International Academic Institute for Science and Technology

International Academic Journal of Science and Engineering
Vol. 2, No. 1, 2015, pp. 186-200.
ISSN 2454-3896

International Academic Journal of Science and Engineering

www.iaiest.com

Resource Management in the Cloud Computing Using a Method Based on Ant Colony Optimization

Pouneh Janmohammadi[a] , Morteza Babazade [b]

[a] *Department of Computer, Babol Branch, Islamic Azad university, Babol, Iran.*

[b] *Department of Computer, Babol Branch, Islamic Azad university, Babol, Iran.*

Abstract

Regarding large amount of tasks and resource constraints, resource management is a significant challenge in cloud computing. There are several reasons, including heterogeneous and dynamic properties of resources and requests in cloud computing environments, which have caused it to be emerged as an NP-complete problem. Also, it is very difficult to manually allocate tasks to cloud computing resources. As a result, an appropriate method is needed to allocate cloud resources to requested tasks in order to increase exploiting cloud resources. So far, many methods and meta-heuristic algorithms such as Round Robin, Max-Min, GA and PSO is used as a solution for this problem. It is only reducing time or costs that is provided in most of presented methods. But in this article, scheduling algorithm aims to process and run users' tasks at the least possible time and cost. Also, the maximum efficiency of resources is taken into account so that all tasks would be distributed evenly among the available resources in order to reduce the completion time of all tasks and to increase resource efficiency. Ant colony optimization algorithm is used to improve the management of resources in the cloud environment. Also, the gravitational algorithm with local search is used to prevent algorithm from fast converging to local optimal and to increase its dynamic ability. Simulation and comparison of the results of proposed strategy with the results of genetic algorithm and ant colony algorithm indicates that the proposed approach could satisfy users and also use resources well.

Keywords: Cloud Computing, Resource Management, Task Scheduling, Ant Colony Optimization (ACO), Gravitational Local Search Algorithm

Introduction:

In recent years, cloud computing has become a popular computing paradigm for web hosting and services on the Internet and this allows users from anywhere in the world to use easily the computing resources, storage and software. There are three main types of cloud computing: Infrastructure as a Service (IaaS), platform as a service (PaaS) and Software as a Service (SaaS). The acceptance and establishment of cloud computing platform have many attractive advantages such as reliability and quality of services (Randles et al, 2011). For consumers, they can buy computing power as they need. From the perspective of the service provider, the main issue is to maximize profit and to minimize operational costs (Yongqiang Gao et al, 2013). Virtualization is one of the key technologies that enable cloud computing (Kusic et al, 2009). Practical services managed by cloud computing system use distributed computing in the virtual machines to achieve efficiency. By virtualization, the cloud provider can deliver the quality of service (QoS) to users while it can achieve the high efficiency of server and energy (Yongqiang Gao et al, 2013).

The resources in cloud computing are divided into two categories, physical and virtual resources. Since resources are limited in cloud computing, resources should be shared and allocated effectively and appropriately between tasks and users' requests. The computing resources and their allocation to user are based on an agreement by which the service provider and the service customer (the user) do with each other (Yongqiang Gao et al, 2013). So, resource management is an important issue that is used for the state of arranging a number of tasks. Discovering, preparing, estimating, allocating, and resource scheduling can be noted as the problems of resource management. Resources scheduling is a schedule of events and resources. Because the shared resources are available at certain times. Events mean to determine the start and end times of an activity depending on required time for it, previous activities, and resource allocation (Sunil Kumar et al, 2014). There are several strategies for making an array of tasks resulting in various cost and efficiency level. So, making decision to find an optimizing strategy for the allocation of resources, especially for limited resources, with any purpose, is considered in the management of resources. It is difficult to manually assign tasks to computing resources in clouds. The main objective is to schedule tasks to compatible resources in accordance with the consistent time that includes finding a proper sequence in which tasks can be performed under reasonable constrained transaction (Yongqiang Gao et al, 2013). Therefore, an efficient algorithm for scheduling job is needed in a cloud environment. A good Scheduler should adopt its scheduling strategy with the changing environment and task type (Chang et al, 2009).

Scheduling tasks considers virtual machines as scheduling units to allocate heterogeneous physical resources for the implementation of tasks. Each virtual machine is an abstract unit of computing capacities and storing provided in the cloud (Ghorbannia and Arian, 1390). Right scheduling can effect greatly on system performance. Effective task scheduling algorithm can meet user's requirements, and improve resource efficiency, thereby it increases the overall performance of cloud computing environment (Ghaffari, 1389). Business and income issues are important in clouds, and costs are calculated by the user agreement. So, both sides want to reduce cost and time. Nowadays, many scheduling strategies are available for cloud computing, many of which have used evolutionary algorithms such as genetic algorithms, particle swarm algorithm, colonial competition algorithm, and ant colony algorithm.

Nadia Hazeri et al (1392) present a scheduling algorithm limited to the quality of multi-service for several workflow MQMW associated with reducing time and cost, and it does the best possible service even when users send several different works to the cloud. Using a scheduling strategy based on the genetic algorithm of simulated inflammation, they assessed all the requirements of servicing to users to search their necessary resources on their virtual machines and provided the best optimal timing of jobs with the

best quality of services (Hazeri, et al., 1392). A study conducted by E. Arianyan, et al (2012) has proposed an improved version of the resource allocation algorithm based on genetic algorithm and it suggests that more parameters should be considered in the resource allocation algorithm. In addition, the importance of attention to weight is provided for decision algorithm parameters. So that this study shows that some factors are more important than others and they should effect greatly on the fitness function of genetic algorithm (Ehsan Arianyan et al, 2012). Hai zhong et al, (2010) used the improved genetic algorithm for automatically scheduling policy. Their algorithm has used the shortest genes, and it has introduced the idea of dividend policy in economics in order to choose optimal allocation or less than optimal for request. They considered a simple sum of CPU, RAM and hard drive in their fitness function. Intended index is resource efficiency, and some factors are not taken into account, such as load balancing and fault tolerance (Hai Zhong et al, 2010). Chen Hong Zhao et al. (2009) have offered an optimized algorithm based on genetic algorithms for scheduling of divisible and independent tasks according to various calculations and needed memory. Suppose a cloud computing system includes m heterogeneous unit (1m>). Arrival time is not known in advance, and each job has the arrival time, the worst-case computation time, and the end opportunity. The standby time is considered equal to its entrance time. Each task has a two-way access to a single process, exclusive access and shared access. Scheduling scheme is centralized, such as a single core processor in the cloud, collecting all the tasks, amount of waiting to distribute and enter another unit. Each process unit has its own distribution line. The main unit communicates with other processing units through this distribution line. The main unit, working in parallel with other units, schedules new arrived jobs and updates periodically the distribution line. Tasks are sorted in ascending order by the value of their final course. Despite achieving good results, reducing the solution space in GA is proposed for future work (Chenhong Zhao et al, 2009).

The basic idea of ACO is inspired by the studies and observations on the foraging behavior of ant colonies (Lin et al, 2012). When an ants group tries to search for the food, they use a special kind of chemical to communicate with each other. That chemical is referred to as pheromone, and when they choose a way to go, most likely, they choose a path marked with stronger and thicker pheromones (Blum et al, 2003). This algorithm is used to find a near-optimal solution for traveling salesman problem (Dorigo and Gambardella, 1997) and quadratic assignment problem (Maniezzo and Color, 1999). This numerical method compared to analytical and genetic methods where the chart constantly changes by time has an advantage: it is an algorithm can be replicated. Therefore, it can freshly change the answer by passing time. So, ACO can be used to optimize any issue requiring to find the shortest route (Dorigo et al, 2006). Medhat Tawfik et al. (2013) used ant colony optimization to allocate incoming batch jobs to virtual machines in VM level. All the VMs in a data center not necessary have a fixed amount of processing power. So, the tasks/ requests (application services) are allocated to these VMs of different processing powers, from the most powerful VM to the lowest. Reducing the time of all jobs is the main purpose. During an iteration of the ACO algorithm each ant k, k=1, ..., m builds a tour executing n (n is 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. In this study, it is assumed that works are independent and there is no limitation or precedence among tasks. They cannot be interrupted or moved to another processor during their execution. The performance parameter such as overall make span time is optimized that means resource utilization ratio has increased and also the cost has decreased (Tawfeek et al, 2013). Given that tasks are considered independently, the precedence of tasks and load balancing are not taken into account. In an article proposed by Xiao Tang Wen et al (2012), a combination of ant algorithm and particle swarm algorithm is used for scheduling of resources in cloud computing. Ant colony algorithm uses pheromone to transfer information, and particle swarm optimization algorithm using the information, local extremum information and global extremum information, chooses next iteration position to direct particles. Ant Colony Optimization is prone to premature convergence and to fall into local optimal solution. Initially,

ants complete the route according to the ant colony optimization, and then they regulate solutions regarding that solutions are local optimization or global optimization, mutations strategies and integrated strategy have been used in the way to overcome the defects of genetic algorithm. Simulation results show that not only does the improved algorithm prevent convergence pace from accelerating, but also it prevents from falling into the trap of local optimal solution. Also, user tasks are presented effectively to appropriate resources in cloud computing leading to improve the exploitation of resources (Xiaotang Wen et al, 2012).

1. Scheduling resources using ant colony optimization algorithm

In this study, a new method based on ant colony optimization algorithm is used to evaluate and optimize the management of resources in a cloud computing. The proposed algorithm aims to enhance constancy and to achieve the optimal solution at less time. Also, as the newest algorithms in this field, gravitational local emulation search algorithm has been used to stop the rapid convergence of ant colony algorithm and falling into the trap of local optimization and to increase scanning capabilities compared to previous algorithms. To achieve the goal, reducing idle time of machine is considered so as it increases machines' efficiency.
Figure 1 shows the general steps of the proposed algorithm. As it can be seen in the flowchart of the proposed method that the proposed method is different from the traditional ant colony algorithm in some ways.

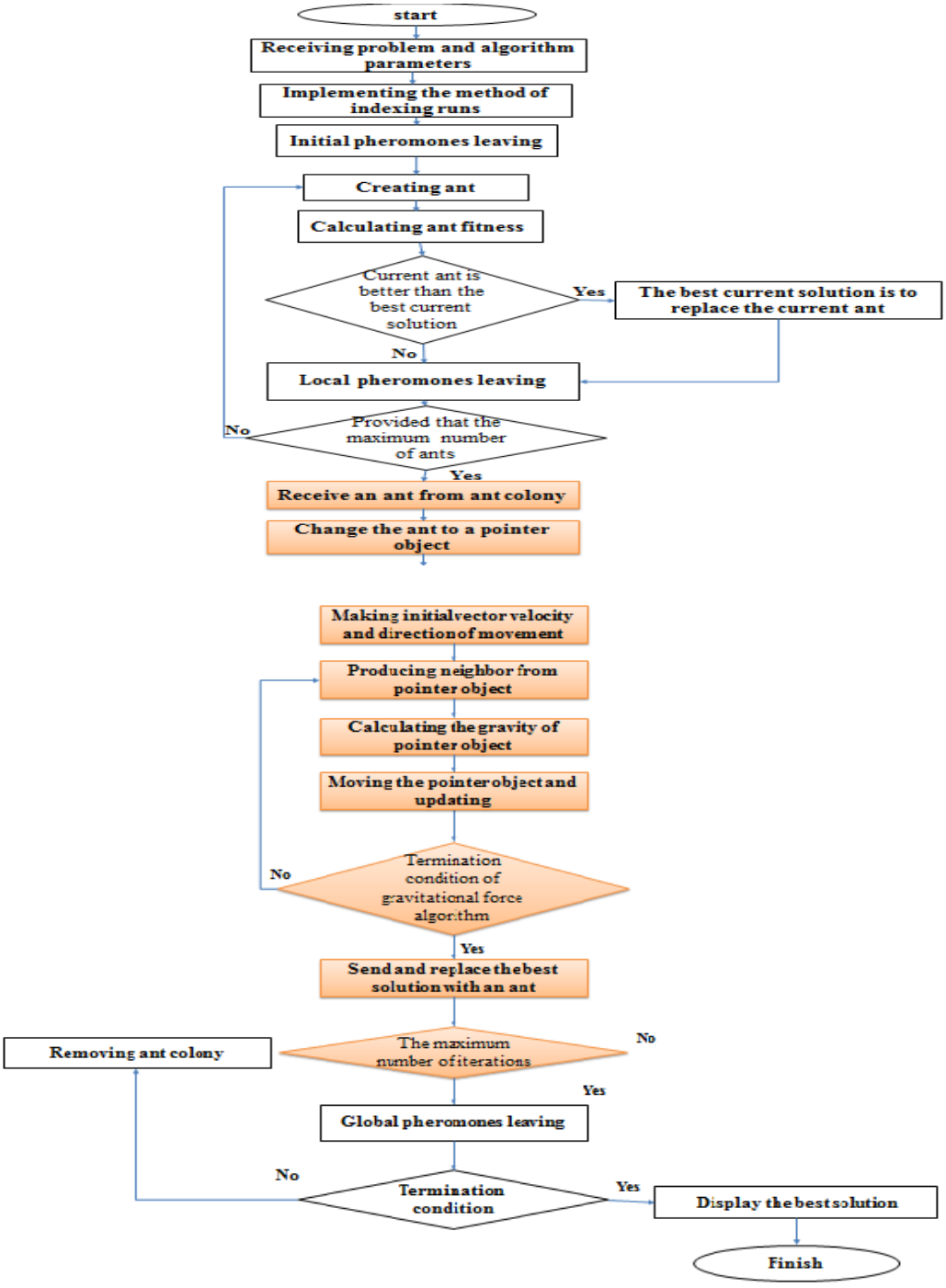

Figure 1: the general steps of the proposed algorithm

There are a number of data centers in a cloud environment, each of which can be located in any part of the geographical environment. Each data center is made up of a number of virtual machines. Virtual machines communicate with their preferred data center, using their communication channels. Each virtual machine is made up of a number of processors for processing information with various processing powers and performance types. Each processor has a different cost to process information. Using the communication channels, processors communicate with the preferred virtual machine. Finally, all these data centers are connected with each other by communication channels in cloud environment. In this environment, users send their requests with different Submitted rates to the resource management system of cloud environment. Then, the cloud system receives users' requests, and by using the resource management algorithm, it allocates resources to process users' tasks. Each user determines the final time and proposed cost for processing all tasks. The user can consider the preference for these two parameters. The preference for the parameters of "proposed time" and "proposed cost", that their values are in the range of [0,1], is calculated in this way that the sum of these two parameters must be equal to 1. In the proposed model, usable bandwidth is also considered in the data center to receive and send data in a cloud environment.

1-1- Indexing implementation and tabling pheromones

At this stage, receiving the size and characteristics of the problem, the proposed algorithm starts to index each implementation from one task, the main reason of which is the reduction of algorithm complexity and the precise performing of next step. For indexing, a unique number is considered for each user's task, and a 4-column matrix is formed. The number of rows is variable and depends on the table of issue. First column shows the performance index number, second column indicates the task number, third column indicates the number of a processor in which the task should be processed and the final column shows the status of implementation. Implementation is in two positions, the processing position with value of 1 and non- processing position with value of zero, having the initial value of zero at the beginning of tabling. In fact, we determine that the implementation of each user is allowed to be processed in which processor. Then, with the help of the indexing table, the pheromones table is build. The pheromones table for the pheromone ant colony algorithm is as an n × n matrix, where n is the number of rows in the indexing table.
After creating the pheromones table at this stage, initial pheromone value is determined for each entry of table cell. The initial value of pheromone is given to the algorithm by the input parameter.

1-2- Creating Ant

Each ant represents a solution in proposed algorithm and determines the processing sequence of execution of users' tasks. We have used a one-dimensional array to show an ant that the size of each array is equal to the number of user s' tasks in the problem, and also, contents of each cell indicate the run number that should be processed in the intended processor for predetermined time span. To create ant, initially, the ant is given one of our valid run. Now, according to the index table, the ant calculates the probability of choosing each unprocessed users' task by equation 1.

$$P_{ij}^{K} = \frac{[T_{ij}]^{\alpha} \times [\frac{1}{n_{ij}}]^{\beta}}{\sum_{l \epsilon N_i^k} [T_{il}]^{\alpha} \times [\frac{1}{n_{il}}]^{\beta}} \quad if\left(j \epsilon N_i^k\right) \qquad \text{(Equation 1)}$$

In equation 1, N ij is sum of the idletime of processor and the time of completion of all tasks. Tij shows the amount of pheromone in pheromone table for entries i and j. β and α shows degree of pheromone sequence dependence and heuristic information related to the proposed algorithm. N_i^k indicates the allowed runs for ant K in the next step. A memory is allocated to each ant that it keeps the observed neighbors (processed run) in memory for calculating next run. After calculating probability, the ant selects one of these runs based on the roulette wheel algorithm, and so on the ant goes for processing the next run, so that all users' runs would be processed. Figure 2 shows an example of ant movement.

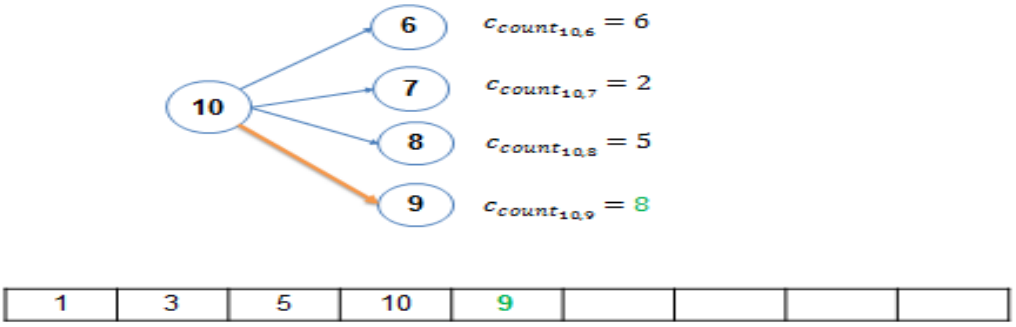

Figure 2: Creating an ant in the proposed algorithm based on pheromone table and machine idle time.

As shown in Figure 2, ant is in the node 10 referring to the index table. Now, ant is to select next node or, in other words, next run according to the index table based on probability that nodes or runs 6,7,8,and 9 are allowed, and because 9 is rated more, it is more lucky to be selected.

1-3- Competence

Competence of each ant in the proposed algorithm is considered as reducing the time of all existing tasks for processing in the cloud computing system and improving users' tasks so that their needs are completely removed. We have also taken in to account the processors idle time to increase the resource efficiency. To calculate the competence of an ant, initially, the total used costs are calculated by using equation 2 to process j user's tasks.

$$P_j = \frac{CT_j}{OP_j} * PC_j \qquad \text{(Equation 2)}$$

In equation 2, OP j indicates the proposed cost j for processing all intended tasks, and CT j represents the calculated cost for processing j user's tasks and PCj represents j user's considered preference for the cost. Finally, Pj is considered as the cost for user j. Then, using equation (3) total required time for processing all j user's tasks is calculated.

$$T_j = \frac{TT_j}{OT_j} * PT_j \qquad \text{(Equation 3)}$$

Equation 3, OTj indicates the proposed time of user j to process all intended tasks and TTj represents the calculated time in resource management system in the cloud environment for processing j user's tasks and PTj represents the considered preference of user j for time. Finally, Tj is the considered time for user j. According to Equation 2 and 3, the considered competence for the user j is calculated by using equation 4.

$$F_j = P_j + T_j \qquad \text{(Equation 4)}$$

In equation 4, P j and Tj are the considered cost and time for user j, respectively, and Fj represents the competence of user j. Also, using equation 5, we considered idle time of each processor. Since the aim is to increasing the efficiency of resources, the lower VM idle time is, the better it is. Also, the Equation 6 is used to reduce the time of completion of total tasks, and to process users' tasks faster.

Also, the computation Idle (i) refers to the calculation of processor idle time, Make span refers to the completion time for processing all tasks of users.

$$idle = \sum_i^m \text{computation}Idle\ (i) \qquad \text{(Equation 5)}$$

$$Makespan = Max_{1 \le i \le n}\{T_i\} \qquad \text{(Equation 6)}$$

Finally, the competence of each ant in the proposed algorithm of total users' competence and completion time for the processing of all tasks and total idle time of processors are considered, so that the lower this value is, the more competent the ant is. Equation 7 presents competence for the ant in proposed algorithm.

$$Fitness = \delta \times Makespan + \sigma \times \sum_{j=0}^{j} F_j + \rho * Idle \qquad \text{(Equation 7)}$$

Because not all these criteria can be improved, preference is considered for each of these criteria, showing the degree of its importance, P, σ and δ values show the coefficient of the importance of each criterion.

1-4- Local pheromones leaving

At this stage, according to equation 8, local updating of pheromone is done only for existing nodes that responding to the current ant.

$$\mathrm{T(i,j)} = (1 - \text{LocalCoefficient}) * T(i,j) + \text{LocalCoefficien} * \mathrm{T_0} \qquad \text{(Equation 8)}$$

In this equation, Local coefficient refers to the local pheromone updating coefficient that is received as incoming parameters. $\mathrm{T_0}$ represents the initial pheromones. T (i, j) is the entry of i and j, the amount of

pheromone in pheromone table. The purpose of this function is to prevent next ants' movements in the current ant's path. Use this function leads to scan new locations in fitted perspective by the algorithm.

1-5- Execution Of Gravitational Emulation Local Search Algorithm

According to the flowchart of the proposed algorithm in Figure 1, at this stage, gravitational emulation local search algorithm is used in order to stop local optimizing and increasing the power of the traditional ant colony optimization algorithm. At this stage, if ant colony algorithm does not find better answer at a specified iteration, this step will run. In section of running gravitational local search algorithm, the algorithm in an iterative loop selects a number of ant based on the ranking method and GELS algorithm is repeated for them. After receiving an ant, GELS algorithm transfers to a pointer object. The object shows a sequence for processing users' tasks in the proposed algorithm.

In fact, for showing each object, a one-dimensional array with the length of the total number of available tasks is used for processing. Each cell is composed of one object, two fields shown as UTi and DPi. First field UTi refers to task number and the second field DPi refers to the number of processor in which it must be processed. Figure 3 shows a considered sample object in the proposed GELS algorithm.

0,0	3,6	2,0	1,2	4,3	5,4

Figure 3: Structure of an object in the space of the algorithm of GELS problem

After receiving an ant and selecting it as a pointer object, velocity vector is created in this stage. To create velocity vector in the first stage, each cell of a vector receives a value based on incoming parameter randomly. In fact, the velocity vector in the proposed algorithm specifies the movement of the pointer object to one of the objects. In fact, the more value of the velocity cell is, the more the pointer object moves toward that object.

Now, the neighborhood should be produced for the pointer object. Neighborhood in the proposed gravitational emulation local search algorithms is defined in this way that, initially, the total array pointer object from beginning to the end is navigated and the source processing the finest operation is selected. Then, one of the fine performed operation is selected randomly and its resource is changed. By changing the source of a fine action and implementing it in another source, it tries to produce a better solution. We create objects with size of velocity vector in the neighborhood of the current pointer object. The first approach is called "First Method". In this method, the object that is faster based on the velocity vector is chosen as a candidate solution and it is considered as inter force to the pointer object. The velocity vector value of the object is updated. Second approach, called "Second Method", the input force of total objects to the pointed object is calculated and the velocity vector value of all objects is updated. Generally speaking, in this method, the velocity vector is edited to determine which object inters the most force to the pointer object, so it moves toward it in the next step.

At this stage, it should be cleared that the pointer moves toward which side, in other words, it is pulled to which object. The first method is called Single Stepping. The object is considered as the new pointer object that has the maximum speed based on the velocity vector. In the second method called Multiple

Stepping, movement is out of problem area; in this way that by moving the contents of two array cells of current pointer object, we try to move out of problem area in order to navigate all problem area and to choose the best solution. Figure 4 shows an example of a movement for Multiple Stepping.

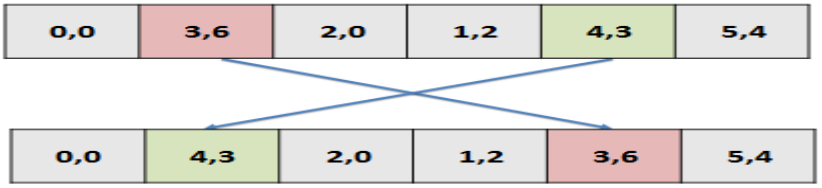

Figure 4: an example of the pointer object movement for Multiple Stepping.

Finally, after ending the algorithm GELS, the best object as Best Solution is sent to the ant algorithm, and the best object is replaced with the ant on which GELS algorithm is performed.

1-6- Global Pheromones leaving

In this stage, after building ant colony, according to equation 9, pheromones are left globally for each entry of pheromone table. This method tries to consider more value for the routes through which ants have followed the best ant.

$$T(i,j) = (1 - GlobalCoefficient) * T(i,j) + GlobalCoefficient * \Delta T(i,j) \qquad \text{(Equation 9)}$$

$$\Delta T(i,j) = \begin{cases} Bestpheromone & if\ (i,j) \in Sbest \\ 0 & otherwise \end{cases} \qquad \text{(Equation 10)}$$

In equation 9, the Global Coefficient refers to global updating coefficient that is an input parameter. $\Delta T(i,j)$ is calculated by equation 10. In equation 10, if the ant moves from run I to run j, and this run of I to j is in the best ant, then $\Delta T(i,j)$ value will be equal to $Bestpheromone$. This value is given to the algorithm by input parameter.

2- The implementation results

In this section, the implementation and results of emulation of the proposed algorithm is described and it has been compared with genetic algorithm and standard ant colony algorithm. Emulation environment of CloudSim has been used to implement the proposed algorithm. The algorithm is implemented on a computer with processor of Intel (R) Pentium (R) 4 CPU 3.00GHz and RAM with 2.00GB. To compare all features of the proposed algorithm, ten sets of test data are designed to cover small, medium and, large cloud environment systems. These sets of test are shown in table 1. To demonstrate the stability of the algorithm and more accurate assessment, any test data 20 has been executed 20 times with the same parameter by the algorithm and average competency has been calculated.

Table 1: set of designed test data

No. of designed test	Characteristics of designed test											
	Characteristics of Users							Characteristics range of each task				
	The number of users	Total tasks	Proposed cost	Transfer rate	Proposed time	[Priority of time, priority of cost]	Geographic area	The number of user's tasks	The size of each task	The number of task's instructions	Type of operating system	Type of architecture
Test 01	14	56	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 02	24	142	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 03	32	154	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 04	52	239	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 05	50	166	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 06	41	173	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 07	40	171	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 08	61	280	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 09	33	158	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]
Test 10	80	375	[1000,2000]	[425,550]	[100,300]	[0,1]	ALL GMT	[2,7]	[600,750]	[2500,3000]	[1,6]	[1,4]

As it's shown in table 1, we have used some ranges for each of the test parameters and users' tasks that one of values in the ranges would be selected randomly. The competence chart comparison resulted from implementing the proposed algorithm, genetic algorithm and, standard ant colony algorithm for data set of Test06 test is shown in Figure 5. The proposed algorithm is better than the compared algorithm in the early iterations, and the proposed method for developing the ant algorithm acts properly for scheduling issues.

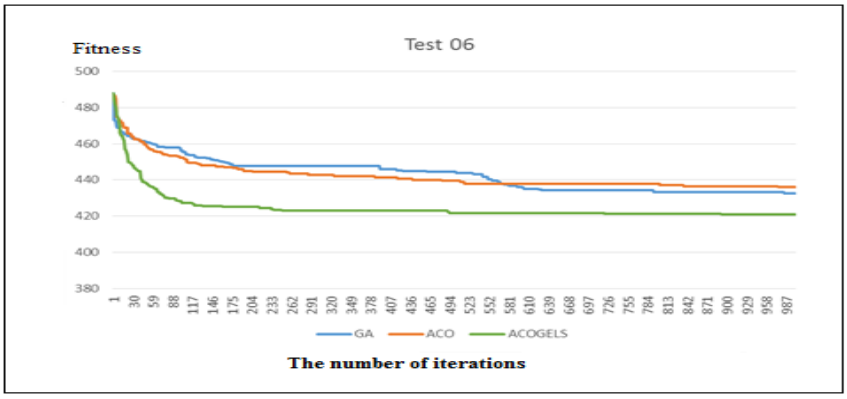

Figure 5: comparing competence chart for Test 06 test data

As it's shown in figure 6, the proposed algorithm has performed so much better than The ant algorithm and genetic algorithm for test07, and also it has improved the best solution in final iterations, and this indicates that the algorithm does not converges too early.

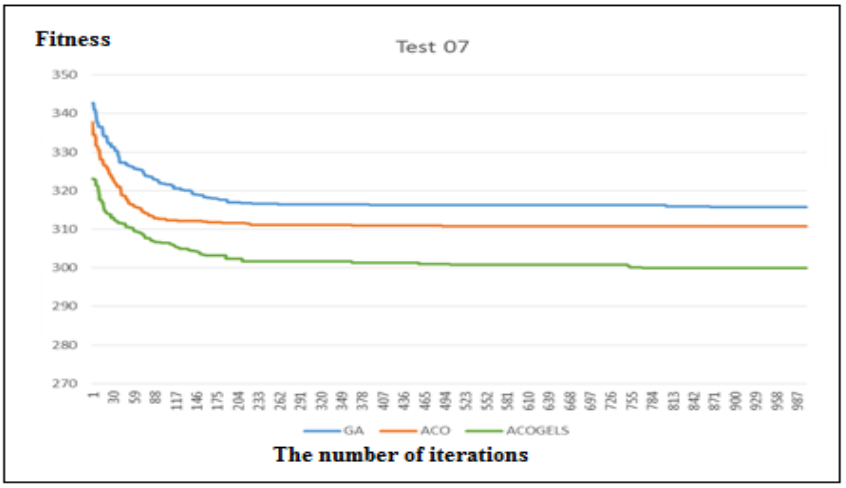

Figure 6: comparing competence chart for Test 07 test data

One way to increase the efficiency of resources on scheduling tasks is to reduce the completing time of processing requested tasks so that the processors can be accessible as soon as possible to process other tasks. This will increase profitability. Figure 7 shows the time obtained for completion of processing the whole tasks by three considered algorithms.

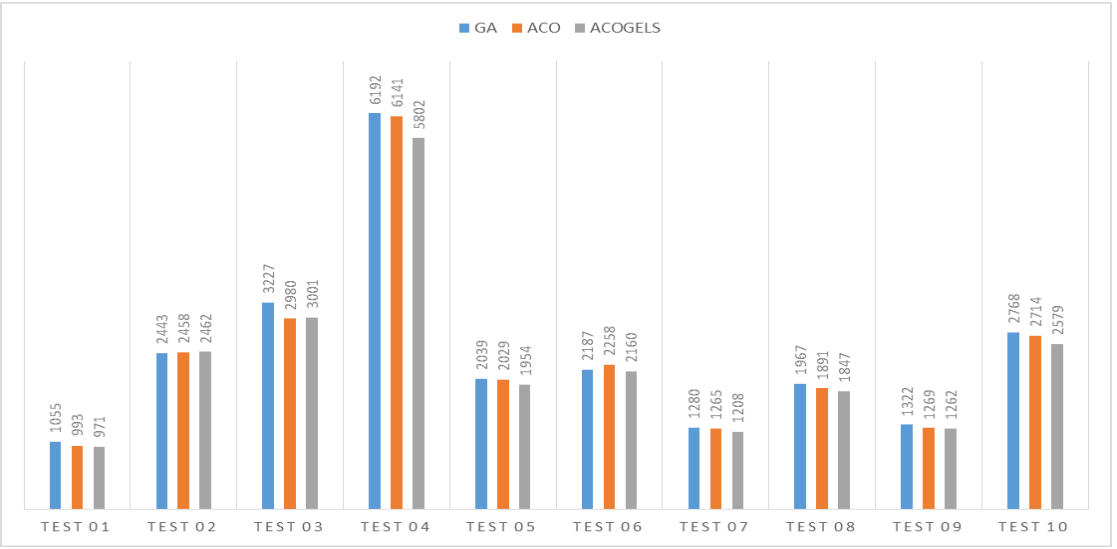

Figure 7: Comparing the completion time of processing users' tasks for ten sets of test of Table 1

Another comparison is to compare the stability of obtained results. Comparison Chart of stability determines the obtained competence of algorithm after 20 different runs with the similar parameters on the intended test data set, that the less different these competences are, the more stable the algorithm is.

As it's shown in a comparison chart of stability in Figure 8, in the proposed algorithm, solutions obtained in each iteration is slightly different that showing confidence coefficient of algorithm very well. But the ant algorithm cannot alone protect the distribution of replies, and it doesn't have good stability. In this test the genetic algorithm is better than the ant colony algorithm.

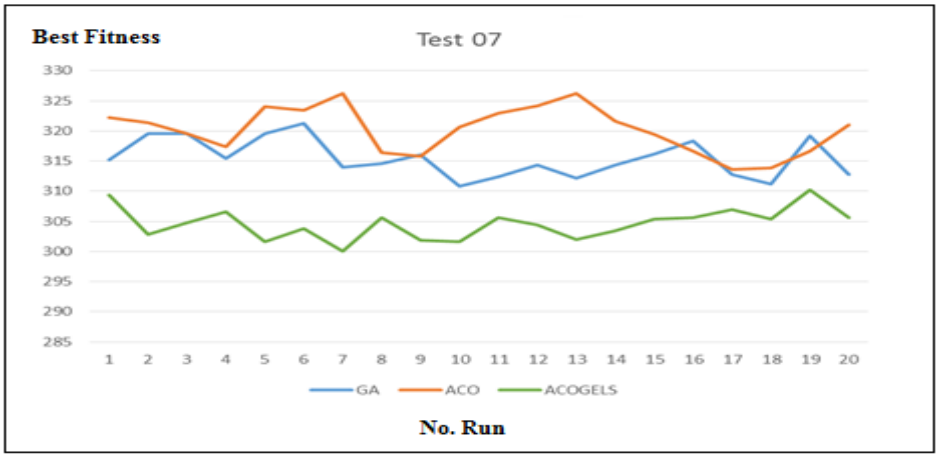

Figure 8: comparing competence chart for Test 07 test data

As it's shown in figure 9, genetic algorithm's replies obtained in each iteration is worse than the previous runs, and the ant algorithm has become better in each iteration. However, the ant algorithm is more stable than the genetic algorithm but it has not performed well compared with the proposed algorithm.

Figure 9: comparing competence chart for Test 10 test data

Summary and Conclusion

This paper investigated the issue of resource management in a cloud environment, and the main challenges for processing users' tasks. Also, the methods that researchers used to manage resources in recent years were evaluated, for example, methods such as using the genetic algorithm, ant colony optimization algorithm, and particle swarm algorithm, etc. Most proposed methods are based on meta-heuristic algorithms and always they have weaknesses such as high running time to achieve the best solution to the problem of resource management, low stability and to ignore all the features of problem area. To solve the problem, a new algorithm is proposed based on ant colony optimization algorithm and GELS algorithm for resource management that emphasizing on better and more stable solution. Experimental results indicate that the proposed algorithm is very effective and it has managed to create a balance among users' standards and also to increase the efficiency of machines. Although there are different algorithms and methods to solve resource management in the cloud, optimizing the use of resources in the cloud environment is a long way to go, researchers may consider the following cases in the future.

Parallel algorithms can be used to solve this challenge. In fact, by parallelizing ant colony algorithm and/or parallelizing gravitational force algorithm in the proposed method, it tries to increase algorithm speed in obtaining replies. Because it can search a bigger problem area with faster speed.

Using and comparing the ant colony algorithm with other local search algorithms such as simulated annealing, hill climbing, searching for harmony and so on, we can obtain the best methods to manage resources in a cloud environment.

Combining the proposed algorithm with Petry nets as a tool for modeling, emulating and evaluating the efficiency of resource management can be considered as future work.

References:

Blum, Christian, Roli, Andrea, "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison",ACM Computing Surveys, Vol. 35,No.3, September 2003,pp.268-308.

C. Lin, G. Wu, F. Xia, M. Li, L. Yao, Z. Pei, Energy efficient ant colony algorithms for data aggregation in wireless sensor networks, J. Comput. System Sci. 78 (6) (2012) 1686–1702.

Chenhong Zhao, Shanshan Zhang, Qingfeng Liu(2009), "Independent Tasks Scheduling Based on Genetic Algorithm in Cloud Computing" ,978-1-4244-3693-4/09, 2009 IEEE.

D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, G. Jiang(2009), "Power and performance management of virtualized computing environments via lookahead control", Cluster Computing 12 (1), pp.1–15.

Ehsan Arianyan, Davood maleki, Alireza Yari" Efficient Resource Allocation in Cloud Data Centers Through Genetic Algorithm" 978-1-4673-2073-3/12/$31.00 ©2012 IEEE.

F. Chang, J. Ren, and R. Viswanathan(2009), "Optimal resource allocation for batch testing" in ICST, 2009 IEEE International Conference on Software Testing Verification and Validation, pp.91-100.

Ghorban Nia, Arash and Arian, Yalda (2011), "presenting a scheduling combinational method based on genetic algorithm in cloud computing environment." The first specialized Conference of intelligent computer systems and their applications.

Hai Zhong, Kun Tao, Xuejie Zhang(2010), “An Approach to Optimized Resource Scheduling Algorithm for Open-source Cloud Systems”,The fifth annual ChinaGrid conference, pp. 124-128.

Hazeri, Nadia: Ahmadi, Mahmoud and Hanani, Ali (2013), "New strategies of scheduling based on the quality of service to users in cloud computing environment", the first national conference on new approaches in computer engineering and information retrieval.

M. Dorigo, L. Gambardella, Ant colony system: A cooperative learning approach to the traveling salesman problem, IEEE Trans. Evol. Comput. 1 (1) (1997) 53–66.

M. Dorigo, M. Birattari, T. Stutzel, "Ant colony optimization", in IEEE Computational Intelligence Magazine, pp.28-39, 2006.

M. Randles, D. Lamb, E. Odat, A. Taleb-Bendiab(2011), ”Distributed redundancy and robustness in complex systems”, J. Comput. System Sci. 77 (2) (2011), pp.293–304.

Medhat A.Tawfeek, Ashraf EL-Sisi, Arabi E.Keshk, Fawzy A.Torkey(2013), “Cloud task scheduling based on Ant colony optimization”, 8th International Conference on Computer Engineering & Systems (ICCES), pp.64-69.

Sunil kumar S.Manvi and Gopal Krishna Shyam (2014), Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey, Journal of Network and Computer Applications, 41, pp 424-440.

V. Maniezzo, A. Colorni, The ant system applied to the quadratic assignment problem, IEEE Trans. Knowl. Data Engrg. 11 (5) (1999) 769–778.

Xiaotang Wen, Minghe Huang, Jianhua Shi(2012), “Study on resource Scheduling based on ACO algorithm and PSO algorithm in cloud computing”, 11th International Symposium on Distributed Computing and Applications to Business, Engineering & Science.

Yongqiang Gao, Haibing Guan, Zhengwei Qi,Yang Hou, Liang Liu, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing", Journal of Computer and System Sciences79,pp.1230–1242, 2013.