Task Scheduling using Modified PSO Algorithm in Cloud Computing Environment

Solmaz Abdi, Seyyed Ahmad Motamedi, and Saeed Sharifian

***Abstract*—** Task scheduling problem is one of the most important steps in using cloud computing environment capabilities. Different experiments show that although having an optimum solution is almost impossible but having a sub-optimal solution using heuristic algorithms seems possible. In this paper three heuristic approaches for task scheduling on cloud environment have been compared with each other. These approaches are PSO algorithm, genetic algorithm and modified PSO algorithm for efficient task scheduling. In all these three algorithms the goal is to generate an optimal schedule in order to minimize completion time of task execution.

***Keywords*—** Cloud environment, Genetic algorithm, PSO, Task scheduling.

I. Introduction

CLOUD computing is developed from the grid computing, distributed computing and parallel computing. It is a new business computing model and will distribute the tasks on the resources pool which is made up with the large number of computers, storage devices [1]. Parallel and distributed system consisting of a collection of interconnected and virtual computers that are dynamically provisioned, and presented as one or more unified computing resources based on service-level agreements established through negotiation between the service provider and consumers [2]. Mainly three types of services are provided by the cloud. First is Infrastructure as a Service (IaaS), which provides cloud users the infrastructure for various purposes like the storage system and computation resources. Second is Platform as a Service (PaaS), which provides the platform to the clients so that they can make their applications on this platform. Third is Software as a Service (SaaS), which provides the software to the users; so users don't need to install the software on their own machines and they can use the software directly from the cloud [3]. Task scheduling is one of the most important and critical problems in cloud computing and many researches have tried to find an optimal solution for scheduling tasks on existing resources in cloud environment. But the problem of task scheduling is a NP-complete problem in general [4]. Nowadays heuristic optimization algorithms are widely used in solving NP-complete problems. An evolutionary algorithm is used to find a sub-optimal solution of the problem in considerable computation time. To reach the solution faster many heuristic based approach are used [5]. Abraham et al. [6] have investigated three different nature based heuristic algorithms which are called genetic algorithm, simulated annealing and Tabu search. Genetic algorithm and simulated annealing are inspired from nature. Genetic algorithm simulates natural selection process. Existing solutions in each generation are evaluated by fitness function and individuals which have better fitness value generate new solutions by crossover and mutation operators. Using genetic algorithm and some modified and improved versions of genetic algorithm for task scheduling are addressed in [7], [8].

One of the newest heuristic algorithms is PSO (particle swarm optimization). This algorithm that has developed by Kennedy and Eberhart [9] is one of the evolutionary algorithms which simulates social behavior of flock of birds or groups of fishes toward their desired destination. In [10] a PSO algorithm is introduced by Shih Tang Lo, et al in which the problem of multi-processor resource scheduling is considered.

In addition to benefits of heuristic algorithms, such as flexibility and acceptable calculations, PSO has single in kind specifications such as easy implementation and consistent performance. Nowadays PSO algorithm is used in different domains such as neural network training, control system analysis and design, structural optimization and so on.

In this paper, PSO algorithm, genetic algorithm and modified PSO algorithm for task scheduling problem in Cloud computing environment are compared with each other. Simulation results show that even if three algorithms show acceptable revenue, but totally Modified PSO Algorithm performance is better than two other algorithms. This paper is organized as follow: in section II the way that scheduling problem is modeled is described, this section represents the definition of Genetic algorithm, PSO algorithm. And also modified PSO algorithm. In section III experimental results are represented, and in section VI conclusions are presented.

II. Scheduling Problem Modeling

The first assumption is that each task that is submitted by user is independent from other tasks. In this paper we assume that we have n tasks which their execution time on each processing machine is known in advance and they should be processed on m computational resources and the goal is to minimize completing execution time and optimizing resource utilization.

If number of tasks is less than resources, tasks are

Solmaz Abdi is with the Amirkabir University of Technology CO 15914 Tehran, Iran. (E-mail: solmazabdi@aut.ac.ir).

Seyyed Ahmad Motamedi and Saeed Sharifian are with the Amirkabir University of Technology CO 15914 Tehran, Iran. (E-mails: motamedi@aut.ac.ir, sharifian_s@aut.ac.ir).

distributed based on first-come-first-serve algorithm. If number of tasks is more than resources task distribution is performed based on scheduling algorithms. In this paper we suppose that number of tasks is more than number of resources, so one task cannot be assigned to different resources, i.e. tasks are not allowed to migrate between resources. To formulate the problem, set of tasks is defined as $T_i i = \{1,2,,3,...,n\}$ which are n independent tasks and $R_j j = \{1,2,3,...,m\}$ is set of computational resources. We suppose that execution time of task i on computational node j is known, and it is equal to jobRunTime (ij). The goal is to find a matrix in which if task i is executed by resource j then x_{ij} element in that matrix is 1 otherwise it is 0 and it minimizes total cost of completion time of task execution on computational resources. In this part two fundamental conditions are considered:

$$\sum_{i}^{m} x_{i,j} = 1 \forall j \in T \tag{1}$$

$$x_{i,j} \in \{0,1\} \forall i \in R, \forall j \in T \tag{2}$$

Makespan is equal to time that execution of last task on processing resource is finished. First limitation ensures that each task is assigned to only one processing resource.

A. Task scheduling using genetic algorithm

Genetic algorithm is one of the most famous evolutionary approaches for optimization which is inspired from genetics and natural selection process. In this approach, characteristics of potential solutions are improved by natural operators in genetic algorithm. Genetic algorithm behaves like biological genetics. This algorithm is one of the most attractive computational models which simulate natural evolution for solving many problems in many areas. Genetic algorithm consists of number of individuals and each of them is a potential solution for solving problem. By manipulating these solutions by genetic operators during different generations, an optimal or suboptimal solution is represented for the problem. Based on existing rules for evolution, only most appropriate solutions in the population have chance to survive to the next generation and creating offspring. The most important aspects of genetic algorithm are: a) defining objective function, b) defining and implementing genetic form of problem, c) defining and implementing genetic operators. In mapping between nature and genetic algorithm, computer is nature, individuals are solution to the problem, population is set of solution, fitness demonstrates quality of solutions, and gene is part of representation of solution. Different steps for genetic algorithm are:

- Initialization

In the first step an initial population is generated randomly. Each individual is considered a potential solution for the problem.

- Fitness function

After creating initial population, fitness function is used for evaluating goodness of individuals. Fitness function for each problem depends on our goal in optimization problem and it is different from one problem to another. In this problem makespan is considered as fitness function.

- Selection and elitism

Elitism means that best individuals go to next generation without any change and selection refers to selecting individuals with better fitness values for creating new individuals which are called offspring. For this problem biased roulette wheel is used for selection.

- crossover

In this step, offspring are generated from selected individuals. The most common way for crossover is that two chromosomes that are chosen as parents, exchange some of their genetic information with each other. Child has some characteristics of its both parents.in this problem one point crossover is used.

- mutation

The second way that genetic algorithm uses for searching cost surface is mutation. In mutation little changes is applied on one or some genes of chromosomes and in this way local minimum is prevented and diversity in the population is kept.

B. Task scheduling using PSO algorithm

B.1. PSO algorithm (particle swarm optimization)

PSO algorithm is one of the newest heuristic optimization algorithms that is proposed by Kennedy and Eberhart [9]. PSO algorithm can be used in optimization problems. In this algorithm behavior of flock of birds or group of fishes in order to get to desired destination is simulated. In this algorithm a population of particles is considered as initial space. Each particle in the population represents a potential solution and it is corresponding to individuals in the evolutionary algorithms. At first a flock of particles is generated randomly, position of each particle represents a potential solution in the problem space. For each particle a position vector is defined which is updated regularly (X^i), moreover there is a velocity vector which is similar to position vector, it is updated consistently too and leads to movement in search space. Based on proposed algorithm by Kennedy and Eberhart [9] the formula that exists for updating position vector is:

$$X_{k+1}^{i} = X_{k}^{i} + V_{k+1}^{i} \tag{3}$$

And presented vector for velocity vector is:

$$v_{k}^{i+1} = w_k v_k^i + c_1 r_1 (pbest_k^i - x_k^i) + c_2 r_2 (gbest - x_i^k) \tag{4}$$

In (4) c_1 and c_2 are constant numbers, r_1 and r_2 are numbers with normal distribution in the range of [0, 1]. Velocity vector changes in range of $[-v_{\max}, +v_{\max}]$.In each generation of this algorithm fitness function is applied on position vector of each particle and goodness of each particle is evaluated. Two defined parameters in (4) are pbest and gbest which play very important roles in the process of PSO. Pbest represents the best position that ith particle has experienced since the process has started and gbest represents the best position that exist between particles from beginning of the algorithm. Using velocity vector that is updated regularly, particle is able to search around its own best position (pbest) and best global position (gbest).

v_{id}^{k} is velocity of ith particle in kth iteration and represents the distance that a particle should travel. c_1 and c_2 are acceleration coefficient. w is inertia weight. Bigger w searches in the new spaces but smaller amounts of w searches in the existing space. If we want to balance between local and global search, inertia weight and acceleration coefficient should be opted for appropriately. In the velocity equation, new velocity for a particle is updated based on its previous velocity and its distance from its own best position and the best position between neighbors, after that particle moves toward its new position based on updated velocity vector. Evaluation of each particle in each iteration is done based on proportion function that is defined formerly. This process repeated until termination criterion of algorithm is satisfied.

B.2 PSO algorithm for task scheduling

At first we should have an accurate mapping between particles of PSO and problem solutions. The goal is how to improve efficiency of our resources and minimizing completion time of tasks. PSO is able to solve many optimization problems in different areas. If we suppose that there are n tasks and we aim to distribute them on m processing machines then particles should be defined in form of $m \times n$ matrixes. In this position matrix, m is number of available nodes at scheduling time and n is number of jobs. Position vector of each particle has two main characteristics:
- All elements of position matrix are 0 or 1.
- In each vector of position matrix there is only one 1 and other elements are zero.

	T1	T2	T3	T4	T5	T6	T7	T8	T9
M1	1	0	1	0	0	0	0	1	0
M2	0	0	0	1	1	0	0	0	1
M3	0	1	0	0	0	1	1	0	0

Fig.1 A typical particle for 9 independent tasks and 3 processing machine

In this matrix in each column, each 1 element depicts the machine that executes task. The second condition ensures that each task should be executed on only one machine. For example in the above position vector, task1 is executed on machine 1, task2 is executed on machine 3, task3 on machine 1 and so on.
Velocity matrix dimensions are exactly similar to position matrix i.e. $m \times n$. And ranges for its elements are $[-v_{\max}, +v_{\max}]$. So for each element in the population velocity matrix is as (5) (v_k is velocity matrix of kth particle):

$$v_k(i,j) \in [-v_{\max}, +v_{\max}] \quad i \in \{1,2,\ldots,m\} \quad j \in \{1,2,\ldots,n\} \quad (\forall i, j) \tag{5}$$

Similar to velocity matrix, pbest and gbest are $m \times n$ matrixes with 0 and 1 elements.
Pbest matrix represents a matrix that shows the best position of particle from beginning of the algorithm and gbest owns the best position between all particles from the beginning of the algorithm. In each step of the algorithm pbest and gbest are updated based on related formulas which are explained in the previous sections.

In each step of the algorithm particles are evaluated by fitness function of the algorithm. If particle's fitness is better than pbest_k, pbest_k is replaced by x_k. In order to update gbest we use pbest of all particles. If there is a pbest that is better than gbest, current gbest is replaced by pbest.
In order to update position vector of particles we use velocity matrix of that particle:

$$v_k^{(t+1)}(i,j) = w.v_k^t(i,j) + c_1 r_1\left(pbest_k^t(i,j) - X_k^t(i,j)\right) + c_2 r_2\left(gbest_k^t(i,j) - X_k^t(i,j)\right) \tag{6}$$

To update position vector of particles based on v_k in each column we compare values of all rows in that column, the element that has biggest value, its corresponding element in position vector turns to 1 and other elements in that column become 0.

Fitness function is defined based on our objective. In this paper we look for minimizing Makespan, i.e. the time that execution of last task is completed. So for each machine we sum job run time of all tasks that are assigned to that machine and the biggest number is said to be fitness of that particle. We repeat this process until stop criterion of algorithm is satisfied which in our algorithm it is equal to number of iterations. Flowchart of PSO algorithm is as Fig. 2.

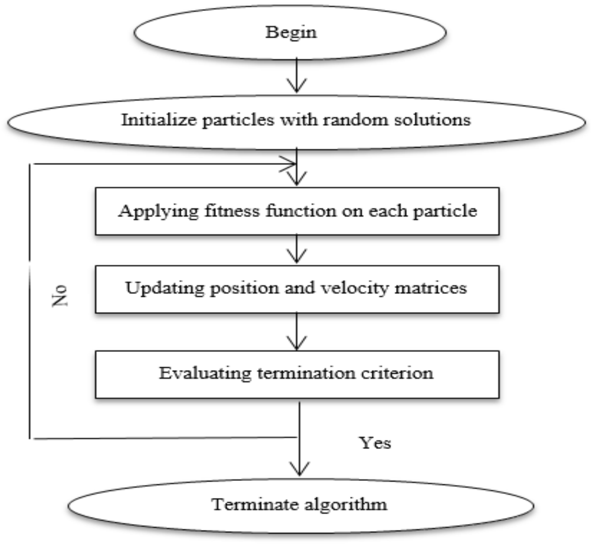
Begin
Initialize particles with random solutions
Applying fitness function on each particle
Updating position and velocity matrices
Evaluating termination criterion
No
Yes
Terminate algorithm

Fig.2 Flowchart of PSO algorithm

C. Modified PSO Algorithm

C.1. smallest Job to fastest processor

in this algorithm following steps are performed: At first we sort existing jobs and existing processors in ascending order, sorting processors is done based on their processing power. After sorting jobs and processors jobs are assigned based on one-to-one mapping.

C.2. Merging SJFP in PSO algorithm

In standard PSO algorithm initial particles are created randomly, but randomness decreases the chance of algorithm to converge to best solution, in order to improve the behavior of PSO algorithm, we merge shortest job to fastest processor algorithm(SJFP) into PSO, i.e. instead of generating initial population randomly we create them considering SJFP algorithm. All other steps are similar to standard PSO algorithm, so the flowchart of this modified PSO algorithm is as Fig. 3.

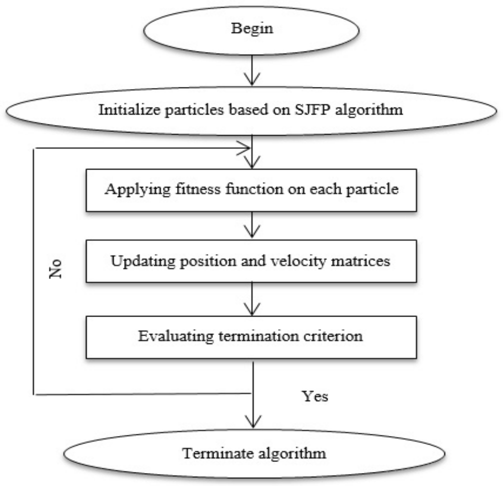
Begin
Initialize particles based on SJFP algorithm
Applying fitness function on each particle
Updating position and velocity matrices
Evaluating termination criterion
No
Yes
Terminate algorithm

Fig. 3 Flowchart of this modified PSO algorithm

III. Experiments and Results

The experimental environment that has been used is Eucalyptus cloud which is implemented on IPCRC center of Amirkabir University; we have used 4 VMs for evaluating performance of these algorithms, each VM has 4 GB of RAM and 30 GB of hard disk and 1 up to 4 cores of CPUs, in which each core has 2.3 GHz processing speed. In order to examine the performance of the proposed algorithm we have implemented Genetic algorithm and standard PSO algorithm too. In this task scheduling problem the criterion that is considered for evaluating performance is makespan. To test the performance of this algorithm we have tested two different scenarios. In the first scenario we changed number of tasks from 100 to 800. The second scenario is performed by changing number of iterations for constant number of tasks. Each experiment is repeated 10 times, and final result is the average of all 10 iterations. The criterion that is considered for the performance of this algorithm is makespan. The improvement percentage is calculated by (7) [11]:

$$\phi = (1 - \frac{\sum Makespan_{PSO}}{\sum Makespan_{GA}}) \times 100 \qquad (7)$$

Table I shows the parameters that are used for GA, and `Table II shows the parameters that are used for PSO and Modified PSO in the first scenario:

TABLE I
GA Parameters

Population size	60
Crossover probability	0.7
Mutation probability	0.01
Maximum no. of iterations	120

TABLE II
PSO And Modified PSO Parameters

Population size	60
w	0.65
c_1	2
c_2	2
Maximum no. of iterations	120

Final results for three algorithms are shown in Table III.

TABLE III
Final Result For Three Algorithms

Scheduling algorithm	Number of Tasks	Number of Resources	Makespan (s)	Improvement
Genetic Algorithm	100	4	665	2.22
PSO			665.85	
Modified PSO			650.23	
Genetic Algorithm	200	4	1100	10.45
PSO			1008	
Modified PSO			985	
Genetic Algorithm	400	4	2010	2.68
PSO			1988	
Modified PSO			1956	
Genetic Algorithm	600	4	3315	3.95
PSO			3225	
Modified PSO			3189	
Genetic Algorithm	800	4	4712	6.66
PSO			4679	
Modified PSO			4398	

The performance of these three algorithms based on noted values is shown in Fig. 4, in which x-axis represents number of tasks and y-axis shows makespan.

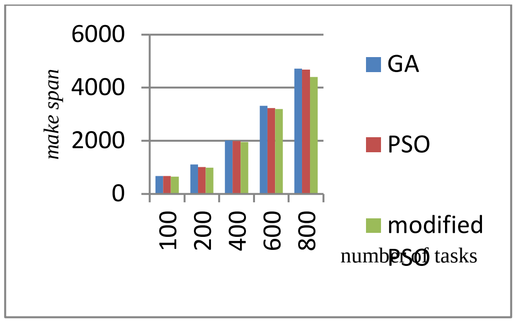

Fig. 4 Comparing performance of different algorithms based on noted values

As it is shown in the Table III and Fig. 4, we have performed experiments for 100,200,400 and 800 tasks on 4 VMs. The criterion that is considered for evaluating performance of these algorithms is makespan i.e. the time in which execution of last task on processing machines becomes completed. Based on presented results in table III and Fig. 4 in comparison between Genetic algorithm and PSO algorithm, in three sets of experiments PSO algorithm shows better result than Genetic algorithm, but Modified PSO algorithm outperforms these two algorithms in all sets of experiments i.e. makespan for this algorithm is has the smallest value in all experiments.

In the second scenario all parameters are the same as table II and III except number of iterations which is changed from 20 to 360 iterations. In this experiment number of tasks is constant and is equal to 400. Experimental results are shown in Fig. 5.

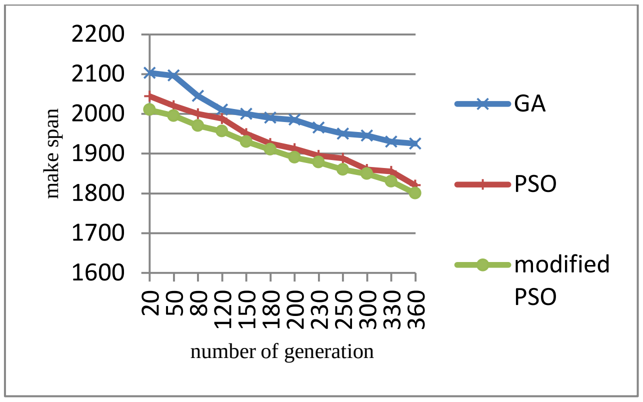

Fig. 5 Experimental results of different algorithms

As it is shown in Fig. 5 by incrementing number of generation makespan decreases, but because of optimized initial particles in the beginning of algorithm, modified PSO algorithm is able to produce solutions with more optimal makespan.

IV. Conclusion

In this paper the problem of task scheduling in cloud computing environment is evaluated. PSO algorithm and Genetic algorithm are most famous algorithms for scheduling tasks in distributed systems. In order to improve the performance of standard PSO algorithm the modified PSO algorithm is suggested, in which SJFP algorithm is merged into standard PSO algorithm for generating initial population in order to minimize makespan. Our experiments depicts that even if both Genetic algorithm and PSO algorithm show acceptable results, it can be said that by and large PSO algorithm shows better results than Genetic algorithm but modified PSO algorithm outperforms these two algorithms from minimizing makespan point of view. This algorithm can be used in cloud computing environment for efficient scheduling of tasks on existing resources, so that completion time of tasks become minimized.

References

[1] China cloud computing. Peng Liu:cloud computing definition and characteristics,http://www.chinacloud.cn/.2009-2-25.

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, I. Brandic, "Cloud Computing and Emerging IT Platforms", Vision, Hype, and Reality for Delivering Computing as the 5th Utility, *Future Generation Computer Systems* 25(6), 599–616 (2009). http://dx.doi.org/10.1016/j.future.2008.12.001

[3] P. Kumar, A. Verma, "Independent Task Scheduling in Cloud Computing by Improved Genetic Algorithm", *International Journal of Advanced Research in Computer Science and Software Engineering*,Vol2, Issue 5, May 2012.

[4] Z. Yingfeng, L. Yulin, "Grid Computing Resource Management Scheduler Based on Evolution Algorithm[j]", *Computer Engineering Conference*, 2003, 29(15):1102175.

[5] P. Roy, M. Mejbah, N. Das. "Heuristic Based Task Scheduling in Multiprocessor Systems with Genetic Algorithm by choosing the eligible processor", *International Journal of Distributed and Parallel Systems (IJDPS)*, Vol3, No.4, July 2012.

[6] Abraham, R. Buyya, and B. Nath." Nature's heuristics for scheduling jobs on computational Grids", *8th IEEE International Conference on Advanced Computing and Communications (ADCOM 2000)*, India, 2000.

[7] H. Yin, H. Wu, J. Zhou, "An Improved Genetic Algorithm with Limited It rat ion for Grid Scheduling", *IEEE Sixth International Conference on Grid and Cooperative Computing*, GCC 2007, Los Alamitos, CA, pp. 221-227, 2007.

[8] R. Verma , S. Dhingra, "Genetic Algorithm for Multiprocessor Task Scheduling", *IJCSMS International Journal of Computer Science and Management Studies*, Vol.1, Issue 02, pp. 181-185, 2011

[9] J. Kennedy, R.C. Eberhart, "Particle swarm optimization", *Proc, IEEE Conf. Neural Netw.*, vol. IV, IEEE, Piscataway, NJ, 1995,pp.1942-1948.

[10] L. Zhang, Y. Chen, B. Yang "Task Scheduling Based on PSO Algorithm in Computational Grid", *2006 Proceedings of the 6th International Conference on Intelligent Systems Design and Applications*, vol-2, 16-18 Oct, 2006,Jinan, China.

[11] T. Chen, B. Zhang, X. Hao, Y. Dai, "Task scheduling in grid based on particle swarm optimization", *The Fifth International Symposium on Parallel and Distributed Computing*, ISPDC '06. pp. 238-245, 2006.