Task Scheduling based on Advance Ant Colony Optimization and Particle Swarm Optimization with Machine learning in the cloud environment

Mrs. NAHLA AHMED FARAJ[1], Prof. RAMESH BABU INAMPUDI[2]

[1] Research Scholar, Department of CSE, Acharya Nagarjuna University, Guntur, India

[2] Professor, Department of CSE, Acharya Nagarjuna University, Guntur, India

[1] E-mail : nahlaalkhtry74@gmail.com,[2] rinampudi@outlook.com

Abstract

Cloud computing technology becomes more and more widely used, and one of the main issues in this cloud environment is associated with task scheduling. Many companies try to develop cloud computing systems and enhancing their services to provide to the users. To decrease non-reasonable task allocation and increase task scheduling performance in a cloud environment, a two-stage strategy model called (MLCCAP) is proposed. At the first stage, is scheduling algorithms, which applied to Advance Ant Colony Optimization (AACO) algorithm and Particle Swarm Optimization (PSO) algorithm to find the optimal resource allocation for each task in the dynamic cloud system. At the second stage, the machine learning algorithm Classifier Chains applied to the results from the two algorithms and generates a new hybrid model considering the size of the tasks and the number of the virtual machines. The experimental result shows that the (MLCCAP) minimize the average makespan compare with traditional task scheduling algorithms.

Keywords: Ant Colony Optimization (ACO), Classifier Chains, Machine learning, Particle Swarm Optimization (PSO), Task scheduling.

1. Introduction

Cloud computing has created a tool to provide massive computing resources on the Internet. In this environment, users request various resources, such as CPU, operation system, memory, access control, networking capacity, and software applications, to the cloud server provider. The cloud server providers provide the user the possible resource based on the cost that the user can pay, to enhance the user's satisfaction and benefit the owners of the resources. In the Cloud computing, different load balancing scheduling is existing, which divides the workload across multiple computing resources such as hard drives, computers, and network to ensure an appropriate utilization of resource consumption. Load balancing scheduling plays an essential role in efficient resource utilization in a cloud computing, which it is divided into two main categories of technique :

- The Batch Mode Heuristic Scheduling Algorithm (BMHA), where jobs are queued in a set and collected as batches as they arrive in the system after which they get started after a

fixed period. Examples on (BMHA) based algorithms are: First Come First Serve (FCFS) [1], Round Robin (RR) [2], Min-Max, and Min-Min [5].

- The on-line mode heuristic scheduling Algorithm (OMHA), in these mode jobs are scheduled individually as they arrive in the system. This technique more feasible in a cloud environment as the systems may have different platforms and execution speed, so the OMHA are more appropriate for a cloud environment.

Because clouds provide a finite pool of virtualized on-demand resources, optimally scheduling them has become an essential and rewarding research topic [4]. Scheduling workload in a heterogeneous cloud environment is very challenging due to limited cloud resources with varying capacities and functionalities [5]. The key issue is how to allocate user tasks to maximize the profit of cloud service providers, while guaranteeing quality-of service (QoS) for all the tasks[6].

Virtual Machines (VMs) are very important service resources for task scheduling in the cloud environment. This paper focus in two kinds of problems i) when larger task is assigned to a VM with low processing ability, its processing time is relatively long and, the task may not be completed before its deadline which is negatively affecting the whole task sequence. ii) When a small task is assigned to a VM with strong processing capacity, which is making a large task stay in a waiting state for an extra time. These two kinds of problems reducing the overall throughput of cloud computing. Perfectly, tasks should be scheduled at their most suitable VMs. One way to do so is to create proper VMs according to the exact requirements of the tasks. Unfortunately, it takes too much time to do so during scheduling. If concrete VMs are dynamically created, the time cost can be significant, sometimes up to half an hour. Therefore, it is too long for users to wait for scheduling their tasks.

In this paper, A hybrid approach, called Multi Label Classifier Chains Advance Ant Colony Optimization and Particle Swarm Optimization (MLCCAP) is proposed. This approach is based on two strategies:

- The first strategy is the scheduling algorithms, which applied on, Advance Ant Colony Optimization (AACO) algorithm and Particle Swarm Optimization (PSO) algorithm to find the optimal resource allocation for each task in the dynamic cloud system.
- The second strategy is the application of the machine learning algorithm Classifier Chains on the results from the two algorithms and generate a new hybrid model considering the size of the tasks and the number of the virtual machines. This strategy not only minimizes the makespan of a given task set, but it also adapts to the dynamic cloud computing system and balances the entire system load.

Embedding machine learning in the cloud environment is a necessary requirement. The machine learning allows scalability of cloud resources, enhance its performance and give its users a better experience using the cloud computing. The need of using ML concepts in cloud computing has been acknowledged by the research communities [14].

In (MLCCAP) framework, the machine learning algorithm required to predict a proper scheduling algorithm for every data center depending on the execution time.

2. Contribution

The major contributions of this paper are as follows. Decrease time cost by selecting a suitable algorithm to the given tasks. With the help of machine learning algorithm, the cloud provider required to predict a proper scheduling algorithm for every datacenter depending on the execution time.

3. Related work

Solymani. Z and Ghavami. B, [9] proposed an algorithm that has two parameters job length and priority jobs. If the algorithm wants to run based on the length of the jobs, it first arranges the tasks in order of length of jobs, then calculates the average length of the jobs, and then calculates the difference in the length of each jobs with the mean. This algorithm selects tasks from the middle and gives credit to each task, and queues for execution. If user wants to run according to the priority, according to the priority given by the user to the job, credit is applied and the tasks are run on the basis of their validity.

Santhosh B. et al.,2011 [10] proposed an advanced max-min scheduling algorithm for cloud computing. Which is a correction in the advanced Max-min algorithm. First, it obtains an average runtime from all available tasks, then selects the closest run-time distance to the average number. Sometimes the biggest task is too big and it causes a system imbalance. Therefore, this method always chooses a task whose length or execution time is closer to the average length or the time it runs all the tasks. This method reduces Makespan and performs load balancing over the Maximin method.

Lin, R. and Q. Li . 2016 [11] proposed a pre-allocation strategy for planning jobs based on Ant colony Optimization in cloud computing called PACO. This algorithm combines an improved ACO algorithm and template size for independent job scheduling. Their proposed algorithm performed well in the simulation environment. The experiments show that PACO can improve task scheduling performance in the cloud environment.

Jain, A. and R. Kumari 2017 [12], Introduce an algorithmic resource to reduce Makespan and increase resource productivity. In this algorithm, it provides scheduling for a large number of independent tasks to examine the performance of the Makespan and CPU parameters based on the particle optimization and the Max-Min algorithm. In more details, this algorithm combines the advantages of both algorithms in single algorithm. The traditional Particle Swarm Optimization algorithm takes a lot of time to find the optimal solution. But by combining with the Max-Min algorithm, the outputs of Max-Min algorithm can be assigned to a VM. This output is then given as input to the PSO algorithm. As a result, it helps to find the optimal solution compare to the existing algorithm.

Zhan and Huo,2012 [13] proposed improved particle swarm optimization (PSO) which the outcome of the evaluation showed that the proposed algorithm can minimize the task

average execution time(ET), and increase the rate of availability of resources in the cloud environment. Where the PSO does not solve large scale optimization, then simulated support algorithm is added into the PSO algorithm which increases the convergence speed of PSO.

4. Motivation

The fundamental problem with the task scheduling is the matchmaking between available resources and resource requests from waiting jobs or applications. With the large number of resources available and the network condition variability in Cloud environment, the computational challenges will become increasingly intricate, in this section some motivations are explained as follows:

1. The system should fully exploit the diversities and dynamicity of computing clusters at massive scale to improve task throughput, reduce the occurrence of task eviction, and autonomously handle component failures. Considering workload and server heterogeneity[7][8] is extremely important when conducting scheduling. Such heterogeneity leads to different resource capacities and unique machine characteristics.

2. For online decision making, real-time decision is urgently required for both user and cloud. Finding the machines that are determined to be most suitable for specific purposes such as workload consolidation and task scheduling. This can be done through the machine learning algorithm that comprehensively considers estimated load, correlative workload performance, and queue states.

3. Performance issues are very difficult to overcome in a white-box or top-down way due to the intricate factors and the complicated combinations that might influence the results. Therefore, learning-based approaches are especially best fit to resource management systems in the cloud, where decisions are often repetitive and the generated training data can be abundantly re-used.

5. Preliminaries

5.1. Machine Learning

Machine Learning is a field of computer science that enables computers to learn and solve problems without being explicitly programmed. Compared with direct human approaches, Machine learning approaches reduce human labor and time, especially when solving complex problems. With the help of machine learning, there are several characteristics that can be used within resource scheduling:

- *Automatic Feature Learning and Selection*. One of the most important advantages of machine learning is the capability of learning proper features after random initializing and training on giving datasets. Machine learning can be used to discover relevant features in disordered datasets, substituting manual feature selection by domain experts or technical staffs. The features that are demonstrated relative to scheduling can be very powerful and critical in targeting the optimal scheduling. Even a large group of parameters can be automatically extracted through a deep neural architecture. It is infeasible for people to find such an optimal setting for a large number of parameters manually.

- *Accurate Prediction and Inference*. Due to the fact that system utility is highly dependent on the prediction accuracy of resource, during resource mapping and scheduling, accuracy is always the top concern. In machine learning fields, accuracy (one of the performance indicators) is used to evaluate the differences between the trained model and the real model. There are a huge number of classical models such as decision tree, support vector machine and neural network[15] [16]. There are many considerations when choosing an algorithm, based on the requirements including accuracy, training time, number of parameters, and number of features.

6. Proposed system (MLCCAP)

A Multi Label Classifier Chains Advance Ant Colony Optimization and Particle Swarm Optimization in (MLCCAP) are divided to two strategies:

6.1. The first strategy

6.1.1. Advanced Ant Colony Optimization

Advanced Ant Colony Optimization: The main idea of ACO derives from the foraging behavior of ant colonies. When ants group tries to search for the food, they use a special genius kind of chemical to communicate with each other. That chemical is referred to as a pheromone. Randomly the ants start to search their foods. Once the ants find a path to the food source, they leave pheromone on the path. An ant can keep track-off the trails of the other ants to the food source by sensing pheromone on the ground. This process continues till most of the ants attract to choose the shortest path as there have been a huge amount of pheromones accumulated on this path [17]. In the task scheduling, During an iteration of the ACO algorithm, each ant k, k = 1, ...,m (m is the number of the ants), builds a tour executing n (n is the number of tasks) steps in which a probabilistic transition rule is applied. The k-ant chooses VM j for next task i with a probability that is computed by Equation 1.

$$D_{y}^{x}(t)=\frac{[\tau i\, j(t)]\alpha * [\eta i\ j]\beta}{\Sigma s \in allowedk\ [\tau is(t)]\alpha * [\eta is]\beta} \quad (1)$$

Where

- τi j(t) shows the pheromone concentration at the t time on the path between task i and VM j.
- allowedk = {0,1, · ·· ·· ·,n−1}.
- ηi j = $\frac{1}{\text{dij}}$ is visibility for the t moment, calculated with heuristic algorithm and di j which expresses the expected execution time and transfer time of task i on VM j can be computed with Eq. (2).

$$d_{ij}=\frac{TL\_Taski}{Pe\_numj * Pe\_mipsj\_VMj}+\frac{InputFilleSize}{VM\_bwj} \quad (2)$$

Where

- TL_Taski is the total length of the task that has been submitted to VMj.
- Pe_numj is the number of VMj processors.

- Pe_mipsj is the MIPS of each processor of VMj Input File Size is the length of the task before execution.
- VM_bwj is the communication bandwidth ability of the VMj.
- The two parameters α and β control the relative weight of the pheromone trail and the visibility information respectively.

Algorithm 1. shows the ACO scheduling [19] where $Cloudlet_{list}$ the list of tasks, VM_{list} the list of virtual machines, and t abu indicates the allowed VMs for ant k in next step also t abu records the traversed VM by ant k.

Algorithm 1. Scheduling based AACO

1: **Require:** α, β , $max_{iterations}$, $Cloudlet_{list}$, VM_{list}
2: **for** *i* in $Cloudlet_{list}$ and *k* in VM_{list} **do**
3: pair $Cloudlet_i, VM_K \leftarrow -\tau_{i,j}(0) = C$ // pheromone(*C*)
4: $VM_K \leftarrow Ant_j \leftarrow randomPick(Ant_{pool})$
5: $Ant^{tabu}\ j \leftarrow add\ (VM_k)$
6: **while** NOT done **do**
7: **for** *k* = 1*tom* **do**
8: $VM_N \leftarrow select(Ant_k, VM_{list}, Cloudlet_{list})$
9: $Ant^{tabu}{}_i \leftarrow add(VMs)$
10: **end for**
11: **for** *k* = 1*tom* **do**
12: $Lk \leftarrow calculate()$ // L_k the length of the current best tour done by the ants.
13: **end for**
14: $\tau_{i,j} \leftarrow update()$ // The local Pheromone value
15: $pheromone_{global} \leftarrow update()$
16: increment (iterations)
17: **end while**

6.1.2. Particle Swarm Optimization (PSO)

Particle Swarm Optimization [18], the main idea of PSO derives from the social behavior of animals such as a flock of birds finding a food source. A particle in PSO is analogous to a bird or fish flying through a search space. The movement of each particle is co-ordinated by a velocity which has both magnitude and direction. Each particle position at any instance of time is influenced by its best position and the position of the best particle in a problem space. The performance of a particle is measured by a fitness value, and in order to measure how well the particles position, the fitness function can be defined: Equation 2

$$f = \text{Min}\left(\frac{\text{VTime}}{\text{VRutilization}}\right)$$

where

VTime to denote the execution time of VMs for executing all of the tasks, VRutilization to denote the resource utilization of VMs during the process of running the tasks.

Algorithm 2. shows the PSO scheduling [27].

Algorithm 2. Scheduling based PSO

```
1: Input: Task, Particles
2: Output: gBest
3: foreach Pi do
4:     pBest = Generate initial position(Pi)
5: foreach pBest of particle Pi do
6:     gBest = Max(pBest1, pBest2, ....)
7: repeat
8:   j ←1 ;
9:    while j ≤ m do
10:     Select the task t j;
11:     Calculate est(t j);
12:     Allocate task(t j);
13:     j++;
14:     foreach particle Pi do
15:         Calculatecur particle f it; (current particle)
16:         if cur particle f it ¡ pBesti f it then
17:        Update(pBesti);
18:        if cur particle fit< gBesti fit then
19:        Update( gBesti);
20:   if the termination condition is met then
21:      Output 4gBesti; Break;
22:  else
23:  foreach particle Pi do
24:   Update( Pi velocity);
25:   Update( Pi position);
26: until the termination condition is met ;
```

6.2. The second strategy

6.2.1 Multi-label classification based ACO and PSO

Multi-label classification has attracted increasing attention in the machine learning community during the past few years. In this paper, classifier chains are used[21]. As its classifier chains selects an order on the label set, a chain of labels, and trains a binary classifier for each label in this order. The difference with binary relevance (BR) is that the feature space used to induce each classifier is extended by the previous labels in the chain. These labels are treated as additional attributes, with the goal to model conditional dependence between a label and its predecessors. CC performs particularly well when being used in an ensemble framework, usually denoted as ensemble of classifier chains (ECC), which reduces the influence of the label order.

7. PERFORMANCE EVALUATION

The implemented of the proposed framework is done by using a 64-bit Windows 7 environment on a machine equipped with an Intel(R) i5-2430 processor with Nvidia (Quadro) HD Graphics 3.10 GHz Processor and 4.096 GB RAM. The proposed framework simulated using the CloudSim toolkit [15]. The results of this simulation show the advantage of the

proposed model compared to the ACO and PSO Algorithms in makespan term. The experiment is implemented with 3 data centers and 50 tasks between 30 and 2700 bytes with different numbers of VMs varying from 2 to 50 under the simulation platform. The resource situation is shown in Table 1. The computation workload of the task is 10000 MIPS.

TABLE 1. PARAMETERS SETTING OF CLOUD SIMULATOR

Type	Parameters	Value
Data Center	Number of Datacenters	3
	Number of Hosts	6
	Datacenter Cost	1-15
Virtual Machine	Total number of VMs	2-50
	VM memory(RAM)	512(MB)
	Bandwidth	1000 bit
Task	Total number of task	50
	Size of task	30-2700 bytes
	Number of PEs requirement	2

To evaluate the performance of the proposed model, the makespan of the (MLCCAP) compared with two basic algorithms. First apply the optimization algorithm when the VMs number has been fixed to 2, and the task sizes are varying between 30, and 2700 bytes. With repeating the optimization while varying the number of virtual machines from 2, to 50. Each algorithm runs 100 times, and the average makespan of these runs is shown in Figures 1, and 2. The function of the machine learning classifier chain algorithm is to choose the future algorithm to be used to schedule the tasks in every data center while having the best task execution time, by using the Gibbs sampler.

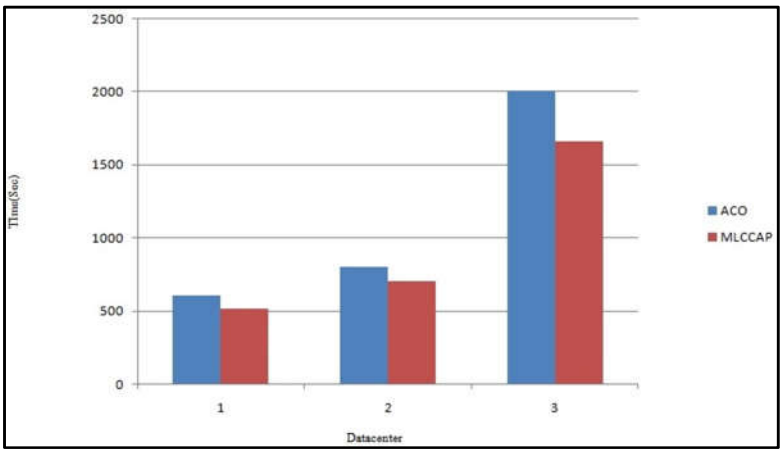

Figure 1. Compare ACO with MLCCAP

From Figures 1, the (MLCCAP) model compared with ACO algorithm for datacenter one, two and three. The results of comparing with ACO, it shows that the time has been reduced 15% on data center one, 12% on data center two and 17% on three data center.

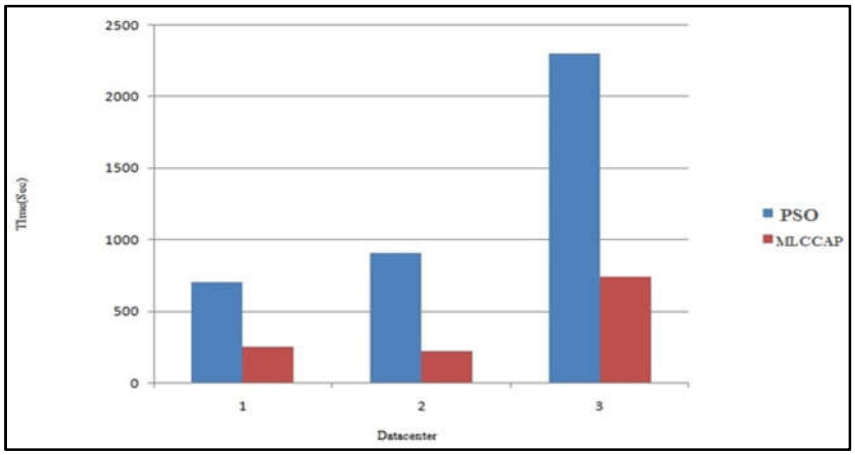

Figure 2. Compare PSO with MLCCAP

From Figures 2, the (MLCCAP) model compared with PSO algorithm for datacenter one, two and three. The results of comparing with PSO, In PSO the decrease on the makespan happens roughly and the result shows 65% , 76% and 68% decrease on the three data centers, respectively.

8. Conclusion

In this paper, an effective scheduling algorithm called (MLCCAP) was proposed to reduce the Makespan for tasks in the cloud computing environment. Using machine learning algorithm Classifier Chains in the proposed model shows that the makespan reduce on the given tasks set compare to traditional ACO and PSO algorithms. The experimental result shows that the (MLCCAP) balance the entire system load effectively, and it clearly shows reducing on the average makespan between 12 % and 76 %. This method can be adapted to existing cloud computing systems for decreasing makespan. Currently proposed model designed for checking makespan, in future, it may can extend with several machine learning techniques such as Neural networks and decision trees with consideration of other relevant metrics such as the capacity of CPU, RAM, and bandwidth.

9. References

[1] J. Li, M. Qiu, Z. Ming, G. Quan, X. Qin, and Z. Gu, "Online optimization for scheduling preemptable tasks on iaas cloud systems,"Journal of Parallel and Distributed Computing, vol. 72, no. 5, pp. 666–677, 2012.

[2] C. S. Pawar and R. B. Wagh, "Priority based dynamic resource allocation in cloud computing," in Cloud and Services Computing (ISCOS),2012 International Symposium on. IEEE, 2012, pp. 1–6.

[3] Y. Han and X. Luo, "An effective algorithm and modeling for information resources scheduling in cloud computing," in Advanced Cloud and Big Data (CBD), 2013 International Conference on. IEEE, 2013, pp.14–19.

[4] Z. H. Zhan, X. F. Liu, Y. J. Gong, J. Zhang, H. S. H. Chung, and Y. Li, “Cloud computing resource scheduling and a survey of its evolutionary approaches,” ACM Comput. Surv., vol. 47, no. 4, pp. 1–33, Jul. 2015.

[5] S. K. Panda and P. K. Jana, “Efficient task scheduling algorithms for heterogeneous multi-cloud environment,” J. Supercomput., vol. 71, no. 4, pp. 1505–1533, Apr. 2015.

[6] 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.

[7] C. Reiss, A. Tumanov, G. R. Ganger et al., “Heterogeneity and dynamicity of clouds at scale: Google trace analysis,” in ACM SoCC, 2012.

[8] I. S. Moreno, P. Garraghan et al., “Analysis, modeling and simulation of workload patterns in a large-scale utility cloud,” IEEE Transactions on Cloud Computing, 2014.

[9] Solymani. Z, Ghavami. B, "The new algorithm of scheduling and allocating resources in the cloud environment", International Conference on Computer Engineering and Information Technology, 2016.

[10] Santhosh B., Manjaiah D. H, and L.Pandma Suresh, "A Survey of Various Scheduling Algorithms in Cloud Environment", International Conference on Emerging Technological Trends [ICETT]. IEEE, 2016.

[11] Lin, R. and Q. Li, "Task Scheduling Algorithm Based on Pre-Allocation Strategy in Cloud Computing", IEEE International Conference on Cloud Computing and Big Data Analysis. 2016.

[12] Jain, A. and R. Kumari, "An Efficient Resource Utilization Based Integrated Task Scheduling Algorithm", 4th International Conference on Signal Processing and Integrated Networks (SPIN), IEEE, 2017.

[13] S.Zhan and H. Huo, “Improved pso-based task scheduling algorithm in cloud computing,” Journal of Information & Computational Science, vol. 9, no. 13, pp. 3821–3829, 2012.

[14] Gurmeet Singh, “Scope of machine learning in cloud computing”, Under the guidance of Dr. Diganta Goswami, 8-11-2010.

[15] S. Haykin, “Neural networks: A comprehensive foundation,” Neural Networks Comprehensive Foundation, pp. 71–80, 2008.

[16] J. A. Suykens and J. Vandewalle, “Least squares support vector machine classifiers,” Neural processing letters, 1999.

[17] T. Liao, T. St¨utzle, M. A. M. de Oca, and M. Dorigo, “A unified ant colony optimization algorithm for continuous optimization,” European Journal of Operational Research, vol. 234, no. 3, pp. 597–609, 2014.

[18] K.-L. Du and M. Swamy, “Particle swarm optimization,” in Search and Optimization by Metaheuristics. Springer, 2016, pp. 153–173.

[19] M. A. Tawfeek, A. El-Sisi, A. E. Keshk, and F. A. Torkey, “Cloud task scheduling based on ant colony optimization,” in Computer Engineering & Systems (ICCES), 2013 8th International Conference on. IEEE, 2013, pp. 64–69.

[20] H.Chen and W. Guo, “Real-time task scheduling algorithm for cloud computing based on particle swarm optimization ,” in International Conference on Cloud Computing and Big Data in Asia. Springer, 2015, pp. 141–152.

[21] J. Read, B. P.fahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," Machine learning, vol. 85, no. 3, pp. 333–359, 2011.

[22] H. Yang, "Improved ant colony algorithm based on pso and its application on cloud computing resource scheduling," in Advanced Materials Research, vol. 989. Trans Tech Publ, 2014, pp. 2192–2195.