International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 36, No. 2, April, 2022, pp. 327-332
Journal homepage: http://iieta.org/journals/ria

GWOCSA Based Algorithm for Allocating Resources to the Tasks in the Cloud

Ellendula Madhukar[1*], Thirumalaisamy Ragunathan[2]

[1] Department of CSE, Sreenidhi Institute of Science and Technology, Hyderabad, Telangana 501301, India
[2] Department of CSE, SRM University, Amaravati, Andhra Pradesh 522503, India

Corresponding Author Email: emadhukar@gmail.com

https://doi.org/10.18280/ria.360218

Received: 14 February 2022
Accepted: 22 April 2022

Keywords:
cloud computing, resource allocation, optimization, GWO, CSA, hybrid algorithms, makespan

ABSTRACT

In the era of enhancing demand for online resources cloud is emerging as the greatest cutting edge technologies to serve the requirements. With volatility in the usage of the cloud resources, it is a tough task to serve the user requirements. Heuristic algorithms, Meta heuristic algorithms are trending these days, as they are helping to find the nearby optimal solutions within a reasonable time. But they suffer from either slow convergence or with premature solutions. It is evident that NP-complete algorithms take exponential time to find an efficient and optimal solution. This paper hybridizes grey wolf optimization along with Crow search algorithm. This balances the exploration and exploitation. The experimental results prove that the proposed algorithm is on par with existing algorithms and at times it shows better performance.

1. INTRODUCTION

Despite the fact that cloud computing has proved to be a transformative technology, the current COVID scenario has placed the technology into a demand zone to meet the requirements of the users to avoid the gathering and to maintain physical distancing. This led to a growing need for online services in the industry, academia, health care, and society. Cloud has aided in maintaining social distance and mob gathering by offering on-demand support for platforms such as Zoom, CISCO Webex, Google meet, Microsoft team, and others

It is possible to get teachers and students together on a common, shared network by introducing cloud computing. Educational institutions are not required to own servers and data centers [1]. Instead, they will use cloud infrastructure to get access to compute resources, databases, servers, and other services as required.

Educational institutions develop interactive classrooms for their students utilizing cloud-based technology. The approach drastically decreases the costs of infrastructure. The cloud technology removes time and geographical limitations, ensuring that information is consistently provided at all times.

The healthcare sector has come a long way in terms of optimizing its data collection activities, from traditional storage to the digitalization of healthcare data. The generation, consumption, preservation, and distribution of healthcare data have changed dramatically [2].

Industry 4.0 is the most recent era in the manufacturing industry, brought about by the Internet of Things and data accessibility. The concept of "Smart Factories" is gaining traction. Increased automation, machine-to-machine, and human-to-machine connection, artificial intelligence, ongoing technical advancements, and digitization of business are all part of the current trend.

The primary purpose of cloud computing is to distribute computing activities to a resource pool made up of a vast number of heterogeneous virtual machines (VMs). Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and other clouds offer the services to worldwide customers, whether stationary or on the go, with a new level of computing tools. In an Internet-based world, these types of services are available on-demand through a pay-per-use/subscription model [3].

Scheduling is essential to improve the efficiency of the cloud. It helps in allocating the resources to the Jobs to complete their execution. Scheduling can be done with deterministic or heuristic algorithms. It is evident that the deterministic algorithms are not suitable for large scale. In contrast, heuristic algorithms are problem-specific. Meta-heuristic algorithms have shown good improvement with the objective function. This led the researchers to work with various metaheuristic algorithms like Ant colony optimization (ACO), Simulated annealing (SA), Particle swarm optimization (PSO), Differential evolution algorithm, Genetic algorithm, and so on.

The two aspects of metaheuristic algorithms are expanded searching (exploration) in the solution space and sharpened search (exploitation) to discover the optimal solution. Balancing these two characteristics is a designing challenge, which in turn shows the efficacy of the metaheuristic algorithms. Concerning these, features of individual metaheuristics are combined to magnify the all-inclusive performance.

Grey wolf optimization algorithm (GWO) [4] is one of the efficient meta-heuristic algorithms. It mimics the headship quality and inimitable(unique) system of the cast about (search) behavior of grey wolf to attack the prey. GWO algorithm is based on population, which initially considers feasible solutions and converges in each iteration towards the optimal solution. Though the exploration and exploitation capabilities of GWO are well balanced, it has the problem of immature exploration. It leads the algorithm to end with a compromised solution. This problem can be avoided with an additional control variable that enhances the capability of exploration. Askarzadeh [5] proposed a Crow search algorithm (CSA) based on crow's behavior. The communication between the crows, hiding the food, and salvaging the food are the intelligent actions of the crows. In CSA, the intelligent actions of crows were adopted. This algorithm avoids the local optima cleverly.

The challenge of managing the resources in the cloud environment is NP-complete. Though many meta-heuristic algorithms have been proposed, they have limitations. Majorly they suffer from premature optimal solutions because of inefficient exploration and exploitation strategies.

To conquer the predicament, researchers conjoined to reduce the limitations. GWO and CSA are combined to bridge the defects of the individual algorithms and amalgamated as GWOCSA [6]. In this paper, the GWOCSA is checked on the cloud for managing the resources and scheduling the tasks. The algorithm has been checked and compared with GWO and CSA algorithms for optimized makespan. The results show that the performance of this hybridized algorithm is promising and gives near-optimal solutions.

The remaining sections are organized as follows. Section 2 discusses related work. Section 3 explains the proposed algorithms and related equations. Section 4 shows the experimental results. Conclusion and future work is shown in section 5.

2. RELATED WORK

In the area of optimization, many heuristic and meta-heuristic theories have been proposed in the recent past. Cloud computing has adopted some of the optimization techniques which combine or improvise one or more meta-heuristic algorithms. To enhance the efficiency of scheduling algorithms, researchers are using swarm intelligence algorithms. To quote a few, "Ant colony optimization, Particle swarm optimization, Genetic Algorithm, Differential evolution algorithm, Simulated annealing, Whale optimization, Grey wolf optimization, Crow search algorithm, Eagle strategy, Moth search algorithm, Artificial bee colony algorithm, Spider Monkey algorithm, Jaya algorithm, Teaching learning based optimization" [1], and so on.

2.1 Ant colony optimization

Tawfeek et al. [7] applied an ant colony optimization algorithm in scheduling and compares ACO with FCFS and proves that ACO shows better results. Li et al. [8] applied ant colony optimization to solve load balancing problems in the cloud environment. They propose an improvised algorithm LBACO.

Dai et al. [9] took into account the effect of strong positive feedback of ACO on the algorithm's convergence rate. The initial pheromone chosen has a significant effect on the convergence rate. The algorithm uses a GA's global search capability to find the best solution and then transforms it into the ACO's initial pheromone.

Azad and Navimipour [10] combined the ACO algorithm with the Cultural algorithm. Makespan and energy consumption are considered as objectives. Particle swarm optimization and ant colony optimization are combined in Ref. [11]. To maintain the population's diversity and the fitness of the particles increased, the authors demonstrate improved working efficiency in fitness, expense, and running time through their approach. They present a more accurate and effective understanding of task scheduling.

However, ACO is strongly reliant on its two constants $\boldsymbol{\alpha}$ and β as a probabilistic algorithm. For combinatorial kinds of problems, it is highly desired to retain the standard values of these parameters. The ACO algorithm's stagnation phase could not be recovered without modifying the values of α. Even if standard values of α and β are preserved, the convergence speed of the solution drastically varies for the same problem and the same number of ants.

2.2 Particle swarm optimization

In the recent past, researchers proposed many algorithms to solve optimization problems. Though individually, they showed their efficiency, either they suffered from lack of exploration or exploitation. The limitation of exploration and exploitation of the algorithms led the researchers to hybridize the algorithms to enhance efficiency. GA and PSO were added to solve the optimization problems [12]. Individuals are generated along with crossover, mutation, and operators of local and global search of PSO. As suggested by Pandey et al. [13], (PSO)-based heuristic for scheduling cloud applications to consider both computing and data transmission costs as the main objectives. The authors compare their proposed algorithm with the "Best resource selection" heuristic. They claim that their algorithm is better than the heuristic algorithm.

Awad et al. [14] proposed a numerical model by considering reliability, execution time, transmission time, and load balancing between tasks and virtual machines using Load Balancing Mutation (balancing) a particle swarm optimization (LBMPSO) based schedule and allocation. They compare their method with standard PSO, random algorithm, and Longest Cloudlet to Fastest Processor (LCFP) algorithm to show an improvement.

Verma and Kaushal [15] proposed a multi-objective optimization-based algorithm by hybridizing particle swarm optimization (HPSO). Under deadline and budget constraints, the HPSO heuristic attempts to optimize two competing goals, namely, makespan and expense. Along with these two opposing goals, the number of resources used to build the workflow schedule is also reduced. The algorithm generates a set of Pareto Optimal solutions from which the consumer can choose the most appropriate one.

Gill et al. [16] suggested BULLET, a PSO-based resource scheduling strategy for scheduling workloads in the cloud to reduce execution expense, time, and energy consumption. BULLET is efficient in reducing the execution time, expense, and energy usage of cloud workloads, as well as other QoS parameters, including availability, efficiency, latency, and resource utilization, according to experimental findings.

The particle swarm optimization (PSO) algorithm has the disadvantages of being easy to fall into a local optimum in high-dimensional space and having a slow convergence rate in the repeated process [17].

2.3 Genetic algorithm

Zhao et al. [18] suggested an efficient genetic algorithm for scheduling independent and divisible tasks that can respond to various computing. They activate the algorithm in heterogeneous environments, where computing and communication capabilities (including CPUs) are heterogeneous. The use of dynamic scheduling is also being considered.

Aziza and Krichen [19] devised a fitness mechanism that aids in the reduction of mission execution costs and minimization of makespan. The extension of GA to two existing scheduling rules, time-shared and space-shared, resulted in two additional policies, TSGA and SSGA. The makespan and overall cost of execution are two metrics used to measure performance.

Ibrahim et al. [20] proposed an Integer Linear Programming (ILP) model that minimizes energy consumption in a Cloud data center to create a dynamic task scheduling algorithm. In addition, an Adaptive Genetic Algorithm (GA) is proposed to reflect the complex nature of the Cloud world and to provide a near-optimal scheduling solution that saves resources.

Zhou et al. [21] suggested MGGS (modified genetic algorithm (GA) combined with greedy strategy) as a new algorithm in this article. To maximize the efficiency of scheduling, the proposed algorithm uses a revamped GA algorithm paired with a greedy approach. MGGS, unlike other algorithms, can find an optimal solution with a lower number of iterations.

2.4 Grey wolf optimization

The grey wolf, also known as the timber wolf [5], is the world's largest wild dog. Wolves exist in clans led by an alpha wolf, with the rest of the pack adhering to a dominance hierarchy. The actions like hunting and finding the location of stay are taken by the alpha wolf. The second level of the wolf pack is beta and is considered subordinate to the alpha. The following levels of the wolves are delta and omega.

The phases of hunting can be considered as follows.

1) Tracking, pursuing, and closing in on the prey. 2) Pursue, encircle, and annoy the prey until it comes to a halt. 3) Attack on the prey.

Grey wolf optimizer is one of the metaheuristic algorithms. It was suggested by Mirjalili et al. [5] who proposed an intriguing meta-heuristic algorithm that mimicked the behavior of grey wolves.

(1) Hierarchical structure

GWO has been mathematically modeled by taking into account the social hierarchy. The best location of the search agent in the solution space is considered as α wolf, β as the second best, and δ as the third. The remaining are known as omega wolves [5].

(2) Encircling

During the hunting wolves encircle the prey. The equation for encircling the prey is denoted by in Eq. (1) and Eq. (2) [4, 5]. Here, the position of the wolves and the prey is represented by a vector. In the equation '*Dist*' indicates the distance between prey and the wolf. GW and X_p correspond to the location of the wolf and the placement of prey respectively.

$$\vec{D} = |\overrightarrow{C * \vec{X}_{prey}}(t) - \overrightarrow{GW}(t)| \tag{1}$$

$$\overrightarrow{GW}(t+1) = \vec{X}_{prey}(t) - \vec{A} \cdot \vec{D} \tag{2}$$

$$\vec{A} = 2\vec{a}\vec{r}_1 - \vec{a} \tag{3}$$

$$\vec{C} = 2 \cdot \vec{r}_2 \tag{4}$$

Eq. (3) & Eq. (4) are vectors of coefficients.

r1 and r2 represent random values between [0,1].

(3) Prey hunting

Grey wolves' encircling the prey is modeled by Eq. (5) and Eq. (6). Guided byα, β, and δ wolves, all the remaining wolves update their position by Eq. (7).

$$\begin{aligned} \vec{D}_\alpha &= |\vec{C} * \vec{X}_\alpha(t) - \overrightarrow{GW}(t)| \\ \vec{D}_\beta &= |\vec{C} * \vec{X}_\beta(t) - \overrightarrow{GW}(t)| \\ \vec{D}_\delta &= |\vec{C} * \vec{X}_\delta(t) - \overrightarrow{GW}(t)| \end{aligned} \tag{5}$$

$$\begin{aligned} \overrightarrow{GW_1}(t+1) &= \vec{X}_\alpha(t) - \vec{A} \cdot \overrightarrow{Dist}_\alpha \\ \overrightarrow{GW_2}(t+1) &= \vec{X}_\beta(t) - \vec{A} \cdot \overrightarrow{Dist}_\beta \\ \overrightarrow{GW_3}(t+1) &= \vec{X}_\delta(t) - \vec{A} \cdot \overrightarrow{Dist}_\delta \end{aligned} \tag{6}$$

$$GW(t+1) = \frac{\sum_{i=1}^{3} \overrightarrow{GW_i}(t+1)}{3} \tag{7}$$

(4) Searching & attacking the prey

The prey gets attacked by grey wolves only when it stops progress. It is described mathematically with a vector *'A'* employed in Eq. (3). *'A'* is a random vector and contains the values in [-*a*, *a*] [4, 5], with '*a*' decreasing from 2 to 0 during the iterations using Eq. (8).

|A| < 1 will lead the wolf to assault the prey with movement towards it and |A| > 1 lead the wolf to move away from the prey [1]. The position of α, β and δ wolves decide the direction of search for the prey. Global (exploration) and local (exploitation) searches rely on A and C vectors.

$$a = 2 - (2 \times t/Max_{iter}) \tag{8}$$

The range of random values for the C vector is [0, 2], which is critical for preventing local optima stagnation. The values of the C vector show random behavior, in turn this helps to explore globally.

Algorithm 1: Grey-wolf-Optimization [4, 5]

```
1.  grey wolves (search agents) initialized
2.  $Pop_s(i=1,2,...,n)$
3.  set the values of a, A and C
4.  All individual Search agent's (Wolf) fitness has to be
    calculated
5.  GWα = The first high-quality search agent/wolf
6.  GWβ = The second high-quality search agent/wolf
7.  GWδ = The third high-quality search agent/wolf
8.  repeat
9.  {
10. (j< Max)
11. for all search agents
12. modify the location of all the search agents using Eq. 7
13. end for
14.  modify a, A, C
15. all the search agent's fitness has to be calculated
16. modify GWα, GWβ, GWδ
17. j=j+1
18. } until(j>Max)
19. end loop
20. return GWα
```

The GWO suffers from poor local search and slow convergence rate [22].

2.5 Crow search algorithms

Crows are intelligent creatures. They have the biggest brain in proportion to their body size. Experiments proved that they show self awareness ingenuity. Crows may use tools to interact in complex ways, and remember the location of their food for many months. Crows watch other birds, observing the place of food storage, and then stealing it after actual bird leaves. If a crow was a victim of theft before, it will take special care including changing hiding areas to prevent being a victim again. This behavior of cleverness is mimicked in finding the optimal solution using a natural-inspired algorithm by Askarzadeh [6]. CSA's operation is based on four important principles: herd living, recall the place of secret food, pursue another individual of their genus, and ultimately protect their accumulation from arbitrary plundering.

(1) Mathematical model for CSA

The position of each crow is represented by an M dimensional vector. Searching for the optimal solution starts from an initial population. Initialize the upper bound of iterations, count of crows, flight length and awareness probability. The fitness will be calculated.

The notation $CPop_k$ represents the initial population of the crow and [$CPop_{j1}$, $CPop_{j2}$, $CPop_{j3}$, ...,$CPop_{jM}$] indicates the position of crow '*j*'.

$Cpop^{i,t}$ indicates the initial location of 'i^{th}' crow at time '*t*'. Whereas '$r_{i,}$' and '*fl*' indicate a random number, and flight length respectively. The secret place is '*sp*'.

If jth crow wants to visit the secret place where the food is hidden, ith crow plans to find the secret place of the crow j. This leads to either of the following. First awareness probability is compared with a random number. The crow updates its position by Eq. (9) [3, 6] if the awareness probability is larger. Otherwise it updates its position with a random crow's position.

$$Cpop^{(i+1,t+1)} = Cpop^{i,t} + r_i \times fl^{i,t} \times (sp^{i,t}\text{-}Cpop^{i,t}) \quad if\ AP>r_j \tag{9}$$

Algorithm 2 shows the pseudo code for CSA.

Algorithm 2: CSA Algorithm [3, 5]

```
1.  CPop_k (k=1,2,...,n) is initialized
2.  fitness of all the crows calculated
3.  Initialize reminiscence of crows
4.  repeat
5.  {
6.  for all crows
7.  Awareness probability (AP) is defined
8.  'rand' is a randomly generated number
9.  if rand >= AP
10. modify the location of crow with equation (9)
11. else
12. take the location of the crow randomly
13. end if
14. end for
15. find the viability of the latest solution
16. find the fitness of each search agent
17. modify the memory of the crows
18. i=i+1
19. }
20. until (i>Max)
21. return the best solution crow
```

The Crow search algorithm also suffers from slow convergence speed and can be trapped into local optima.

3. POSED PROCEDURE TO UPDATE THE POSITION OF SEARCH AGENTS

3.1 GWOCSA

Hasty convergence is a flaw in GWO algorithm. This is attributable to the alpha, beta, and delta positions of the search agent's updates. This has a small potential to be exploited as well. By combining the GWO with the CSA, these flaws can be overcome. This strikes an appropriate balance between discovery and exploitation. The proposed algorithm effectively maximizes the advantages of the two algorithms, resulting in broad universal applicability. The CSA algorithm uses flight [3, 6]as a control parameter. This helps in both ways, i.e., in global and local searching. Small values guide to go for local search and large values for global search.

The poor global searching capability can be overcome by accommodating a constant. This is adapted from CSA. Eq. (10) and Eq. (11) represent this scenario. The constant '*fl*' guides to search in the global space and local space based on the selected value. This helps to stabilize both exploration and exploitation.

Algorithm 3 represents the pseudo-code for GWOCSA. In the GWO based approach, updating of the search agent's position will be done in view of α, β, and δ wolves location. However, in the proposed hybrid model, the update of the grey wolf is done in accordance with Eq. (10). This helps greatly in avoiding local optima and helps to search in the global space.

Algorithm 3: GWOCSA [3]

```
1.  Initialize the grey wolves GW_i (i=1,2,...,n)
2.  a, A, and C initialized.
3.  find the fitness of all the search agents (Wolf)
4.  find GW_α and GW_β
5.  while (t<Max)
6.  for all the search agents
7.  if AP>random number
8.  modify the location of the present search agent by Eq. (10)
9.  else
10. modify the location of the present search agent by Eq. (11)
11. end if
12. end for
13. modify AP using Eq. (12)
14. modify 'a' using Eq. (13)
15. modify A, C
16. modify the fitness of all search agents.
17. modify GWα, GWβ
18. t=t+1
19. end while
20. return GWα
```

However, to modify the location of the search agent is done based on an adaptive parameter. If AP is greater than a randomly generated number, Eq. (10) helps to calculate latest location of the agent. Otherwise, Eq. (11) is used to update the position. Eq. (12) shows the adaptive balance probability.

$$\overrightarrow{GW}_{(iter+1)} = \overrightarrow{GW} + f1.rand.((\overrightarrow{GW}_1 - \overrightarrow{GW}) + (\overrightarrow{GW}_2 - \overrightarrow{GW}))/2 \tag{10}$$

$$\overrightarrow{GW}_{(iter+1)} = \overrightarrow{GW} + fl \times rand((\overrightarrow{GW}_1 - \overrightarrow{GW})) \quad (11)$$

$$AP = 1 - (\frac{1.01 \times t^3}{Max\_iteration^3}) \quad (12)$$

In the proposed model, a new update method is adopted. Instead of using Eq. (8), to change the value of 'a', Eq. (13) is used. This helps to improve the overall performance.

$$a = 2 - cos\ (rand(\,)) \times \frac{1}{Max\_iteration} \quad (13)$$

4. EXPERIMENTAL RESULTS

The proposed algorithm is checked in MATALB R2020A software for its efficiency on Intel core™i7 CPU@1.80-GHz with 8 GB of RAM using synthetic data. As it is an NP-Complete problem, the algorithm gives various outputs based on the random numbers. In this setup five jobs and five heterogeneous virtual machines are considered. Table 1 presents the initial population. Table 2 represents the execution time of each job on every VM. Table 3 shows the final allocation of the algorithm.

Table 1. Initial population

Initial population				
Job2	Job3	Job5	Job4	Job1
Job3	Job2	Job1	Job5	Job4
Job4	Job2	Job3	Job5	Job1
Job1	Job3	Job5	Job4	Job2
Job4	Job3	Job1	Job5	Job2
Job3	Job5	Job2	Job4	Job1
Job4	Job5	Job1	Job3	Job2
Job1	Job3	Job2	Job5	Job4
Job3	Job4	Job1	Job2	Job5
Job4	Job3	Job2	Job1	Job5
Job1	Job5	Job3	Job2	Job4
Job3	Job4	Job1	Job2	Job5
Job2	Job4	Job1	Job3	Job5
Job5	Job1	Job2	Job3	Job4
Job4	Job5	Job2	Job1	Job3
Job3	Job2	Job5	Job1	Job4
Job3	Job4	Job5	Job1	Job2
Job1	Job4	Job2	Job5	Job3
Job5	Job3	Job2	Job1	Job4
Job3	Job5	Job1	Job2	Job4

Table 2. Execution times of jobs on VMS

	VM1	VM2	VM3	VM4	VM5
JOB1	28	26	28	24	17
JOB2	17	16	24	25	24
JOB3	28	20	15	25	15
JOB4	28	27	26	20	18
JOB5	21	26	28	24	15

Table 3. Final allocation-GWOCSA

VM1	VM2	VM3	VM4	VM5	Completion time
JOB5	JOB2	JOB3	JOB4	JOB1	89

The best makespan of the given problem after 10 iterations calculated by GWOCSA=124, with GWO=122 and with CSA=128. After iteration 120, the best makespan is 89. GWOCSA algorithm converges quickly. GWO and CSA suffer from slow convergence. Based on the values obtained from results it can be concluded that the proposed algorithm competes with GWO and CSA algorithms. It has proven that at times its efficiency is better than the other two algorithms.

Figure 1 shows the comparison of the GWOCSA with GWO and CSA. From the figure, it can be observed that at the initial stages GWO shows a better solution. CSA algorithm does not converge quickly. However, the proposed algorithm balances the exploration and exploitation and reaches an optimal solution. The makespan after 20 iterations was 123 with GWO,130 with CSA, and 120 with GWOCSA. GWOCSA converges to the optimal solution after 120 iterations. The convergence of the GWO algorithms takes after 160 iterations, and with CSA algorithm happens after 180 iterations.

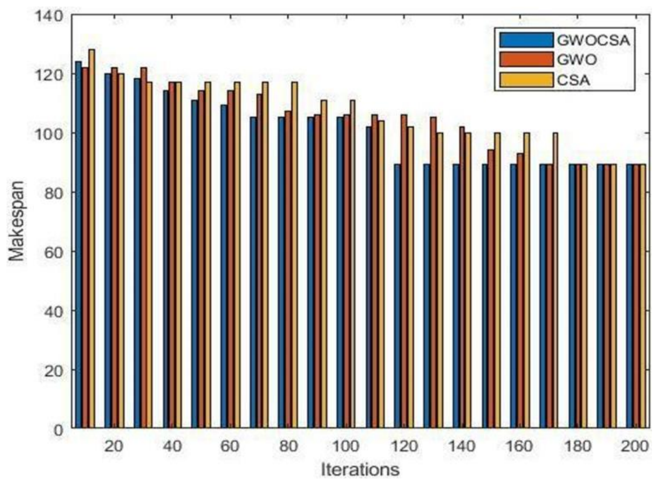

Figure 1. Comparison of makespan

5. CONCLUSION

Present paper discusses the drawbacks of existing meta-heuristic algorithms. GWO and CSA algorithms are considered to enhance the drawbacks of the algorithms when considered individually. Exploration and exploitation are balanced with the help of hybridization. This work can be further enhanced by applying a multi-objective function, in which multiple objectives can be addressed. The proposed algorithm is tested on MATLAB. The proposed algorithm converges quickly when compared with GWO and CSA algorithms. The present work can be extended further. Interested researchers may test on real clouds like OPENSTACK and CLOUDSTACK to check for the efficiency.

REFERENCES

[1] https://cloudacademy.com/blog/surprising-ways-cloud-computing-is-changing-education/.

[2] https://www.healthitoutcomes.com/doc/ways-cloud-computing-is-impacting-healthcare-0001.

[3] Xu, F., Liu, F., Jin, H., Vasilakos, A.V. (2013). Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions. Proceedings of the IEEE, 102(1): 11-31. https://doi.org/10.1109/JPROC.2013.2287711

[4] Mirjalili, S., Mirjalili, S.M., Lewis, A. (2014). Grey wolf optimizer. Advances in Engineering Software, 69: 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[5] Askarzadeh, A. (2016). A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. Computers & Structures, 169: 1-12. https://doi.org/10.1016/j.compstruc.2016.03.001

[6] Arora, S., Singh, H., Sharma, M., Sharma, S., Anand, P. (2019). A new hybrid algorithm based on grey wolf optimization and crow search algorithm for unconstrained function optimization and feature selection. IEEE Access, 7: 26343-26361. https://doi.org/10.1109/ACCESS.2019.2897325

[7] Tawfeek, M.A., El-Sisi, A., Keshk, A.E., Torkey, F.A. (2013). Cloud task scheduling based on ant colony optimization. 2013 8th International Conference on Computer Engineering & Systems (ICCES), Cairo, Egypt, pp. 64-69. https://doi.org/10.1109/ICCES.2013.6707172

[8] Li, K., Xu, G., Zhao, G., Dong, Y., Wang, D. (2011). Cloud task scheduling based on load balancing ant colony optimization. 2011 Sixth Annual Chinagrid Conference, Liaoning, China, pp. 3-9. https://doi.org/10.1109/ChinaGrid.2011.17

[9] Dai, Y., Lou, Y., Lu, X. (2015). A task scheduling algorithm based on genetic algorithm and ant colony optimization algorithm with multi-QoS constraints in cloud computing. 2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics, Hangzhou, China, pp. 428-431. https://doi.org/10.1109/IHMSC.2015.186

[10] Azad, P., Navimipour, N.J. (2017). An energy-aware task scheduling in the cloud computing using a hybrid cultural and ant colony optimization algorithm. International Journal of Cloud Applications and Computing (IJCAC), 7(4): 20-40. https://doi.org/10.4018/IJCAC.2017100102

[11] Chen, X., Long, D. (2019). Task scheduling of cloud computing using integrated particle swarm algorithm and ant colony algorithm. Cluster Computing, 22(2): 2761-2769. https://doi.org/10.1007/s10586-017-1479-y

[12] Kao, Y.T., Zahara, E. (2008). A hybrid genetic algorithm and particle swarm optimization for multimodal functions. Applied Soft Computing, 8(2): 849-857. https://doi.org/10.1016/j.asoc.2007.07.002

[13] Pandey, S., Wu, L., Guru, S.M., Buyya, R. (2010). A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. 2010 24th IEEE International Conference on Advanced Information Networking and Applications, pp. 400-407. https://doi.org/10.1109/AINA.2010.31

[14] Awad, A.I., El-Hefnawy, N.A., Abdel_kader, H.M. (2015). Enhanced particle swarm optimization for task scheduling in cloud computing environments. Procedia Computer Science, 65: 920-929. https://doi.org/10.1016/j.procs.2015.09.064

[15] Verma, A., Kaushal, S. (2017). A hybrid multi-objective particle swarm optimization for scientific workflow scheduling. Parallel Computing, 62: 1-19. https://doi.org/10.1016/j.parco.2017.01.002

[16] Gill, S.S., Buyya, R., Chana, I., Singh, M., Abraham, A. (2018). BULLET: Particle swarm optimization based scheduling technique for provisioned cloud resources. Journal of Network and Systems Management, 26(2): 361-400. https://doi.org/10.1007/s10922-017-9419-y

[17] Li, M., Du, W., Nian, F. (2014). An adaptive particle swarm optimization algorithm based on directed weighted complex network. Mathematical Problems in Engineering, 2014: 434972. https://doi.org/10.1155/2014/434972

[18] Zhao, C., Zhang, S., Liu, Q., Xie, J., Hu, J. (2009). Independent tasks scheduling based on genetic algorithm in cloud computing. 2009 5th International Conference on Wireless Communications, Networking and Mobile Computing, Beijing, China, pp. 1-4. https://doi.org/10.1109/WICOM.2009.5301850

[19] Aziza, H., Krichen, S. (2018). Bi-objective decision support system for task-scheduling based on genetic algorithm in cloud computing. Computing, 100(2): 65-91. https://doi.org/10.1007/s00607-017-0566-5

[20] Ibrahim, H., Aburukba, R.O., El-Fakih, K. (2018). An integer linear programming model and adaptive genetic algorithm approach to minimize energy consumption of cloud computing data centers. Computers & Electrical Engineering, 67: 551-565. https://doi.org/10.1016/j.compeleceng.2018.02.028

[21] Zhou, Z., Li, F., Zhu, H., Xie, H., Abawajy, J.H., Chowdhury, M.U. (2020). An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments. Neural Computing and Applications, 32(6): 1531-1541. https://doi.org/10.1007/s00521-019-04119-7

[22] Wang, J.S., Li, S.X. (2019). An improved grey wolf optimizer based on differential evolution and elimination mechanism. Scientific Reports, 9(1): 7181. https://doi.org/10.1038/s41598-019-43546-3