BF-PSO-TS: Hybrid Heuristic Algorithms for Optimizing Task Schedulingon Cloud Computing Environment

Hussin M. Alkhashai
Department of Computer Science, Faculty of Computers & Information Cairo University, Cairo, Egypt

Fatma A. Omara
Professor, Department of Computer Science, Faculty of Computers &Information, Cairo University, Cairo, Egypt

***Abstract*---Task Scheduling is a major problem in Cloud computing because the cloud provider has to serve many users. Also, a good scheduling algorithm helps in the proper and efficient utilization of the resources. So, task scheduling is considered as one of the major issues on the Cloud computing systems. The objective of this paper is to assign the tasks to multiple computing resources. Consequently, the total cost of execution is to be minimum and load to be shared between these computing resources. Therefore, two hybrid algorithms based on Particle Swarm Optimization (PSO) have been introduced to schedule the tasks; Best-Fit-PSO (BFPSO) and PSO-Tabu Search (PSOTS). According to BFPSO algorithm, Best-Fit (BF) algorithm has been merged into the PSO algorithm to improve the performance. The main principle of the modified BFSOP algorithm is that BF algorithm is used to generate the initial population of the standard PSO algorithm instead of being initiated randomly. According to the proposed PSOTS algorithm, the Tabu-Search (TS) has been used to improve the local research by avoiding the trap of the local optimality which could be occurred using the standard PSO algorithm. The two proposed algorithms (i.e., BFPSO and PSOTS) have been implemented using Cloudsim and evaluated comparing to the standard PSO algorithm using five problems with different number of independent tasks and resources. The performance parameters have been considered are the execution time (Makspan), cost, and resources utilization. The implementation results prove that the proposed hybrid algorithms (i.e., BFPSO, PSOTS) outperform the standard PSO algorithm.**

Keyword---Cloud computing; task scheduling; CloudSim; Particle Swarm Optimization; Tabu search; Best-Fit

I. INTRODUCTION

In recent years, there has been a rapid development of science and technology, including in computer science, where the development and modernization of the Cloud computing and Big Data in general service have been used[1]. So, this growth leads to more demand for this technology because of the different demands by the users with the evolution of the Web services, which provide the user with all the needed requirements and measuring anywhere and anytime (e.g., servers, storage, networks, operating systems). These services could be offered by Cloud computing[2].

Cloud computing fosters elasticity and seamless capability of interchanging the information and resources between the user and providers through the Internet. Therefore, Cloud computing can help enterprises to improve the creation and delivery of IT solutions by providing them with easy access to services in a cost effective and flexible manner. On the other hand, the Cloud computing applications span many levels, including business, technology, government, smart grids, intelligent transportation networks, disaster management, automation, and data analysis[3]. Cloud computing service providers make the large-scale network servers from the data center to be large-scale virtual resources. Infrastructure as a Service (IaaS) level is the delivery of hardware (e.g., storage and network), and associated software (OS, visualization technology, file system), and provide a lot of computational capacities to serve remote users in a flexible and efficient way. In this model, the resources are provisioned in the form of Virtual Machines (VMs) deployed within the Cloud equipment consisting of pooling data, physical resources etc. for fulfilling the requests. Therefore, resource management sub systems in the Heterogeneous Distributed Computing System (*HDCS*) are designed to schedule the incoming tasks in getting the service[4].

On the other hand, there are some challenges facing Cloud Computing. One of the major challenges is the task scheduler.

Task scheduling is the most important issue in the Cloud computing because the user will have to pay for using the resources on the basis of time. The goal of task scheduling is to distribute the load evenly in the system by maximizing resource utilization and reducing execution time.

Typically, the numbers of tasks and resources are tremendous in the Cloud computing environment, especially for big data applications. Therefore, some efforts have been made towards this issue. For instance, formulating the scheduling problem as a general task graph to be executed on different VMs to detract execution time and maximize the resource utilization, using heuristic algorithms [5].

Kennedy and Eberhart [6], have presented a self-adaptive global search-based optimization algorithm, called PSO, in terms of time and resources. The algorithm is similar to other population based algorithms as Genetic algorithm without direct recombination of individuals of the population. Instead, it relies on the social behavior of the particles. In every generation, each particle adjusts its trajectory based on its best position called (local best) and the position of the best particle

called (global best) of the entire population. These concepts increase the random nature of the particle and converge quickly to a global minimum with a reasonable good solution. *PSO* algorithm has become popular because of its simplicity and its effectiveness in a wide range of applications. There are some applications that have used *PSO* algorithm to solve the NP-Hard problems like scheduling and allocation task problem, the data mining problem, and the environmental engineering problem[7].

In this paper, two hybrid heuristic algorithms, *BFPSO* and *PSOTS*, have been proposed to find a suitable task scheduling based on particle swarm algorithm in the Cloud computing environment. The main issues of these hybrid algorithms are reducing execution time (Makspan), reducing cost and maximizing resources utilization.

According to the first hybrid (*BFPSO*) algorithm, the BF algorithm has been merged into *PSO* for generating the initial population instead of generating it randomly in the standard *PSO*.

According to the second hybrid(*PSOTS)* algorithm, *TS* algorithm has been merged to the standard *PSO* to improve local research by avoiding the trap of the local optimality. This leads to improve the performance.

The proposed hybrid algorithms, *BFPSO, PSOTS,* have been implemented using the open source Cloudsim3.0.3 simulator with different number of independent tasks and different number of VMs.

The rest of this paper is organized as follows: Section 2 related work is discussed. In Section 3, the problem has been explained. In Section 4, problem statement is discussed. Sections 5 and 6 describe the proposed algorithms in detail, and its implementation and performance evaluation. Finally, section 7 presents the conclusions, and future research directions.

II. Related Work

In general, task scheduling is an important issue in any business, especially in Cloud computing, where it is considered as one of the main challenges facing the cloud computing because the service provider has to serve many users applications at different times and from different places. Therefore, schedule these tasks to be working on Cloud computing environment has to be more flexible and quick way. Thus, much research has been done in this field.

SolmazAbdi, Seyyed Ahmad Motamedi, and SaeedSharifia[8] have proposed a task schedule algorithm. The main principle of this algorithm is that the existed jobs and processors are sorted in ascending order. The processors are sorted based on their processing power. After sorting the jobs and processors, jobs are assigned using one to one mapping. After that, the Shortest Job to Fastest Processor (*SJFP*) algorithm is applied to generate initial population for the *PSO* algorithm. They used 4 VMs for evaluating performance of these algorithms, each VM has 4 GB of RAM and 30 GB of hard and 1 up to 4 of CPUs, in which each CPU has 2.3 GHz processing speed. In this task scheduling problem the criterion that is considered for evaluating performance is makespan. To test the performance of this algorithm, there are two different scenarios. In the first scenario, they 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 the final result is the average of all 10 iterations. The criterion that is considered for the performance of this algorithm is Makspan.

Gomathi B. Karthikeyan[9] has proposed a Hybrid Particle Swarm Optimization (*HPSO*) based scheduling heuristic to balance the load across the complete system while trying to minimize the makespan of a given task sets. In the *HPSO* algorithm, velocity of particles can be updated with vector differential operator from Differential Evolution (DE). 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. The differential operator is used to provide additional exploration capability in search space. Particle is actually shifted to new location only if the new location gives better fitness value, then it is shifted to new location. The implementation results showed that *HPSO* algorithm is more efficient when compared with the standard *PSO* algorithm.

Shaobin Zhan, HongyingHuo Shenzhen[10] have improved the PSO algorithm to scheduling resources in the Cloud computing. This Improvement has been satisfied by introducing the simulated annealing heuristic with its fast random global searching ability for each iteration of the PSO algorithm to improve convergence rate, and guarantee the accuracy of the original PSO algorithm, which avoids sinking into local optima. They considered the estimation of both Genetic Algorithm and Simulated Annealing and builds up A calculation. As indicated by sort of the undertakings, the diverse weight of parameters can be given to discovering assets that fulfill the QOS of the errands and their desires. Calculation steps first execute ventures of GA after which toughening comes which helps to enhance nearby hunt capacity of GA. This calculation proficiently finishes the looking of assets and assignment process in distributed computing.

Through experiments, the results show that this algorithm can reduce the average running time, and raise the rate resources availability.

Ali Al-maamari, Fatma A. Omara[11] have merged the Cuckoo algorithm into the *PSO* algorithm, called *PSOCS*, to generate an optimal scheduling of tasks in the Cloud computing environment in order to complete the tasks in a minimum execution time, as well as, resources utilization. Cuckoo Search (*CS*) heuristic has been used to improve the load balance and avoid the trap of the local optimality. According to the proposed *PSOCS* algorithm, tasks have been assigned to a set of distributed VMs such that total execution time was minimized and system throughput was maximized.

III. Scheduling Problem Modelling

The goal of the task scheduling is to minimize the tasks' execution time, as well as, the execution cost, and optimize resource utilization. According to the work in this paper, the

number of tasks is considered more than the number of resources (i.e., $n>>m$), and tasks cannot be assigned to different resources, that is, tasks are not allowed to migrate between resources.

To formulate the problem, consider the set of tasks is defined as $Ti = \{1, 2, \dots n\}$where n is the number of independent tasks and $Rj = \{1, 2, \dots m\}$is the set of computational resources. Suppose the execution of task i on VM j is defined as $CTij$. The total execution time (i.e., Makspan) $CTij$ is defined using the following equation[8],[12]:

$$Makspan = CTmax\ (i, j), i \in T,$$

$$i = 1,2, \dots n\ and\ j \in VM, J = 1,2, \dots m \qquad (1)$$

Where $CTmax$ is the maximum time for completing task i on a virtual machine j.

In addition, ***the total cost*** of executing all tasks on the available resources is calculated using equation (2) [13]:

$$Total\ Cost = \frac{Task\ length\ *\ Cost\ per\ seconds}{VM\ mips} + Processing\ Cost \qquad (2)$$

On the other hand, the utilization of resource represents the ratio between the total busy time of Virtual Machine and the total finish execution time of the parallel application. Therefore, Resources Utilization (U) for processing all tasks on VMs is represented by Equation (3)[5][11].

$$U = \frac{final\ VM\ available\ time}{\#VMs * schedulingtime} * 100 \qquad (3)$$

IV. PROBLEM STATEMENT

In the Cloud computing environment, a huge-scale computing task is typically split into several tasks, and finding a map between tasks and resources is achieved through visualization technologies. The major problem of task scheduling is to define the best mapping of n independent tasks into m heterogeneous resources(i.e., VMs) with the goal of minimizing the execution time (***Makespan***), and cost resources and maximizing resources utilization.

The goal of task scheduling is to create a mapping between tasks (T) and resources (R). i.e.;

$$f: T \rightarrow R$$

In this paper, the *PSO, BFPSO, and PSOTS* algorithms have been applied to find f. Note that, in this study, the following assumptions have been considered:

1- *tasks are independent and different sizes*
2- *The resources (i.e., VMs) are heterogeneous*
3- *The available resources are of exclusive usage and cannot be shared among different tasks.* It means that the resources (i.e., VMs) cannot consider other tasks until the completion of the current tasks is in progress.

V. THE PROPOSED ALGORITHM

The principles of the proposed *BFPSO* and *PSOTS* algorithms for task scheduling problem are based on *PSO*. First, the basics of the standard *PSO* algorithm will be introduced.

A The Standard PSO Algorithm

The basic idea of the *PSO* algorithm is that if one of the birds finds a good path to the prediction location, more followers would be attracted. This helps to improve the search for best path because all populations are involved. According to the PSO, every particle in the swarm behavior has two characters; (1) A position x which notates the suggested location. (2) A velocity v which defines the amount of moving. Each particle travels over the whole search space and remembers the best position found. The communication is made between particles such that each bird could determine the location and velocities based on the best solutions discovered by others. Each position and velocities are scored by a fitness function f to quantify the best solution. Specifically, at iteration k, particles keep two values for each track; local best position (P_LB) and global best position (P_GB), both of which are calculated from f. Therefore, the update function is defined using Equations(4) and (5)[5].

$$V(t+1) = wV(t) + c_1 * R1 * (pbest(t) - p(t)) + c_2 * R2 * (gbest(t) - p(t)) \qquad (4)$$

$$p(t+1) = p(t) + V(t+1) \qquad (5)$$

Where V is the particle velocity, the variable W is called the inertia weight factor, p is the current solution, ***p best*** is a local solution, ***g best*** is a global solution, and $R1$,$R2$ are the random numbers uniformly distributed on the interval [0,1].The variables $C1$, $C2$ are the learning factors.

The pseudo code of the standard *PSO* algorithm is as follows [7].

PSO Algorithm

Set of particle dimension as equal to the size of ready tasks in **{ ti } ∈ T**

1. *Initialize particles position randomly from* ***PC= 1, ..., j*** *and velocity* ***vi*** *randomly.*
2. *For each particle, calculate its fitness value as in Equations (3), and (4)*
3. *If the fitness value is better than the prior* ***p_best****, set, the current fitness value as the new* ***p****_best.*
4. *After Steps 3 and 4 for all particles, select the best particle as* ***g_best****.*
5. *For all particles, calculate the velocity using equation (3) and update their positions using equation (4).*
6. *If the stopping criteria or maximum iteration is not satisfied, repeat starting from step 3.*

The flowchart of the standard *PSO* is presented in Figure 1.

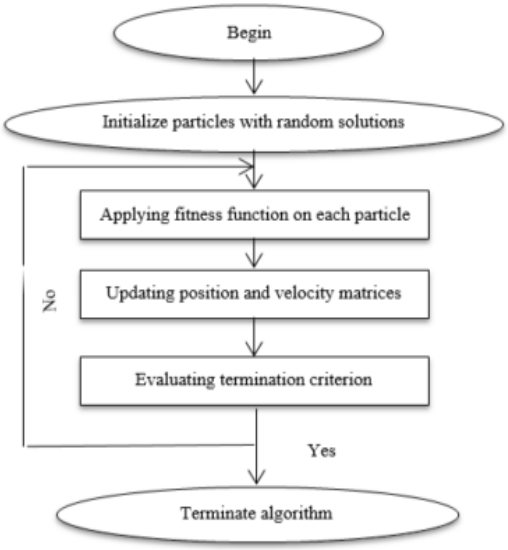

Fig. 1. Flowchart of Standard PSO Algorithm [13]

B *Best-Fit -PSO Algorithm*

In the standard *PSO* algorithm, initial particles are created randomly. Unfortunately, randomness could decrease the chance of the algorithm to converge to the best solution. In order to improve the behavior of the standard *PSO* algorithm, the Best-Fit (***BF***) algorithm has been merged into *PSO* algorithm, that is, instead of generating initial population randomly; it is created according to the *BF* algorithm. All other steps of the standard *PSO* algorithm are not changed. The flowchart of the modified *BFPSO* algorithm is shown in as Figure 2. For this purpose, the initial population is selected after sorting the tasks by priority, and allocating the ***m*** on to the suitable resources (i.e., VMs) with minimum running time according to *BF* algorithm[14].

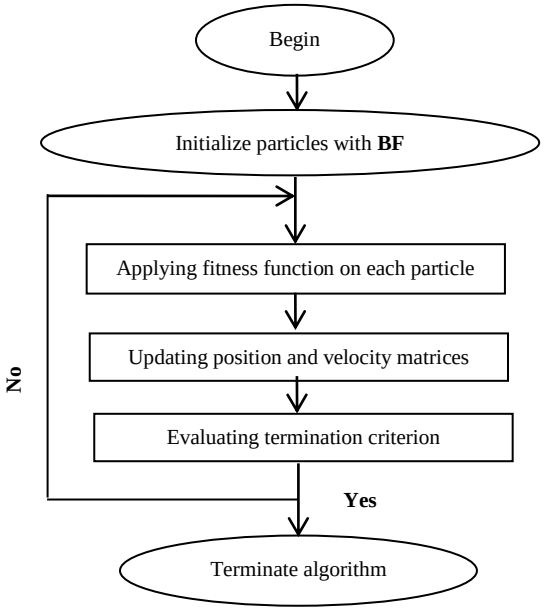

Fig. 2. Flowchart of BFPSO algorithm

C *Tabu Search Based on PSO Algorithm*

TS are a meta-heuristic procedure for solving problems. It is designed to guide other algorithms to escape the trap of local optimally. It has been applied to solve task scheduling and other optimization problems[15].

TS is a neighborhood search algorithm which employs intelligent search and flexible memory technique to avoid being trapped at the local optimum, and speed up the search process. The *TS* algorithm has been merged to the standard *PSO* algorithm to improve the local search where *PSO* algorithm searches and sends the results to the *TS* algorithm. The *TS* applies the neighborhood technology, adds the result to the Tabu list, compares the result and selects the best solutions. This situation is repeated until the best solution is obtained.

According to the example in Table [1], the neighborhood technology assigns with two minimum local execution time for any particle (i.e., VM3, VM4), and moves the tasks among VM1, VM3, and VM4, and uses Tabu list to restrict the search space to selects better solution [14],[16].

TABLE I. SHOW THE DISTRIBUTE THE TASK'S ON VM'S WITH TABU SELECT

Particle 0	T0	T1	T2	T3	T4	T5	T6	T7	T8	T9	Time
Instance 0	VM0		VM0	–	–	VM0	–	–	VM0	VM0	93.54
Instance 1	–	VM1	–	VM1	–	–	–	–	–	–	15.84
Instance 2	–	–	–	–	–	–	VM2	–	–	–	18.25
Instance 3	–	–	–	–	–	–	–	VM3	–	–	7.92
Instance 4	–	–	–	–	VM4	–	–	–	–	–	13.32

The flowchart of the modified PSO algorithm using TS algorithm is shown in Fig. 3.

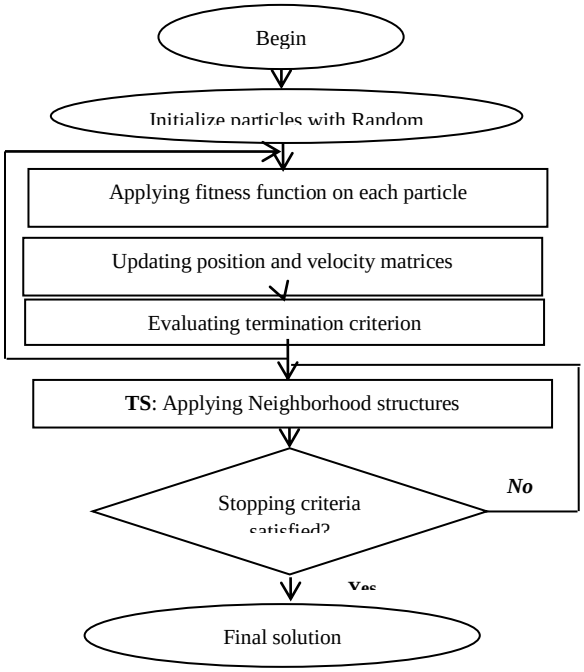

Fig. 3. Flowchart of PSOTS algorithm

VI. THE PERFORMANCE EVALUATION

To evaluate the proposed hybrid algorithms, a comparative study has been conducted among the proposed *BFPSO, PSOTS* task scheduling algorithms and the standard *PSO* algorithm with considering the performance parameters of the standard *PSO* which are Makespan, cost of execution, as well as resource utilization.

A Experimental Settings

Cloudsim3.0.3 is an open source simulator which has been developed by Gridbus project team and the grid Laboratory of the University of Melbourne in Australia. The Cloudsim can run on Linux and Windows systems [6]. Cloudsim has been used to implement and compare the proposed *BFPSO, PSOTS* task scheduling algorithms with respect to the standard *PSO* algorithm. This simulation mainly validates the advantage of the Makespan, cost and the resource utilization among these scheduling algorithms in the Cloud computing environment.

B Performance Evaluation

To evaluate the performance of the three algorithms; the standard *PSO, BFPSO* and *PSOTS* algorithms, 15 Virtual machines are considered with 25, 50, 75,100, and 125 cloudlets (i.e., tasks).

It should be noted that the generation of the tasks using Cloudsim has been done randomly within a specific range of the tasks' length. This range has been defined to be 10000 to 75000 in our implementation.

The simulation results of the Makespan, resources utilization, and resources cost of the three algorithms, *PSO, BFPSO, and PSOTS*, using 15 Virtual machines and 25,50,75,100,125 tasks, are described in Tables[2], [3], [4],and Figs(4),(5),(6) and (7) respectively.

TABLE II. MAKESPAN, COST, AND RESOURCES UTILIZATION OF STANDARD PSO TASK SCHEDULING ALGORITHM

No Of VM	*No of Task*	*Makespan*	*Utilization*	*Cost*
15	25	4.092052352	0.590684092	2447.8
	50	8.049770726	0.460671215	3459.6
	75	10.06010162	0.600100115	6348.2
	100	15.05739967	0.481335388	7656.6
	125	18.32775419	0.419473994	9158.6

TABLE III. MAKESPAN, COST, AND RESOURCES UTILIZATION OF BFPSO TASK SCHEDULING ALGORITHM

No Of VM	*No of Task*	*Makespan*	*Utilization*	*Cost*
15	25	3.857488357	0.661309163	2170.8
	50	7.02620354	0.501536643	3430.6
	75	9.220183209	0.677329305	6110.2
	100	14.49099158	0.518613507	7463.4
	125	18.03631718	0.442760167	8972.4

TABLE IV. MAKESPAN, COST, AND RESOURCES UTILIZATION OF PSOTS TASK SCHEDULING ALGORITHM

No Of VM	*No of Task*	*Makspan*	*Utilization*	*Cost*
15	25	3.609294627	0.70798141	2047.8
	50	6.928386256	0.57958908	3259.6
	75	9.194306359	0.70554259	5648.2
	100	14.91445344	0.55196318	7256.6
	125	17.60515986	0.48542686	8458.6

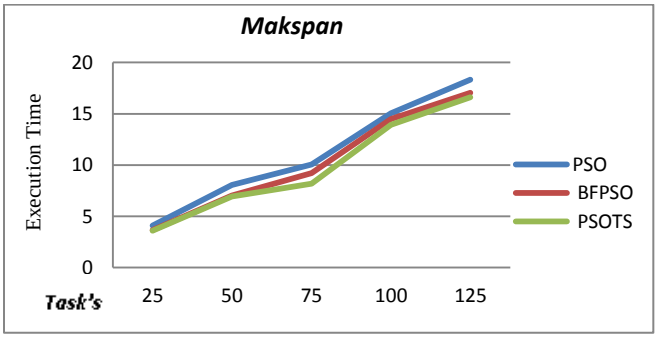

Fig. 4. The comparison of Makspan using three algorithms

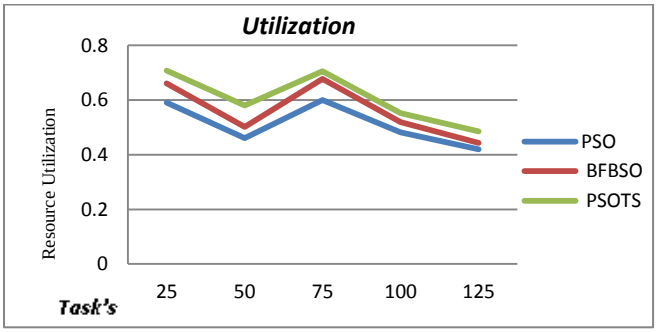

Fig. 5. The comparison of resource utilization using three algorithms

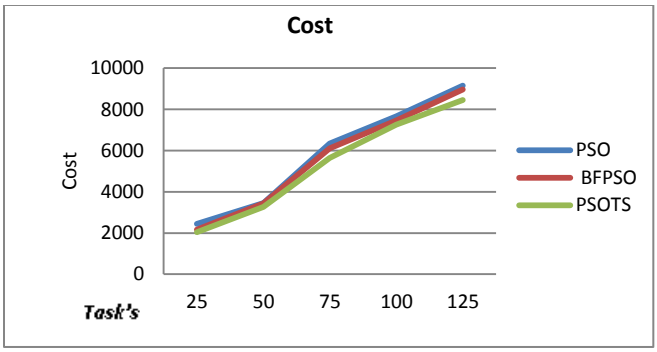

Fig. 6. The comparison of cost using three algorithms

According to the implementation results, it is found that the proposed *BFPSO* and *PSOTS* algorithms outperform the standard PSO with respect to Makespan by 5.32% and 7.85%

in average respectively. With respect to resource utilization, the two algorithms outperform the standard PSO by 8.54% and 14.68% on an average, respectively. With respect to cost, the proposed algorithms outperform the standard PSO by 3.18% and 16.34%, respectively in average (see Table [5]).

It notes that there is a variation when number of task 50 is increased to 75 tasks. This may happen because of the generated tasks with different length, and because VMs are heterogeneous.

TABLE V. AVERAGE IMPROVEMENT OF THE PROPOSED TWO ALGORITHMS RELATIVE THE STANDARD PSO

IMPROVE		
Parameter	BFPSO Vs PSO	PSOTS Vs PSO
Makspan	5.32%	7.85%
Resource Utilization	8.54%	14.68%
Cost	3.18%	16.34%

VII. CONCLUSION

In this paper, the problem of task scheduling in the Cloud computing environment is concerned. *BF*, *PSO* and *TS* algorithms are most famous algorithms for scheduling tasks in the distributed systems. In order to improve the performance of the standard *PSO* algorithm, the modified *PSO* algorithm is suggested, in which *BF* algorithm is merged into standard *PSO* algorithm for generating initial population in order to obtain a good initial selection. On the other hand, *TS* algorithm has been merged to standard *PSO* algorithm to avoid trapped and make an optimal local search to get a good solution by reducing the execution time (*Makespan*), cost of processing, and resources utilization.

The implementation results show that both proposed algorithms (i.e., *BFPSO* and *PSOTS*) outperform the standard *PSO* with respect to the Makespan, resource utilization and cost. Also, the proposed *PSOTS* algorithm has satisfied better results than that the proposed *BFPSO* algorithm. Unfortunately, this improvement has been satisfied at the expense of the time complexity of *PSOTS* algorithm.

In the future work, we plan to improve *PSO* by considering other greedy algorithms.

REFERENCES

[1] Shawish and M. Salama, "Inter-cooperative Collective Intelligence: Techniques and Applications," a book chapter of Internet of Intelligent Things: Bringing Artificial Intelligence into Things and Communication Networks, pp. 39–68, 2014.

[2] An Oracle White Paper "Oracle Cloud Computing," May, pp. 1–22, 2010. It is avaiable in https://www.mendeley.com/catalog/oracle-cloudcomputing/?utm_source=desktop&utm_medium=1.16.1&utm_campaign=open_catalog&userDocumentId=%7B6f362332-4682-41ce-a166-2c26a3c6decc%7D.

[3] R. Buyya, "Introduction to the IEEE Transactions on Cloud Computing," IEEE Trans. Cloud Comput., vol. 1, no. 1, pp. 3–21, 2013.

[4] Ghorbannia Delavar and Y. Aryan, "HSGA: A hybrid heuristicalgorithm for workflow scheduling in cloud systems," Cluster Comput,(The Journal of Networks, Software Tools and Applications) springer, vol. 17, no. MARCH 2014, pp. 129–137, 2014.

[5] G. Zhao, "Cost-Aware Scheduling Algorithm Based on PSO in Cloud Computing Environment," International Journal of Grid and Distributed Computing, vol. 7, no. 1, pp. 33–42, 2014.

[6] S. Zhan and H. Huo, "Improved PSO-based Task Scheduling Algorithm in Cloud Computing," Journal of Information and Computational Science (JICS)., pp. 3821–3829, 2012.

[7] S. Pandey, L. Wu, S. M. S. M. S. M. Guru, and R. Buyya, "A Particle Swarm Optimization-Based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments," 2010 24th IEEE Int. Conf. Adv. Inf. Netw. Appl., pp. 400–407, 2010.

[8] S. Abdi, S. A. Motamedi, and S. Sharifian, "Task Scheduling using Modified PSO Algorithm in Cloud Computing Environment," International Conference on Machine Learning, Electrical and Mechanical Engineering (ICMLEME'2014) Jan. 8-9, 2014 Dubai (UAE)

[9] K. Gomathi, B Krishnasamy, "TASK SCHEDULING ALGORITHM BASED ON HYBRID PARTICLE SWARM OPTIMIZATION IN CLOUD COMPUTING ENVIRONMENT," Journal of Theoretical and Applied Information Technology ,vol. 55, no. 1, 2013.

[10] Singh, Raja Manish Paul, Sanchita Kumar, Abhishek, "Task Scheduling in Cloud Computing : Review," (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (6) , 2014.

[11] Al-maamari and F. A. Omara, "Task Scheduling using Hybrid Algorithm in Cloud Computing Environments," International Journal of Grid Distribution Computing Vol. 8, No.5, (2015), pp.245-256 .

[12] S. Saha, "A Survey on Resource Management in Cloud Computing," (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (3) , 2014.

[13] R. Sahal and F. A. Omara, "Effective Virtual Machine Configuration for Cloud Environment," The 9th International Conference on INFOrmatics and Systems (INFOS2014) - 15-17 December Parallel and Distributed Computing Track Effective, pp. 15–20, 2014.

[14] Yi, Pan Ding, Hui Ramamurthy, "A Tabu search based heuristic for optimized joint resource allocation and task scheduling in Grid/Clouds," 2013 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Kattankulathur, 2013.

[15] Laguna, Manuel Barnes, J. Wesley Glover, Fred W., "Tabu search methods for a single machine scheduling problem, " Journal of Intelligent Manufacturing,1991.

[16] S. S. Manimegalai, "Task Scheduling Using Two-Phase Variable Neighborhood Search Algorithm on Heterogeneous Computing and Grid Environments," Arabian Journal for Science and Engineering, © King Fahd University of Petroleum andMinerals 2015.