Tasks Scheduling with Heterogeneity System in the Cloud Computing using ACO Algorithm

Mrs. Nahla Ahmed farag[1], Prof. Ramesh Babu Inampudi[2]
[1,2](Department of Computer Science and Engineering)

Abstract

In Cloud computing the resources are managed dynamically based on the need and demand for resources for the task. Task scheduling is a serious problem in the cloud computing that needs to be optimized. Several research studies have been conducted to improve cloud computing task scheduling using Ant algorithms. In this paper, a cloud task scheduling called Classify Ant Colony Optimization (CACO) algorithm compared with the traditional Ant Colony Optimization(ACO) algorithms to present the dynamic allocation of resources under fourcategories time,cost,cost-time, time-costand the ways each of this scheduling algorithm adapts to handle the load and have high-performance computing, therefore paper focuses on the concept that at every decision point an ant decides which task to schedule and where to map it. The experimental results show that the proposed (CACO) algorithm can effectively achieve good performance, load balancing, and improve the resource utilization.

Keywords - *Task Scheduling, Ant Colony Optimization(ACO.*

I. INTRODUCTION

Cloud computing, which is the sum of utility computing and grid computing, is viewed as the resource pool that can be used for scheduling [1]. The users can make a solution of the problem of mass data at any time, any place, by integrating and scheduling the computation and storage resources. Cloud computing has the characteristics of scalable, stretching, transparent, economy etc. [2]. The developers and users can reasonably change resource allocation strategy and virtual machine usage strategy in the cloud platform, even build an exclusive private cloud platform. Task scheduling affects the overall efficiency of cloud platform, and proper task scheduling is required so as to improve the efficiency and to minimize the execution time. The overall performance of cloud systems depends on the scheduling algorithm, and how to efficiently and rationally allocate the finite, heterogeneous and geographically distributed resources to meet the end-users requirements are an urgent issue for cloud service providers. In cloud computing, the purpose behind task scheduling is to specify a particular resource from all the available resources so that the efficiency of the computing environment increases [3].The process of task scheduling in the cloud computing is an NP-complete problem. In this process, the cloud scheduler receives the tasks from the users and maps them to available resources taking into consideration tasks characteristics, attributes, and requirements as well as the resource parameters and properties. So, a good task scheduling algorithm achieves an efficient utilization of resource with maximum profit and a high-performance computing. In the early 1990s, Ant Colony Optimization (ACO) was introduced by M. Dorigo and colleagues. The foraging behavior of ant colonies can be replicated in the simulation and inspires a class of ant algorithms known as ''Ant Colony Optimization'' (ACO). Ant colony, self-organized group of ants follows very intelligent approach for finding the shortest route between source to the Destination by Stigmergy and pheromone techniques. ACO is novel nature-inspired Meta heuristic for the solution of hard combinatorial optimization (CO) problems and it belongs to the class of Meta heuristics [5] which are approximate algorithms used to obtain good enough solutions to hard combinatorial optimization problems in a reasonable amount of computation time. This characteristic of real ant colonies is exploited in artificial ant colonies in order to solve CO problems. Dynamic and Elasticity Algorithm to perform the task scheduling by Ant colony Optimization to perform task scheduling among the Systems existing in the Data centers are presented.

The paper summarized as follows. Section I givesan introduction to cloud computing. Section II explains about related works. Overview of ACO is explained in Section III.The proposed work is explained in Section IV. Comparison with related work is explained in Section V. Section VI presents the experimental evaluation.

II. RELATED WORK

Previous researchers have proposed task scheduling techniques to minimize task execution time, maximize system performance and minimize cost using optimization algorithms such as particle swarm optimization (PSO), genetic algorithm (GA), bee colony optimization (BCO), ant colony optimization (ACO) [20, 21,22]. These techniques have contributed to further developments of ideal solutions. With changing cloud computing environment, VMs are limited to handle the volume of the task that often arrives data centers. Thus, methods applied by existing researchers still need to be handled and to be improved with some new methods.For conventional applications with independent tasks, some simple methods such as Min-Min and Max-Min [9, 10] are used to meet the QoS. In paper [11], in order to reduce the time, Suraj Pandey used Partical Swarm Optimization (PSO) to schedule the workflow.Time was also treated as optimization objective in the literature [12] and [13]. Paper [14] proposed Multiobjective Differential Evolution(MODE). WeiNeng Chen and Jun Zhang [15] used ACO to solve workflow with various QoS requirements in grid computing. Genetic-based optimization techniques also have been used to solve scheduling problems in Grid environments, for review, readers are referred to [16, 17, 18]. Although these approaches worked effectively in a grid environment, they couldn't be directly applied to solve scheduling problems in cloud computing, because cloud computing is more commercialized than grid computing. In cloud computing, Meng Xu [19] introduced a Multiple QoS constraining scheduling strategy of Multi-Workflows (MQMW) to solve workflow scheduling problem. Zhang Jun Wu[18] used ACO, GA, and PSO to solve workflow scheduling problem, the experimental results show that the performance of ACO based scheduling algorithm is better than others.

III. ANT COLONY OPTIMIZATION (ACO)

In the early 1990s, Ant Colony Optimization (ACO) was introduced by M. Dorigo and colleagues. ACO algorithm has been successfully applied to many NP-hard optimization problems [5, 6]. One of the first problems to which ACO was successfully applied was the traveling salesman problem (TSP) [7]. In the ACO algorithm, many cooperating agents called artificial ants are used to search the optimal solution. More specifically, the ACO algorithm can be depicted as the following:

A) Initialization of the algorithms. Parameters used in this algorithm are initialized. Intensity of pheromone is set to a small constant.

B) Initialization of ants. Assume that there is a group of M artificial ants in the algorithm. For scheduling of tasks, the number of ants taken is less than or equal to number of tasks [8]. At the beginning, all ants are set to the initial state. Then at each iteration, each ant randomly selects a constructive direction and terminates with a final tour.

C) Solution construction. M ants build M solutions to the problem based on both pheromone and heuristic information, but only the optimal path is reserved.

D) ***Local pheromone updating***. Local pheromone is updated in this step according to a certain updating rule.

E) Global pheromone updating. Total pheromone is updated in this step according to a certain updating rule.

F) Terminal test. Only if the optimal solution is found or the algorithm runs meeting the execution time or the number of the iterations has reached the max number of iterations can the algorithm end, or return to step B) until one the above three conditions is satisfied.

IV. PROPOSED ALGORITHM

In contrast with many algorithms for mapping and scheduling problems, CACO algorithm starts from the concept that it is actually performing a heuristic scheduling. Heuristics simply optimize the priority function of a classic list formulation. In other cases, list scheduling is used after the heuristic search to evaluate the quality of the mapping chosen for the tasks. List scheduling exploits a heuristic priority function also in these cases. In this paper, four kinds of situation are declared. Each situation differs another in method of computing QoS.

A)The time-concerned type. If a task is a time-concerned task, users of the task only concern with the task processing time without taking cost into consideration. These tasks are usually time emergency.

B) The cost-concerned type. If a task is a cost-concerned task, users of the tasks only concern with the cost they need to pay for task processing without taking processing time into consideration. These tasks are not urgent. Tasks are sorted in descending order according to the average processing cost.

C) The time-tended type. Users of this kind value both cost and processing time. Therefore, QoS of these tasks includes both cost and processing time. But time is more important to them.

D) The cost-tended type. Users of this kind value both cost and processing time. Therefore, QoS of these tasks includes both cost and processing time. But cost is more important to them.

1. Initialize the values for pheromone

Solution =null;

Iteration value = 1;

2. Randomly allocate the jobs for VMs &VM value inserted into the table.

3. Repeat the following task until the completion of Ants Tour.

a. construct the solution.

b. calculate the distance between the Sources to destination.

c. According to the result pheromone updated.

d. Again calculate distance .

e. for every edge apply the locale pheromone update.

f. replace the optimal solution with the current solution.

g. increase the iteration value by one.

4. Display output as a current solution.

Fig. 1Pseudo code of CACO Procedure

CACO(Classify Ant Colony Optimization) formulation, the algorithm focuses on the concept that at every decision point an ant decides which task to schedule and where to map it. Thus, every ant constructs step by step the mapped and scheduled task graph, choosing one node after the other based on the above four situations. Each ant constructs a complete schedule in N steps, where N is the number of jobs, the ant selects a new job from a candidate list. The candidate lists include all the jobs that have satisfied dependence constraints, and for which all the required resources are available in the current timeframe. When a job is selected, the availability of the related resources is updated to the current scheduling time plus the execution time of the selected job. With appropriate choices, can always reach the optimal solution for the RCSP [23]. Following the ACO approach, each ant generates its own scheduling. The results (overall execution times) are then evaluated, and the pheromone matrix is updated following the standard update policies. In subsequent iterations, the ants will converge to the shortest schedules. We adapted this formulation to perform scheduling, mapping and partial dynamic configuration.

V. COMPARISON WITH RELATED WORK

1)Exact Integer Linear Programming (ILP) formulations for scheduling [24][25] have been proposed, but they are not applicable for large instances of the problems.In the proposed algorithm is preferred to obtain sub-optimal results in an acceptable time,cost,cost-time,time-cost.

2)Resource Constrained Scheduling problem (RCSP) is the list based algorithm [26], which uses a priority list to determine the order in which operations should be scheduled. In the proposed algorithm classifier decide the task priority than an ant takes local decision.

3) GAs [27], TS and SA [28], [29] have also been used to solve hardware /software partitioning and mapping problems , which tries to find the best solution performing local moves, has been successfully adopted in many formulations. These works, however, assume that the hardware is static; the programmable components cannot be reconfigured. In the proposed algorithm an ant chooses the VM based on preknowledge.

4) ACO has been recently applied separately to both scheduling [30], [31] and hardware /software partitioning for multiprocessor systems with static programmable logic. In the proposed algorithm, however, differs from these works for several aspects. We consider scheduling and mapping simultaneously.

The main difference is that the proposed ACO is wrapped by the scheduler, and thus each step is actually a scheduling step.

VI. EXPERIMENTAL EVALUATION

The implementation of the proposed algorithm CACO consists of two components, Classifier and Scheduling. The experiment was implemented in CloudSim which is a toolkit for simulating cloud computing scenarios. The edition is CloudSim 3.0.3 simulator on windows 7 OS with Intel Core (i5) 2 Duo CPU 2.40GHz processor. Eclipse is used to run CloudSim 3.0.3. In our simulation scenario, there areeight datacenters whose system architecture are X86 and the OS is Linux. There are 3-4 hosts in this datacenter. The bandwidth is 10000. The detailed data are shown in the Table 1.

Table 1:Testing Environment

Datacenter Parameter List	
Number of Datacenter	8
Number of Hosts	3-4
Virtual Machine Parameter	
Number of VMs	50
VM Memory (RAM)	256 - 2048
Total Number of Tasks	100 - 1000

The Proposed CACO Algorithm tested under different Conditions by varying Parameters. As shown in Figure 2, The Proposed Algorithm minimizes the Average Make Span when compared with Existing Algorithms ACO.In the Proposed Algorithm, with increasing the Number of Jobs Average Make span of the CACO Algorithm Decreases.

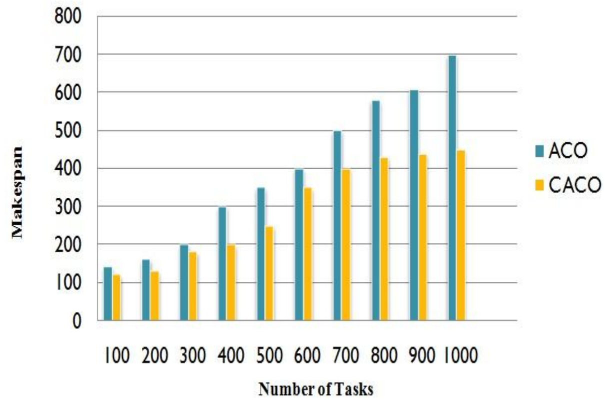

Fig.2 Comparison of Average of Make Span for the Existing and Proposed Algorithms.

The Figure 3 shows that the CACO can effectively give an optimized solution, which increases speed and efficiency of managing the tasks queue by minimizing the waiting time of tasks, and reduce the queue length.

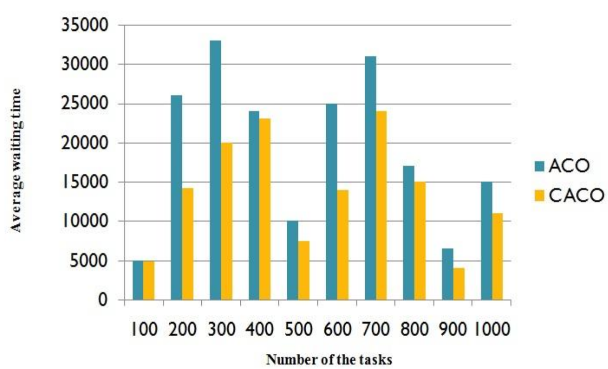

Fig. 3 Average waiting time using ACO and CACO.

Figure 4 shows the impact of the workflow size on the allocation cost in the application scheduling. As Figure 4 indicates that the allocation cost of the proposed algorithm in all instances is around 18% less than the traditional ACO algorithm. The low cost in the proposed algorithm is explained by the fact that it selects a slot with the given four conditions (time, cost, cost-time ,time-cost) for running the eligible task from the whole existing. However, the ACO algorithm randomly selects a subset of the slots for scheduling the whole tasks. In the ACO algorithm, as long as the executions of the parent tasks are not completed, child-tasks will not be submitted.

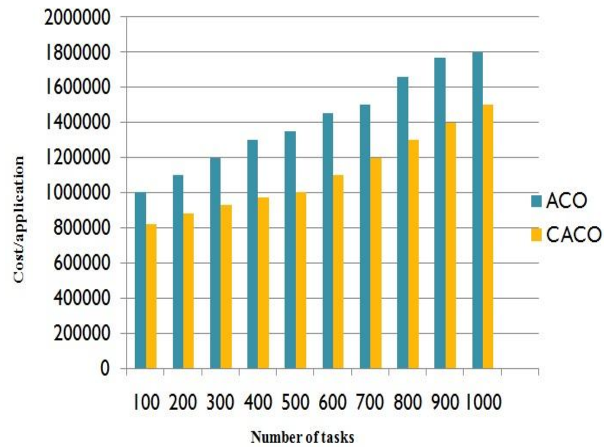

Fig. 4 Workflow size impact on workflow allocation cost.

The Experimental results shown that the Proposed Algorithm Average Make Span, Average waiting timeand allocation cost is less than Existing Algorithms.

VII. CONCLUSION AND FUTURE STUDIES

In this paper, for the cloud task scheduling problem, we introduce Classify Ant Colony Optimization algorithm(CACO) by improving basic model of the Ant Colony Optimization (ACO) algorithm. The result of the experiment indicates that thealgorithm can effectively solve the problem of task scheduling in the cloud.In the Future this research can be the basis for makingother scheduling policies to reduce,processing time and reduce the load on the data centers.Further, it can be extended to other policies and other metricscan be used to have a more optimized policy.

REFERENCES

[1] P Well, T Grance. The NIST definition of cloud computing . NIST special pubulication, 2011.

[2] Arya Lokesh Kumar, Verma Amandeep. Workfilow scheduling algorithms in cloud environment-A survey.//Proceedings of Engineering and Computational Sciences (RAECS). Chandigarh, India, 2014:1-4.

[3] S.Sindhu and S. Mukherjee, "Efficient task scheduling algorithms for cloud computing environment," in High Performance Architecture and Grid Computing. Springer, 2011, pp. 79–83.

[4] M.Dorigo, C. Blum / Theoretical Computer Science 344 (2005) 243 – 278.

[5] Dorigo, M., Di Caro, G, "The Ant Colony Optimization metaheuristic", MIT press, (2004)1, pp.25-64.

[6] Dorigo, M., Gambardella, L.M, "Ant Colony System: A cooperative Learning Approach to the Traveling Salesman Problem". IEEE Transaction on Evolutionary Computation,(l999) 1, pp. 53-66.
[7] Dorigo, M., Maniezzo, V., Colorni, A. "The ant system: “Optimization by a colony of cooperating agents"", IEEE Transactions on Systems, Man, and Cybernetics, part B, vol. 26, no. 1, 1996, pp.29-41.
[8] Mala Kalra, Sarbjeet Singh, "A review of metaheuristic scheduling techniques in cloud computing". Egyptian Informatics Journal (2015) 16, pp.275-295.
[9] D.B. Tracy, J. S.Howard, and B. Noah. Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing, vol. 61, no. 6 , 2001, pp. 810 - 837.
[10] J.Yu, R. Buyya. Workflow Scheduling Algorithms for Grid Computing. Metaheuristics for Scheduling in Distributed Computing Environments, F. Xhafa and A. Abraham (eds), ISBN:978-3-540- 69260-7, Springer, Berlin, Germany, 2008.
[11] Suraj Pandey1, LinlinWu1, Siddeswara Guru2, Rajkumar Buyya1. A Particle Swarm Optimization (PSO)-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments. Technical Report,CLOUDS-TR-2009-11,Cloud Computing and Distributed Systems laboratory, The University of Melbourne Australia, October,2009.
[12] R.Sakellariou, H. Zhao, E. Tsiakkouri, and M. D. Dikaiakos. Scheduling Workflows with Budget Constraints. CoreGRIDWorkshop on Integrated research in Grid Computing. Technical Report TR-05-22, University of Pisa, Dipartimento Di Informatica,Pisa, Italy, November 2005,pp: 347-357.
[13] J.Yu, R. Buyya, and C. K. Tham. A Cost-based Scheduling of Scientific Workflow Applications on Utility Grids. Proc. of the 1st IEEE International Conference on e-Science and Grid Computing, Melbourne, Australia, December 2005, pp: 140-147
[14] KhaledTalukde, MichaelKirley, Rajkumar Buyya. Multiobjective Differential Evolution for Scheduling Workflow Applications on Global Grids. Concurrency and Computation: Practice and Experience. Wiley Press, New York, USA.21 (13), pp: 1742-1756,2009.
[15] Wei-Neng Chen, Jun Zhang. An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem with Various QoS Requirements. 2009 IEEE Transactions on systems, Man, And Cyberetics.
[16] J.Yu and R. Buyya. Scheduling Scientific Workflow Applications with Deadline and Budget Constraints using Genetic Algorithms. Scientific Programming Journal, IOS Press, 2006, 14(3-4), pp: 217-230.
[17] Kim S, Weissman JB. A genetic algorithm based approach for scheduling decomposable data Grid applications. ICPP.IEEE Computer Society: Silver Spring, MD, 2004, pp: 406–413.
[18] Ye G, Rao R, Li M. A multiobjective resource scheduling approach based on genetic algorithms in Grid environment. Fifth International Conference on Grid and Cooperative Workshops, Hunan, China,2006; 504–509.
[19] Meng Xu, Lizhen Cui, Haiyang Wang, Yanbing Bi. A Multiple QoS Constrained Scheduling Strategy of Multiple Workflows for Cloud Computing. 2009 IEEE International Symposium on Parallel and Distributed Processing with Applications.2009, pp: 629-634.
[20] M.Gupta and G. Sharma, “An efficient modified artificial bee colonyalgorithm for job scheduling problem,” in International Journal ofSoft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-1, Issue6. Citeseer, 2012.
[21] R.F. de Mello, L. J. Senger, and L. T. Yang, “A routing load balancingpolicy for grid computing environments,” in Advanced InformationNetworking and Applications, 2006. AINA 2006. 20th InternationalConference on, vol. 1. IEEE, 2006, pp. 6–pp.
[22] K.-L. Du and M. Swamy, “Particle swarm optimization,” in Search and Optimization by Metaheuristics. Springer, 2016, pp. 153–173.
[23] S.Hartmann and R. Kolisch. Experimental evaluation of state-of-theart heuristics for the resource-constrained project scheduling problem. European Journal of Operational Research, 127(2):394–407, 2000.
[24] Kent Wilken, Jack Liu, and Mark Heffernan. Optimal instruction scheduling using integer programming. In PLDI ’00: ACM SIGPLAN Conference on Programming Language, Design and Implementation, pages 121–133, 2000.
[25] R.Niemann and P. Marwedel. An algorithm for hardware/software partitioning using mixed integer linear programming. Design Automation for Embedded Systems, 2(2):125–163, March 1997.
[26] Thomas L. Adam, K. M. Chandy, and J. R. Dickson. A comparison of list schedules for parallel processing systems. Commun. ACM, 17(12):685–690, 1974.
[27] J.I.Hidalgo and J. Lanchares. Functional partitioning for hardwaresoftware codesign using genetic algorithms. In 23rd Euromicro Confer-ence, pages 631–638, 1997.
[28] T.Wiangtong, P.Y.K. Cheung, and W. Luk. Comparing three heuristic search methods for functional partitioning in hardware–software codesign. Design Automation for Embedded Systems, 6(4):425–449, 2002.
[29] P.Eles, Z. Peng, K. Kuchcinski, and A. Doboli. System level hardware/ software partitioning based on simulated annealing and tabu search.Design Automation for Embedded Systems, 2:5–32, 1997.
[30] G.Wang, W. Gong, B. DeRenzi, and R. Kastner. Ant colony optimizations for resource- and timing-constrained operation scheduling. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 26(6):1010–1029, June 2007.
[31] D.Merkle, M. Middendorf, and H. Schmeck. Ant colony optimization for resource-constrained project scheduling. IEEE Transactions on Evolutionary Computation, 6(4):333–346, August 2002.