International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Multi-Job Scheduling and Optimization Model for Cloud Computing

Dzikunu Andrews Dodzi Kobla[1*], Kenneth Owusu[2], Amos Peprah[3], Koukoyi Ebenezer[4]

[1]Zhejiang Normal University, Jinhua, Zhejiang, CN.
[2] Hubei University of Technology, CN,
[3]Nanjing University ofPosts and Telecommunications Nanjing, CN.
[4]Southwest University of Science and Technology Mianyang, CN.

* Corresponding Author: adzikunu[at]zjnu.edu.cn

Abstract: ***Among the most frequently encountered problems in the cloud computing system seems to be independent job scheduling. The Grey Wolf Optimization and Artificial Bee Colony (GWO-ABC) approach is a brand-new hybrid optimization technique that is proposed in this work. The GWO-ABC is used to enhance multi-job scheduling inside a cloud computing system. It imitates the hunting and grouping behaviour of wolves and ants. The proposed multi-objective model seems to be the foundation of the GWO-ABC scheduler. In aspects of cost, degree of imbalance, makespan, resource utilisation, and energy usage, the experiment outcomes on the trialed data demonstrated that the GWO-ABC scheduler outperforms the outcomes of the Round Robin (RR) method, conventional Whale Optimization Algorithm (WOA), and Vocalisation of Humpback WOA (VWOA). The results of the simulation clearly show that the GWO-ABC method is better than the algorithms that are currently being used.***

Keywords: Optimization, Job Scheduling, Cloud Computing, Bee Colony, Exploitation

1. Introduction

The term "Cloud" refers to a network of linked computers that contains multiple unified computing resources. Cloud computing, which aims to supply the necessary virtualized capabilities for back-end operations in numerous multi-tier functionalities, has nowadays become an essential component of information technologies. Additionally, cloud computing can offer access to a variety of applications, as well as its Data Centers (DCs) may offer ample storage for keeping the cloud customer's vital data as well as some security solutions. Cloud computing, which aims to supply the necessary virtualized capabilities for back-end operations in numerous multi-tier functionalities, has nowadays become an essential component of information technologies [1]. The IT industry is changing and becoming more uniform thanks to cloud computing. It can arrange on-demand exposure to a configurable commodities pool that has been shared and available to all users, with little need for cloud provider collaboration or administration [2]. This technology has some gains, including the ability to enhance the cost, time, storage gains, and stack adjusting in the marketplace. With this invention, all apps can continue to operate on a virtual system, and all services are shared among the VMs [3]. Each application is unique. Cloud computing job scheduling requires an effective method. Scheduling systems can be divided into heuristic and meta-heuristic scheduling strategies [4-6], as demonstrated in figure 1. Single- and multi-objective scheduling systems are further subcategories of meta-heuristic techniques [7-8]. Cloud Service Providers (CSPs) and cloud customers, who frequently have competing goals in the workflow and job scheduling issues, seem to be the two primary players in the cloud computing industry. A novel hybrid Grey Wolf Optimization and Artificial Bee Colony (GWO-ABC) technique has been put forth in this study as a solution to this issue.

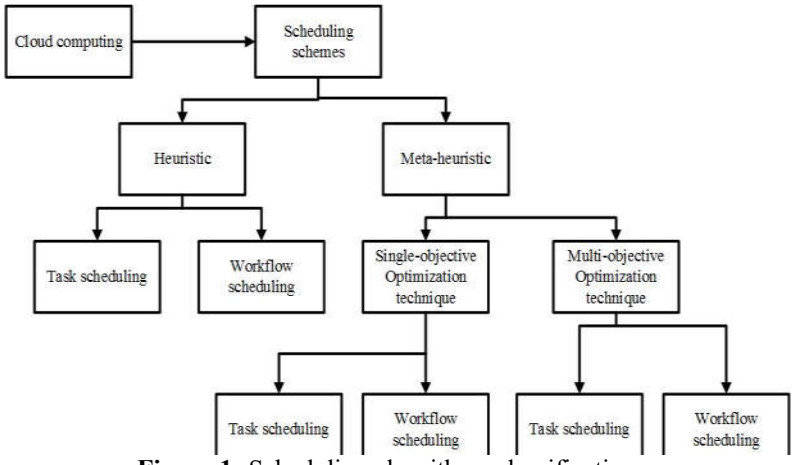

Figure 1: Scheduling algorithms classification

1.1. Workflow and Job Scheduling

A workflow could be described using a Directed Acyclic Graph (DAG), where nodes represent edges and tasks indicate priority. Any process can be data-, computation-, or communication-intensive. There are two types of processes: non-deterministic and deterministic. Earlier, job and input/output interactions must be understood. The latter involves workflow design during operation. Workflow and job scheduling problems include assigning the right virtual machine (VM) to each task.

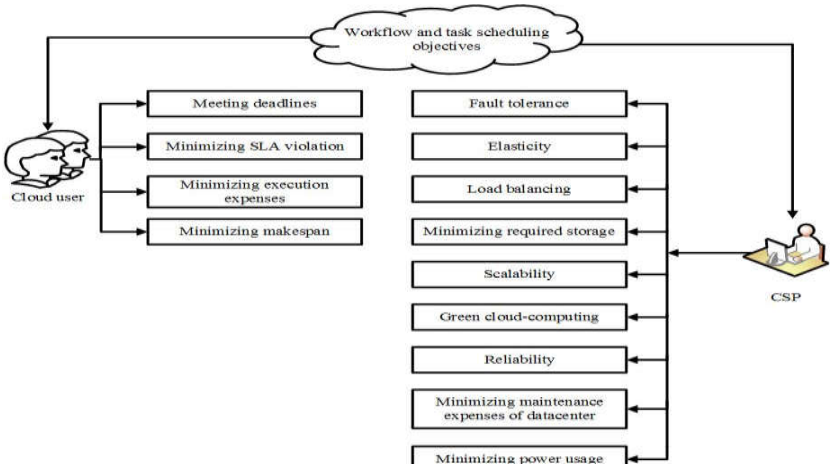

Figure 2: Workflow and job scheduling objectives

Several independent or working clouds may make up a multi-cloud system. By distributing the scheduling queries over various clouds, multi-cloud setups typically try to reduce dependence on any one CSP and boost effectiveness and throughput.

2. Related Works

It can be difficult to select a VM from a variety of data hubs with a variety of characteristics like lowered energy usage, ideal response time, cost minimization, etc. in a cloud Infrastructure as a Service (IaaS) setting due to the diversification of offerings concerning resources and innovation. As a result, V. Karunakaran [9] presented a hybrid method predicated on a non-dominated sorting genetic and gravitational search algorithm (NSGA and GSA) that simplifies VM choice for scheduling systems.

Qi Qi et al. [10] firstly developed and published a Multi-task Deep Reinforcement Learning (DRL) technique for scalable parallel Task Scheduling (MDTS). To address multi-objective work scheduling issues in cloud computing systems, LaithAbualigah and Ali Diabat[11] have developed a unique hybrid antlion optimization technique with elite-based differential evolution. This solution was dubbed MALO. Thus, in an attempt to address the task scheduling issue, Abdullah Alzaqebah et al. [12] used the Grey Wolf Optimization (GWO) method with alterations to the fitness value to manage multiple objectives in a unique fitness; the duration and expense seem to be the criteria considered in the fitness. As a result, Raja Masadeh et al. [13] proposed the vocalisation of humpback whale optimization algorithm (VWOA), a novel metaheuristic optimization technique. Thus, SaharSaeedi et al. [14] have suggested a method to resolve the workflow scheduling issue using the Improved MaO Particle Swarm Optimization (I_MaOPSO) technique, which takes into account four competing goals, including reliability maximisation and expense, time, and energy usage reductions.

As a result, Sarah E. Shukri et al. [15] suggest an enhanced variant of the Multi-Verse Optimizer (EMVO) as a better job scheduler in this domain. Lei Shi et al. [16] have thus researched the issue of multi-job-linked job scheduling to reduce the jobs' completion time. As a result, S. Velliangiri et al. [17] have presented Hybrid Electro Search with a Genetic Algorithm (HESGA) to enhance task scheduling behaviour by taking into account factors like load balancing, and resource consumption, makespan, and multi-cloud expense. The proliferation and growth of technological devices (smartphones, laptops, and tablets), along with rapid internet connectivity, have made the cloud computing concept reasonable and reduced IT complexity [18]. It's crucial to ensure the security and dependability of these systems for safety-critical tasks. Qing-Hua Zhu et al. [19] suggest an innovative scheduling technique called matching and multi-round allocation (MMA) to maximise the total cost and makespan for all contributed tasks responsive to security and performance requirements to overcome trust restrictions in a diverse multi-cloud environment.

3. 3. Multi-Objective Optimization Problems (MOPs)

Multi-objective optimization involves competing goals (MOPs). Because the purposes often compete, a collection of tradeoff options, referred to as non-dominated alternatives for MOPs, may be obtained.

3.3.1 Cloud Model

This research focuses on job scheduling applications; moreover, this research presumes that VMs had enough memory to carry out the job scheduling based on the profiling findings for memory usage they acquired in their task as well as the VM kinds provided by Amazon EC2. Presumed that the processing power in regards to floating point operations per second (FLOPS) for each type of VM is either known or can be guessed. This system uses this data to determine how long a task will take to

complete on a specific VM. Thus, the execution time duration is computed by Eqn. (1).

$$E_{t_i}^{VM_i} = {}^{S_{t_i}}/_{P_{VM_i} \times (1 - PD_{VM_i})} \quad (1)$$

Where, $E_{t_i}^{VM_i}$ indicates the execution time duration, S_{t_i} represents the task size in flop unit and PD_{VM_i} indicates VM's performance degradation percentage.In addition, the time taken for the information to be transmitted from the parent task $t_i$to the child task $t_k$is computed by Eqn. (2).

$$TT_{t_{ik}} = {}^{DS_{t_i}^{out}}/_{\gamma} \quad (2)$$

Where $TT_{t_{ik}}$ represents the transmission time from $t_i$to t_k, $DS_{t_i}^{out}$ indicates the output data size, and γ is bandwidth.
This study assumes that a data center handles the complete job scheduling procedure. Therefore, the bandwidth on VMs was the same, and there is no transmission time between the two job scheduling processes running on the same VM. Eqn. (3) computed the total processing time.

$$Pt_{t_i}^{VM_i} = E_{t_i}^{VM_i} + \left[\sum_1^j TT_{t_{ik}} \times S_j\right] \quad (3)$$

Where, $Pt_{t_i}^{VM_i}$ indicates total processing time, j represents the task's number of edges, and S_j task size with the number of edges.

3.1. Multi-job Scheduling

The fitness variables are a key component of the job scheduling approach. In this study, the fitness function was dependent on four factors: round trip latency, task weight, energy use, and resource usage. Moreover, the fitness function at greater value is expressed in Eqn. (4).

$$F = 0.25[T_W + R_U + (1 - E_d) + (1 - T_L)] \quad (4)$$

Certain tasks in a CC system call for a lot of CPU resources, while others call for fewer CPU assets and higher storage.$R_U$comprises two variables: Memory Cost (MCost), and CPU Cost (Ccost), which are described as Eqn. (5).

$$R_U = 0.5[CCost\ (i) + MCost(i)] \quad (5)$$

$$CCost\ (i) = C_i \times C_t \times t_{ix} \times C_{base} \quad (6)$$

Here, when a resource has been used in its lowest utilization level to maintain the system running and ready to process inbound requests, the basic cost was represented by C_{base} , t_{ix} represents the makespan in which the task i is performed in VMi , C_i represents the VMi cost, and C_t indicated the expense of CPU transfer to complete the necessary tasks. The cost and MCost are computed by Eqn. (6) and (7), correspondingly.

$$MCost\ (i) = M_i \times M_t \times t_{ix} \times M_{base} \quad (7)$$

Where M_{base} denotes the memory's base cost, M_i indicates memory costs, t_{ix} indicates the task time duration running in VMi, and M_t denotes the constant value, which indicates the cost related to memory transmission. The amount of time it takes to compute and how much energy it uses are related.As a result, Eqn (8) defines the amount of energy (ECost (i)) needed by Task time (Ti) inVMi.

$$ECost\ (i) = C_{ix} \times P_e^i \quad (8)$$

Where, C_{ix} represents energy consumption time, and power is indicated as P_e^i , which is used during implementation Ti . Moreover, VMi overall consumed energy to complete all tasks (TC(i)) is computed by Eqn. (9).

$$TC(i) = \sum_{i=1}^{VM} ECost\ (i) \quad (9)$$

Round trip time (RTT) denotes the entire step's latency (L), which includes the commencement of reception, transmission, and response-waiting time. Latency is computed by Eqn. (10).

$$L = E_{ET} + E_{TT} + (2 \times d) \quad (10)$$

Where E_{ET} represents expected execution time, E_{TT} represents expected transmission time, and d indicates delay. The four essential components of E_{ET} our task size, are CPU speed, RAM speed, and bandwidth. Moreover, E_{ET} and E_{TT} are computed by Eqn. (11) and (12), respectively.

$$E_{ET} = {}^{T_s + S_{RAM} + S_{CPU}}/_{Bandwidth} \quad (11)$$

Where, T_s indicates task size, S_{CPU} represents CPU speed, and S_{RAM} indicates RAM speed.

$$E_{TT} = {}^{T_s}/_{B_R} \quad (12)$$

Here, B_R indicates Bit rate.The final task weight (T_W) seems to be a significant variable that is impacted by task prioritizing. Task weight and user type are the two major characteristics of this property. User types are a representation of the user privilege classes. Additionally, it includes a variety of categories (like class W, class X, class R, and class T). To employ the resources accessible, any task seems to have a priority weight which assigns it a medium, low, or high priority as shown in table 1.

Table 1: Tasks weight

Classes	W	X	R	T
Weight	0.4	0.3	0.2	0.1
Priority	U	H	M	L

In table.1, U indicates urgent, H represents high, M represents medium, and L indicates large. The suggested mathematical model uses precedence weight factors of 0.4,

0.3, 0.2, and 0.1. These numbers have been selected so that their total equals one. The final task weight (T_W) is expressed in Eqn. (13).

$$T_W = U_T \times W_t \tag{13}$$

Where U_T indicates user type, and W_t indicates task weight.

4. Proposed Approach

4.1. Grey Wolf Optimization (GWO)

GWO seems to be a technique for bionic optimization. It imitates grey wolf hunting behaviour, with a definite division of work and mutual collaboration. Grey wolves usually live in packs of 5-12 individuals and have a strict dominating hierarchy predicated on wolf leadership abilities. For hunting, the enclosing of prey approach is used. For iteration i, the mathematical framework for this method is shown below Eqn. (14) and (15).

$$\vec{F} = \left|\vec{Y} \times \overrightarrow{Q_p}(i) - \vec{Q}(i)\right| \tag{14}$$

$$\vec{Q}(i+1) = \overrightarrow{Q_p}(i) - \vec{D}.\vec{F} \tag{15}$$

Here, $\vec{D}$ and $\vec{Y}$ are coefficient vectors, which is described as $\vec{D} = 2\vec{d}.\overrightarrow{v_1} - \vec{d}$ an $\overrightarrow{Y} = 2.\overrightarrow{v_2}$. Where, the random vectors $\overrightarrow{v_1}, \overrightarrow{v_2}\ \varepsilon\ (0,1)$ and $\overrightarrow{d} = d_1\left(1 - {}^{i}/_{maxi}\right)$, linearly decrease from d_1 to zero; d_1 the value was set as 2 in actual GWO. Moreover, $maxi$ represents a maximum number of iterations. The GWO's hunting process has been headed by three finest solutions i.e., $\propto, \beta\gamma$ wolves. Thus, these 3 leading solution positions have been saved in the pack and the remaining ω wolves update their positions predicated on them. This position updating technique's mathematical model is represented in Eqn. (16).

$$\vec{Q}(i+1) = {\left(\overrightarrow{Q_1} + \overrightarrow{Q_2} + \overrightarrow{Q_3}\right)}/{3} \tag{16}$$

Where, $\overrightarrow{Q_1}, \overrightarrow{Q_2}, and\ \overrightarrow{Q_3}$ are computed by Eqn. (17)

$$\begin{aligned} \overrightarrow{Q_1} &= \overrightarrow{Q_\propto}(i) - \overrightarrow{D_1}.\overrightarrow{F_\propto} \\ \overrightarrow{Q_2} &= \overrightarrow{Q_\beta}(i) - \overrightarrow{D_1}.\overrightarrow{F_\beta} \\ \overrightarrow{Q_3} &= \overrightarrow{Q_\gamma}(i) - \overrightarrow{D_1}.\overrightarrow{F_\gamma} \end{aligned} \tag{17}$$

Where, $\overrightarrow{F_\propto}$, $\overrightarrow{F_\beta}$, and $\overrightarrow{F_\gamma}$ are computed by Eqn. (18)

$$\begin{aligned} \overrightarrow{F_\propto} &= \left|\overrightarrow{Y_1} \times \overrightarrow{Q_\propto}(i) - \vec{Q}\right| \\ \overrightarrow{F_\beta} &= \left|\overrightarrow{Y_2} \times \overrightarrow{Q_\beta}(i) - \vec{Q}\right| \\ \overrightarrow{F_\gamma} &= \left|\overrightarrow{Y_3} \times \overrightarrow{Q_\gamma}(i) - \vec{Q}\right| \end{aligned} \tag{18}$$

4.2. Hybrid GWO-Artificial Bee Colony (ABC) Optimization

The artificial bee colony (ABC) seems to be probably of the newest algorithms, inspired by honey bees' clever foraging behavior. In ABC technique, the colony of artificial bee is made up of three types of bees: bystanders, employed bees, and scouts. The search technique for both engaged and an observer bee in ABC is driven by upgrading a random component in the solutions vector with another solution vector as shown in Eqn. (19).

$$b_{k,l} = \vec{Q}(i+1)_{k,l} + \varphi_{k,l}\left(\vec{Q}(i+1)_{k,l} - q_{i,j}\right) \tag{19}$$

Where, b_{kl} represents the new solution attained by mutating l^{th} the dimension parameter of two distinct solutions in a pack and φ_{kl} represents the random number, which varies between -1 and 1.

Because GWO makes use of the finest solutions in the organizational hierarchy, combining it with ABC will result in a powerful algorithm that combines the benefits of both. The hybrid GWO-ABC technique's abstraction is shown in algorithm.1 and figure 3.

Algorithm.1: Abstraction of GWO-ABC

1. Initialize grey wolf population $\vec{Q}_p = (q_1, q_2, \dots, q_n)$, where, $(p \in N)$
2. Initialize $d, \vec{D}, \vec{Y}$ and $i = 1$.
3. Compute each search agent's fitness $f(Q_a)$, where $(a \in N)$
4. $\overrightarrow{Q_\alpha}$ = finest search agent
5. $\overrightarrow{Q_\beta}$ = 2nd finest search agent
6. $\overrightarrow{Q_\gamma}$ = 3rd finest search agent
7. while $(i < Max - iterations)$ do
8. **for** every search agent **do**
9. Upgrade present search agent's position by Eqn. (3)
10. **end for**
11. Upgrade: $d, \vec{D}, and\ \vec{Y}$
12. Compute: all search agent's fitness
13. Upgrade: ABC's search technique
14. Update: search agent's fitness $\vec{Q}(i+1)$ in Eqn. (6)
15. **end while**
16. **else**, return Q_α

The following aspects demonstrate the unique hybrid optimization algorithm (GWO-ABC)'s ability to resolve optimization issues:

1. Throughout iterations, GWO-ABC can keep the finest solutions because of the social structure.
2. The proposed hunting method makes use of the potential answers to determine where the prey is located.
3. The h and F parameters values are used to aid GWO-ABC in transitioning between exploitation and exploration without any problems.
4. Iterations are split equally between the exploitation and exploration phases.
5. m and q are the two key GWO-ABC variables that need to be changed.

4.3. GWO-ABC Based Job Scheduling

These limits encourage algorithmic exploration. GWO-ABC assigned tasks to VMs. The iteration technique was continued based on the present solution using GWO-ABC, which starts with the solution's random population and deems the present solution the finest alternative. GWO-stages ABC were:

- **Initialization Phase:** The search agent's population $\vec{Q}_p = (q_1, q_2, \dots, q_n)$ was generated randomly in this phase.
- **Fitness Computation Phase:** The fitness function has been computed using a function (4). Predicated on assessment, the finest search agent $\overrightarrow{Q *_p}$ has been picked.
- **GWO Phase:** Following the generation of the original population, the methodology moves forward according to the standard GWO and upgrades its variables as well as the search agent's locations.
- **Encircling Prey Phase:** The prey's position is presumed to be fixed throughout this period. Depending on the most recent best agent, which was depicted as in Eqn. (20), other wolves (search agents) change their positions.

$$\vec{F} = \left|\vec{Y} \times \overrightarrow{Q *_p}(i) - \vec{Q}(i)\right| \quad (20)$$

- **ABC Phase:** As was previously mentioned, in ABC, bees (both active and passive observers) disseminate data among the possible solutions inside the pack as well as alter existing solutions.
- By picking random adjacent solutions as well as a place for information sharing, improves exploration opportunities.
- **Exploitation Phase:** The algorithm determines a search agent's new position during this phase. The spiral technique using Eqn (21) updates the agent's position by the use of the spiral updating position technique.

$$\vec{Q}(i+1) = \left|\overrightarrow{Q_p}(i) - \vec{Q}(i)\right| \times cos(2\pi x) \times e^{wx} + \overrightarrow{Q_p}(i) \quad (21)$$

Where w is constant for evaluating logarithmic spiral's shape, and x the value is (-1,1)

- **Exploration Phase:** Utilizing Eqn (22) and (23) an arbitrarily chosen search agent, the search agent's position will be upgraded.

$$\vec{F} = \left|\vec{Y} \times \overrightarrow{Q_{rand}} - \vec{Q}\right| \quad (22)$$

Where $\overrightarrow{Q_{rand}}$ indicates the random position vector.

$$\vec{Q}(i+1) = \overrightarrow{Q_{rand}} - \vec{D}.\vec{F} \quad (23)$$

4.4. Experimental and Evaluation

The effectiveness of the proposed GWO-ABC for scheduling separate tasks is empirically assessed. A computer system with an Intel Core i-7 processor, Windows 10 OS, and 16 GB of RAM has been used for the experimentation. Regarding energy consumption, degree of imbalance, makespan, cost, and resource usage, the GWO-ABC outcomes are contrasted with other existing methods like WOA, VWOA, and RR [13] to determine the presented model's effectiveness.

These metrics were employed for evaluating the presented GWO-ABC model performance in job scheduling.

1. **Energy usage:** It seems to be the term for how a process uses resources as well as the CPU to determine how much power was consumed.
2. **Makespan:** It displays the overall execution time needed to complete all separate tasks.
3. **Resource usage:** The consumption of resources should be increased because it benefits cloud computing service providers.
4. **Cost:** Cost is an indicator of the expense incurred during task execution on particular VMs.
5. **Degree of imbalance (DI):** Utilizing Eqn (24), DI calculates the imbalance between the VMs.

$$DI = {(T_{max} - T_{min})}/{T_{avg}} \quad (24)$$

Where T_{max} represents the maximum execution time, T_{min} denotes the minimum execution time, and T_{avg} indicates the average execution time.

4.5. Results

In this research, the GWO-ABC scheduling method's performance is compared to that of the conventional WOA, VWOA, and RR scheduling techniques over a range of independent jobs (100–500) with varied numbers of chosen VMs (8, and 16). The configuration consists of two hosts, two data centres, 16 VMs, and various job counts (100, 200, 300, 400, and 500). Every case is run 30 times. Each instance's mean is calculated. Meta-heuristic optimization methods include VWOA and WOA. Thus, the GWO-ABC percentage improvement over other approaches for 16VMs was computed using Eqn. (25). The exploration stage refers to the process of searching as widely as possible for the search space's potential areas.

$$Improvement = \left(1 - {\sum Makespan_{GWO-ABC}}/{\sum Makespan_{other\ methods}}\right) \times 100 \quad (25)$$

The capacity to conduct local searches around the potential places found during the exploratory phase was what the exploitation stage denotes. Because of the helpers who expand the search areas and force wolves to join their groups, GWO-ABC possesses great exploitation and exploration abilities. Additionally, it features stochastic operations that enable it to scan the search spaces broadly and arbitrarily.

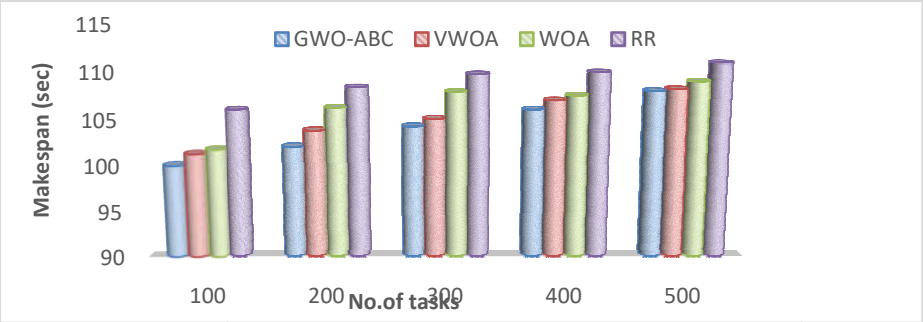

Figure 4: Tasks vs makespan comparison on 8 VMs.

Figure 4 and figure5 display the simulation findings. These numbers demonstrate that the GWO-ABC performs better than standard WOA, VWOA, and RR in regards of makespan because of its capacity for exploitation and exploration. The exploration stage depends on the limitless amount of wolves and bees present in each cluster. GWO-ABC can give good exploitation, exploration, and localized optimal aversion with the use of the variable$|\bar{A}|$. While the RR method performs worst due to its reliance on temporal quantum size and lack of consideration for resource and task data. If the time quantum was large, this results in a long waiting period. This results in a long makespan.

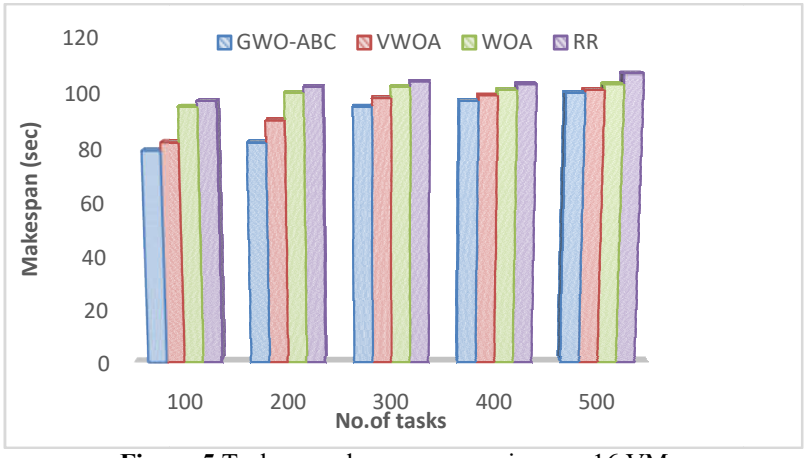

Figure 5. Tasks vsmakespan comparison on 16 VMs.

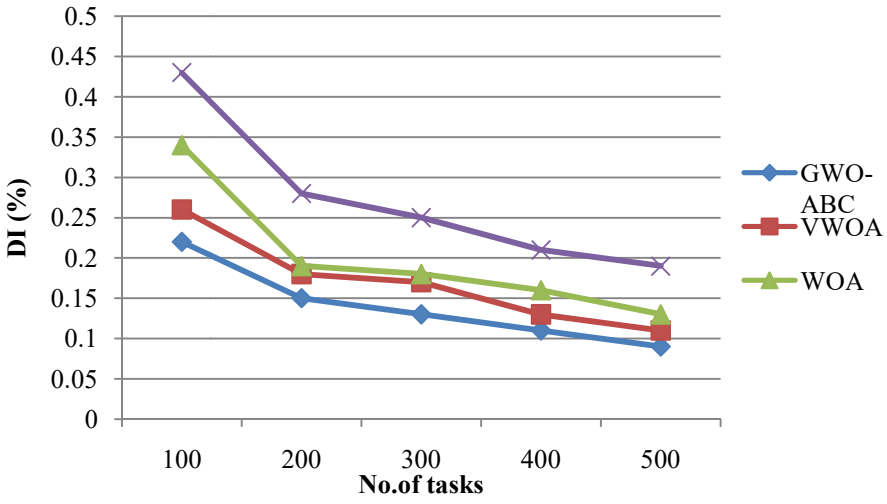

Figure 6: Mean DI on 16 VMs.

Figure 6 shows the degree of imbalance (DI) for the 4 approaches and 100–500 workloads on 16 VMs. GWO-ABC has less overall imbalance than WOA, RR, and VWOA. The technique assigns workloads with larger weights to available and efficient VMs based on task duration, priority, and VM capability. Every VM's execution time would decrease. GWO-ABC utilizes less energy than WOA, RR, and VWOA. GWO-ABC utilizes 28, 31, and 84% less energy than WOA, RR, and VWOA. Energy use depends on execution time. GWO-ABC utilized less energy due to its short lifespan. RR's long makespan required more energy.

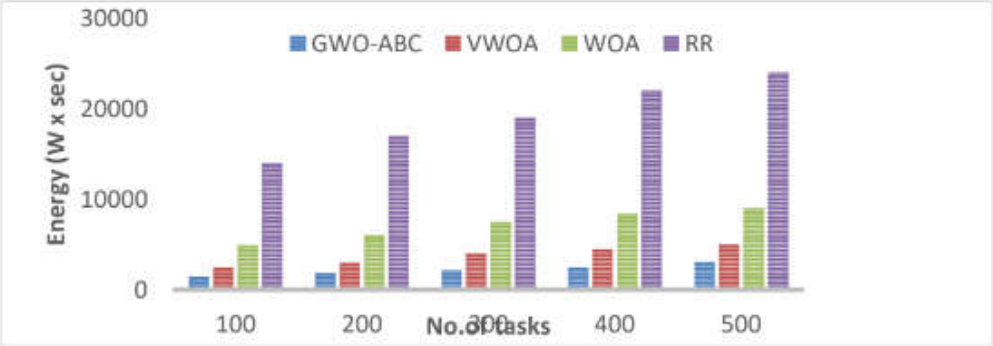

Figure 7: Outcome of energy on 16 VMs.

Table 2 shows 100-500 tasks on 8 and 16 VMs and their scheduling costs. Multiple tasks affect efficiency and resources. It gets harder to find a global optimum solution when the number of tasks is huge. In contrast, when there are numerous resources, it is easy to discover a global optimum solution since there are many tasks and few resources, which raises scheduling costs, waiting time and makespan. GWO-work ABC's scheduling execution costs are 20%, 22%, and 27% lower than WOA, VWOA, and RR.

Table 2: Resource utilization on 16 VMs.

No. of tasks	Resource usage (%)			
	GWO-ABC	VWOA	WOA	RR
100	76	72	70	55
200	81	78	71	52
300	85	82	73	45
400	90	88	77	42
500	93	90	79	39

When contrasted to WOA, VWOA, and RR methods, the average resource usage of GWO-ABC was raised by 8%, 23%, and 35%, correspondingly. Because of its high levels of exploitation and exploration, GWO-ABC used more resources. In comparison to the RR method, which lacks data about resources and tasks, it also has current data about all resources and tasks. The RR method sends tasks to VMs in a loop and does not take into account selecting the best VM for the task.

5. Conclusion

Job scheduling for cloud computing has benefited from the use of numerous meta-heuristic and heuristic methods. The hunting and grouping behaviours of wolves and ants are imitated in this work by the novel hybrid optimization method GWO-ABC. According to this approach, each cluster should have three of the best solutions. Therefore, there have been three localized optimal solution solutions that are the finest. As a result, the first existing solution was regarded as the finest one, and the searching agent has been used to identify the overall ideal solution. The adoption of the multi-objective numerical model enhances the scheduling of various jobs in a cloud computing system. The main goals of GWO-ABC are to reduce costs, manufacturing time, and energy use, and to increase resource utilisation. The algorithm has been assessed using the CloudSim tool. According to simulation outcomes, the proposed GWO-ABC method outperforms the current WOA, VWOA, and RR methods regarding makespan, DI, cost, energy usage, and resource usage. Additionally, every cluster's 3 top solutions can optimally schedule new solutions to be produced. This research project will be expanded for dependent activities in the future. Additionally, the suggested GWO-ABC could be modified to address different optimization issues.

References

[1] Cheng CL, Li J, and Wang Y 2015 An Energy-Saving Task Scheduling Strategy Based On Vacation Queuing Theory In Cloud ComputingTsinghua Sci. Technol. **20**(1) pp 28-39doi: 10.1109/TST.2015.7040511.

[2] B. Keshanchi, A. Souri, and N. J. Navimipour, "An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: Formal verification, simulation, and statistical testing," J. Syst. Softw., vol. 124, pp. 1–21, Feb. 2017, doi: 10.1016/j.jss.2016.07.006.

[3] G. Lovász, F. Niedermeier, and H. de Meer, "Performance Tradeoffs of Energy-Aware Virtual Machine Consolidation," Clust. Comput, vol. 16, no. 3, pp. 481–496, Sep. 2013, doi: 10.1007/s10586-012-0214-y.

[4] S. Midya, A. Roy, K. Majumder, and S. Phadikar, "Multi-Objective Optimization Technique For Resource Allocation And Task Scheduling In Vehicular Cloud Architecture: A Hybrid Adaptive Nature-Inspired Approach," J. Netw. Comput. Appl., vol. 103, pp. 58–84, Feb. 2018, doi: 10.1016/j.jnca.2017.11.016.

[5] H. Y. Shishido, J. C. Estrella, C. F. M. Toledo, and M. S. Arantes, "Genetic-based algorithms applied to a workflow scheduling algorithm with security and deadline constraints in clouds," Comput. Electr. Eng., vol. 69, pp. 378–394, Jul. 2018, doi: 10.1016/j.compeleceng.2017.12.004.

[6] M. Abdullahi, M. A. Ngadi, S. I. Dishing, S. M. Abdulhamid, and B. I. Ahmad, "An Efficient Symbiotic Organisms Search Algorithm With Chaotic Optimization Strategy For Multi-Objective Task Scheduling Problems In Cloud Computing Environment," J. New. Comput. Appl., vol. 133, pp. 60–74, May 2019, doi: 10.1016/j.jnca.2019.02.005.

[7] F. Ebadifard and S. M. Babamir, "Optimizing Multi-Objective Based Workflow Scheduling In Cloud

Computing Using Black Hole Algorithm," in 2017 3rd International Conference on Web Research (ICWR), Tehran, Iran, Apr. 2017, pp. 102–108. doi: 10.1109/ICWR.2017.7959313.
[8] I. Pietri, Y. Chronis, and Y. Ioannidis, "Multi-Objective Optimization Of Scheduling Dataflows On Heterogeneous Cloud Resources," in 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, Dec. 2017, pp. 361–368. doi: 10.1109/BigData.2017.8257946.
[9] V. Karunakaran, "A Stochastic Development Of Cloud Computing Based Task Scheduling Algorithm," J. Soft Comput. Paradigm JSCP, vol. 1, no. 01, pp. 41–48, 2019.
[10] Q. Qi et al., "Scalable Parallel Task Scheduling for Autonomous Driving Using Multi-Task Deep Reinforcement Learning," IEEE Trans. Veh. Technol., vol. 69, no. 11, pp. 13861–13874, Nov. 2020, doi: 10.1109/TVT.2020.3029864.
[11] L. Abualigah and A. Diabat, "A Novel Hybrid Antlion Optimization Algorithm For Multi-Objective Task Scheduling Problems In Cloud Computing Environments," Clust. Comput., vol. 24, no. 1, pp. 205–223, Mar. 2021, doi: 10.1007/s10586-020-03075-5.
[12] A. Alzaqebah, R. Al-Sayyed, and R. Masadeh, "Task Scheduling based on Modified Grey Wolf Optimizer in Cloud Computing Environment," in 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS), Amman, Jordan, Oct. 2019, pp. 1–6. doi: 10.1109/ICTCS.2019.8923071.
[13] R. Masadeh, A. Sharieh, and B. Mahafzah, "Humpback whale optimization algorithm based on vocal behavior for task scheduling in cloud computing," Int. J. Adv. Sci. Technol., vol. 13, no. 3, pp. 121–140, 2019.
[14] S. Saeedi, R. Khorsand, S. Ghandi Bidgoli, and M. Ramezanpour, "Improved Many-Objective Particle Swarm Optimization Algorithm For Scientific Workflow Scheduling In Cloud Computing," Comput. Ind. Eng., vol. 147, p. 106649, Sep. 2020, doi: 10.1016/j.cie.2020.106649.
[15] S. E. Shukri, R. Al-Sayyed, A. Hudaib, and S. Mirjalili, "Enhanced multi-verse optimizer for task scheduling in cloud computing environments," Expert Syst. Appl., vol. 168, p. 114230, Apr. 2021, doi: 10.1016/j.eswa.2020.114230.
[16] L. Shi et al., "Multijob Associated Task Scheduling for Cloud Computing Based on Task Duplication and Insertion," Wirel. Commun. Mob. Comput., vol. 2021, pp. 1–13, Apr. 2021, doi: 10.1155/2021/6631752.
[17] S. Velliangiri, P. Karthikeyan, V. M. Arul Xavier, and D. Baswaraj, "Hybrid electro search with genetic algorithm for task scheduling in cloud computing," Ain Shams Eng. J., vol. 12, no. 1, pp. 631–639, Mar. 2021, doi: 10.1016/j.asej.2020.07.003.
[18] Derrick, Yeboah, George Kofi Agordzo and Lü Ye. "New Paradigm of Computing (Distributed Computing)." (2020).
[19] Q.-H. Zhu, H. Tang, J.-J. Huang, and Y. Hou, "Task Scheduling for Multi-Cloud Computing Subject to Security and Reliability Constraints," IEEECAA J. Autom. Sin, vol. 8, no. 4, pp. 848–865, Apr. 2021, doi: 10.1109/JAS.2021.1003934.