating the foraging process of ants. At first, ants choose a path randomly. When the ants reach their desired targets, they calculate the path of fitness, at which point the ants set pheromone on the path according to fitness. Lastly, in order to focus the ants on the high fitness path and to achieve the optimal solution as frequently as possible, it is necessary to update the pheromone and behavior choices.

1) THE TRANSITION PROBABILITIES OF BEHAVIOR CHOICE

When tasks are scheduled through the ant colony algorithm, the steps are as follows. First, input the number of tasks, the task deadline and budget costs, the number of resources, their ability, and other relevant parameters. Secondly, each task is assigned an ant. When the task T_i is successfully allocated to resource R_j task T_i will be recorded through the taboo table. Third, the above steps are repeated for the next task, which is not in the taboo table until all tasks are scheduled completely. This process of assigning tasks to resources mimics the process of the ant forming a path. The choice process of assigning tasks to resources has an important relationship with pheromone and heuristic information. Therefore, in order to achieve the optimal solution, the following method was used to achieve behavioral choices: We assumed that $g_k(T_i, R_j)$ is the resource set that meets the deadline and the budget constraints of task T_i in the $k-th$ iteration. Consequently, the probability of task T_i scheduled to resource R_j is as formula 9 in the $k-th$ iteration.

$$P_k(T_i, R_j) = \begin{cases} \dfrac{[\tau(T_i, R_j)]^{\alpha}[\eta(T_i, R_j)]^{\beta}}{\sum_{h \in g_k(T_i, R_j)} [\tau(T_i, h)]^{\alpha}[\eta(T_i, h)]^{\beta}}, & R_j \in g_k(T_i, R_j) \\ 0, & otherwise \end{cases} \tag{9}$$

Here $\tau(T_i, R_j)$ is the pheromone of the task T_i assigned to the resource R_j in the path. $\eta(T_i, R_j)$ is the heuristic information, which is set to the reciprocal of start time of task T_i. The parameters α and β are the weight factors of the heuristic information and pheromone. They represent the relative important degree of the heuristic information and pheromone.

2) THE FITNESS FUNCTION

When an ant traverses all tasks, this forms a path. This path is a feasible solution to the problem. In order to ensure the quality of the solution, avoid falling into local optimum and instead achieve the optimal solution as much as possible; a fitness function is used to evaluate the quality of feasible solutions. The fitness function needs to be set based on the optimization model problems. According to the scheduling optimization model in this paper, there are two scheduling targets of makespan and costs minimized. Therefore, the fitness function of the formula 10 is as the evaluation function:

$$Fit(x) = \gamma e^{(-F(x))} + \delta e^{-B(x)} \tag{10}$$

Here, γ and δ are the weight factors of the performance and cost, $\gamma > \delta$, and $\gamma, \delta \in (0, 1)$. $F(x)$ and $B(x)$ are the performance and cost objective functions, respectively. Their values are smaller, and the values of makespan and cost are smaller, which means the fitness of the method is higher.

3) UPDATING PHEROMONE

If the fitness of a path is high, the pheromone of the path should strengthen to allow more ants to find this path. As such, it is necessary to update the pheromone for each side of the path. The updating rule is shown as formula 11.

$$\tau(T_i, R_j) = (1 - \rho) \cdot \tau(T_i, R_j) + \Delta\tau(T_i, R_j) \tag{11}$$

where ρ is the pheromone evaporation factor, and $\Delta\tau(T_i, R_j)$ is the incremental amount of the pheromone.

The fitness of the path is higher, and the incremental amount is bigger.

$$\Delta\tau(T_i, R_j) = \begin{cases} Q(\gamma e^{(-f(x))} + \delta e^{(-b(x))}), & (T_i, R_j) \in Path_l \\ 0, \quad otherwise \end{cases} \tag{12}$$

wherein Q is the constant and its value is taken as 100. $F(x)$ and $B(x)$ are smaller, and the incremental amount of pheromone is higher. The good solution will be enhanced and the poor solution will be reduced by the pheromone update. After several iterations, more and more ants will tend toward the optimal path. The pheromone evaporation factor ρ is used to prevent the local optimum.

The implementation process of the ant colony optimization – scheduling algorithm is shown as the pseudo-code of Algorithm 1.

Algorithm 1 Multi-Objective Ant Colony Optimization Scheduling Method

Input:
$T_1, T_2, \ldots, T_i, \ldots, T_K, R_1, R_2, \ldots, R_j, \ldots, R_N, iter_{max}$
Output:
map (T_i, R_j)

```
BEGIN
  Initialize ants distribution among R_j;
  DO
    FOR each ant do
      FOR each T_i do
        Select next route;;
      END FOR
      Evaluate fitness of individual path by formula 10;
      IF r_j meets the optimization problem Then
        Output the map (T_i, R_j);
        Update pheromone along its path by formula 11 and 12;
      END IF
      END FOR
    UNTIL iter_max
END
```

Where $iter_{max}$ is the maximum number of iterations, which has been set to 100 in this paper.

B. COMPLEXITY ANALYSIS OF ANT COLONY OPTIMIZATION SCHEDULING METHOD

The method's time complexity has been divided into two parts. The first is to find the optimal path. The algorithm complexity is $O(K)$. The second part is the optimization judgment to meet the cost and performance constraints. The complexity is $O(KN)$. So the overall complexity of the algorithm is $O(KN)$.

For space complexity, it is clear that the number of tasks, the number of resources and the number of ants are all constants. At the same time, the method does not involve dynamic variables, or recursive operations. Therefore, it does not need extra space and the space complexity is $O(1)$.

VI. SIMULATION EXPERIMENTS

There are two kinds of experiments. One is simulation experiments, the other is the real application instance experiments. Some simulation experiments were designed using Cloudsim 3.0 [41] to verify the performance of PBACO.

A. EXPERIMENTS AND PARAMETERS SETUP

The experiments compared the scheduling policy of this paper with the original colony algorithm, the classical heuristic algorithm Min-Min algorithm [42], [43] and FCFS scheduling. The original ant colony algorithm was only used to schedule without considering budget constraints. The Min-Min algorithm has an advantage on scheduling time and focuses on optimizing completion times.

Our experiments generated a data center using Cloudsim 3.0. There were 100 hosts and 10 virtual machines on each host. The parameter setup of VMs in the data center is shown in Table 2. The parameter setup of tasks is shown in Table 3.

TABLE 2. The parameter setup of VMs in data center.

Parameter	Value
CPU computing ability	1860 MIPs, 2660 MIPs
RAM	4096 MB
Band Width	100 M/s
Storage	10 G

TABLE 3. The parameter setup of task in data center.

Parameter	Value
Length (CPU)	[400, 1000] MIPs
File size	[200, 1000] MB
Output size (Memory)	[20, 40] MB
The number of tasks	[100, 600]

Additionally, the number of ants in the ant colony algorithm and in the original ant colony algorithm is 10 in the experiments. The number of iterations is 100. Our experiments tested and compared the performance of 10 groups of different α, β and ρ parameters. We then selected the best set of parameters as the parameters in experiments. α, β were taken 3 and 2, respectively. ρ has been set to 0.01.

B. EXPERIMENT METRICS

The experiments used four performance-evaluation indicators: makespan, which is the total time of all tasks and is used to evaluate scheduling performance; cost of user, which was different because of different scheduling methods and resources; deadline violation rate, which is the feedback effect of QoS because a perfect stable system requires feedback to verify its performance; the fourth indicator is resource utilization.

The cost was calculated using resource cost and the completion of tasks, and the price refers to Amazon cloud service charges price. $C_{base} = 0.17/hour$, $M_{base} = 0.05/GB/hour$. Additionally, transmission costs for CPU and memory have been set as follows: $C_{Trans} = 0.005$, $M_{Trans} = 0.50$ [42].

The deadline violation rate is also shown as the scheduling performance, which was obtained using this form. If the running time (response time + completion time) of task T_i is greater than the deadline D_i, the task is considered to violate the deadline constraints. The deadline violation rate D_v is calculated as formula 13:

$$v = \frac{n_d}{K} * 100\% \tag{13}$$

where n_d is the number violating the deadline time in K task.

C. MAKESPAN

100-600 tasks were submitted 10 times repeatedly. The results output the execution time of each task, after which point makespan was calculated to get the mean value. There are two different arrival rates of 10 and 80 per second. The results are shown in Figs. 2 and 3.

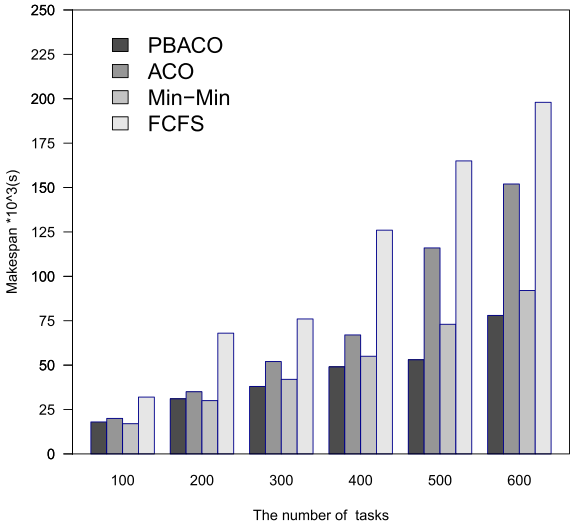


FIGURE 2. Makespan, the arrival rate = 10.

Figs. 2 and 3 show the makespan of four methods with different arrival rates. First, the different arrival rates affect the makespan results. When the arrival rate is bigger, the response time is greater. Therefore, the makespan is higher when the arrival rate is 80 per second.

Secondly, the makespan is different for different methods. It is clear that the performance of PBACO is the best. When the number of tasks is lower, the makespan for PBACO is similar to the Min-Min method. However, as the number of tasks increase, the PBACO method shows a great advantage. At its worst, PBACO is approximately equal to the Min-Min algorithm, which has advantages regarding completion time. At its best, PBACO increases nearly 56.6% relative to the FCFS algorithm. Because this method uses the evaluation function of the performance to assess the quality of the solution, it is easier to find the optimal solution than the original colony algorithm. This method is also superior to the original colony algorithm.

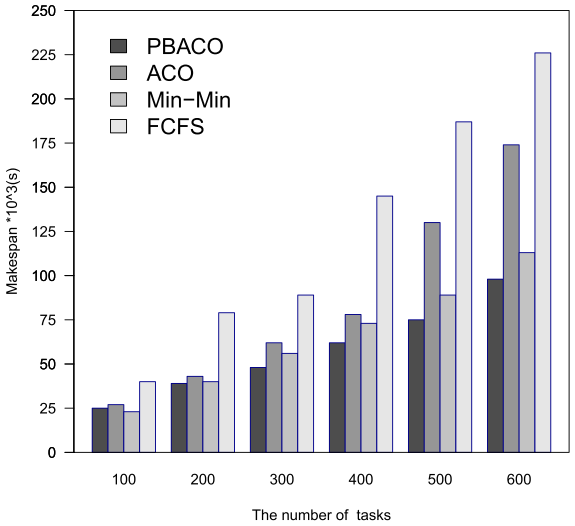


FIGURE 3. Makespan, the arrival rate = 80.

D. COSTS

The second experiment verifies the costs of the four methods, 200 and 600 tasks at different deadlines. PBACO used the resource cost model and cost constraints in this paper, while the other three methods only considered the deadline, without considering the budget cost constraints. The results are shown in Figs. 4 and 5.

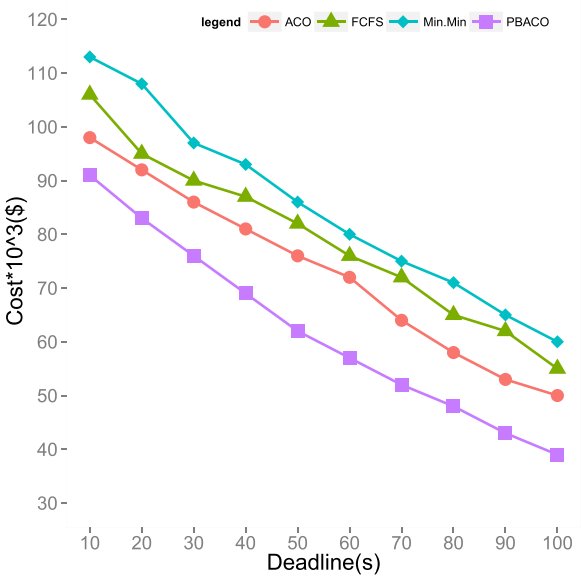


FIGURE 4. Costs with different deadline, $K = 200$.

Figs. 4 and 5 show the costs of 200 and 600 tasks. When the number of tasks is less, the costs of various methods are not very different. PBACO is the best. The reducing range is

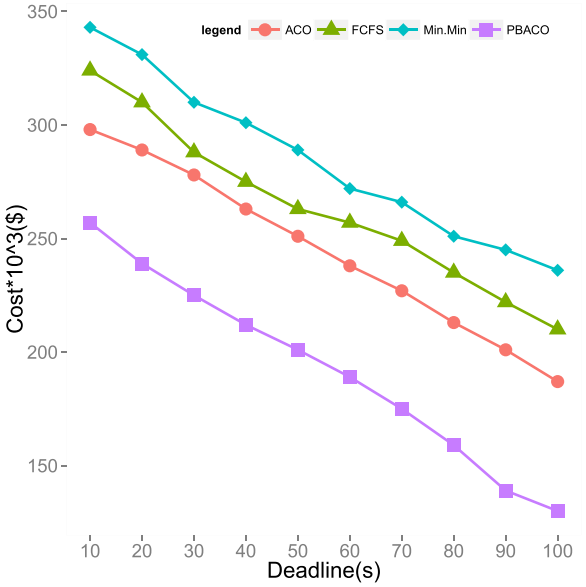


FIGURE 5. Costs with different deadline, $K = 600$.

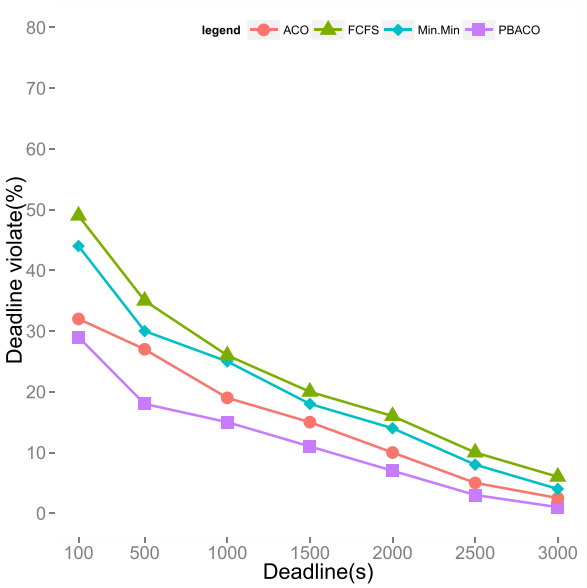


FIGURE 6. The violation rate of deadline, $K = 200$.

from 7% to 23% compared to other methods. However, as the number of tasks increased, the differences of the four methods likewise increased. PBACO had more obvious advantages; it reduced by 38% relative to the Min-Min algorithm at its best. This is because PBACO increases the cost constraints relative to other methods and uses the cost function to evaluate and adjust the quality of the solution.

Figs. 4 and 5 also reflect the relationship between resource costs and deadlines. In a tight deadline, the task requires better and more performance resources for completion. Therefore, the corresponding costs will be higher, which is consistent with the resource cost model.

E. THE VIOLATION RATE OF DEADLINE

The third experiment verified scheduling QoS through the deadline violation rates. It selected 200 and 600 tasks and set some of the larger tasks beyond the deadline. The deadline violation rates for different task deadline are shown in Figs. 6 and 7.

Figs. 6 and 7 show the deadline violation rates for different task deadlines. It is obvious that the deadline violation rates have been lower with a lower number of tasks. Fewer tasks have more resources to select. At the same time, the deadline violation rates relating to tight deadlines are high for all four methods, and decline when the deadlines loosen.

PBACO is still the best of the four methods and reduces by 34% compared to FCFS algorithm at its best. The PBACO method takes into account the deadline constraints; at the same time it evaluates and adjusts the quality of the solutions through the evaluation function of the performance target.

F. THE RESOURCE UTILIZATION

Ten resources were selected and scheduled 100 times by the four methods. Each resources instance was used as the

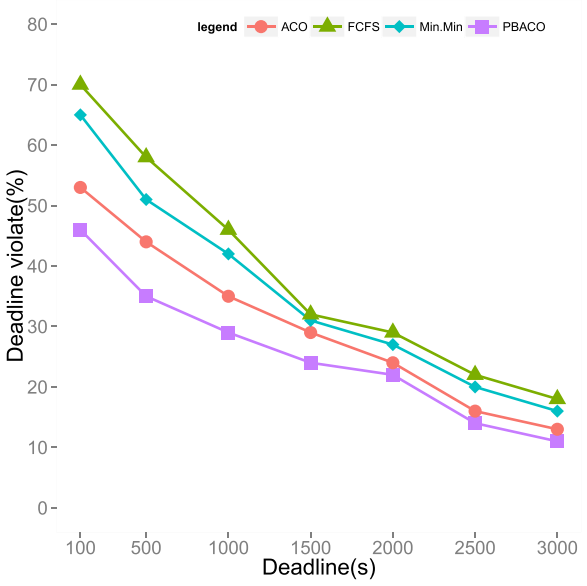


FIGURE 7. The violation rate of deadline, $K = 600$.

resource utilization for each scheduling. The results are shown in Figs. 8 and 9.

Figs. 8 and 9 shows the performance of resource utilization and load balancing with different deadlines. When the deadline is tight, the tasks tend to schedule several resources R1, R4, and R9. Therefore, the resource utilization and load balancing performance are worse. The reason of the tasks tending to schedule R1, R4, and R9 is as follows. The CPU computing ability for these three resources is all 2660 MIPs, and the remaining resources are 1860 MIPs. The original load of these three resources is lower than the others.

However, PBACO is still the best of the four methods. The difference for the resource utilization of PBACO is less than

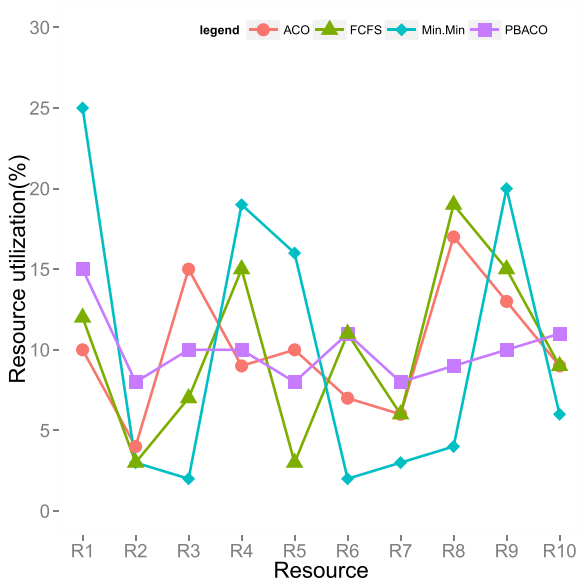


FIGURE 8. The resource utilization, the deadline ≥100.

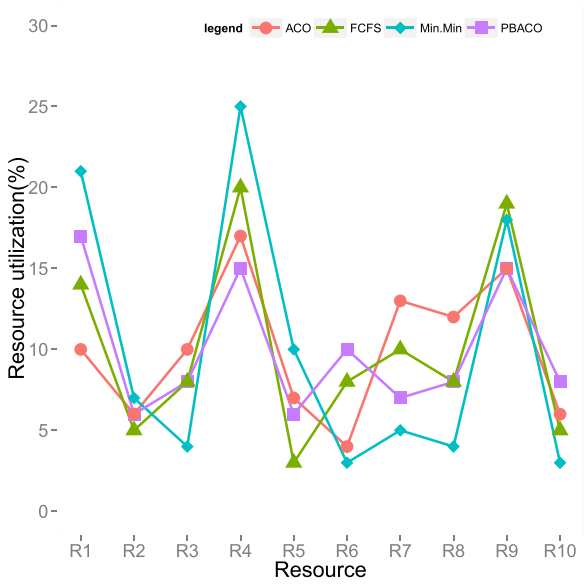


FIGURE 9. The resource utilization, the deadline <100.

the other methods. This means PBACO achieve load balancing as often as possible because the PBACO performance evaluation function evaluates and adjusts the quality of the solution in a timely manner and avoids falling into the local optimal solution. Thus it prevents the task and always tends to select good resource performance, which makes these good resources overload, consequently impacting efficient task execution. At the same time, PBACO prevents some light resource load very light so that it does not waste resources.

VII. APPLICATION INSTANCE EXPERIMENTS

In order to further verify the performance of PBACO, some application instance experiments were designed. These application instances were used as IaaS applications in three papers [31], [32], [44]. Zuo *et al.* [32] also used bionic intelligence (PSO) algorithm for multi-objective optimization scheduling whose research goal and method are very similar to PBACO. Therefore, PBACO were compared with the methods of SLPSO-SA(SLPSO-based scheduling approach) and SPSO-SA(standard PSO-based scheduling approach) in paper [32].

The application instance 1 (termed app 1 in the following tables and figures) includes eight small applications, parameters are as shown in table 4. And the application instance 2 and 3 (termed app 2 and 3 in the following tables and figures) includes five and eight larger applications, respectively. There parameters are as shown in table 5. All the deadline and runtime of each application are integer, and are uniformly distribution because PBACO, SLPSO-SA and SPSO-SA need restrict the search space.

TABLE 4. The parameter setup of application instances.

Parameter	Value(App 1)	Value(App 2 and 3)
Number of tasks	unif[1,5]	unif[1,50]
VM instance type	unif[1,3]	unif[1,3]
Deadline(hours)	unif[1,5]	unif[1,168]
Runtime(hours)	unif[1,Deadline]	unif[1,Deadline]

TABLE 5. The parameter setup of cloud resources.

Resources	Number(App 1)	Number(App 2 and 3)
CPU	20	512
Memory	40GB	1024GB

The experiments still used these four performance-evaluation indicators: makespan, cost, deadline violation rate and resource utilization. Resource utilization is the average CPU and memory utilization, respectively, which are calculated as formula 14 and 15.

$$U_{CPU} = \frac{\sum_{i=1}^{K} C_i}{total_{CPU}} * 100\% \tag{14}$$

$$U_{Memory} = \frac{\sum_{i=1}^{K} M_i}{total_{Memory}} * 100\% \tag{15}$$

where K is the number of tasks. $total_{CPU}$ and $total_{Memory}$ are the the total number of CPU and memory.

A. MAKESPAN

First, the experiment obtained the makespans of three application instances. The results are shown in Figs. 10 and 11. The values of application instance 1, 2 and 3 are very different. Therefore, in order to show clearly, two figures are used to show the results.

Figs. 10 and 11 show the makespan of three methods. PBACO is the best of three methods. This is because the main optimization goal of PBACO is to minimize the makespan, while SLPSO-SA regards the profit as the main optimization goal. Therefore, the results are reasonable and show the effectiveness of PBACO.

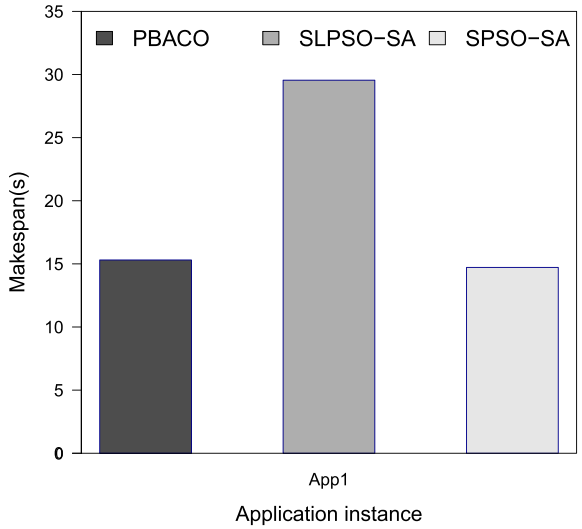


FIGURE 10. Makespan of Application instance 1.

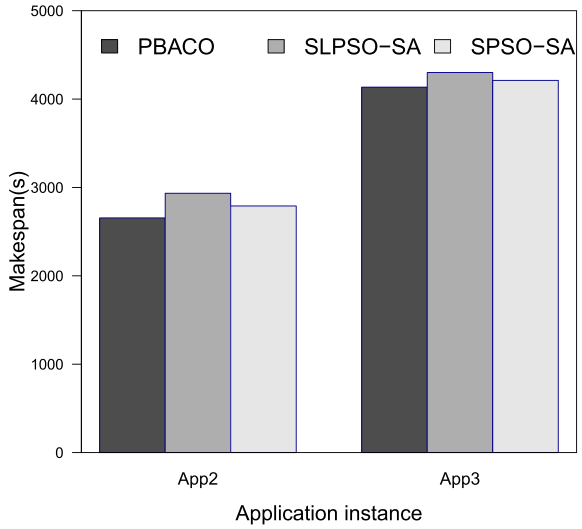


FIGURE 11. Makespan of Application instance 2 and 3.

B. COST

Second, the calculation method of cost and the parameters are slightly different between PBACO and SLPSO-SA. So here, this experiment used the price and standards of Amazon cloud service charges price. The main goal SLPSO-SA is profit. In order to facilitate comparison, SLPSO-SA and SPSO-SA also use cost, which is the cost to pay for completing application instances 1, 2 and. The results are shown in figs. 12 and 13.

From figs. 12 and 13, it is very clear that the cost of PBACO is worse than SLPSO-SA and SPSO-SA. Especially, SLPSO-SA has very good performance in the instance 2 and 3. This is because the main optimization goal of SLPSO-SA is profit, accordingly, it also have a good advantage on cost.

C. THE AVERAGE VIOLATION RATE OF DEADLINE

Third, the deadline violate of SLPSO - SA was obtained according to the paper [32], and PBACO was obtained according to the formula 13. Here, the experiment used the average deadline violate of three application instances. The results are shown in fig. 14.

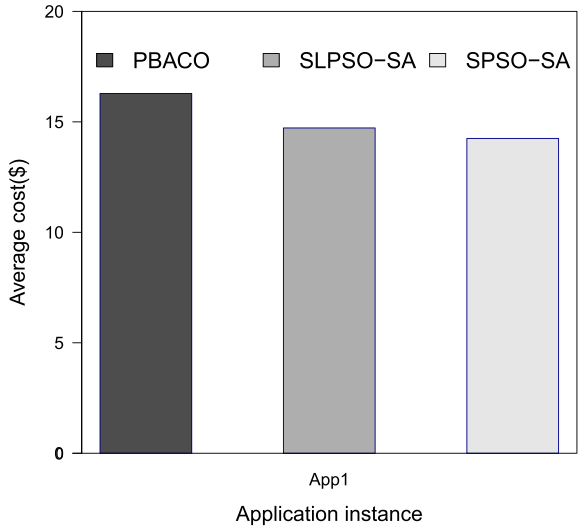


FIGURE 12. Cost of Application instance 1.

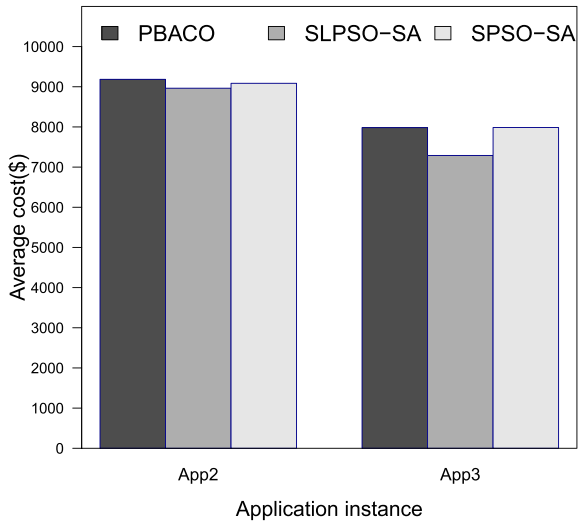


FIGURE 13. Cost of Application instance 2 and 3.

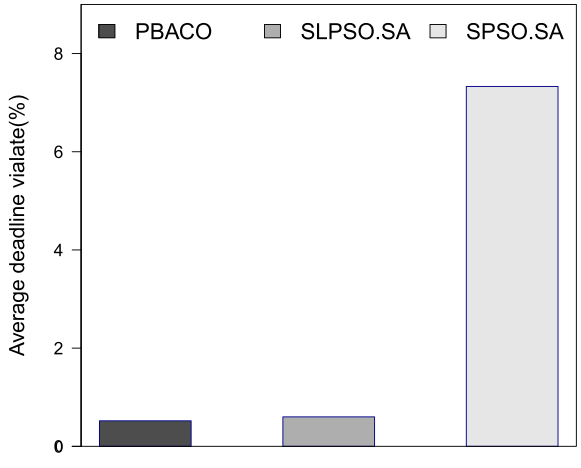


FIGURE 14. The average violation rate of deadline.

Fig. 14 shows that the deadline violate of PBACO and SLPSO-SA both are very good. And they are similar. PBACO only has a little advantage. Both of them are far better than SPAO-SA. This is because both of them were considered the deadline constraints so that they are greatly superior to SPAO-SA.

D. THE RESOURCE UTILIZATION

Finally, the experiment obtained the average CPU and memory utilization of three application instances. The results are shown in figs. 15 and 16.

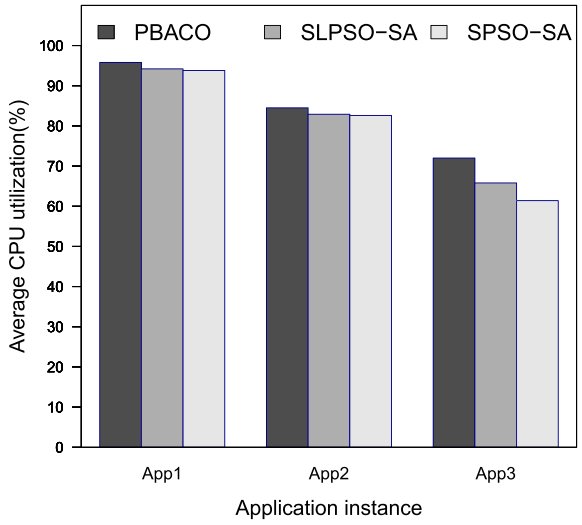


FIGURE 15. The average CPU utilization.

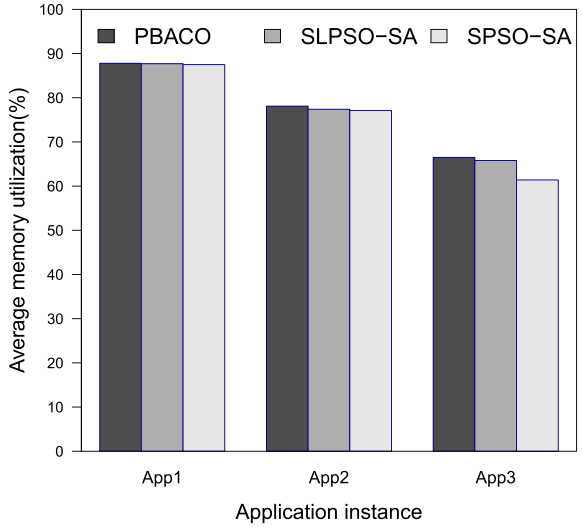


FIGURE 16. The average memory utilization.

From figs. 15 and 16, it is very clear that the average CPU utilization of PBACO is the highest in three methods. What is more, PBACO has more and more obvious advantages on application instance 3. However, for the average memory utilization, three methods are similar, PBACO only has slightly advantages on application instance 3.

VIII. CONCLUSIONS

A multi-objective optimization scheduling method (PBACO) was proposed based on the ant colony algorithm. First, the PBACO resource cost model can be used to define the task demand for resources in detail. This model reflects the relationship between resource costs and task costs. Secondly, a multi-objective optimization model was proposed, based on the model. Its main objective was to optimize the scheduling of performance and user costs. Thirdly, an improved ant colony algorithm was proposed to solve the optimization problem. In order to prevent the ant colony algorithm from falling into a local optimal solution, this method used the performance and budget constraint functions to evaluate the costs and provide feedback on the quality of the solution. It then adjusted the quality of the solution according to the results of the evaluation and feedback. Experimental results show that the PBACO method has a great advantage in terms of makespan. Even in the worst case, PBACO is approximately equal to Min-Min algorithm, which has advantages regarding completion time. At its best, PBACO increases nearly 56.6% relative to the FCFS algorithm. The PBACO methods are still better than other similar methods regarding other metrics, such as cost, the violation rate of deadline and resource utilization, proving the effectiveness of PBACO.

REFERENCES

[1] C. Zhu, V. C. M. Leung, X. Hu, L. Shu, and L. T. Yang, "A review of key issues that concern the feasibility of mobile cloud computing," in *Proc. IEEE Int. Conf. Cyber, Phys., Soc. Comput. (CPSCom)*, Aug. 2013, pp. 769–776.

[2] Y. Chen, A. Zhang, and Z. Tan, "Complexity and approximation of single machine scheduling with an operator non-availability period to minimize total completion time," *Inf. Sci.*, vol. 25, no. 1, pp. 150–163, Dec. 2013.

[3] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-efficient scheduling heuristics for deadline constrained workloads on hybrid clouds," in *Proc. IEEE 3rd Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Nov./Dec. 2011, pp. 320–327.

[4] S. K. Garg, R. Buyya, and H. J. Siegel, "Time and cost trade-off management for scheduling parallel applications on utility grids," *Future Generat. Comput. Syst.*, vol. 26, no. 8, pp. 1344–1355, Oct. 2010.

[5] H. Kellerer and V. A. Strusevich, "Fully polynomial approximation schemes for a symmetric quadratic knapsack problem and its scheduling applications," *Algorithmica*, vol. 57, no. 4, pp. 769–795, Aug. 2010.

[6] C.-W. Tsai, W.-C. Huang, M.-H. Chiang, M.-C. Chiang, and C.-S. Yang, "A hyper-heuristic scheduling algorithm for cloud," *IEEE Trans. Cloud Comput.*, vol. 2, no. 2, pp. 236–250, Apr./Jun. 2014.

[7] Z. Tang, L. Jiang, J. Zhou, K. Li, and K. Li, "A self-adaptive scheduling algorithm for reduce start time," *Future Generat. Comput. Syst.*, vols. 43–44, no. 3, pp. 51–60, Feb. 2015.

[8] B. Tripathy, S. Dash, and S. K. Padhy, "Dynamic task scheduling using a directed neural network," *J. Parallel Distrib. Comput.*, vol. 75, no. 5, pp. 101–106, Jan. 2015.

[9] H. Topcuoglu, S. Hariri, and M.-Y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 13, no. 3, pp. 260–274, Mar. 2002.

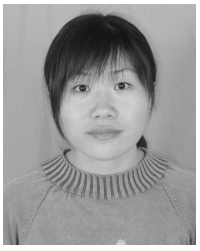
[10] Y. Wang and W. Shi, "Budget-driven scheduling algorithms for batches of MapReduce jobs in heterogeneous clouds," *IEEE Trans. Cloud Comput.*, vol. 2, no. 3, pp. 306–319, Jul./Sep. 2014.

[11] M. A. Rodriguez and R. Buyya, "Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds," *IEEE Trans. Cloud Comput.*, vol. 2, no. 2, pp. 222–235, Apr./Jun. 2014.

[12] M. Hu and B. Veeravalli, "Dynamic scheduling of hybrid real-time tasks on clusters," *IEEE Trans. Comput.*, vol. 33, no. 12, pp. 2988–2997, Dec. 2015.

[13] K. Li, X. Tang, and K. Li, "Energy-efficient stochastic task scheduling on heterogeneous computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 11, pp. 2867–2876, Nov. 2014.

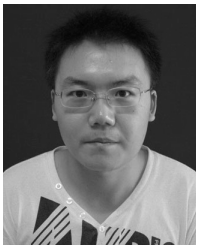
[14] S. Shin, Y. Kim, and S. Lee, "Deadline-guaranteed scheduling algorithm with improved resource utilization for cloud computing," in *Proc. 12th Annu. IEEE Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2015, pp. 814–819.
[15] B. Zhang, X. Wan, J. Luo, and X. Shen, "A nearly optimal packet scheduling algorithm for input queued switches with deadline guarantees," *IEEE Trans. Comput.*, vol. 64, no. 6, pp. 1548–1563, Jun. 2015.
[16] P. Agrawal and S. Rao, "Energy-aware scheduling of distributed systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 4, pp. 1163–1175, Oct. 2011.
[17] J. Mei, K. Li, and K. Li, "Energy-aware task scheduling in heterogeneous computing environments," *Cluster Comput.*, vol. 17, no. 2, pp. 537–550, Jun. 2014.
[18] X. Zhu, L. T. Yang, H. Chen, J. Wang, S. Yin, and X. Liu, "Real-time tasks oriented energy-aware scheduling in virtualized clouds," *IEEE Trans. Cloud Comput.*, vol. 2, no. 2, pp. 168–180, Apr./Jun. 2014.
[19] L. Zuo, L. Shu, C. Zhu, and Z. Zhou, "A resource evaluation model based on entropy optimization toward green cloud," in *Proc. 9th Int. Conf. Semantics, Knowl., Grids (SKG)*, Oct. 2013, pp. 74–81.
[20] F. Farahnakian *et al.*, "Using ant colony system to consolidate VMs for green cloud computing," *IEEE Trans. Services Comput.*, vol. 8, no. 2, pp. 187–198, Mar./Apr. 2015.
[21] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1107–1117, Jun. 2013.
[22] R. A. M. Razali, R. A. Rahman, N. Zaini, and M. Samad, "Virtual machine migration implementation in load balancing for cloud computing," in *Proc. 5th Int. Conf. Intell. Adv. Syst. (ICIAS)*, Kuala Lumpur, Malaysia, Jun. 2014, pp. 1–4.
[23] X. Tang, K. Li, Z. Zeng, and B. Veeravalli, "A novel security-driven scheduling algorithm for precedence-constrained tasks in heterogeneous distributed systems," *IEEE Trans. Comput.*, vol. 60, no. 7, pp. 1017–1029, Jul. 2011.
[24] S. Di, C.-L. Wang, and F. Cappello, "Adaptive algorithm for minimizing cloud task length with prediction errors," *IEEE Trans. Cloud Comput.*, vol. 2, no. 2, pp. 194–207, Apr./Jun. 2014.
[25] J. Zhao, K. Yang, X. Wei, Y. Ding, L. Hu, and G. Xu, "A heuristic clustering-based task deployment approach for load balancing using Bayes theorem in cloud environment," *IEEE Trans. Parallel Distrib. Syst.*, to be published.
[26] Z. Liu, W. Qu, W. Liu, Z. Li, and Y. Xu, "Resource preprocessing and optimal task scheduling in cloud computing environments," *Concurrency Comput., Pract. Exper.*, vol. 27, no. 13, pp. 3461–3482, Sep. 2015.
[27] Q. Zhu and G. Agrawal, "Resource provisioning with budget constraints for adaptive applications in cloud environments," *IEEE Trans. Services Comput.*, vol. 5, no. 4, pp. 497–511, Jun. 2012.
[28] R. Duan, R. Prodan, and X. Li, "Multi-objective game theoretic scheduling of bag-of-tasks workflows on hybrid clouds," *IEEE Trans. Cloud Comput.*, vol. 2, no. 1, pp. 29–42, Jan./Mar. 2014.
[29] J. Hu, J. Gu, G. Sun, and T. Zhao, "A scheduling strategy on load balancing of virtual machine resources in cloud computing environment," in *Proc. 3rd Int. Symp. Parallel Archit., Algorithms, Program. (PAAP)*, Dec. 2014, pp. 89–96.
[30] S. K. Panda and P. K. Jana, "A multi-objective task scheduling algorithm for heterogeneous multi-cloud environment," in *Proc. Int. Conf. Electron. Design, Comput. Netw., Autom. Verification (EDCAV)*, Jan. 2015, pp. 82–87.
[31] R. Van den Bossche, K. Vanmechelen, and J. Broeckhove, "Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, Miami, FL, USA, Jul. 2010, pp. 228–235.
[32] X. Zuo, G. Zhang, and W. Tan, "Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 2, pp. 564–573, Apr. 2014.
[33] R. Madivi and S. S. Kamath, "An hybrid bio-inspired task scheduling algorithm in cloud environment," in *Proc. Int. Conf. Comput., Commun., Netw. Technol. (ICCCNT)*, Jul. 2014, pp. 1–7.
[34] F. Ferrandi, P. L. Lanzi, C. Pilato, D. Sciuto, and A. Tumeo, "Ant colony heuristic for mapping and scheduling tasks and communications on heterogeneous embedded systems," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 29, no. 6, pp. 911–924, Jun. 2010.
[35] Y. Zhou and X. Huang, "Scheduling workflow in cloud computing based on ant colony optimization algorithm," in *Proc. 6th Int. Conf. Bus. Intell. Financial Eng. (BIFE)*, Nov. 2013, pp. 57–61.
[36] L. Zuo, S. Dong, C. Zhu, L. Shu, and G. Han, "A cloud resource evaluation model based on entropy optimization and ant colony clustering," *Comput. J.*, vol. 58, no. 6, pp. 1254–1266, Jan. 2015.
[37] R. Chaukwale and S. S. Kamath, "A modified ant colony optimization algorithm with load balancing for job shop scheduling," in *Proc. 15th Int. Conf. Adv. Comput. Technol. (ICACT)*, Sep. 2013, pp. 1–5.
[38] W. Daun, X. Fu, F. Wang, B. Wang, and H. Hu, "Cloud computing task scheduling model based on improved ant colony algorithm," *Comput. Eng.*, vol. 41, no. 2, pp. 12–16, Feb. 2015.
[39] Y. Zha and J. Yang, "Task scheduling in cloud computing based on improved ant colony optimization," *Comput. Eng. Design*, vol. 34, no. 5, pp. 1716–1719, May 2013.
[40] W. Daun, X. Fu, F. Wang, B. Wang, and H. Hu, "QoS constraints task scheduling based on genetic algorithm and ant colony algorithm under cloud computing environment," *J. Comput. Appl.*, vol. 34, no. S2, pp. 66–69, Nov. 2014.
[41] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, 2011.
[42] G. Liu, J. Li, and J. Xu, "An improved min-min algorithm in cloud computing," in *Proc. Int. Conf. Modern Comput. Sci. Appl.*, 2013, pp. 47–52.
[43] C. Zhu, X. Li, V. C. M. Leung, X. Hu, and L. T. Yang, "Job scheduling for cloud computing integrated with wireless sensor network," in *Proc. IEEE 6th Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Dec. 2014, pp. 62–69.
[44] S. He, L. Guo, and Y. Guo, "Real time elastic cloud management for limited resources," in *Proc. 4th IEEE Int. Conf. Cloud Comput.*, Washington, DC, USA, Jul. 2011, pp. 622–629.



LIYUN ZUO received the B.S. degree in engineering and management from Zhengzhou University, in 2003, and the M.S. degree from the Huazhong University of Science and Technology, in 2006. She is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, South China University of Technology. She is a Professor with the Guangdong University of Petrochemical Technology, China. Her research interests include cloud computing, resource evaluation, and scheduling optimization.

LEI SHU (M'07) received the Ph.D. degree from the National University of Ireland, Galway, Ireland, in 2010. Until 2012, he was a Specially Assigned Researcher with the Department of Multimedia Engineering, Graduate School of Information Science and Technology, Osaka University, Japan. Since 2012, he has been with the Guangdong University of Petrochemical Technology, China, as a Full Professor. Since 2013, he served as a Ph.D. Supervisor with the Dalian University of Technology and a Master Supervisor with the Beijing University of Posts and Telecommunications. Meanwhile, he is also the Vice Director of the Guangdong Provincial Key Laboratory of Petrochemical Equipment Fault Diagnosis, China. He is the Founder of the Industrial Security and Wireless Sensor Networks Laboratory. He has published over 200 papers in related conferences, journals, and books. He has an H-index of 17. His research interests include: wireless sensor networks, multimedia communication, middleware, security, and fault diagnosis. He is a member of the IEEE ComSoc, EAI, and ACM. He received the Globecom 2010 and ICC 2013 Best Paper Award. He serves as the Editor-in-Chief of *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, and an Associate Editor for a number of famous international journals. He served as more than 50 various Co-Chair for international conferences/workshops, e.g., IWCMC, ICC, ISCC, ICNC, and Chinacom, in particular, the Symposium Co-Chair of IWCMC 2012 and ICC 2012, the General Chair of Chinacom 2014 and Qshine 2015, and the Steering Chair of InisCom 2015; a TPC Member of more than 150 conferences, e.g., DCOSS, MASS, ICC, Globecom, ICCCN, WCNC, and ISCC.

SHOUBIN DONG received the Ph.D. degree in electronics engineering from the University of Science and Technology of China, in 1994. She was a Visiting Scholar with the School of Computer Science, Carnegie Mellon University, from 2001 to 2002. She is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. She is the Deputy Director of the Communication and Computer Network Laboratory of Guangdong Province, China. Her main research areas include high performance computing, big data processing, and next-generation Internet.



CHUNSHENG ZHU (S'12) received the B.E. degree in network engineering from the Dalian University of Technology, China, in 2010, and the M.Sc. degree in computer science from St. Francis Xavier University, Canada, in 2012. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, The University of British Columbia, Canada. He has over 50 papers published or accepted by refereed international journals (e.g., the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, the IEEE SYSTEMS JOURNAL, and the IEEE ACCESS) and conferences (e.g., the IEEE Globecom and the IEEE ICC). His current research interests are mainly in the areas of wireless sensor networks and cloud computing.

TAKAHIRO HARA (M'98–SM'06) received the B.E., M.E., and Dr.E. degrees from Osaka University, Osaka, Japan, in 1995, 1997, and 2000, respectively. He is currently an Associate Professor with the Department of Multimedia Engineering, Osaka University. He has published more than 100 international journal and conference papers in the areas of databases, mobile computing, peer-to-peer systems, WWW, and wireless networking. His research interests include distributed databases, peer-to-peer systems, mobile networks, and mobile computing systems. He is a member of five learned societies, including Senior Member of ACM. He served as a Program Chair of the IEEE International Conference on Mobile Data Management (2006 and 2010) and the IEEE International Conference on Advanced Information Networking and Applications (2009). He was a Guest Editor of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS of the Special Issues on Peer-to-Peer Communications and Applications. He served and serves as a PC Member of more than 120 international conferences, such as the IEEE ICNP, WWW, DASFAA, ACM MobiHoc, and ACM SAC.