HTSCC: A Hybrid Task Scheduling Algorithm in Cloud Computing Environment

Rasha A. Al-Arasi [1], Anwar Saif [2]
[1] Sana'a University, Department of Computer Science, Sana 'a, Yemen

rasha.ali66@gmail.com

[2] Sana'a University, Department of Information Systems, Sana'a, Yemen

anwarsaif.ye@gmail.com

ABSTRACT

Nowadays, cloud computing makes it possible for users to use the computing resources like application, software, and hardware, etc., on pay as use model via the internet. One of the core and challenging issue in cloud computing is the task scheduling. Task scheduling problem is an NP-hard problem and is responsible for mapping the tasks to resources in a way to spread the load evenly. The appropriate mapping between resources and tasks reduces makespan and maximizes resource utilization. In this paper, we present and implement an independent task scheduling algorithm that assigns the users' tasks to multiple computing resources. The proposed algorithm is a hybrid algorithm for task scheduling in cloud computing based on a genetic algorithm (GA) and particle swarm optimization (PSO). The algorithm is implemented and simulated using CloudSim simulator. The simulation results show that our proposed algorithm outperforms the GA and PSO algorithms by decreasing the makespan and increasing the resource utilization.

Indexing terms/Keywords

Cloud computing, Task scheduling, Genetic algorithm (GA), Particle swarm optimization (PSO), Makespan, Resource utilization.

Subject Classification: Distributed Computing Classification

Language: English

Date of Submission: 09-08-2018

Date of Acceptance: 25-08-2018

Date of Publication: 29-08-2018

DOI: 10.24297/ijct.v17i2.7584

ISSN: 2277-3061

Volume: 17 Issue: 02

Journal: International Journal Of Computers & Technology

Publisher: CIRWORLD

Website: https://cirworld.com

1. INTRODUCTION

Cloud computing is a novel technology that enhances the usage of the virtualized resources on the internet for the end user. The main idea behind cloud computing is providing on-demand various resources and services from service providers with high availability and scalability in a distributed system [1]. Cloud computing consists of a collection of a huge number of computing resources such as virtual machines, network bandwidth, processing, and storage [2]. Provisioning of these resources on demand is one of the major objectives of the cloud computing task scheduling. Task scheduling problem(TSP) is the most critical challenge in the cloud environment [3]. TSP is a nondeterministic polynomial time(NP)-hard problem and is responsible for allocating tasks of an application to computation resources efficiently [4]. Thus, many techniques have proposed to solve the NP-hard problem, but there is no specific technique provides a solution with polynomial time for this problem. However, the meta-heuristic techniques have attracted the attention to get optimal solutions for this problem. Meta-heuristic is a population-based technique. Algorithms such as ant colony optimization(ACO), genetic algorithm (GA), particle swarm optimization(PSO), and imperialism competitive algorithm (ICA) are examples of algorithms that use this technique [5]. Genetic algorithm and particle swarm optimization are the most popular meta-heuristic techniques to solve the TSP. PSO is a population-based stochastic optimization technique that gains its popularity due to the ease of implementation, quick convergence for large problems, and few parameters to adjust. However, PSO algorithm might get a non-optimal solution because there is a possibility of becoming trapped in a local search in the last iteration. The drawback of the GA- based algorithms is the slowness to reach the optimal solution for the large size problems. Hence, the proposed algorithm inherits the advantages of PSO and GA algorithms to address the TSP. The proposed HTSCC algorithm improves the local search by using the GA mutation operator and expected to work with the different size of tasks. These features of the proposed algorithm reduce the makespan and increase the resources utilization. The rest of this paper is organized as follows: Section 2 presents related work. In Section 3, the proposed method in details is described. Section 4 discusses the experimental results and evaluates the proposed work. Then, the conclusion and future work are summarized in section 5.

2. RELATED WORK

Tasks scheduling is a hot and major research area in the distributed environment like cloud computing. It is a challenging issue in which a lot of research works have been carried out. Many meta-heuristic techniques were proposed to solve the TSP using various strategies.

A. Al-maamari and F. A. Omara [3], proposed an algorithm that combines PSO with other local search strategy called Cuckoo search. The algorithm performs well in reducing the increasing the resource utilization and completion time.

A. Chhabra [6] proposed a hybrid PSACGA algorithm by amalgamate the features of ACO, PSO and operators of GA to overcome the limitation of each individual algorithm. The proposed algorithm shows a good performance in optimizing the flow time and makespan of parallel job scheduling problem.

Q. Meng, L. Zhang, and Y. Fan [7] have proposed a method to overcome trapping in a local search by combining the PSO algorithm with GA and simulated annealing(SA).

A. S. Kumar and M. Venkatesan [8] presented a hybrid algorithm called HGPSO for task scheduling based on the priority of the task. The proposed algorithm takes input from an on-demand queue and evaluates the appropriate resources for the user tasks.

A cuckoo search-based task scheduling method presented by M. Agarwal and G. M. S. Srivastava [2] in the cloud computing for the optimization of makespan.

A. Al-maamari and F. A. Omara [9] have applied task scheduling method based-PSO for cloud computing to reduce the makespan and maximize resource utilization. The proposed algorithm is combining between cuckoo search and dynamic PSO. thus the algorithm is called MDAPSO.

M. Shojafar et al. [10] presented a FUGE algorithm to minimize cost and makespan in tasks scheduling for the cloud. Based on different QoS parameters various types of chromosomes were created. The FUGE algorithm used fuzzy theory to calculate crossover operation and fitness value of these chromosomes.

K. Zhu, H. Song, L. Liu, J. Gao, and G. Cheng [11] proposed a genetic algorithm-based approach called multi-agent genetic algorithm (MAGA) for balancing the load between virtual machines. MAGA is an amalgamation of multi-agent techniques and GA that improves the optimization and reduces convergence time as compared to basic GA.

O. Udomkasemsub et al. [12] presented a workflow scheduling using the random-draft PSO(RDPSO) algorithm for workflow scheduling in the cloud environment to minimize makespan and cost.

C. T. Joseph, K. Chandrasekaran, and R. Cyriac [13] proposed a novel approach for mapping virtual machines(VMs) to suitable physical machines to maximize resource utilization using a family genetic algorithm (FGA). R. Aron, I. Chana, and A. Abraham [14] introduced a novel PSO strategy in grid environment for secure scheduling of jobs on appropriate resources.

For the aim of reducing makespan and increasing resource utilization, F. Ebadifard and S. M. Babamir [15] proposed a static task scheduling technique for independent tasks on cloud computing based on PSO algorithm.

N. Dordaie and N. J. Navimipour [16] presented a hybrid Particle swarm optimization and hill climbing algorithm for task scheduling in cloud computing. This algorithm reduces the makespan compared to the PSO and HEFT-B algorithms.

S. A. Hamad and F. A. Omara [17] modified the GA task scheduling to introduce a task scheduling algorithm that minimizes the cost of tasks and completion time and maximizes resource utilization.

M. Jaeyalakshmi and P. Kumar [18] have introduced an algorithm to solve task scheduling and resource allocation problem in cloud computing based on bat algorithm. Bat algorithm optimized the performance and efficiency of the system by tunning multibed objectives such as makespan, load balancing, deadline, and execution cost.

R. Babukartik and P. Dhavachelvan [19], proposed a hybrid algorithm which combines the advantages of ACO and Cuckoo search to minimize the makespan. The proposed algorithm can be used in high power and scientific computing.

R. Muthuram and G. Kousalya [20], introduced efficient genetic algorithms based scheduling algorithm to optimized the makespan and execution cost. This algorithm begins with an initial population and implements the different genetic operators such as selection, crossover, and mutation.

Z. Tarek, M. Zakria, and F. A. Omara [21] proposed a modified particle swarm optimization (MPOS) for solving the TSP with an objective of reducing the cost.

3. PROPOSED METHOD

In this section, we have introduced the GA and PSO algorithms and then presented HTSCC algorithm in detail.

3.1 . Particle swarm optimization (PSO)

Particle swarm optimization (PSO) is a population-based stochastic optimization which was introduced by James Kennedy and Russell Eberhart in 1995 [22]. The development of this algorithm was inspired from social behavior of particles like bird flocking and fish schooling. Nowadays the PSO algorithm is used in different domains such as scheduling, communication networks, neural networks, power systems, security and military, fuzzy system control and so on [23]. Compared to other meta-heuristic optimization algorithms the advantages of PSO are that it is easy to implement and there are few parameters to adjust. The PSO algorithm preserves a swarm of particles where the particles represent potential solutions and the population of potential solutions represents swarms. Each particle has a position in source multidimensional search space. The position of a particle is determined according to its own personal best experiences of a particle (Pbest) and the common best experience (Gbest) among a number of swarms [24]. In addition to the position, the particles have velocity. In every iteration of PSO position and velocity for every particle is updated according to simple mechanisms. The algorithm is shown in figure1.

3.2 . Genetic algorithm

John Holland discovered genetic algorithms (GA) in the 1960s. GA is an evolutionary method based on Darwin's theory of "SURVIVAL OF THE FITTEST" [25]. In the GA algorithm, the chromosomes are a group of genes and represent a group of candidate solutions to the problem space. Figure 2 depicts the procedure of this algorithm. The algorithm begins by generating an initial population of chromosomes randomly. The chromosomes which are selected depend on evaluating of a fitness function. Then, apply the operators of GA in each generation. The operators are selection, crossover(reproduction), and mutation. The selection process uses Darwin's theory to select an intermediate population (mating pool) according to their fitness for the next generation. There are different selection methods to select the best chromosomes(parents) such as tournament selection (TOS), roulette wheel selection (RWS), linear rank selection (LRS), selection of Boltzmann, truncation selection (TRS) and others [26]. Various crossover operators such as one-point, two-point, and uniform crossover can be applied over parents to generate a new offspring. As well, various mutation operators are also used to mutate new offspring such as Move, Swap, Move, Swap, etc.

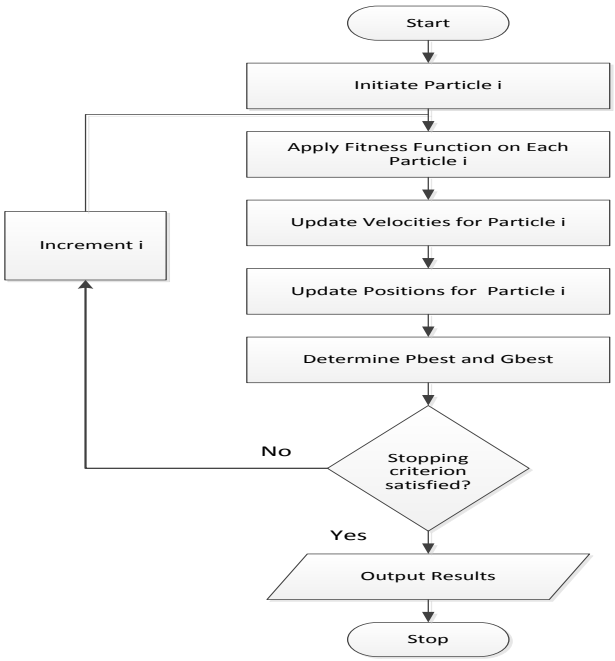

Fig 1: The flow chart of the PSO algorithm

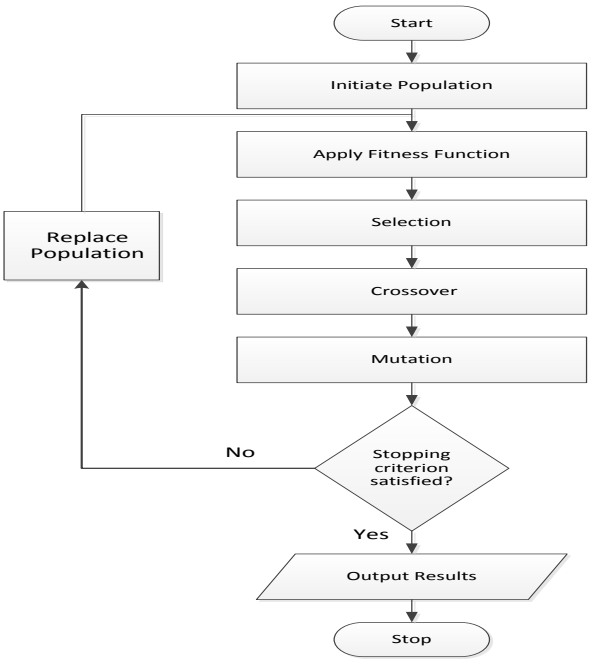

Fig 2: The flow chart of the GA algorithm

3.3 . The Proposed hybrid HTSCC algorithm

The HTSCC algorithm is an optimization algorithm that consolidates the features of GA and PSO algorithms to overcome the drawbacks of these two algorithms. The flowchart of the HTSCC algorithm shown in figure 3.

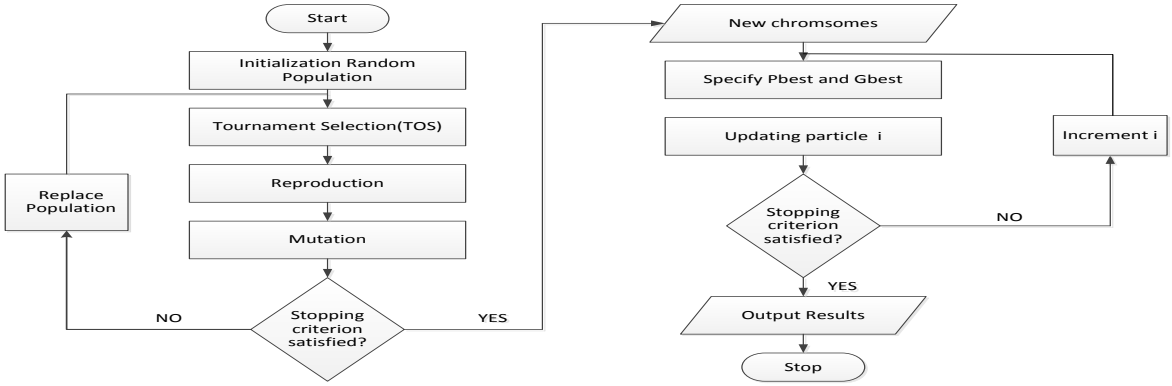

Fig 3: The flow chart of the HTSCC algorithm

The HTSCC algorithm is mainly divided into two phases: in the first phase, initializes the population randomly and applies GA operators tournament selection(TOS), reproduction (crossover), and mutation sequentially. The second phase applies the PSO method over GA to get optimal solutions. The detailed steps are shown in figure 4.

Step 1: The Initialization of a random population and determining the parameters of the proposed algorithm such as a number of iterations and populations. These populations are called chromosomes which represent the random solutions in the first iteration.

Step 2: Apply GA method.

Step 2.1: Tournament selection is used to select an intermediate solution(chromosomes) for crossover operator depending on relative fitness. The TOS aim is to overcome the limitations of the population size. In this step, not all the chromosomes are selected. Only, the best two chromosomes are selected at random from the population. The random number is then selected between 0 and 1. The non-selected chromosomes could be selected again when they returned to the population.

Step 2.2: Single point crossover operator is applied to the selected chromosomes with 0.9 probability crossover (PC) to generate two offspring. The reproduction point is selected randomly from the rang of chromosome length.

```
//phase 1:GA method
Step 1: Initialize population randomly
Step 2:  GA method
            //Apply GA operators
        2.1: Tournament selection (TOS)
            (depending on relative fitness )
        2.2: Reproduction
            (Apply PC, carry out crossover )
        2.3: Mutation
            (Using PM)
Step 3: Repeat until the termination criterion reached
//phase 2: PSO
Step 4:   PSO Method
            // Apply position and velocity operators of the PSO
        4.1: Calculate Pbest and Gbest
        4.2: Update velocity
        4.3: Update position
Step 5: Repeat until the termination criterion reached and Get best solution
```

Fig 4: Proposed algorithm

Step 2.3: Mutation operator takes place after crossover to make modification in the chromosomes which yield from reproduction. In the HTSCC algorithm, the probability of mutation(PM) is (1/ (number of tasks)). The mutation aims to produce chromosomes with the best fitness value than existent chromosomes. Also, improving the local search ability. The chromosomes are enhanced progressively in each generation by applying the GA operators.

Step 3: Steps 2.1 to 2.3 will be repeated until the termination criterion reached.

Step 4: Apply the PSO method.

The resulting chromosomes from the first phase are fed to the second phase as particles. The particles represent several solutions in PSO. In this phase, the operators of PSO such as determine Pbesti and Gbest, a velocity of particles, and the position of particles are applied in each iteration.

Step 4.1: In each iteration, Pbesti and Gbest are calculated. The Pbesti is the best position of particle and Gbest is the best position of entire particles in the population. In the beginning, the fitness value is evaluated against each particle Xi. If particles' fitness is better than Pbesti, Pbesti is replaced by Xi. Likewise, if Pbesti better than Gbest, Gbest is replaced by Pbesti according to the following:

If Xi < Pbesti

{Pbesti = Xi}

If Pbesti < Gbest

{Gbest = Pbesti}

Step 4.2: After determining the value of Pbesti and Gbest, velocity and position of particles are updating accordingly. The velocity of each particle is updated by equation 1.

Step 4.3: Equation 2 shows how each particle's position (X_i^{k+1}) is updating during the search in the solution space.

$$V_i^{k+1} = W V_i^k + a_1 rand_1 * (Pbest_i - X_i^k) + a_2 rand_2 * (Gbest - X_i^k), [27] \quad (1)$$

$$X_i^{k+1} = X_i^k + V_i^{k+1}, [27] \quad (2)$$

Where

V_i^{k+1}	velocity of particle i at iteration k+1
W	inertia weight
V_i^k	velocity of particle i at iteration k
a1,a2	acceleration coefficients
rand1,rand2	random number between 0 and 1
$Pbest_i$	best position of particle i
X_i^k	position of particle i at iteration k
Gbest	best position of entire particles in a population
X_i^{k+1}	position of particle i at iteration k+1

$a_1 rand_1 * (Pbest_i - X_i^k)$ cognitive component

$a_2 rand_2 * (Gbest - X_i^k)$ social component

Step 5: Steps 4.1 to 4.3 will be repeated until the termination criterion reached. The minimum returned Gbest presents the best solution of TSP.

4. PERFORMANCE EVALUATION

In this section, the proposed HTSCC algorithm is compared with original GA and PSO algorithms in terms of resource utilization and makespan.

4.1 . Environment setting

In our experiments, we have used CloudSim to implement the proposed HTSCC algorithm and other algorithms. The CloudSim is a toolkit simulation for modeling the cloud computing environment [28]. It is an easy to use simulator and flexible in defining configurations parameters. We have used some parameters in our proposed algorithm which have a considerable impact on the performance of the algorithm. The parameters used by HTSCC algorithm along with their values are given in Table 1.

Table 1. Simulation parameters setting

Parameter	Value
Number of VMs	5 -10
Number of tasks	10 - 40
RAM(MB)	128 - 4096
BW(mbps)	700 - 1500
MIPS	200 - 600
No. of processor	5

Two scenarios were used to test the performance of the proposed algorithm, the first scenario uses 5 virtual machines while the second one uses 10 virtual machines. The number of tasks in both scenarios is between 10 and 40.

In this table, the parameters were used to specify the specifications of VMs. We were used a different number of VMs, 5VMs as scenario one and 10VMs as scenario two with a different number of tasks between (10 to 40) in the two scenarios. The performance metrics experienced in these experiments are makespan and resource utilization. The setting parameters for the proposed HTSCC algorithm are shown in Table 2.

Table 2. HTSCC algorithm parameters

Parameter	Value
Maximum iteration	80
Population size(no. of solution)	80
Execute times	10
Crossover operator	Single point
Crossover probability	0.9
Mutation operator	Random
Mutation probability	1/(number of tasks)
a1=a2	1.49445
r1,r2	Random numbers between 0 and
W	0.9 to 0.4

The HTSCC start with 80

random solutions and the termination condition is set to 80. The simulation experiment is executed 10 times and the average of the results of the makespan and resource utilization were compared with GA and PSO algorithms.

4.2. Experimental result

In order to evaluate the performance of the proposed HTSCC algorithm in reducing the makespan and increasing resource utilization, we have two different scenarios. In the first scenario, we used 5 VMs and change the number of tasks from 10 to 40. The second scenario, 10 VMs and change the number of tasks from 10 to 40. The final results of executed the three algorithms in terms of makespan and resource utilization which depicted in Table 3.

The results in table 3 show that the proposed HTSCC achieves better results than GA and PSO. It has the minimum makespan and the maximum resource utilization among the three algorithms. Each experiment is repeated 10 times, and the average results were calculated and demonstrated in Table 4.

Table 3. The simulation results of the two sceinarios

	Scheduling algorithm	Number of VMs	Number of tasks	Makespan(sec)	Resource utilization(rate)		Number or VMs	Number of tasks	Makespan(sec)	Resource utilization(rate)
First Scenario:	HTSCC	5	10	48.5	0.491053	Second Scenario	10	10	32.9	0.346789
	GA			74.1	0.395652				52.2	0.211538
	PSO			64.1	0.387838				40.7	0.245000
	HTSCC		20	86.5	0.555801			20	58.4	0.350577
	GA			142.3	0.350000				79.6	0.321221
	PSO			136.2	0.423944				68.2	0.268354
	HTSCC		30	123.3	0.518440			30	73.8	0.514617
	GA			226.2	0.382123				93.3	0.366667
	PSO			179.1	0.403530				87.4	0.367816
	HTSCC		40	155.2	0.513242			40	96.7	0.471577
	GA			210.1	0.418095				105.4	0.445121
	PSO			192.2	0.472251				101.9	0.455876

Table 4. Average result of 10 experiments.

Scheduling algorithm	Makespan(sec)	Resource utilization(rate)
HTSCC	84.4125	0.470262
GA	122.9	0.361302
PSO	108.725	0.37808

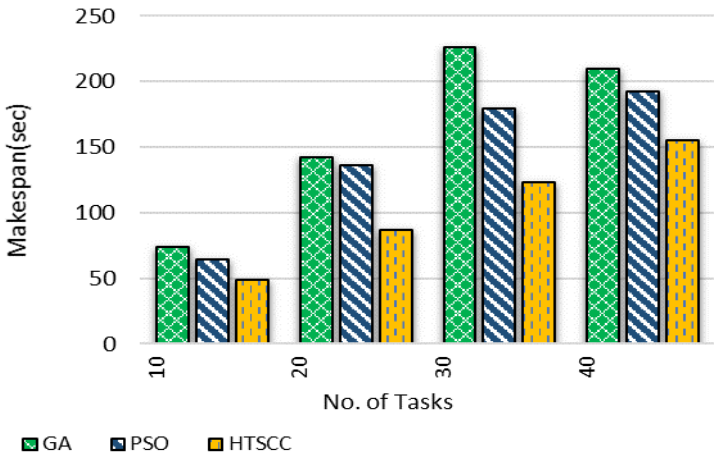

Fig 5: The makespan of GA, PSO and HTSCC algorithms with 5VMs

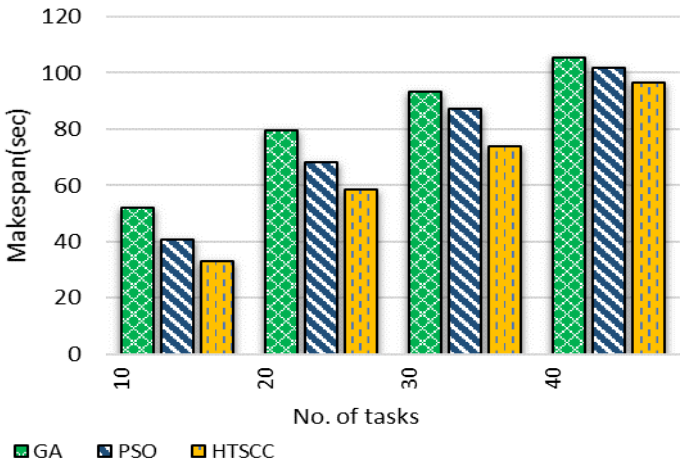

Fig 6: The makespan of GA, PSO and HTSCC algorithms with 10VMs

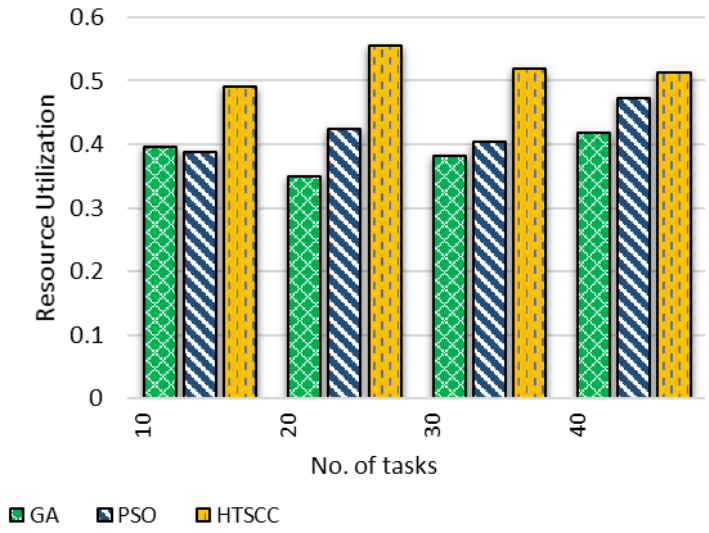

Fig 7: The resource utilization of GA, PSO and HTSCC algorithms with 5VMs

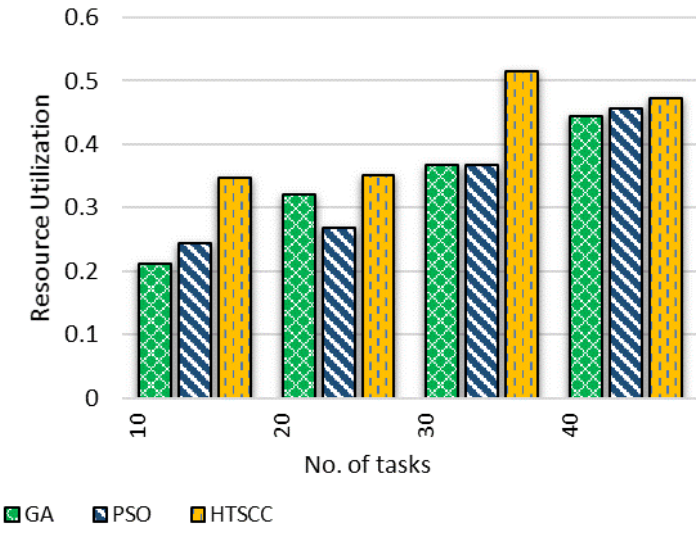

Fig 8: The resource utilization of GA, PSO and HTSCC algorithms with 10VMs

Figures 5 and 6 show, the HTSCC algorithm achieved considerable enhancement of 31.32% and 22.36% better than GA and PSO algorithms respectively. Also, in term of resource utilization, figures 7 and 8, they show an improvement up to 23.17% and 19.6% of HTSCC over the GA and PSO respectively.

These results show the significance of the proposed algorithm in obtaining the optimal solution faster and with fewer resources than the other algorithms by combining the fast convergence and appropriate diversity.

5. CONCLUSION AND FUTURE WORK

The work in this paper presents a hybrid task scheduling algorithm to ensure distribution of the tasks through VMs in an efficient way to increase resource utilization and decrease makespan of the application in the cloud computing environment. The proposed HTSCC algorithm makes use of the advantages of the GA and PSO algorithms in order to maximize resource utilization and minimize makespan. Simulation results of the HTSCC show that the makespan can be enhanced by about 31.32% and 22.36% while resource utilization is enhanced by about 23.17% and 19.6% compared to the GA and PSO respectively. In the future work, the proposed algorithm will be tested over dynamic workflow applications where the user can change the parameters of the workflow task during the runtime. Moreover, the proposed algorithm will be enhanced to address multi-objective metrics such as cost, speed up and efficiency.

REFERENCES

1. Senyo, P.K., E. Addae, and R. Boateng, Cloud computing research: A review of research themes, frameworks, methods and future research directions. International Journal of Information Management, 2018. 38(1): p. 128-139.
2. Agarwal, M. and G.M.S. Srivastava, A Cuckoo Search Algorithm-Based Task Scheduling in Cloud Computing, in Advances in Computer and Computational Sciences. 2018, Springer. p. 293-299.
3. Al-maamari, A. and F.A. Omara, Task scheduling using hybrid algorithm in cloud computing environments. Journal of Computer Engineering (IOSR-JCE), 2015. 17(3): p. 96-106.
4. Singh, P., M. Dutta, and N. Aggarwal, A review of task scheduling based on meta-heuristics approach in cloud computing. Knowledge and Information Systems, 2017: p. 1-51.
5. Yang, X.-S., et al., Swarm intelligence and bio-inspired computation: theory and applications. 2013: Newnes.
6. Chhabra, A., Hybrid PSACGA Algorithm for Job Scheduling to Minimize Makespan in Heterogeneous Grids, in Industry Interactive Innovations in Science, Engineering and Technology. 2018, Springer. p. 107-120.
7. Meng, Q., L. Zhang, and Y. Fan, A Hybrid Particle Swarm Optimization Algorithm for Solving Job Shop Scheduling Problems, in Theory, Methodology, Tools and Applications for Modeling and Simulation of Complex Systems. 2016, Springer. p. 71-78.
8. Kumar, A.S. and M. Venkatesan, Task scheduling in a cloud computing environment using HGPSO algorithm. Cluster Computing, 2018: p. 1-7.
9. Al-maamari, A. and F.A. Omara, Task scheduling using PSO algorithm in cloud computing environments. International Journal of Grid and Distributed Computing, 2015. 8(5): p. 245-256.
10. Shojafar, M., et al., FUGE: A joint meta-heuristic approach to cloud job scheduling algorithm using fuzzy theory and a genetic method. Cluster Computing, 2015. 18(2): p. 829-844.
11. Zhu, K., et al. Hybrid genetic algorithm for cloud computing applications. in Services Computing Conference (APSCC), 2011 IEEE Asia-Pacific. 2011. IEEE.
12. Udomkasemsub, O., L. Xiaorong, and T. Achalakul. A multiple-objective workflow scheduling framework for cloud data analytics. in Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on. 2012. IEEE.
13. Joseph, C.T., K. Chandrasekaran, and R. Cyriac, A novel family genetic approach for virtual machine allocation. Procedia Computer Science, 2015. 46: p. 558-565.
14. Aron, R., I. Chana, and A. Abraham, A hyper-heuristic approach for resource provisioning-based scheduling in grid environment. The Journal of Supercomputing, 2015. 71(4): p. 1427-1450.

15. Ebadifard, F. and S.M. Babamir, A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment. Concurrency and Computation: Practice and Experience, 2018. 30(12): p. e4368.
16. Dordaie, N. and N.J. Navimipour, A hybrid particle swarm optimization and hill climbing algorithm for task scheduling in the cloud environments. ICT Express, 2017.
17. Hamad, S.A. and F.A. Omara, Genetic-based task scheduling algorithm in cloud computing environment. International Journal of Advanced computer Science and Applications, 2016. 7(4): p. 550-556.
18. Jaeyalakshmi, M. and P. Kumar, Task Scheduling Using Meta-Heuristic Optimization Techniques in Cloud Environment. International Journal Of Engineering And Computer Science, 2016. 5(11).
19. Babukartik, R. and P. Dhavachelvan, Hybrid Algorithm using the advantage of ACO and Cuckoo Search for Job Scheduling. International Journal of Information Technology Convergence and Services, 2012. 2(4): p. 25.
20. Muthuram, R. and G. Kousalya, GAF–Genetic Algorithm based Framework for Cloud Resource Scheduling.
21. Tarek, Z., M. Zakria, and F.A. Omara, Pso optimization algorithm for task scheduling on the cloud computing environment. International Journal of Computers and Technology, 2014. 13(9).
22. Kennedy, J. and R. Eberhart, Particle swarm optimization 1995 IEEE International Conference on Neural Networks Proceedings. 1942, Vols.
23. Poli, R., An analysis of publications on particle swarm optimization applications. Essex, UK: Department of Computer Science, University of Essex, 2007.
24. Imran, M., R. Hashim, and N.E.A. Khalid, An overview of particle swarm optimization variants. Procedia Engineering, 2013. 53: p. 491-496.
25. Kao, Y.-T. and E. Zahara, A hybrid genetic algorithm and particle swarm optimization for multimodal functions. Applied Soft Computing, 2008. 8(2): p. 849-857.
26. Saini, N., Review of Selection Methods in Genetic Algorithms. International Journal Of Engineering And Computer Science, 2017. 6(12): p. 22261-22263.
27. Bansal, J.C., Particle Swarm Optimization, in Evolutionary and Swarm Intelligence Algorithms. 2019, Springer. p. 11-23.
28. Calheiros, R.N., et al., CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and experience, 2011. 41(1): p. 23-50.

Author' biography with Photo

Rasha Al-Arasi received the B.S. in Information system and primary M.S. degrees in computer science from the faculty of computer and information technology in 2010 and 2016, respectively. Currently, She works as a tutor in the field of computer and information technology at Sana'a University. Recently, She has published many journals and conference articles locally and overseas.

Anwar Saif: received his Ph.D. degree from University Putra Malaysia (UPM) in 2012. Currently, he is an assistant professor at the faculty of Computer Science and Information Technology, Sana'a University, Yemen. His research interests in wireless communication, VoIP, Distributed systems and cloud computing.