Performance Evaluation of Various VM Migration based Nature-inspired Mechanisms in Cloud Environment

ARCHANA[1]
NARANDER KUMAR[2]

BBAU - Babasaheb Bhimrao Ambedkar University Lucknow (A Central University)
DCS - Department of Computer Science
Vidya Vihar Raibareli Road - Lucknow - Uttar pradesh 226025 - India
[1] (first)archana03lko@gmail.com
[2*] (second)nk_iet@yahoo.co.in

Abstract. The number of cloud users is rapidly growing in a cloud computing environment, which increases the need for resources. Virtual machine migration, which involves moving the overloaded host to another one, can be used to handle the growing demand for resources successfully. The Bat method, PSO (Particle Swarm Optimization), Cuckoo Search (CS), and Genetic algorithm (GA) are some of the popular meta-heuristics algorithms used in this paper to minimize migration time and makespan value of Virtual Machine(VM) Migration. The main goals of this work are to accomplish VM migration with shorter migration times, smaller makespan values, and higher VM throughput values. Additionally, compare the methods' performance to determine which algorithm is more efficient for minimizing migration time in cloud environments. The fitness value, migration time, makespan, and throughput performance characteristics have been calculated for various task sizes and execution iterations. According to calculated performance characteristics, the Bat algorithm outperformed the other three. The Bat algorithm's migration time is better by 2% to PSO, 6% compared to Cuckoo Search, and 50% compared to GA. Also, when performing VM migration in a cloud computing environment, the Bat algorithm outperforms PSO, CS, and GA in terms of makespan, fitness value, and throughput value.

Keywords: Bat algorithm, Particle swarm optimization, Cuckoo search algorithm, Genetic algorithm, meta-heuristics, virtual machine migration, cloud environment.

(Received November 21st,2022 / Accepted December 11st, 2022)

1 Introduction

Cloud computing provides a low-cost, personalized computing framework and services while assuring quality. These resources are adaptable and can be increased in size as the user needs them. The partitioning of a computer into various VMs with various operating systems and application environments is known as virtualization. Each VM operates independently in this environment and is unaffected by the others. The isolation and security of the programmed are both maintained by this feature. A crucial part of the cloud environment is played by virtualization. Day by day, the number of users increases as more customers migrate to cloud data centres [22].

Due to the limitation of resources, this increasing number of users leads to a significant issue in managing the VMs efficiently. Let's suppose a request is being sent repeatedly to a few servers, and after some time, some of these servers may get overloaded, and others may become idle. Hardware issues could affect every virtual computer on it. The question then becomes how to distribute tasks evenly among servers and prevent VM failure. A process to prevent the host from overloading and distributing tasks uniformly is known as VM migration, which migrates the VMs on overburden hosts to lowburden hosts [24]. In order to migrate virtual machines and reduce their migration time and makespan in a cloud environment, this study employs a number of well-known Meta heuristics techniques, including the Bat method, Cuckoo Search method, PSO method, and Genetic algorithm. To determine whether Meta heuristics method performs best when performing virtual machine migration, comparative performance analysis has also been done. The following are the goals of this paper:

- To reduce the amount of time it takes to migrate VMs in a cloud environment.
- To reduce the makespan of VM migration.
- To maximize the throughput of the Virtual machine.
- To perform the comparative analysis of some popular meta-heuristic algorithm concerning VM migration.

2 Related work

When the use of resources goes extensive, there is a need to handle it. A genetic algorithm is employed to control virtual machines' issues to move the overloaded task to another VM [10]. A bio-inspired algorithm is suggested to execute virtual machine placement that will decrease power consumption and network latency and improve financial profit [4]. The forecasting of VM's necessary resources serves as the foundation for the hybrid technique that is suggested. The performance of the suggested technique was assessed during a live VM migration and compared to other recent techniques [19]. An optimization model for virtual machine selection is provided to maximize the use of medical resources in a cloud environment. Using the provided methodology reduces the overall execution time of medical searches [1].

To reduce service lag for several cloudlets in edge cloud computing scenarios, a PSO model is recommended. Here, emphasis is placed on lowering both transmission and processing delays [20]. Along with discussing the impact of workload aspects, a thorough assessment of the various live VM migration methodologies is covered [12, 21, 5, 26, 2]. The consolidation of virtual machines is offered using an integrated method. To effectively increase VM utilization, this suggested strategy migrates the overloaded host to the idle host and performs collocation of the underloaded host to another underloaded host [14].

A GA technique is presented to balance the workload and optimize virtual machine allocation in the cloud computing framework [15]. An adaptive management strategy is shown to improve the adaptive management capabilities of virtual machine placement in the cloud computing environment. According to the simulation results, this technique performs better and can optimize global [13]. A hybrid approach that uses attribute-based resource allocation to assign and carry out resource consumption per user requirements efficiently is proposed [25]. Find the overloaded and underloaded hosts using the interquartile range to effectively control the overloaded and underloaded VM server [16]. A hybrid strategy is suggested to reduce energy consumption during VM migration [8, 17].

A novel algorithm that was based on the Gene Aggregation Algorithm was proposed. This suggested technique was utilized to reduce the number of genes on each chromosome, efficiently allocate between virtual machines, and reduce communication costs [6, **?**, **?**, **?**]. Two cooperative methods were presented to evenly distribute the physical resources of the cloud system while balancing the demand. The suggested algorithm aids in lowering migration, decreasing power consumption, and preventing performance deterioration [18]. An integrated bio-inspired algorithm was presented to optimize job allocation and reduce VM migration, migration cost, and energy consumption [23].

According to the research, it is necessary to do a comparative analysis of a few well-known meta-heuristics algorithms in their respective fields to reduce the migration time and makespan of VM migration in cloud environments. This work uses the Bat, PSO, CS, and GA algorithms to address the problems of VM migration in a cloud environment, which could fill a research gap. The following is how this paper is structured. The introduction is presented in Section 1, and Section 2 contains the related work. Section 3 describes the problem formulation, and section 4 explains the methodology. Section 5 shows the implementation or simulation results of all four meta-heuristics algorithms (Bat, PSO, CS, and GA), comparative analysis, and a discussion. Section 6 includes the conclusions, and future perspectives and references are presented in Section 7.

3 Problem formulation

This paper uses the CloudSimPlus simulator to perform a simulation of an aforementioned meta-heuristics algorithm. The problem is minimizing the migration time, lowering the makespan value and increasing the resource migration's throughput. Four performance metrics were calculated, and each method's results have been compared to indicate which algorithm provides the shortest migration and makespan value and performs better VM migration in cloud environments. All performance metrics have been mathematically formulated and are listed below:

3.1 Fitness value

This parameter is considered to calculate the fitness value of all employed algorithms, i.e. Bat, PSO, CS and GA algorithms.

$$objfun = min(f + xd * xd) \tag{1}$$

Here objfun represents the objective function, x indicates each task, and d indicates the dimension of migration.

3.2 Migration time

This parameter is used to calculate the migration time of all four algorithms. Here mT represents the migration time, eM shows the end of VM migration, and sM shows the start time of the VM migration.

$$mT = eM - sM \tag{2}$$

3.3 Makespan

This parameter is evaluated to calculate the makespan of the algorithms for VM migration. mS Represents the migration time, VM_MIPS shows a million instructions per second of the virtual machine, $CLOUDLET_FIleSize$ shows the cloudlet file size, and VM_BW indicates the bandwidth of virtual machines.

$$mS = (((mT)/4 * VM_{MIPS}) +$$

$$(\text{CLOUDLET}_F IleSize/VM_{BW}))/1000 \quad (3)$$

3.4 Throughput

This parameter is evaluated to find out the throughput of the algorithms. It helps to calculate how many instructions migrated per millisecond on a virtual machine.

$$tP = (nPop * maIt)/mT \tag{4}$$

Here tP shows the throughput, nPop indicates the number of resources or task size and maIt indicates the maximum number of running iterations.

4 Methodology

The benefit of using a meta-heuristic algorithm is that they can quickly locate the best solution. The goal of this work is to reduce the virtual machine migration's makespan value and migration time. Therefore, some popular meta-heuristics algorithm was employed in this paper. This section shows the detailed working of the aforementioned meta-heuristics algorithms used in this paper. Here explained the bat algorithm, PSO algorithm, CS algorithm and GA algorithm briefly. This section helps to understand how these algorithms work with their respective steps. Algorithms are listed below:

4.1 Bat algorithm

A recent or novel meta-heuristics approach is the Bat algorithm, which is based on how bats use echolocation. All bats employ echolocation, which helps them avoid obstacles, locate prey, and locate the crevices where they live in the dark. This technique allows them to perceive distance. The features of bats, which individually generate a long, piercing sound pulse and hear the echo that bounces back off of nearby objects, served as the inspiration for the bat algorithm. All bats' considered frequency (fe), position(p), velocity (vi), loudness (lu), and pulse rate (pr) are recorded in the bat algorithm. These criteria assist in the search and discovery of the solution [11].

Steps to perform bat algorithm:

- Initialize the population of bats and vi, fe, lu and pr.
- While (t<maxiteration)
- Generate new solutions by adjusting fe, vi and location.
- If (random>pr)
- Select a solution among the best solution.
- If (random<lu fe (xi) <fe (x1))
- Accept the new solution, increase the pr and reduce lu.
- End if.
- Rank the bats and find the current best.
- End the while
- Display the best solution.

4.2 PSO algorithm

It is a Meta-heuristics algorithm base on the nature-inspired movement behaviour of birds and fish flocking to search the food or resources. Each individual is denoted as a particle, and the population or group on whole particles is considered a swarm. Randomly initialized each particle which searched for a feasible solution. Each particle updates its velocity and position to move towards for optimal or best solution. The particle updates its position according to the best individuals closer to the solution [3]. This criterion helps to search for and discover the solution.

A step-by-step working of the PSO algorithm has listed below:

- Set each particle's initial position and velocity in the search space.
- Use the objective or fitness function to determine each particle's fitness. If fitness value exceeds best fitness value (wbest)

  new value=wBest
- For each particle, calculate velocity and position position=vm$_i^t + z_i^t$

  Velocity(z$_i^t + 1) = uz_i^t + c_1 rd_1(lBest_i^t - vm_i^t)$ $+ c_2 rd_2((wBest_i^t - vm_i^t))$

  (vm_i^t represents the old position, z_i^t shows the velocity of each particle, $z_i^t + 1$ represents the updated velocity, c represents the acceleration factor,u=inertia weight, rd_1, rd_2=random value, $lBest_i^t$ shows the local best of each particle and $wBest_i^t$ shows the global best)
- Evaluate fitness by using the fitness function
- Find the current best
- update(t)=t+1

  (t shows the termination or iteration)
- Output $wBest_i^t$ and vm_i^t
- Stop when it reaches the termination condition; otherwise, go to step 3.

4.3 Cuckoo Search algorithm

It is a meta-heuristic method that is based on cuckoo bird traits. To rear their newly born chicks by the host species, each cuckoo lays eggs in the host species that are identical to it in terms of egg colour, shape, and texture. In 2009, Xin-she Yang and Suash Deb, motivated by the cuckoo bird character, invented the cuckoo search algorithm. An egg in a nest represents each solution in the cuckoo search method, and any newly discovered solutions are also represented by cuckoo eggs. Therefore, this situation aims to develop a new, potentially superior solution to replace the existing, less-than-ideal one in the nests. In CS algorithm used the levy flight mechanism instead of a simple random walk. To determine the length of a random walk's steps, a heavy flight takes the heavy-tailed probability distribution into account.

$$X_i^t + 1 = X_i^t + \alpha * levy(\lambda) \tag{6}$$

The CS algorithm suggests three main rules, which are as follows:

- Each cuckoo can only lay one egg at a time in a nest that is selected at random.
- The best nests with top-notch eggs (solutions) are taken into consideration for the next generation.
- There are a fixed number of host nests that can be reached.

The host bird has a probability of discovering an alien egg (0, 1), in which case it can either toss the egg or give up on the nest and start a new one somewhere else [7].

4.4 Genetic algorithm

It is one of the more well-known Meta heuristic algorithms, created by J.H. Holland in 1992. Fitness selection, chromosomal representation, and biologically inspired operators are some of its components. Three biologically inspired operations are selection, mutation, and crossover. The main idea underlying the GA is to combine the best solutions from past explorations of the solution space with new ones. GA is carried out in the following steps in a computing environment:

- Using a heuristic method or randomly generating an initial population of chromosomes. Chromosomes in the population determine the potential answer.
- Each chromosome's fitness value is assessed.
- Use the genetic operators of mutation, crossover, and selection now.
- From this developing population, calculate each chromosome's fitness value once more.

- Put an end to the procedure once all chromosomes have been mapped to the same intersection or when it has reached its maximum number of iterations [9] .

5 Simulation results and discussion

The cloudSimPlus simulator has been used to employ all four mentioned meta-heuristic algorithms in a cloud environment. The performance parameters calculated during simulation are migration time, fitness value, makespan, and throughput, which help to compare these algorithms. The performance metrics, as mentioned earlier calculated using different numbers of tasks size and iterations. Here, consider the 10, 50, 100, 150, and 200 no. tasks size and for iteration 100, 200, 400, 600, 800, and 1000 considered. The Bat, PSO, CS and GA algorithms run for all mentioned number of iterations and task sizes to perform VM migration. The simulation parameters required to implement all four meta-heuristics algorithms are listed in Table 1.

Table 1: Simulation parameters

Type	*Parameters*	*Value*
Tasks	Tasks size number of iterations	10-200 100-1000
Broker	Number of brokers	4
Datacenter	No. of data centers	2
CloudletLength	Length of data	10,000
VM	No. of VMs RAM RAM Operating system	4 10000 MB 10000 MB Windows 10
Hosts	No. of hosts MIPS Storage RAM Bandwidth	3 1000 10,00,000 (15,000,500000,25000)MB 16000L (mb/s)
Batalgorithm	Loudness Pulse rate	0,1 0,1
PSOalgorithm	Interia weight Acceleration coefficient	1 1.3, 2.7
CSalgorithm	PA (probability) Pulse rate	0.25 0,1
GAalgorithm	Mutation rate Crossover rate	0.5 0.7
Parametersforallfouralgorithms	Dimensions Upper bound Lower bound	5 5.12 -5.12

All four algorithms' performance is evaluated in six dimensions: (i) migration time with a different number of iterations, (ii) fitness value with a diverse iterations, (iii) makespan value with a numerous number of iterations, (iv) throughput value with diverse iterations, (v) migration time with a varying number of tasks, and (vi) makespan value with a numerous number of tasks, Table 2. shows the calculated values of the first performance dimension (migration time with a different number of iterations) of all four algorithms and the graphical representation in fig 1. Here iteration values vary from 100, 200, 400, 600 and 1000 at task 50. The graph of fig 1 is generated on the CloudSimPlus simulator, where the numbers of distinct iterations are represented on the x-axis and migration time in The calculated values of the second performance dimension (fitness value with a different number of iterations) of all four algorithms are enlisted in table 3. Here iteration values vary from 100, 200, 400, 600 and 1000 at task 50. The graphical representation of table 3 is shown in fig 2, where the numbers of distinct iterations are represented on the x-axis and the fitness values is shown on the y-axis. The calculated result demonstrates that the Bat algorithm outperformed PSO, CS and GA to optimize the fitness value of VM migration, moving from minimum to the maximum number of iterations. The calculated values of

Table 2: Comparison of Migration time (in ms) with different numbers of iterations at task size 50

No.ofIterations	*Algorithm*			
	Bat	PSO	CS	GA
100	3	5	17	30
200	5	7	8	35
400	7	9	13	47
600	9	11	16	54
800	10	12	22	58
1000	12	14	18	62

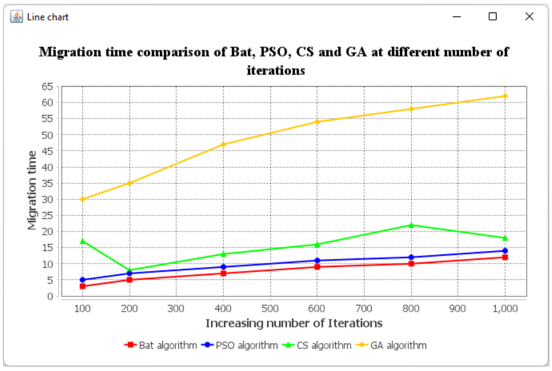

Figure 1: A graphical comparison of the migration time of Bat, PSO, CS and GA algorithms at minimum to maximum iteration

the second performance dimension (fitness value with a different number of iterations) of all four algorithms are enlisted in table 3. Here iteration values vary from 100, 200, 400, 600 and 1000 at task 50. The graphical representation of table 3 is shown in figure 2, where the number of distinct iterations are represented on the x-axis and the fitness values is shown on the y-axis. The calculated result demonstrates that the Bat algorithm outperformed PSO, CS and GA to optimize the fitness value of VM migration, moving from minimum to the maximum number of iterations. The simulation

Table 3: Comparison of Fitness value with different number of iterations at task size 50

$No.of Iterations$	$Algorithm$			
	Bat	PSO	CS	GA
100	1.725	2.1295	2.1430	4.0
200	0.1011	1.6339	2.4462	3.61
400	1.2033	1.1113	1.4688	4.0
600	0.5086	0.9184	1.8123	5.0
800	0.7907	0.8219	1.0493	4.21
1000	0.9618	0.9968	1.2149	5.0

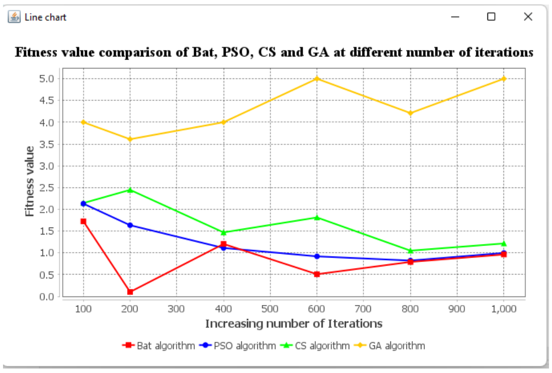

Figure 2: A graphical comparison of the fitness values of Bat, PSO, CS and GA algorithms at minimum to maximum iteration

values of the third performance dimension (makespan value with a different number of iterations) of all four algorithms are enlisted in Table 4. Here iteration values vary from 100, 200, 400, 600 and 1000 at task 50. The graphical representation of table 4 is shown in fig 3, where the numbers of distinct iterations are represented on the x-axis and the makespan values in milliseconds is shown on the y-axis. The calculated result demonstrates that the Bat algorithm outperformed PSO, CS and GA to minimize the makespan value of VM migration by moving to a minimum to a maximum number of iterations. The calculated values of the fourth performance dimension (throughput value with a different number of iterations) of all four algorithms are enlisted in table 5. Here iteration values vary from 100, 200, 400, 600 and 1000 at task 50. The graphical representation of table 5 is shown in fig 4, where the numbers of distinct iterations are represented on the x-axis and the throughput values in instruction per milliseconds is shown on the y-axis. The calculated result indicates that the Bat algorithm outperformed as compared PSO, CS

Table 4: Comparison of Makespan value (in ms) with different number of iterations at task size 50

$No.of Iterations$	$Algorithm$			
	Bat	PSO	CS	GA
100	15	23	69	120
200	22	29	32	140
400	30	39	54	188
600	38	45	67	217
800	40	51	88	234
1000	49	59	75	248

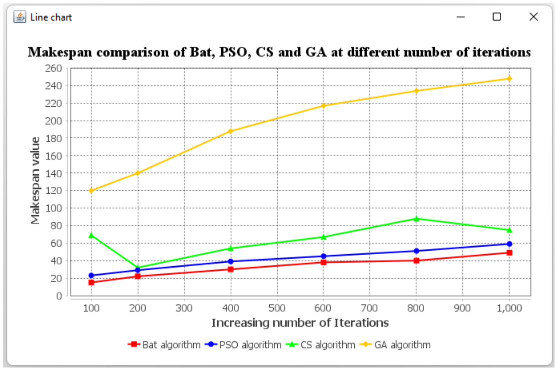

Figure 3: A graphical comparison of the makespan value of Bat, PSO, CS and GA algorithms at minimum to maximum iteration

and GA to maximize the throughput value of VM migration, moving to a minimum to a maximum number of iterations. The calculated values of the fifth per-

Table 5: Comparison of throughput value with different number of iterations at task size 50

$No.of Iterations$	$Algorithm$			
	Bat	PSO	CS	GA
100	333	217	72	41
200	454	344	312	71
400	666	512	370	106
600	789	666	447	138
800	1000	784	454	170
1000	1020	847	666	201

formance dimension (migration time with a different number of tasks at iteration 1000) of all four algorithms are enlisted in table 6. Here task sizes vary from 10, 50, 100, 150and 200 at iteration 1000. The graphical representation of table 6 is shown in fig 5, where the numbers of distinct task sizes are represented on the x-axis and the migration time in milliseconds is shown

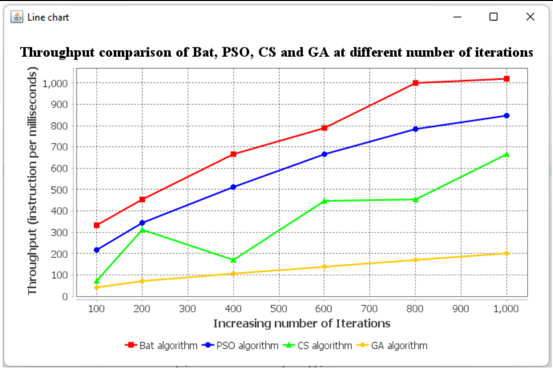

Figure 4: A graphical comparison of the throughput value of Bat, PSO, CS and GA algorithms at minimum to maximum iteration

on the y-axis. The calculated result demonstrates that the Bat algorithm outperformed PSO, CS and GA to minimize the migration time of VM migration, either moving to a minimum to a maximum number of task sizes. The calculated values of the last performance

Table 6: Comparison of Migration time with different numbers of tasks at iteration 1000

No.of Iterations	*Algorithm*			
	Bat	PSO	CS	GA
10	6	6	7	34
50	12	15	20	64
100	19	24	50	90
150	29	33	53	92
200	33	39	50	116

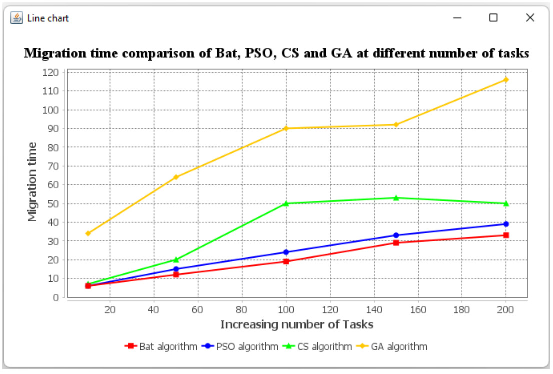

Figure 5: A graphical comparison of the migration time of Bat, PSO, CS and GA algorithms at minimum to maximum task size

dimension (makespan value with a different number of tasks at iteration 1000) of all four algorithms are enlisted in table 7. Here task size varies from 10, 50, 100, 150 and 200 at iteration 1000. The graphical representation of table 7 is shown in fig 6, where the numbers of distinct task sizes are represented on the x-axis and the makespan value in milliseconds is shown on the y-axis. The calculated result demonstrates that the Bat algorithm outperformed compared to PSO, CS and GA to minimize the makespan of VM migration, either moving to a minimum to a maximum number of task sizes.

All the four algorithms better in their perspectives on

Table 7: Comparison of Makespan with different number of tasks at iteration 1000

No.of Iterations	*Algorithm*			
	Bat	PSO	CS	GA
10	26	27	31	138
50	49	62	83	256
100	79	97	202	361
150	119	133	213	368
200	133	159	200	465

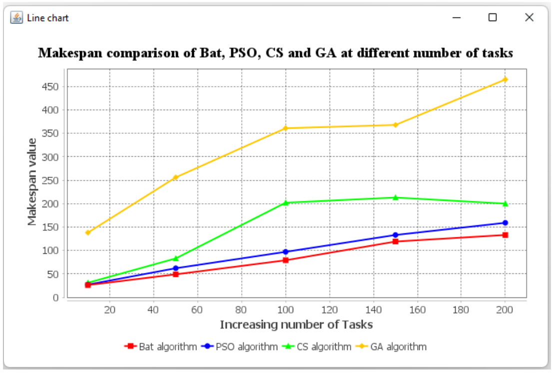

Figure 6: A graphical comparison of the makespan value of Bat, PSO, CS and GA algorithms at minimum to maximum task size

performing VM migration, but when these four algorithms have been compared, it was discovered that Bat is superior to PSO, CS, and GA algorithms, PSO is superior to CS and GA algorithms, and CS outperforms GA algorithm in terms of resolving the VM migration issue.

6 Conclusion

This study employs some popular meta-heuristics algorithms, i.e. Bat algorithm, PSO, the Cuckoo Search method and the Genetic algorithm, to keep down the migration time, minimize the makespan value and increase the throughput of VM migration in a cloud based computing. Here is a performance comparison of the methods mentioned above, which have been applied using varying tasks and iterations from the smallest to the largest. To compare and determine which metaheuristics algorithm is best for VM migration, performance factors include the fitness value, migration time, makespan, and throughput value. When performing VM migration, the migration time through the Bat algorithm takes 12 milliseconds, PSO takes 14 milliseconds, CS takes 18 milliseconds, and GA takes 62 milliseconds, with a maximum of 1000 iterations. At a maximum iteration of 1000, the Bat method has makespan (in milliseconds) and throughput (in instructions per millisecond) values of 49 and 1020 for PSO 59 and 847, CS 75 and 666, and GA 248 and 201, respectively. According to the simulation results mentioned above, the Bat algorithm performs well in reducing migration time, decreasing makespan value, and improving the throughput value of VM migration in a cloud based computing. Another technique superior to the Bat algorithm for accomplishing VM migration in a cloud based computing can be suggested for future perspective.

References

[1] Abdelaziz, A., Anastasiadou, M., and Castelli, M. A parallel particle swarm optimisation for selecting optimal virtual machine on cloud environment. *Applied Sciences*, 10(18):6538, 2020.

[2] Affonso, E. T., Nunes, R. D., Rosa, R. L., Pivaro, G. F., and Rodríguez, D. Z. Speech quality assessment in wireless voip communication using deep belief network. *IEEE Access*, 6:77022–77032, 2018.

[3] Affonso, E. T., Rodríguez, D. Z., Rosa, R. L., Andrade, T., and Bressan, G. Voice quality assessment in mobile devices considering different fading models. In *2016 IEEE International Symposium on Consumer Electronics (ISCE)*, pages 21–22. IEEE, 2016.

[4] Akram, S., Ghaleb, S., Ba, S., and Siva, V. Survey study of virtual machine migration techniques in cloud computing. *Int. J. Comput. Appl*, 177:18–22, 2017.

[5] Braiki, K. and Youssef, H. Data center resource provisioning using particle swarm optimization and cuckoo search: A performance comparison. In *International Conference on Advanced Information Networking and Applications*, pages 1138–1149. Springer, 2020.

[6] Gopu, A. and Venkataraman, N. Virtual machine placement using multi-objective bat algorithm with decomposition in distributed cloud: Moba/d for vmp. *International Journal of Applied Metaheuristic Computing (IJAMC)*, 12(4):62–77, 2021.

[7] Goyal, K., Jain, V., and Verma, P. An analysis on virtual machine migration issues and challenges in cloud computing. *Int J Comput Appl*, 975:8887, 2018.

[8] Jiang, Y., Wang, J., Shi, J., Zhu, J., and Teng, L. Network-aware virtual machine migration based on gene aggregation genetic algorithm. *Mobile Networks and Applications*, 25(4):1457–1468, 2020.

[9] Joshi, A. S., Kulkarni, O., Kakandikar, G. M., and Nandedkar, V. M. Cuckoo search optimization-a review. *Materials Today: Proceedings*, 4(8):7262–7269, 2017.

[10] Karthikeyan, K., Sunder, R., Shankar, K., Lakshmanaprabu, S., Vijayakumar, V., Elhoseny, M., and Manogaran, G. Energy consumption analysis of virtual machine migration in cloud using hybrid swarm optimization (abc–ba). *The Journal of Supercomputing*, 76(5):3374–3390, 2020.

[11] Katoch, S., Chauhan, S. S., and Kumar, V. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, 80(5):8091–8126, 2021.

[12] Kaur, G. and Sachdeva, R. Virtual machine migration approach in cloud computing using genetic algorithm. In *Advances in Information Communication Technology and Computing*, pages 195–204. Springer, 2021.

[13] Kaur, K. and Kumar, Y. Swarm intelligence and its applications towards various computing: A systematic review. In *2020 International conference on intelligent engineering and management (ICIEM)*, pages 57–62. IEEE, 2020.

[14] Le, T. A survey of live virtual machine migration techniques. *Computer Science Review*, 38:100304, 2020.

[15] Li, S. and Pan, X. Adaptive management and multi-objective optimization of virtual machine in cloud computing based on particle swarm optimization. *EURASIP Journal on Wireless Communications and Networking*, 2020(1):1–12, 2020.

[16] Liu, F., Ma, Z., Wang, B., and Lin, W. A virtual machine consolidation algorithm based on ant colony system and extreme learning machine for cloud data center. *IEEE Access*, 8:53–67, 2019.

[17] Muhammad, A., Xu, Y., Yan, C., and Chen, N. A balanced virtual machine migration mechanism based on genetic algorithm. In *2017 2nd International Conference on Machinery, Electronics and Control Simulation (MECS 2017)*, pages 173–176. Atlantis Press, 2016.

[18] Muhammad, K., Gao, S., Qaisar, S., Abdul, M., Muhammad, A., Usman, A., Aleena, A., and Shahid, A. Comparative analysis of meta-heuristic algorithms for solving optimization problems. In *2018 8th International Conference on Management, Education and Information (MEICI 2018)*, pages 612–618. Atlantis Press, 2018.

[19] Naik, B. B., Singh, D., and Samaddar, A. B. Fhcs: Hybridised optimisation for virtual machine migration and task scheduling in cloud data center. *IET Commun.*, 14(12):1942–1948, 2020.

[20] Nashaat, H., Ashry, N., and Rizk, R. Smart elastic scheduling algorithm for virtual machine migration in cloud computing. *The Journal of Supercomputing*, 75(7):3842–3865, 2019.

[21] Paulraj, G. J. L., Francis, S. A. J., Peter, J. D., and Jebadurai, I. J. A combined forecast-based virtual machine migration in cloud data centers. *Computers & Electrical Engineering*, 69:287–300, 2018.

[22] Rodrigues, T. G., Suto, K., Nishiyama, H., and Kato, N. A pso model with vm migration and transmission power control for low service delay in the multiple cloudlets ecc scenario. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2017.

[23] Rodríguez, D. Z., Rosa, R. L., Costa, E. A., Abrahão, J., and Bressan, G. Video quality assessment in video streaming services considering user preference for video content. *IEEE Transactions on Consumer Electronics*, 60(3):436–444, 2014.

[24] Shirvani, M. H., Rahmani, A. M., and Sahafi, A. A survey study on virtual machine migration and server consolidation techniques in dvfs-enabled cloud datacenter: Taxonomy and challenges. *Journal of King Saud University-Computer and Information Sciences*, 32(3):267–286, 2020.

[25] Singh, G. and Gupta, P. A review on migration techniques and challenges in live virtual machine migration. In *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pages 542–546. IEEE, 2016.

[26] Singh, S. and Singh, D. A bio-inspired vm migration using re-initialization and decomposition based-whale optimization. *ICT Express*, 2022.

[27] Talwani, S. and Singla, J. A comprehensive review of virtual machine migration techniques in cloud computing. In *Proceedings of the International Conference on Innovative Computing & Communications (ICICC)*, 2020.

[28] Vadivel, R. and Sudalaimuthu, T. Cauchy particle swarm optimization (cpso) based migrations of tasks in a virtual machine. *Wireless Personal Communications*, pages 1–18, 2021.

[29] Zhang, F., Liu, G., Fu, X., and Yahyapour, R. A survey on virtual machine migration: Challenges, techniques, and open issues. *IEEE Communications Surveys & Tutorials*, 20(2):1206–1243, 2018.