Analysis of Swarm Intelligent Algorithms in Homogeneous Cloud Environment for Task Scheduling

[1]Jai Bhagwan
[1]*Department of Computer Science & Engineering*
Guru Jambheshwar University of Science & Technology
Hisar, India
[1]jaitweet@gmail.com

[2]Sanjeev Kumar
[2]*Department of Computer Science & Engineering*
Guru Jambheshwar University of Science & Technology
Hisar, India
[2]sanjukhambra@yahoo.co.in

***Abstract*— Swarm intelligent algorithms can solve many types of real-life problems. These algorithms are developed by the inspiration of social activities of insects and other animals. In the current time, swam intelligent algorithms are playing a significant role in cloud computing for workflow as well as independent task scheduling problems. Scheduling can be divided into two major classes such as static and dynamic. In dynamic scheduling, the quantity of data is not known before the scheduling. Cloud computing is a method of sharing the pool of assets like memory, hardware storage, network, etc. which is known as virtualization. These services are offered based on pay-per-model. In this paper, PSO, ACO, and Cat Swarm Optimization-based algorithms are studied and compared for the independent task scheduling. In few years, these algorithms have been castoff to crack the task scheduling in the zone of cloud technology and found effective. The relative results show that the Cat Swarm Optimization is effective enough than others.**

Keywords—Ant Colony Optimization (ACO), Cat Swarm Optimization (CSO), Cloud Computing, Meta-Heuristic Techniques, Particle Swarm Optimization (PSO), Virtual Machines (VMs) Quality of Service (QoS)

I. Introduction

Swarm Intelligence is the most powerful approach to solve the NP-Hard problems effectively. These techniques work on the principle of insects or animals' social behaviour [1]. The swarm intelligent techniques have been developed in the last few decades. Many soft computing algorithms have been introduced to crack the issues of cloud computing environments like task scheduling, load balancing, energy efficiency, faster response, and many more. Cloud computing is among the latest technologies and the service providers use it as pay on demand. Cloud technology is a modified version of distributed and parallel computing which provides the solution through virtualization [2]. Three foremost categories of cloud computing are: a) infrastructure as a service (IaaS), platform as a service (PaaS), and Software as a Service (SaaS) [4]. IaaS is used to provide hardware services, PaaS is used to provide application development services and SaaS is used to provide a platform where applications can be used by the end-users. Task scheduling is a major factor to expand the efficacy of a cloud [5]. It can be achieved at the optimum level using meta-heuristic algorithms [6]. A detailed performance analysis of the meta-heuristic algorithms like ACO, PSO, and CSO has been done in this paper.

A. *Ant Colony Optimization*

It is a meta-heuristic technique that was built on the behaviour of the real ants. The undeviating path is found by the ants by releasing the pheromone in order to search their food [3][6].

B. *Particle Swarm Optimization*

PSO is a meta-heuristic procedure that works based on the social behaviour of a flock of birds. In this algorithm, a population member is termed a particle and the entire population is known as a swarm. The population is initialized randomly and each particle searches its food in the entire search space by remembering its best position as well as its neighbours. The positions are updated by using a velocity factor's previous value, global best, and local best particle values [7][9][10].

C. *Cat Swarm Optimization*

Cat Swarm Optimization is another meta-heuristic algorithm that works based on the properties of a real cat animal. This algorithm is having two modes like seeking mode and tracing mode. The seeking mode is also called global searching mode whereas the tracing mode is known as a local searching mode. There is a mixing ratio (MR) value that decides how many cats will go to seeking mode and tracing mode for searching their target. In the seeking mode, the current cat replicates its N number of copies. Where, N is a number that is decided with the Seeking Memory Pool (SMP) factor. Finally, the best cat is identified among various random solutions and replaced with the original current cat. In tracing mode, the best cat is identified using a velocity value and other factors like the current position of a cat and a global best cat. In the end, the best cat is picked out from both modes and stored in the memory for further calculation [16].

The rest of the paper is organized as: in section II, the literature review is described; section III is showing the simulation settings section IV denotes experimental results and discussion, and finally, section V summarized the conclusions.

II. Literature Review

Various papers have been studied and described in this section. In [2], the authors compared the scheduling policy designed by Ant Colony Optimization with Round-Robin and FCFS algorithms. The CloudSim toolkit was used for experiments and the results indicated that the ACO algorithm outperformed Round-Robin and FCFS algorithm in relation to makespan and degree of imbalance. The authors in [4] proposed a Multi-objective Ant Colony Optimization scheme for the placement of virtual machines. The proposed technique was tested with a MOGA and found as competitive. The proposed algorithm was found better than BPA method and Max-Min Ant Scheme. The performance parameters were power consumption, resource wastage, and running time. In [5], a task scheduling policy using Improved Ant Colony Optimization was proposed. At first, the ACO was improved for better convergence. In order to improve the pheromone updating strategy, a coefficient was introduced named reward and punishment. A load-balanced coefficient was also incorporated to balance the load of virtual machines. The simulation studies described that the proposed Ant Colony Optimization algorithm was found efficient in terms of convergence speed, completion time, virtual machines utilization, and load balance. In order to minimize the makespan, a task scheduling scheme was planned based on Ant Colony Optimization in [6]. The experiments were carried out in the CloudSim toolkit by taking 100 – 500 tasks and the proposed policy was found better than the default policy. In [7] paper, an Improved PSO was introduced for mapping a large number of tasks. The suggested technique was found efficient than Honey Bee, Ant Colony Scheme and, RR methods in relation to load balance, imbalance degree, and makespan. The simulation was carried out using the CloudSim tool. In the paper [8], a heuristic initialization-based Particle Swarm Optimization method was introduced. The approaches used for initialization are Largest Job to Efficient Processor and Smallest Finishing Time. The proposed algorithms LJFP-PSO and MCT-PSO were found efficient in respect to makespan, processing time, energy consumption, etc. as compared to PSO, Max-Min, and other comparative algorithms. The results were simulated in MATLAB software. In paper [9], a hybrid task mapping algorithm was projected by the mixture of Particle Swarm Optimization and Hill Climbing method. The proposed method is efficient than the HEFT-B and PSO algorithm in the account of makespan. A hybrid task scheduling method was proposed by using Reformed Particle Swarm Optimization and Fuzzy theory in the paper [10]. The performance metrics were set as makespan, computation time, and imbalance degree. To overcome the problem of local optima of the PSO, the crossover mutation operator was used. To enhance the global searching capacity, modified velocity techniques and roulette wheel selection method. The simulation tool was the CloudSim toolkit. The proposed method named FMPSO was outperformed FUGE, SGA, MGA, SPSO, and MPSO algorithms. In the paper [11], the author introduced an improved PSO for optimizing resource scheduling and improving efficiency. The experiments verified that the Improve PSO worked better in relation to tasks execution time, Resources Utility, and other QoS parameters as compared to RR and PSO algorithm. The paper [12] presented an Enriched Particle Swarm Optimization for refining the task scheduling effectiveness. A ranging and tuning function was introduced to improve efficiency. BAT algorithm was also combined for optimum solutions. The simulation results conclude that the RTPSO-B is efficient than ACO, GA, and PSO in relations of resource utilization, makespan, and cost. In [13], an MFOSF-PSO method was introduced and discussed. The newly designed algorithm was found efficient in rapports of makespan, deadline, cost, and resource operation QoS parameters. The assessment of the proposed technique was done with First Come First Serve, Min-Min, and PSO algorithms. In [14] an algorithm named MaOPSO was offered to resolve the workflow scheduling issues. A total of four improvements were done in order to enhance the efficiency of the algorithm in stabilities of better exploration and exploitation. The experiments clearly show that the recommended technique is good than other existing algorithms. The authors proposed a Binary variety of Particle Swarm Optimization in order to reduce cost, better load balancing, and low time complexity [15]. After the simulation, it was found that the projected framework has done a great job in positions of makespan, load balancing, etc. In [16], a hybrid CSO algorithm was proposed to crack the dynamic task mapping in the cloud system. The Simulated Annealing along with orthogonal Taguchi approaches enhanced the performance of the CSO in relation to QoS parameters like makespan and cost. The introduced procedure was found better than modified GA, modified PSO, and modified ACO. The scientists of [17] introduced an Improved CSO algorithm with the help of the LDIW equation. This improves the local searching of the conventional CSO procedure. The trials were performed with the help of the CloudSim toolkit. The CSO-LDIW method was found healthier than the traditional CSO and PSO-LDIW in the account of makespan. The authors in [18] proposed a Multi-objective CSO procedure to lessen energy consumption and faster convergence. The MOCSO algorithm was found better than MOPSO after experiments. The QoS parameters were CPU idle time, makespan, and cost. In [19], the researchers compared the CSO and PSO for workflow scheduling and the CSO was found better in terms of convergence means the solution was provided by the CSO in less number of iterations with lesser computation cost.

III. SIMULATION SETTINGS

The simulation experiments were executed on a personal computing machine having the hardware and software configuration: CPU - Intel Core i3 5th Gen. 2.0 GHz, RAM - 4 GB, HDD- 1 TB, and OS - Windows 10 64 bit. The simulation parameters are summarised in Table 1 [20].

TABLE I. SIMULATION PARAMETERS

Parameter	**Values**
System Architecture	x86
VMM	Xen
OS	Linux
Number of Cloudlets Cloudlets Length Type	500-1300 Random
Numbers of VMs	3, 5 and 8
CPU (PEs Number)	1
RAM per VM	1024 MB
Bandwidth	1000 bps
Processing Elements per VM	1000 MIPS
Image Size	10000 MB
Policy Type	Time Shared
ACO Properties	
No. of Initial Ants (m)	100
No. of Iterations	300
Q, Alpha, Beta, Gamma, Rho	1, 2, 1, 4, 0.05 Respectively
PSO Properties	
No. of Particles	100

No. of Iterations	300
Local and Global Weights (C1 and C2)	1.5
CSO Properties	
No. of Cats	100
Iterations	300
Weight (C1)	1.5
r1 (Random Variable)	[0,1]
Mixing Ratio [MR]	Random [0, 1] i.e. 0.2 – 0.3

For performance measurement, the following parameters are used in this research paper.

A. *Makespan*

Makespan [20] is the finished time of a group of tasks which is calculated by the following Equation (1).

$$Makespan = Max\ (CT_i,)\ T_i \in Tasks \qquad (1)$$

Where, CT_i is completion time of Task Ti

B. *Cost*

Cost can be calculated [20] by Equations (2), (3), and (4).

$$Total\ Cost = \frac{MF+CF}{2} \qquad (2)$$

Where, MF is movement factor and CF is cost factor.

$$MF = \left[\frac{1}{No.\ of \frac{Hosts}{Datacenters}} \sum_{x=1}^{VMx} \left(\frac{Number\ of\ Migrations}{Used\ VM}\right)\right] \qquad (3)$$

$$CF = \sum_{x=1}^{VMx} \left(\frac{Processing\ Cost \times Memory\ of\ Tasks}{VM \times Datacenter}\right) \qquad (4)$$

C. *Fitness Function*

The fitness function [20] used in this research is defined by the following Equation (5).

$$F_X = \frac{1}{Datacenter \times VMj} \left[\sum_{i=1}^{DCi} \sum_{j=1}^{VMj} \frac{1}{VM} \frac{CPU\ Utilized}{CPUij} + \frac{Memory\ Utilized}{Memoryij} + \frac{Makespan\ Utilized}{Makespanij} + \frac{Bandwidth\ Utilized}{Bandwidthij}\right] \qquad (5)$$

IV. Simulation Results and Discussion

For simulation, the CloudSim toolkit has been used. An environment has been created using 3, 5, and 8 VMs. A set of 500, 800, and 1300 independent tasks have been taken to test the performance of the ACO, PSO, and CSO algorithms.

Table II is representing the makespan results achieved after several times executions of each algorithm.

TABLE II. Makespan Comparison (In Sec.)

Scenarios	VMs	ACO	PSO	CSO
Scenario – 1 500 Tasks	3	500.43	203.33	180.13
	5	323.03	183.29	152.79
	8	303.13	159.07	149.27
Scenario – 2 800 Tasks	3	615.22	380.15	357.84
	5	579.29	278.55	241.13
	8	459.79	235.29	201.29
Scenario – 3 1300 Tasks	3	1265.51	681.99	647.43
	5	1051.39	549.42	500.11
	8	829.13	419.19	389.17

Fig. 1, 2, and 3 are representation the virtual machines and makespans at the x-axis and y-axis respectively.

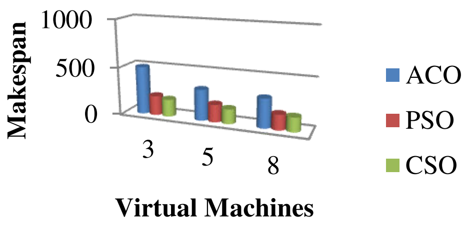

Fig. 1. Makespan Evaluation of 500 tasks.

Fig. 1 is representing that the computation makespan of the CSO algorithm is lesser than all other algorithms.

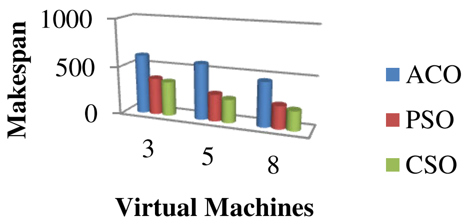

Fig. 2. Makespan Evaluation of 800 tasks.

Fig. 2 is demonstrating the results of the makespan and it can be seen that for each set of VMs, the CSO outperforms other algorithms.

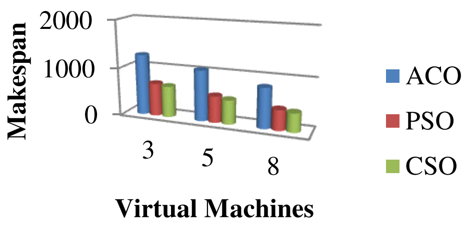

Fig. 3. Makespan Evaluation of 1300 tasks.

Finally, it can be seen that for all scenarios, the CSO algorithm is giving efficient results as compared to the ACO, PSO. The reason behind the better results is that the CSO method is having good convergence and better global searching properties.

Table III is demonstrating the cost comparison of various algorithms used in this research with respect to processing cost.

TABLE III. Cost Comparison (In Indian Rupees)

Scenarios	VMs	ACO	PSO	CSO
Scenario – 1 500 Tasks	3	51.78	39.07	33.29
	5	79.23	53.27	45.57
	8	87.41	64.30	57.08
Scenario – 2 800 Tasks	3	70.29	55.44	47.39
	5	103.13	81.25	70.24

	8	139.27	99.27	83.29
Scenario – 3 1300 Tasks	3	107.13	97.50	87.37
	5	143.29	119.35	100.13
	8	180.23	157.60	145.43

Fig. 4, 5, and 6 are showing the virtual machines at the x-axis and computation cost at the y-axis.

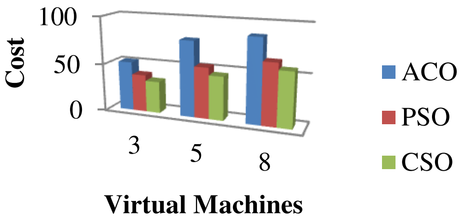

Fig. 4. Cost Evaluation of 500 tasks.

In Fig. 4, it can be seen that the cost consumption of the CSO is lesser than other algorithms.

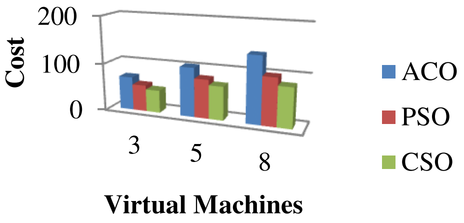

Fig. 5. Cost Evaluation of 800 tasks.

Fig 5 is demonstrating that the CSO algorithm is beating all other algorithms in comparison to the computation cost.

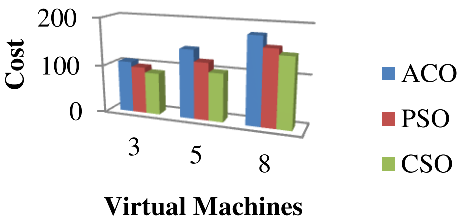

Fig. 6. Cost Evaluation of 1300 tasks.

Fig. 6 is representing that the CSO algorithm is working fine in respect to the computation cost as compared to other algorithms with all sets of VMs.

The reason behind the success of the CSO algorithm as compared to the ACO and PSO is efficient migration of the tasks among under-loaded and fully-loaded virtual machines. It can be clearly identified that the homogeneous environment can give slightly better results on account of the makespan, but it is quite expensive as compared to the heterogeneous cloud environment, the experiments of the heterogeneous cloud environment can be found in the paper [21] in terms of cost in Indian Paisa.

V. CONCLUSTION AND FUTURE SCOPE

Cloud computing is the most demanding technology of the current time. Many algorithms have been developed by the scientist for achieving QoS. In this paper, the most famous algorithms are compared for QoS parameters like makespan and cost. The PSO algorithm is working fine as compared to the ACO algorithm. But the simulation results are showing that the Cat Swarm Optimization algorithm is performing outstandingly as compared to these two ACO and PSO algorithms. The CloudSim toolkit was used for the purpose of experiments. In a homogeneous environment, the makespan can be decreased by enhancing the computation power via virtual machines but this leads to higher costs as compared to the heterogeneous cloud environment.

In the future, a new technique can be compared or an extended version of the CSO algorithm can be proposed.

REFERENCES

[1] M. Dorigo, M. Birattari, and T. Stiitzle, “Ant Colony Optimization: Artificial Ants as a Computational Intelligence Technique,” IEEE Computational Intelligence Magazine, pp. 28-39, November, 2006.

[2] M. A. Tawfeek, A. El-Sisi, A. E. Keshk, and F. A. Torkey, “Cloud Task Scheduling Based on Ant Colony Optimization,” 8th International Conference on Computer Engineering and Systems (ICCES), pp. 64-69, 2013.

[3] J. Sun, S. Xiong, and F. Guo, “A New Pheromone Updating Strategy in Ant Colony Optimization,” Proceedings of the Third International Conference on Machine Learning and Cybernetics, pp. 620-625, Shanghai, 2004.

[4] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, “A Multi-Objective Ant Colony System Algorithm for Virtual Machine Placement in Cloud Computing,” Journal of Computer and System Sciences, vol. 79, pp. 1230-1242, 2013.

[5] X. Wei, “Task Scheduling Optimzation Strategy using Improved Ant Colony Optimization Algorithm in Cloud Computing,” Journal of Ambient Intelligence and Humanized Computing, October, 2020.

[6] L. Wang, and L. Ai, “Task Sheduling Policy Based on Ant Colony Optimization in Cloud Computing Environment,” Proceedings of 2nd International Conference on Logistic, Informatics and Service Science, pp. 953-957, 2013.

[7] H. Sleh, H. Nashaat, W. Sabeer, and H. M. Harb, “IPSO Task Scheduling Algorithm for Large Scale Data in Cloud Computing Environment,” IEEE Access, vol. 7, pp. 5412-5420, 2019.

[8] S. A. Alsaidy, A. D. Abbood, and M. A. Sahib, “Heuristic Initialization of PSO Task Scheduling Algorihm in Cloud Computing,” Journla of King Saud University- Computer and Information Sciences, Article in Press.

[9] N. Dordaie, and N. J. Navimipour, “A Hybrid Particle Swarm Optimization and Hill Climbing Algorithm for Task Scheduling in the Cloud Environments,” ICT Express, vol. 4, pp. 199-202, 2018.

[10] N. Mansouri, B. M. H. Zade, and M. M. Javidi, “Hybrid Task Scheduling Strategy for Cloud Computing by Modified Particle Swarm Optimization and Fuzzy Theory,” Computers & Industrial Engineering, vol. 130, pp. 597-633, 2019.

[11] W. Qi, “Optimization of Cloud Computing Task Execution Time and User QoS Utility by Improved Particle Swarm Optimization,” Microprocessors and Microsystems, vol. 80, 2021.

[12] R. Valarmathi, and T. Sheela, “Ranging and Tuning Based Particle Swarm Optimization with Bat Algorithm for Task Scheduling in Cloud Computing,” Cluster Computing, vol. 22, pp. 11975-11988, 2019.

[13] M. Bansal, S. K. Malik, “A Multi-Feceted Optimization Scheduling Framework Based on Particle Swarm Optimization Algorithm in Cloud Computing,” Sustainable Computing: Informatics and Systems, vol. 28, 2020.

[14] S. Saeedi, R. Khorsand, S. G. Bidgoli, and M. Ramezanpour, “Improved Many-Objective Particle Swarm Optimization Algorithm for Scientific Workflow Scheduling in Cloud Computing,” Computers & Industrial Engineering, vol. 147, 2020.

[15] J. P. B. Mapetu, Z. Chen, and L. Kong, “Low-Time Complixity and Low-Cost Binary Particle Swarm Optimization Algorithm for Task Scheduling and Load Balancing in Cloud Computing,” Applied Intelligence, vol. 49, pp. 3308-3330, 2019.

[16] D. Gabi, A. S. Ismail, A. Zainal, Z. Zakaria, and A. Al-Khasawneh, “Hybrid Cat Swarm Optimization and Simulated Annealing for Dynamic Task Scheduling on Cloud Computing Environment,” Journal of Information and Communication Technology, vol. 17, no. 3, pp. 435-467, 2018.

[17] D. Gabi, A. S. Ismail, and N. M. Dankolo, “Minimized Makespan Based Improved Cat Swarm Optimization for Efficient Task Scheduling in Cloud Datacenter,” HPCCT: Proceedings of the 3rd High Performance Computing and Cluster Technologies Conference, pp. 16-20, June, 2019.

[18] S. Bilgaiyan, S. Sagnika, and M. Das, “A Multi-Objective Cat Swarm Optimization Algorithm for Workflow Scheduling in Cloud Computing Environment,” International Journal of Soft Computing, vol. 10, no. 1, pp. 37-45, 2015.

[19] S. Bilgaiyan, S. Sagnika, and M. Das, “Workflow Scheduling in Cloud Computing Environment using Cat Swarm Optimization,” IEEE International Advance Computing Conference (IACC), pp. 680-685, Feb., 2014.

[20] J. Bhagwan, and S. Kumar, “An HC-CSO Algorithm for Workflow Scheduling in Heterogeneous Cloud Computing System,” International Journal of Advanced Computer Science and Applications, vol. 12, no. 6, pp. 485-492, 2021.

[21] J. Bhagwan, and S. Kumar, “Performance Evaluation of Meta-Heuristic Algorithms for Task Scheduling in Cloud Environment,” International Journal of Advanced Research in Science, Communication and Technology, vol. 5, no. 2, pp. 87-93, 2021.