Journal of Theoretical and Applied Information Technology
10th September 2013. Vol. 55 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

TASK SCHEDULING ALGORITHM BASED ON HYBRID PARTICLE SWARM OPTIMIZATION IN CLOUD COMPUTING ENVIRONMENT

[1]GOMATHI B, [2]KARTHIKEYAN KRISHNASAMY
[1]Asstt Prof., Department of IT, Hindusthan College of Engg. and Tech., Coimbatore, India
[2] Prof., Department of IT, Sri Krishna College of Engg. and Tech., Coimbatore, India
E-mail: [1]gomathi.babu@gmail.com , [2]karthiaish1966@gmail.com

ABSTRACT

Cloud computing environment can offer dynamic and elastic virtual resources to the end users on demand basis. Task scheduling should satisfy the dynamic requirements of users and also need to utilize the virtual resources efficiently in cloud environment, so that task scheduling in cloud is an NP-Complete problem. In this paper, we present a Hybrid Particle Swarm Optimization (HPSO) based scheduling heuristic to balance the load across the entire system while trying to minimize the makespan of a given task sets. Finally, experimentation results show that hybrid PSO is more effective as compared to using existing simple PSO algorithm.
Keywords: *Task scheduling, Particle Swarm Optimization, Cloud computing, Makespan, Resource Utilization*

1. INTRODUCTION

Cloud[1,2] provides the greater flexibility for user to use resources as a service by low cost, so it increases the computational power and storage capacity for user. Task scheduling in cloud environment is responsible to assign user tasks to appropriate resources for execution. Since the cloud resources are abstracted, virtualized and dynamically scalable, Task scheduling which is NP-Complete problem has always been an important issue in cloud environment. In order to schedule the tasks quickly and efficiently, a task scheduling based on HPSO is proposed to meet the task requirements of users and improve the utilization of resources while minimizing the makespan of the given task sets.

Particle Swarm Optimization (PSO)[3] is an evolutionary algorithm that simulates the behavior of a flock of birds to a desired place. Due to its simplicity and its effectiveness, PSO is used in wide range of application with lowest computational cost. Due to problems in PSO like premature convergence in later generations, we present HPSO by using vector differential operator in Differential Evolution (DE) algorithm. This HPSO is used to improve the resource utilization while try to find optimal schedule for given task set in the cloud environment.

The rest of this paper is organized as follows. Section 2 gives related work. In section 3, we describe makespan minimization problem. Section 4 presents task scheduling algorithm that uses HPSO. Section 5 presents an experimental evaluation of proposed method. Section 6 concludes the paper and discussed some future enhancement.

2. RELATED WORK

Task scheduling is a challenging problem in distributed system since it involves heterogeneous environment. To obtain good method to solve this problem, research is conducted on various heuristics approaches for task scheduling in distributed system. GA for multiprocessor scheduling was proposed by [5] which scheduled the task on a multiprocessor system to minimize the finishing time of schedule. Individuals consisted of multiple lists, with each list representing the tasks assigned to one processor. The approach proposed in this paper restricts action of genetic operators to ensure the validity of evolved individuals. As a result, some parts of the search space may be unreachable. Zomaya etal[6] proposed a dynamic load balancing strategy based on a genetic algorithm. To speedup the scheduler and reduce the chance of processor becoming idle, tasks in the sliding window could be considered for execution

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and characteristics of tasks known in advance. Adaptive threshold helped to balance the load of the processor dynamically.

Yajun etal[7] combined FCFS and GA to balance the load of sequential tasks under grid environment in order to achieve minimum execution time and maximum load utilization. A sliding window technique was presented to trigger the switch between FCFS and GA as well as helped to make rapid task assignment. To avoid the premature convergence, low convergence speed and local optima, hybrid Adaptive GA was proposed [8] to adjust the crossover and mutation probability adaptively and non-linearly to expand the search space as well as to improve the convergence. Sandeep etal[9] proposed fuzzy-GA optimization to schedule the job in hadoop framework in order to improve the resource utilization. The revised scheduling algorithm was used to predict the execution time of tasks for better load balancing across nodes in the cloud environment, but efficiency of the prediction was highly affected by the choice of the task vector.

Lei Zhang etal [10] proposed Particle Swarm Optimization for task scheduling problem to generate an optimal schedule in grid environment in order to complete the tasks in a minimum time as well as to utilize the resources in an efficient way. Yin etal [11] proposed HPSO algorithm to assign tasks to a set of distributed processors such that total execution cost and communication cost were minimized and system throughput was maximized. HPSO used hill climbing heuristic to speed up the convergence. Task assignment with load balancing using hybrid PSO in distributed system was proposed in [12]. HPSO combined PSO for global optimistic result and Simulated Annealing (SA) for local optimistic result. It was cost effective when compared to other variants of PSO.

The task scheduling algorithm based on PSO [13] improved the utility of resources in Grid environment. This algorithm considered dependent tasks and dynamic heterogeneous resources for which global optimal solution was adjusted at runtime. Lizheng etal [14] proposed PSO for task scheduling in cloud environment in order to minimize the processing time. It converged faster in larger task set. Due to loss of diversity, PSO suffered in premature convergence. So, we use HPSO with differential operator to avoid the premature convergence in task scheduling problem. In this paper, we focus on minimizing the makespan as well as maximizing the resource utilization.

3. PROBLEM FORMULATION

The scheduling of tasks on the cloud resources have several objectives. We focus on minimizing makespan as well as maximizing resource utilization. Therefore, fitness function is defined to evaluate the dual optimization criteria. In this paper, we consider that cloud environment has heterogeneous resources with different processing capability. The processing time of task may vary according to the task scheduling on different resources.

To formulate the problem, we denote the set of n independent tasks as *Ti*, where i={0,1…n} and set of m processors as R_j, where j={0,1,…m}. Assume that the execution time $P_{i,j}$ for task i on processor j is known and resource utilization for each processor is represented as *Ri(utilization)*. In permutation matrix x, entry $x_{i,j}$=1 if task i is assigned to processor j, otherwise $x_{i,j}$=0. This gives the assurance that each task is assigned to exactly one processor.

Our first objective is minimizing the makespan, which is longest task completion time among all processors in cloud.Another factor to consider is average resource utilization, which is defined as sum of the utilization of all processors divided by the total number of processors. Equation 1 and 2 represent makespan *MS* and average resource utilization $\bar{Ri}$ respectively. We evaluate both objective functions using single fitness function in equation 3 as shown below.

$$MS = \max_{1 \le i \le m} \sum_{j=1}^{n} P_{i,j} * x_{i,j} \quad (1)$$

$$\bar{Ri}(utilisation) = (\sum_{i=1}^{m} Ri(utilisation)) / m \quad (2)$$

$$fv = \min MS \Big/ \max \bar{Ri}(utilisation) \quad (3)$$

Subject to,

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

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

$$Ri(utilisation) = (\sum_{j=1}^{n} P_{i,j}) / MS, \forall i \in R$$

Journal of Theoretical and Applied Information Technology
10th September 2013. Vol. 55 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4. TASK SCHEDULING BASED ON PSO

4.1. Classical PSO

PSO is an evolutionary algorithm that simulates social behavior of bird flocking to find food source of fish schooling to protect themselves from a predator. Each solution candidate called a particle in PSO which is flying through a search space. In PSO population represents the number of particles in the search space. The velocity of each particle directs movement of flying particle in search space. The position of particle depends on its best position (pbest) and position of the best particle (gbest) in an entire population. The whole population is initialized randomly. The fitness value of particles which is used to measure the performance of particle is evaluated and optimized in each generation. In each generation, velocity and position of particle is updated as follows:

$$V^{i}_{k+1}=W_k V^{i}_{k}+c_1 r_1 \quad (P^{i}_{k}-X^{i}_{k}) \quad +c_2 r_2 \quad (P^{g}_{k}-X^{i}_{k}) \tag{4}$$

$$X^{i}_{k+1} = X^{i}_{k} + V^{i}_{k+1} \tag{5}$$

Where, the variables *r1* and *r2* are random numbers between 0 and 1. *c1* and *c2* are acceleration co-efficients. W is the inertia weight. V^{i}_{k} is the velocity of particle i at iteration k. X^{i}_{k} represents the current position of particle i at iteration k. The classical PSO updates the velocity of a particle using three components: inertia of previous velocity provides momentum of particle as well as control the balance between exploration and exploitation in the search space, social component represents the cooperation of the particles in moving towards the global best position found in the search space and cognitive component represents the private experience of the particle itself. In the HPSO, cognitive term is replaced as weighted difference of the position values of any two randomly chosen different particles in the swarm. This replacement which is explained in the next section, helps to avoid the premature convergence which is caused by rapid loss of diversity within the swarm.

4.2. Particle Representation

To solve a task scheduling problem, each solution should map with one of the particle in the population. Each particle is defined as *n* dimensional vector which is responsible for *n* tasks in the task scheduling. Figure 1 depicts the representation of particle in task scheduling.

Task0	*Task1*	*Task2*	*Task3*	*Task4*
Resource2	*Resource0*	*Resource1*	*Resource0*	*Resource1*

Figure 1: Resource Mapping in HPSO Particle

4.3. Solution Representation

The initial population of particle is generated for algorithm randomly. The initial position and velocity of each particle is generated using the following formula [15]:

$$X_0^{k} = X_{min} + (X_{max} - X_{min}) * r \tag{6}$$

$$V_0^{k} = V_{min} + (V_{max} - V_{min}) * r \tag{7}$$

where $X_{min} = V_{min} = -0.4$ and $X_{max} = V_{max} = 4.0$ and r is a uniform random number between 0 and 1. As the position of particle is a continuous value, we need to convert continuous value into discrete permutation using Small Position Value (SPV) rule [15] for task scheduling in HPSO algorithm. By applying SPV rule, continuous position vector $X^{i}_{k}=\{ X^{i}_{1,} X^{i}_{2...} X^{i}_{n}\}$is transformed as dispersed permutation vector $S^{i}_{k}=\{S^{i}_{1},S^{i}_{2},\ldots S^{i}_{n}\}$. To find resource allocation for each task, we should convert permutation vector S_k into resource vector $R^{i}_{k}=\{ R^{i}_{1,} R^{i}_{2...} R^{i}_{n}\}$ by using the following equation,

$$R^{i}_{k} = S^{i}_{k} \bmod m \tag{8}$$

The Table1 illustrates the solution representation of a particle for 9 tasks and 3 resources.

4.4. The Hybrid PSO Algorithm

Task scheduling which is NP-Complete problem is a complicated issue in achieving better performance in cloud environment. In this section, we present a hybrid PSO for scheduling tasks to minimize the makespan of given task set while improving the efficiency of resources. In hybrid PSO, velocity of particles can be updated with

Table 1: Solution Representation of a Particle

Dimension	X_i^k	S_i^k	R_i^k
0	0.89	2	2
1	-0.11	1	1
2	3.15	7	1
3	-0.39	0	0
4	3.41	8	2
5	2.64	5	2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

6	3.00	6	0
7	1.03	3	0
8	1.52	4	1

vector differential operator from Differential Evolution (DE). In the proposed method, cognitive term in the velocity equation is replaced by the term containing the weighted difference (δ) between the position vectors of any two randomly chosen distinct particles from the whole population as shown below [4]:

$$\delta = X^k - X^j \quad (9)$$

$$V^i_{k+1} = W_k V^i_k + \beta\delta + c_2 r_2 (P^g_k - X^i_k) \text{ if rand}(0,1) < C \quad (10)$$

$$= V^i_k, \text{ otherwise}$$

Where *CR* is the crossover constant, δ is the n dimensional difference vector and β is a scale factor in (0,1). The differential operator is used to provide additional exploration capability in search space. The new trial location T_i is created for particle i as shown below:

$$T_i = X^i_k + V^i_{k+1} \quad (11)$$

Unlike PSO, particle is actually shifted to new location only if the new location gives better fitness value as shown below, so HPSO helps to avoid premature convergence due to loss of diversity[4].

$$X^i_{k+1} = T_i \quad \text{if } fv(T_i) < fv(X^i_k) \quad (12)$$

$$= X^i_k \quad \text{otherwise}$$

Finally, if particle gets stagnated at any place in solution space, then it is shifted to new location using equation (6). The pseudo code of HPSO is presented in Figure 2.

5. EXPERIMENTAL EVALUATION

In this section, we present the metric of comparison and experimental setup to test the proposed algorithm on the simulated cloud environment. Initially, test runs were based on the following parameters such as number of tasks is 100, number of resources is 5,number of generation is 100, population size is 10, crossover constant *CR* is 0.9 and scale factor β is 0.8. The performance of proposed algorithm was compared with classical PSO algorithm based on two metrics such as makespan and average resource utilization.

As classical PSO and hybrid PSO are stochastic and result may be different for particular problem, each experiment was repeated 20 times and average has been calculated. We tested and observed these algorithms in terms of such performance metrics under different parameters. Three tests were performed for each set of parameters as shown below.

1. *Set dimension of particle as number of ready tasks*
2. *Initialize particles position and velocity vectors randomly using equation (6) and (7)*
3. *By using SPV rule, find discrete vector* $S^i_k=\{S^i_1,S^i_2,\ldots S^i_n\}$ *from the continuous position vector* $X^i_k=\{X^i_1, X^i_2 \ldots X^i_n\}$ *for each particle and then map discrete vector* S^i_k *elements of each particle into resource vector* $R^i_k=\{R^i_1, R^i_2 \ldots R^i_n\}$ *according to equation (8)*
4. *Evaluate the fitness of each particle using equations from (1) to (3)*
5. *If fitness value is better than personal best pbest, update pbest by current fitness value. Select the best particle as gbest.*
6. *Update velocity and position of each particle using equation (9) through (12)*
7. *If a particle gets stagnated for predetermined number of iteration, then particle is shifted to new location using equation (6)*
8. *If maximum iteration is reached or stopping criteria is satisfied, then stop. Otherwise repeat from step 3*

Figure 2 Pseudo Code of HPSO for Task Scheduling

As classical PSO and hybrid PSO are stochastic and result may be different for particular problem, each experiment was repeated 20 times and average has been calculated. We tested and observed these algorithms in terms of such performance metrics under different parameters. Three tests were performed for each set of parameters as shown below.

5.1. Variation In Number Of Tasks

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

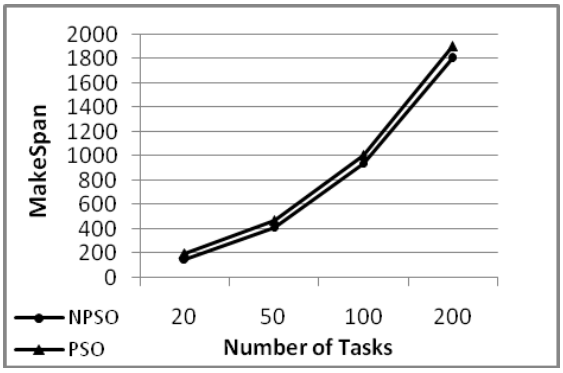

Figure 3. Makespan Vs Number of Tasks

This section gives effects on makespan and average resource utilization by HPSO and PSO when number of tasks was changed from 20 to 100. When the number of tasks was increased, makespan produced by PSO and HPSO were increased linearly when larger number of tasks has to be scheduled. Algorithms have taken longer time to complete all the tasks. As we have seen in figure 3, HPSO algorithm performed better than PSO algorithm. In figure 4, resource utilization using HPSO was varying from 94 to 99 percent while PSO utilized the resources ranges from 91 to 97 percent. HPSO increased the resource utilization up to 99 percent when number of tasks was increased, then indicating that HPSO is able to work better in the case of more tasks.

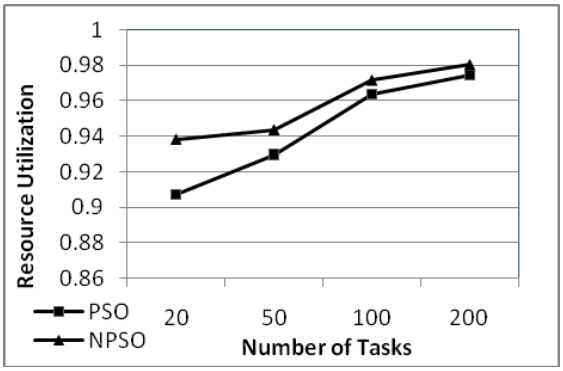

Figure 4. Resource utilization Vs Number of Tasks

5.2. Variation In Number Of Resources

In this section, the effect on the performance of HPSO was observed when number of resources was varying for the same set of tasks. Figure 5 shows that makespan was significantly reduced as the number of resources was increased from 5 to 20. Even though, there was larger number of tasks to be scheduled, extra resources were handled the excess load in the system.

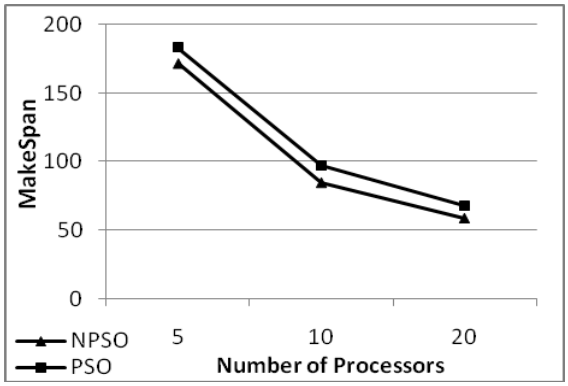

Figure 5. Makespan Vs Number of Processors

Unlike the improvement in makespan, figure 6 shows that average resource utilization reduced as the number of resources increased. It shows that it is difficult to balance load across the system when large number of tasks were needed to balance out across a larger system. Therefore, improving the makespan by increasing number of resources results as a cost of lower resource utilization.

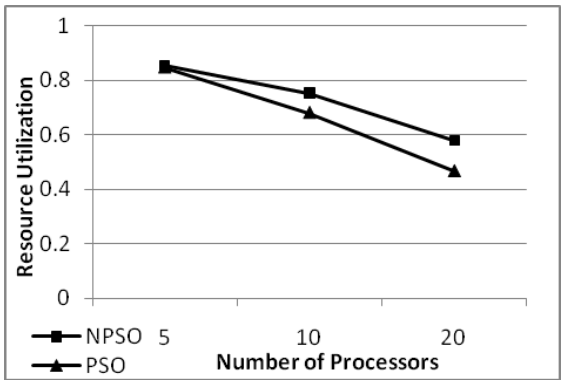

Figure 6. Resource utilization Vs Number of Processors

5.3 Convergence Analysis

Figure 7 shows that makespan reduced as the number of iterations was increased. It shows that the quality of the task scheduling improved after each iteration. However, it is shown that PSO is getting struck in premature convergence, but HPSO gave faster convergence as well as reduction of makespan than PSO.

In summary, the above shows that HPSO is suitable to handle the task scheduling in cloud environment. The different parameters were changed to identify the optimal set of parameters for HPSO to improve the performance of task scheduling problem.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

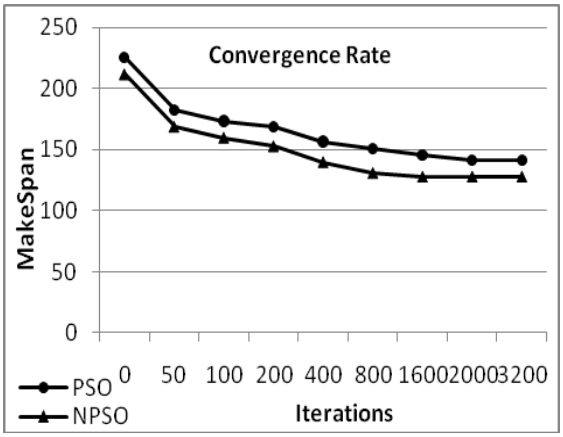

Figure 7. Convergence Analysis

6. CONCLUSION

The proposed task scheduling technique using HPSO has been effective to minimize the makespan as well as maximize the resource utilization in cloud environment. It is found that HPSO based task scheduling can achieve better load balancing as compared to PSO based scheduling. As part of our future work, we would like to integrate HPSO based task scheduling on the map-reducing framework in cloud environment.

REFERENCES:

[1] G. Boss, P. Malladi, D. Quan, L. Legregni, and H. Hall, "Cloud computing," *Technical Report, IBM* High Performance on Demand Solutions, 2007.

[2] Dikaiakos,M, katsaros,D, Mehra,P,Vakali.A, Cloud Computing: Distributed Internet Computing for IT and Scientific Research", *IEEE Transactions on Internet Computing*, pp. 10-13, 2009.

[3] J. Kennedy and R. C. Eberhard, "Particle swarm optimization", Proc. of IEEE *Int'l Conf. on Neural Networks*, pp.1942-1948, Piscataway, NJ, USA,1995.

[4] Swagatam Das,Amit Konar,uday K.Chakraborty, Improving Particle Swarm Optimization with Differentially Perturbed Velocity,*ACM*,2005.

[5] Edwin S.H.Hou, Nirwan Ansari, Hong Ren, A Genetic Algorithm for Multiprocessor Scheduling, *IEEE Transactions on Parallel and Distributed Systems*, Vol.5, No.2,February 1994.

[6] Zomaya A Y, Teh Y-H. Observations on using genetic algorithms for dynamic load-balancing, *IEEE Transactions on Parallel and Distributed Systems*, pp. 899-912, 2001.

[7] Yajun Li, Yuhang Yang, Maode Ma, Liang Zhou, A hybrid load balancing strategy of sequential tasks for grid computing environments, *Future Generation Computer systems*, pp.819-828,2009.

[8] Youchan Zhu and Xueying Guo, Grid Dependent Tasks Scheduling Based on Hybrid Adaptive Genetic Algorithm, *Global Congress on Intelligent Systems*, 2009.

[9] Sandeep Tayal, Tasks Scheduling optimization for the Cloud Computing systems, *International Journal of Advanced Engg. Sciences and Technologies*, Vol No.5, IssueNo.2, pp. 111-115, 2011.

[10] L. Zhang, Y. Chen, R. Sun, S. Jing, and B. Yang. "A task scheduling algorithm based on PSO for grid computing". *International Journal of Computational Intelligence Research*, vol.4, 2008.

[11] P.Yin, S.Yu, P.Wang, Y.Wang, A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems, *Computer standards and interfaces*, 2006,28, pp.441-450

[12] P.Visalakshi and S.N.Sivanandam, Dynamic Task Scheduling with Load Balancing using Hybrid Particle Swarm Optimization, *International Journal for Open Problems Compt. Math*. Vol.2, 2009.

[13] Tingwei Chen, Bin Zhang, Xianwen Hao, Yu Dai, Task Scheduling in Grid Based on Particle Swarm Optimization, *Proceedings of the Fifth International Symposium on Parallel and Distributed Computing (ISPDC'06)*,2006.

[14] Lizheng Guo, Shuguang, shigen, changyuan, Task Scheduling Optimization in cloud Computing based on Heuristic Algorithm, *Journal of Networks*, Vol 7,2012.

[15] M. Fatih Tasgetiren, Yun-Chia Liang, Mehmet Sevkli, and Gunes Gencyilmaz, "Particle Swarm Optimization and Differential Evolution for Single Machine Total Weighted Tardiness Problem," *International Journal of Production Research*, pp.4737-4754 vol. 44, no. 22, 2006.