ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

HYBRID HENRY GAS-HARRIS HAWKS MODIFIED-OPPOSITION ALGORITHM FOR TASK SCHEDULING IN CLOUD COMPUTING

NORA OMRAN ALKAAM[1, 2], ABU BAKAR MD. SULTAN[1], MASNIDA HUSSIN[1], AND KHAIRONI YATIM SHARIF[1]

[1]Dept. of Software Engineering and Information System, Universiti Putra Malaysia, 43400, Malaysia
[2]Iraqi Ministry of Higher Education and Scientific Research, Iraq

E-mail: [1]nora.omran20@gmail.com, [1]gs60414@student.upm.edu.my

ABSTRACT

Objectives: Cloud computing environments allow users to remotely access computational resources and data computing services online. Task scheduling requires the development of reliable and efficient methods for mapping tasks to resources, making it an essential component of cloud computing. Effective task scheduling is critical for increasing operational efficiency since it entails carefully assigning tasks to resources to ensure optimal performance. This accurate coordination not only increases productivity, but it also optimizes resource allocation. Cloud computing solutions can improve overall system performance, reduce processing times, and increase efficiency by enhancing job scheduling.
Methods: This study introduced a (Henry Gas-Harris Hawks Modified-Opposition) (HGHHM) algorithm to enhance the Henry Gas Solubility algorithm based on two components: Harris Hawks Optimization (HHO) and a modified comprehensive opposition-based learning (MOBL). This proposed HGHHM algorithm utilized the HHO method as a local search strategy to enhance the quality of approved solutions. While, MOBL improves the less effective solutions by carefully calculating their opposite equivalents and wisely choosing the most advantageous option. This approach facilitated the enhancement of suboptimal solutions, leading to an overall enhancement in the efficiency of the selected techniques.
Results: CloudSim was used to test the HGHHM algorithm on HPC2N dataset. Thus, the suggested HGHHM algorithm's simulated makespan and resource utilization outperformed previous algorithms, in our experiments, we utilized datasets of varying sizes from 500 to 4000.
Conclusions: By using the HGHHM algorithm, this research improves cloud job scheduling efficiency and reliability by improving makespan and resource consumption. These findings confirm hybrid meta-heuristic techniques' efficacy and emphasize the need to balance exploitation and exploration to avoid local optima entrapment. Nevertheless, the study is limited in its scope as it does not take into account other factors such as energy consumption and cost.

Keywords: *Cloud Computing; Henry Gas Solubility Optimization; Harris Hawks Optimization; Task Scheduling.*

Acronyms

HGHHM: Henry Gas-Harris Hawks Modified-Opposition
HHO: Harris Hawks optimization algorithm
HGSO: Henry gas solubility optimization algorithm
MOBL: Modified comprehensive opposition base learning
NP: Non-deterministic polynomial time
OBL: Opposition-based learning
QoS: Quality of service
RU: Resource utilization

1. INTRODUCTION

Significant technological developments in recent years for data processing and storage have been observed due to extensive internet use. These advanced technologies have developed the current cloud computing concept, which transfers computational tasks and data from personal computers (PCs) and laptops to large-scale data centers [4, 12]. This innovative platform lets users conveniently retrieve global information online at any given time and location. However, the precise and reliable mapping of tasks allocations to resources is one of the primary concerns in cloud computing. The mapping process involves task scheduling, which is widely recognized as a Non-deterministic Polynomial Time (NP)-hard issue [3, 20]. Additionally this task scheduling concern is further compounded by its inherent complexity, dynamic characteristics, significant resource consumption, and unpredictability in resource availability [44].

Cloud computing requires the capability to manage a substantial number of concurrent users effectively. The quality of service (QoS) ability to meet all user requests efficiently and effectively is also crucial. Hence, an effective task scheduling procedure can achieve these requirements within a designated timeframe. The central aim of scheduling is to strategically assign tasks to appropriate resources, in order to meet particular criteria or objectives. The user is responsible for initiating the submission of tasks to the cloud scheduler, which is a critical element in the process. By utilizing a cloud-based information service, the system evaluates the present condition of accessible resources. Tasks are allocated strategically to various resources based on their specific needs, ensuring optimal and efficient use of the available computer infrastructure [49, 13]. Numerous studies investigated task scheduling algorithms to address cloud computing concerns in potentially constructing an optimal task schedule sequence, including particle swarm, Harris Hawks, and Crow algorithms [18].

While heuristic strategies offer an effective approach to task scheduling, it's important to note that this method doesn't always guarantee an optimal solution. Therefore, meta-heuristic algorithms are considered the most optimal method for addressing complex issues, which have proved significantly superior to other methods. These methods can locate approximately optimal solutions within a polynomial time frame instead of exponential time [22, 41].

Cloud computing requires efficient management of simultaneous users and meeting quality of service requirements. We are motivated to tackle this issue because of the persistent difficulties encountered by cloud computing systems in maximizing resource consumption, managing concurrent users, and meeting quality of service standards. Furthermore, the field of cloud computing technology is experiencing significant advancements, which in turn present novel prospects and complexities. This inspires us to examine these advancements and devise novel approaches that utilize emerging technology and enhance system efficacy. Furthermore, the increasing demand for cloud computing in various industries necessitates the development of scalable and efficient solutions.

This paper addresses these challenges by introducing a novel hybrid meta-heuristic approach, HGHHM, designed to optimize task scheduling in cloud computing environments. By enhancing resource utilization efficiency and reducing makespan. This approach holds significant importance for improving the performance and scalability of cloud systems, thus advancing the field's capability to meet the demands of modern computing. The suggested method is an integrated HGSO algorithm using the Harris Hawks optimization (HHO) and modified comprehensive opposition-based learning (MOBL) to increase the local search of the HGSO algorithm. Consequently, the primary objective of this study was to propose an enhanced Henry gas solubility optimization (HGHHM) approach for optimizing cloud computing task scheduling. This approach aims to address multi-objective functions, specifically to reduce the makespan (MKS) and increase the resource utilization (RU). The following briefly describes the primary contributions of this paper:

1. Modeling the Scheduling Problem: Considering the changing availability of cloud resources, this study formulates the scheduling problem as an optimization task.

2. An Improved Multi-objective Scheduling Model is Introduced: Our novel multi-objective scheduling approach aims to maximize resource consumption and minimize makespan at the same time, maximizing overall system efficiency.

3. Creation of the HGHHM Algorithm: We introduce the HGHHM algorithm, an enhanced version of the HGSO algorithm that combines the Harris Hawks algorithm with an altered

comprehensive opposition-based learning approach. With this improvement, the inherent shortcomings of conventional HGSO are effectively addressed, greatly enhancing the performance and scalability of cloud systems by balancing exploitation and exploration to avoid local optima entrapment.

The remaining sections of this work are structured as follows: Section 2: Motivation and Related Works provides a comprehensive overview of relevant research carried out in the field. The concept of the algorithm utilized for this investigation is summarized in section 3, background study. In Section 4, we outline the process of formulating the scheduling problem as an optimization task. Section 5 provides a detailed explanation of the key components of the HGHHM algorithm. In Section 6, we elucidate the temporal complexity of the suggested approach. Section 7: Experimental environment, this section offers a comprehensive examination and elucidation of the data, along with the presentation of the experimental findings. Section 8 represents the final part of our investigation, where we provide concluding remarks and suggest possible paths for future research on this topic.

2. MOTIVATION AND RELATED WORKS

There are various shortcomings in the existing literature that allow us to develop a new task scheduling algorithm. As a result, this section focuses on the limitations of present approaches, as well as the types of objectives in current works: single and multi-objective, and how they affect job scheduling.

2.1 Meta-Heuristic Scheduling Strategies in the Cloud

Despite the advancements in cloud computing, the challenges of efficiently scheduling tasks and distributing resources persist. The limitations of present methods often limit their effectiveness in meeting the evolving requirements of modern computer environments. The primary challenges faced by current task scheduling algorithms encompass scalability issues, extended makespans, and inefficient resource utilization. The task scheduling challenge is further complicated by the inability to efficiently manage several simultaneous users and meet various quality of service (QoS) requirements as we will discuss in the upcoming related works. These limitations emphasize the necessity for innovative approaches to address these problems and enhance the scalability and efficiency of cloud computing systems [27, 43].

To begin our review, we studied a range of meta-heuristic approaches designed to maximize efficiency by striking a balance between strategies for exploration and exploitation. In practical, task scheduling studies using meta-heuristic algorithms were significantly investigated. For instance, Elaziz *et al.* introduced a modified Moth algorithm using Differential Evolution (MSDE). [2] Even though the study revealed that the algorithm outperformed other algorithms based on performance metrics, this approach was limited due to the high time complexity.

In another study, Attiya *et al.* proposed an annealing-based Harris Hawks optimization technique (HHOSA) [10]. The study indicated that the HHOSA approach developed significant MKS reduction. However the algorithm only operates on a single objective.

In addition Sa *et al.* provided an upgraded discrete symbiotic organism search method with meta-heuristics for the best work scheduling in a cloud computing environment. The experiment was carried out using the CloudSim simulator, and the simulation results showed that, especially for large search spaces, the performance of the suggested method was significant when compared to the benchmark technique for both the makespan and response time, Nevertheless, it frequently becomes trapped in local optima as a result of the enormous size of the makespan and response time values [42].

Similarly Alboaneen *et al.* discovered an optimization framework for optimizing joint task scheduling and virtual machine (VM) placement [5]. The study highlighted that the simulation results enhanced resource utilization, reduced costs, shortened the MKS, and lowered the degree of imbalance. However the algorithm work with small sizes of tasks.

In a recent study [13] the Johnson Sequencing algorithm is used to schedule tasks across three servers in order to minimize the makespan. The algorithm strategically determines the optimal order for task execution on each server while accounting for job interdependencies and individual server processing times. Despite this, further investigation is required to assess the scalability of this algorithm.

Singh et al. introduced the crow–penguin optimizer method [46]. The multi-objective formulation optimized the maximum value by maximizing QoS and resource utilization while minimizing load and MKS. However, the algorithm used large number of resources with small size of tasks. Meanwhile, Abd Elaziz and Attiya presented an HGSWC approach that was based on a whale optimization algorithm (WOA), HGSO, and comprehensive opposition-based learning (COBL) for optimal task scheduling [1]. The study demonstrated higher HGSWC MKS performance than HGSO and WOA algorithms. Despite this, the algorithm's convergence requires more refinement, therefore there is opportunity for improving the makspan.

Considering that these studies provided substantial evidence regarding MKS, we proposed and evaluated a novel HGHHM algorithm in this study to address the cloud task scheduling issue. The proposed solution was built upon the modified MOBL and HHO with HGSO operators.

Finally, [38, 39] creating a novel dynamic scheduling method through the manipulation of the cloud job schedule's gaps. The algorithm that regulates the gaps in the schedule of cloud jobs. The performance analysis demonstrates that the recommended method outperforms compared algorithms in terms of flow time, makespan time, and total tardiness. Nevertheless, the total delay rises as the number of tasks increases.

The cloud ecosystem consists of two primary entities: cloud providers and cloud clients. Prior studies research on work scheduling has mostly concentrated on incentives for either cloud users or providers of cloud services. Most optimization issues focus on a single objective, however these problems are typically multi-objective. In the cloud, Users in the cloud industry want cost-effectiveness and high quality services, however the providers aim to maximize resource usage and profitability.

Therefore, in the following sections (section 4) , we explicitly outline the task scheduling problem as an optimization challenge, specifically considering its multi-objective nature and the consequential effects on both users and providers in the cloud industry.

2.2 Single and Multi-Objective Functions

An optimization problem can be categorized as either a single-objective or multi-objective function, depending on the number of criteria involved. The goal of single-objective function optimization is to identify the best solution given a single criterion. However in practical situations, pursuing one goal could conflict with another, resulting in less-than-ideal outcomes. On the other hand, multi-criteria optimization aims to produce a collection of solutions that concurrently fulfill several conditions, providing a more thorough method of problem-solving [11] Multi-objective optimization improves decision-making processes by taking many objectives into account at once. This allows for a more sophisticated examination of trade-offs and the discovery of Pareto-optimal solutions that successfully balance conflicting objectives [40].This method works especially well in complicated systems such as cloud computing, where it is necessary to optimize multiple performance measures at the same time in order to get the best results. Zakaria et al, suggested a hybrid approach to task scheduling problems that combines Tabu Search (TS) and Cat Swarm Optimization (CSO). On the single parameter, the suggested algorithm (TS-CSO) performs better in makspan than the other algorithms, as demonstrated by the implementation results.

However, the other goal is not optimized, which could have an impact on the system's total effectiveness [51]. In the same context Fu et al., proposed particle swarm optimization genetic hybrid algorithm to reduce makespan [21]. On the other hand, studies such as that of Tanha et al. adopt a multi-objective perspective, proposed Genetic with thermodynamic simulated annealing intends to improve schedule length ratio, speed, efficiency, and makespan as well. By taking into account multiple objectives at once, this method provides a more thorough optimization strategy, which eventually improves system performance as a whole [47]. Additionally Manikandan et al, proposed Whale optimization algorithm with bee algorithm to Enhanced makespan, energy consumption, resource utilization and computation cost ultimately leading to improved overall system performance [52].

As a result, the concept of multi-objective optimization in scheduling in a cloud computing environment has become increasingly popular, indicating a significant advancement in handling the intricate and varied optimization objectives in cloud-based task scheduling.

3. BACKGROUND STUDY

This section explores the theoretical basis for the algorithms used in the study. It introduces numerous key concepts drawn from conventional algorithms, including HHO (Harris Hawks Optimization) and HGSO (henry gas algorithm). Furthermore, the section gives a brief explanation of the proposed MOBL. By discussing these fundamental theories, the section provides the foundation for understanding how these algorithms are adapted and integrated into the proposed HGHHM approach to effective task scheduling in cloud computing.

3.1 The Concept of Harris Hawks Optimization Algorithm (HHO)

The HHO algorithm is an original meta-heuristic algorithm that addresses global optimization problems. Generally, the HHO exhibits behavioral patterns similar to hawks in their natural habitat as they pursue and capture their prey. Compared to other meta-heuristic approaches, this algorithm performs a search process in two distinct phases using various methodologies (see **Fig. 1**) [24]. Approximately two fundamental phases are present in this algorithm: exploitation and exploration. The phases in this study involved effectively utilizing four robust exploitation strategies of HHO algorithm, which enhanced the precision and effectiveness of the proposed algorithm. Therefore, this strategic integration significantly contributed to a more powerful and adaptable algorithm [36]. Following is a brief clarification of the HHO stages:

Exploration

The algorithm now initiates the Exploration phase, determined by the value of the prey's escape energy E. If the cardinality of set E (|E|) is 1 or more, the algorithm will start the Exploration phase as clarify in the pseudo code.

Transform from exploration to exploitation

Based on the prey's escape energy E, the HHO algorithm can switch between various exploitative behaviors after transitioning from exploration to exploitation.

Exploitation

The hawks will utilize one of the four tactics to capture their prey, which in the algorithm context refers to identifying the optimal solution. It indicates that they will surround the target from various directions with varying levels of force based on the prey's remaining energy. The four strategies are (Soft besiege, Hard besiege, Soft besiege with progressive rapid dives, Hard besiege with progressive rapid dives), as mentioned in the pseudo code of HHO [48 , 33, 36].

Algorithm 1 HHO

```
Algorithm input: N (population size), s_ma
(maximum iterations).
Outputs: The position of the solution and th
fitness value Initialize the random population X
for i = 1, 2,..., N.
Generate a starting population Xi for i = 1, 2,..., N
Carry out the following actions till the termina
condition is met:
Determine fitness values.
Determine the optimal solution $X_b$.
Define $X_b$ as the optimal position of the rabbi
For each hawk X
update the beginning energy $E_0$ and jump strengt
J
$E_0$=2 $r_5$-1, J = 2(1 -$r_5$), $r_5$ is a random variabl
Update the variable E using the followin
equation:
        $E = 2E_0(1 - \frac{s}{s_{max}})$

If the absolute value of E is greater than or equal
to 1 : ➡ Exploration phase

 Update the location variable using the followin
equation

$Xi(s+1) = \begin{cases} Xr(s) - r1\,|\,Xr(s) - 2r_2X(s)\,|\,. & Q \geq 0.5 \\ \left(Xb(s) - X_{Avg}(s)\right) - \omega & . \; Q < 0.5 \end{cases}$
If the absolute value of E is less than 1 do the
next steps: ➡ Exploitation phase

 If (r ≥0.5 and |E|≥ 0.5) then ⟹ Soft besiege

Update the location variable using the following
equation

    $X(s+1) = \Delta X(s) - E\,|\,J \times X_b(s) - X(s)\,|$
          $\Delta X(s) = X_b(s) - X(s).$

r ≥0.5 and |E|< 0.5) then ⟹ Hard besiege
Update the location variable using next equation

        $X(s+1) = X_b(s) - E \times |\,\Delta X(s)|.$

Else if (r < 0.5 and |E|≥ 0.5) then ⟹
    Soft besiege with progressive rapid dives
Update the location variable using next equation

 $X(s+1) = \begin{cases} Yd \;\; if\; F(Yd) < F(X(s)). \\ Zd \;\; if\; F(Zd) < F(X(s)) \end{cases}$
Else if (r < 0.5 and |E|< 0.5) then
```

```
⟹ Hard besiege with progressive rapid dives
Update the location variable using next equation
$X(s+1) = \begin{cases} Yd' \ \text{if } F(Yd') < F(X(s)). \\ Zd' \ \text{if } F(Zd') < F(X(s)) \end{cases}$
Return $X_b$ (best solution )
  End
```

Figure 1: The HHO Algorithm [8, 9, 24]

3.2 The Concept of Henry Gas Solubility Algorithm (HGSO)

The HGSO algorithm is commonly employed to simulate Henry's law [1]. This method is well-known for its strong exploratory capabilities. The core structure of HGSO consists of multiple phases to augment its exploitation potential further (see **Fig. 2**) [17]. A Harris Hawks algorithm (known for its exceptional exploitation capabilities) was meticulously incorporated into the HGSO. This synergistic hybridization could significantly enhance the capabilities of the Henry gas optimization technique.

Algorithm 2 HGSO

```
Create a population ($i$=1, 2,...,) with $N$ individuals, Define the number of gas types as $i, Hj, Pi, j, Cj, l1, l2$, and $l3$.
Partition the population of agents into distinct groups (clusters) based on their Henry's constant value ($Hj$), which is consistent throughout each group.
Assess each cluster j.
Obtain the optimal gas $Xi$ within each cluster, as well as the most effective search agent $Xbest$.
While (stopping criteria not met (i.e. s < $Maxiteration$))
    For each search agent: Update the location of the individual search agent.
        $Xij(s+1) = Xij(s) + Fg \times r \times \eta \times (\mathbf{X_{ib}}(s) - Xij(s)) + Fg \times r \times \alpha \times (Sij(s) \times \mathbf{X_{ib}}(s) - Xij(s))$
    End for
    Update the Henry's coefficients for each type of gas.
        $Hj(s+1) = Hj(s) \times \exp(-Cj \times (1/T(s) - 1/T^{\theta})), \quad T(s) = \exp(-s/iters)$
    Provide the updated solubility values for each gas
        $Sij(s) = K \times Hj(s+1) \times Pij(s)$
    Rank and choose the number of agents with the worst performance:
        $Nw = N \times r \times (\mathbf{c_2} - \mathbf{c_1}) + \mathbf{c_1}, \ \mathbf{c_1} = 0.1$, and $\mathbf{c_2} = 0.2$
    Adjust the position of the most ineffective agents.
        $Gij = \mathbf{G_{ij}^{min}} + r \times (\mathbf{G_{ij}^{max}} - \mathbf{G_{ij}^{min}}), \quad i = 1, 2, \ldots, Nw$
    Revise the optimal gas $Xi,best$, and the optimal search agent $Xbest$
End while
s=s+1
Return $Xbest$
```

Figure 1: The HGSO Algorithm [1, 21]

3.3 Modified Comprehensive Opposition Base Learning (MOBL)

Meta-heuristic algorithms have effectively incorporated the underlying opposition-based learning (OBL) concept into several schemes to enhance performance.

Definition: Assume that x is a real number in the range [a, b]. Hence, the definition of the opposite number $\bar{X}$ is given as follows [15, 45].

$$\bar{X} = a + b - x \qquad (1)$$

Comprehensive opposition-based learning (COBL) is an expanded version of the conventional OBL approach, which enables the meta-heuristic algorithm to achieve convergence toward optimal solutions. The primary objective of COBL is to transform the given solution X_i to one of its opposing solutions [1, 14]. This study updated the COBL by shifting the solution X_i to the extended opposition ($\bar{X}^{eo}$). When the existing solutions were contrasted with their opposing solutions, the modified comprehensive opposition-based learning (MOBL) technique attempted to determine the optimum solutions [10]. The opposite solution $\bar{X}j$ of X_j is calculated as follows:

$$\bar{X}j = UB_j + LB_j - X_j, \text{where } j = 1, 2.. \qquad (2)$$

Where UB and LB are upper and lower values. Therefore, the MOBL is expressed using the following equations:

$$\bar{X}_j^{eo} = \begin{cases} \bar{X}j + (UB_j - \bar{X}j) x \text{ rnd } X_i > (LB_j + UB_{j)})/2 \\ (LB_j + (\bar{X}j - LB_j) x \text{ rnd } X_i < (LB_j + UB_{j)})/2 \end{cases}$$
(3)

Where $j = 1, 2 \ldots DIM$. Lastly, The optimal or near-optimal solutions were chosen from the collection of existing solutions Xi and their opposing solutions $\bar{X}_i$.

4. PROBLEM FORMULATION

Our empirical findings highlight the enormous difficulty associated with task scheduling in cloud computing systems, where the goal is to effectively distribute many jobs among available computing resources to optimize different goals as makespan, cost, and energy consumption [43]. Formally defining the task scheduling problem is often accomplished by conceiving the cloud system (CS) as consisting of a number of physical machines (PMs), referred to as Npm, each of which hosts a group of virtual machines (VMs), referred to as NVM. The computing power, memory size, storage capacity, and network bandwidth of each virtual machine (VM) define it. Let us consider a task having the following properties, indexed as lth = 1, 2, 3..., NT, where NT is the total number of tasks assigned to virtual machines (VMs): [Snl, T_lenl, Etl, Prl] = T (1).Where T_lenl is the task's length (measured in millions of instructions) and Snl is the l-task's serial number. The lth task's priority is indicated by Prl. The estimated time to complete the lth Task is Etl. The following equation provides the ETC matrix for NT tasks and NVM virtual machines.

$$ETC_{ij} = \begin{bmatrix} ETC_{1,1} & ETC_{1,2} & \ldots & ETC_{1,vms} \\ ETC_{2,1} & ETC_{2,2} & \ldots & ETC_{2,vms} \\ \ldots & \ldots & \ldots & \ldots \\ ETC_{n,1} & ETC_{n,2} & \ldots & ETC_{n,vms} \end{bmatrix} \quad (4)$$

Where the element $\boldsymbol{ETC_{ij}}$ represents the estimated time Et for lth task on the jth VM, and it is defined as:

$$ETC_{ij} = \frac{T\_lenl_i}{MIPS_j} \quad (5)$$

Where $MIPS_j$ denotes the processing capacity of the jth VM.

Thus, the mapping of a set of tasks from a given tasks onto a set of virtual machine in this study in a way that optimizes both the task scheduling's resource consumption and makespan at the same time. The following equation (8) illustrate how this topic is stated as a multi-objective optimization problem with the goals of maximizing resource usage and minimizing service time.
The makespan (MKS) can be expressed as shown in Equation (6).

$$MKS = \max_{j \in 1, 2, vms} \sum_{i=1}^{n} ETC_{i,j} \quad (6)$$

The **RU** can be calculated by Eq. (7):

$$\text{Resource Utilization} = \frac{\sum_{i=1}^{N} Tvmi}{makespan * N} \quad (7)$$

Tvmi represents the length of time consumed by the VMi to complete all jobs, while N stands for the total number of resources [26, 27].

Consequently, our goal is minimizing MKS and maximize resource utilization, the fitness function is formalized as follows:

$$Fv = Min\ MKS \quad \& \quad Fv = max\ RU \quad (8)$$

5. THE PROPOSED ALGORITHM

The suggested HGHHM method successfully utilized the capabilities of the HHO to eliminate task scheduling constraints. By acting as local operators, these HHO operators significantly improved HGSO's performance. Therefore, this methodology serves to effectively mitigate the drawbacks associated with each distinct meta-heuristic approach. The HHO algorithm was chosen because the integration of its four exploitation tactics improves our algorithm's flexibility and efficacy, aligning closely with our goals through balanced exploration and exploitation, hierarchical adaptability, and parallelism. While selecting HGSO for its strong exploration capabilities.

The combination of HHO's balanced exploration and exploitation with HGSO's excellent exploration capabilities preventing it from getting stuck in local optima which offers a superior algorithm that excels at exhaustively exploring the solution space, which is vital to our project's success.

The initial set of N integer solutions X in the proposed HGHHM algorithm acquired a size n, representing the number of tasks. Each solution contained values confined to the range [1, vms] (vms = virtual machines). The fitness value is then computed to determine the quality of each solution using equation (8).

The optimal Xb value was identified while the effectiveness of solution X was increased by combining HGSO and HHO. Depending on the probability of the fitness value, the ith solution was modified using the HGSO and HHO operators and then using MOBL to enhance the worst solutions. This update process proceeded until the necessary conditions for termination were fulfilled.

The subsequent sections provide a more comprehensive analysis of the stages involved in the proposed HGHHM algorithm. **Fig. 3** illustrates the steps of our new algorithm and **Fig. 4** displays the structure of the cloud scheduling technique presented, which is based on a modified HGSO.

5.1 First Stage

The representation stage: the proposed HGHHM algorithm produces solutions (Xi, i= 1, 2, 3... N) as follows:

$$\bar{X}ij = floor\left(\left(LBij + \alpha * (UBij - LBij)\right)\right), \alpha \in [0,1], j = 1,2,.,vm \quad (9)$$

Following the task scheduling description, the lesser limit LB was assigned to 1 while the maximum limit UB was fixed to VMs from Equation (9). The real values in this scenario were converted to integers using the floor function. This conversion process was also employed to obtain the integer for Xi.

5.2 Second Stage

The update stage: the computation of the fitness value (Fv) was initially performed for each potential solution X. Therefore, the optimal solution Xb was determined. The probability of each solution is then measured based on its fitness values as follows:

$$Pri = \frac{Fvi}{\sum_{i=1}^{N} Fvi} \quad (10)$$

Depending on the value of Pri, the solution Xi is updated using either the HHO or HGSO operators as follows:

$$Xi(s+1) = \begin{Bmatrix} Using\ operators\ of\ HHO & if\ Pri \geq r_{pr} \\ Using\ operators\ of\ HGSO & if\ Pri < r_{pr} \end{Bmatrix} \quad (11)$$

Where rpr is a random number ∈ [0, 1]. Depending on Pri, the rpr value is adjusted as follows:

$$r_{pr} = Lpr + rnd * (Upr - Lpr) \quad (12)$$

Where Upr and Lpr are the highest and lowest Pr values, respectively. The subsequent step involves determining the worst Nw solutions as follows:

$Nw = N \times r \times (c_2 - c_1) + c_1, c_1 = 0.1$ and $c_2 = 0.2$ (13)

5.3 Third Stage

Subsequently, the MOBL was used by shifting the solution X to the extended opposition ($\bar{X}^{eo}$). Consequently, the reflected OBL was formed using Equation (1). The most optimal solutions were then selected from the collection of existing solutions X and their opposing $\bar{X}$.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Lastly, the termination requirements were verified. If these requirements were met, the proposed HGHHM algorithm was stopped, and Xb was returned. Otherwise, the updating stage was repeated. **Fig. 5** depicts the suggested system model for the HGHHM strategy, which executes a scheduling procedure that enhances both makespan time and resource utilization.

Proposed HGHHM Algorithm

```
Initialize: value of N, which represents the
number of solutions, values of n, smax, and
m, which respectively represent how
many tasks and the total number of
iterations, and the number of machines.

1. Assign starting values to the parameters of HGHHM.
2. Generate a random integer solution (X) consisting of N*n elements.
3. Set i=1
4. Reiterate
5. Calculate the fitness value (Fv) of Xi, where i ranges from 1 to N, as given in equation (8).
6. Ascertain the optimal resolution Xb
7. For I :1 to N
8. Calculate the probability Pri using equation (10) and $r_{pr}$ using equation (12).
        If Pri ≥ $r_{pr}$ then
9. Update Xi using the Exploitation of HHO as in algorithm 1
10. Else
11. Update Xi using the Exploration of HSGO as in algorithm 2
12. End if    13. End for
13. Determine the worst solution using (13)
14. Update the position of the worst agents using MOBL as in Eq (3)
15. Update the current iteration (s=s+1)
16. Until s > s max    18.Return Xb
```

Figure 3: The Pseudo Code of the Proposed Cloud Scheduling Algorithm

6. COMPLEXITY TIME OF HGHHM ALGORITHM

The temporal complexity of HGSO is $O(t \times n \times d) \times O(\text{obj})$, where t is the maximum number of iterations, n is the number of populations, d tells us how dimensional the solution space is, and obj is the objective function's

No Yes

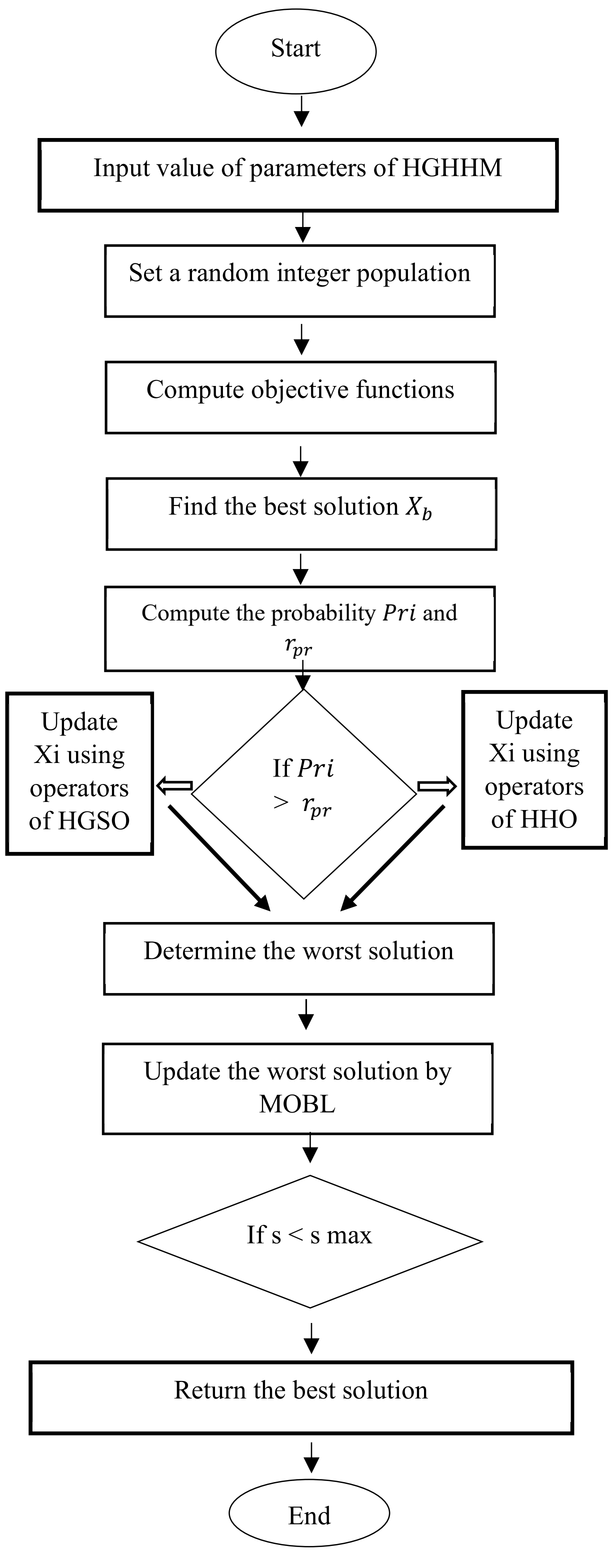

Start
Input value of parameters of HGHHM
Set a random integer population
Compute objective functions
Find the best solution X_b
Compute the probability Pri and r_{pr}
Update Xi using operators of HGSO
If $Pri > r_{pr}$
Update Xi using operators of HHO
Determine the worst solution
Update the worst solution by MOBL
If s < s max
Return the best solution
End

Figure 4: The structure of the Proposed Algorithm

computational complexity. Building on this basis, comparable features are shown by the temporal complexity analysis of the suggested HGHHM algorithm. Figure 4 and the HGHHM pseudo-code indicate that this algorithm's time complexity can be written as $(t\times n\times d)\times O(\text{obj})$ [1, 21].

Every agent in the HGHHM iteration is required to select a position update strategy using the given algorithmic equations. The process by which the algorithm explores and exploits the solution space depends on these update mechanisms. HGHHM's temporal complexity remains $(t\times n\times d)\times O(\text{obj})$, as the position update techniques are similar to those in HGSO. HGHHM's complexity enables effective handling of large-scale optimization problems, balancing the need for full search capabilities with practical computing restrictions.

7. EXPERIMENTAL ENVIRONMENT

The experimental design, simulation tool, dataset, and the performance metric used in this study are presented in this section.

7.1 Platform for Simulating Cloud Computing

The University of Melbourne, Australia, developed CloudSim, a collection of simulation toolsets for cloud computing that operates on discrete events [7, 23]. This platform emulates various diverse cloud computing resources, users, and scheduling approaches. The CloudSim facilitated the developing and evaluating scheduling algorithms through the simulation and modelling of several variables in parallel programming tools. These programs included brokers and intermediaries for clients, applications, assets, and resources. The CloudSim is also an extendable simulation framework, which the primary steps in the simulation are as follows [32, 37].

7.1.1. Simulation initialization: Before the CloudSim simulation, the CloudSim object required the CloudSim.init(...) function as the initial setup [35].

7.1.2. Create cloud computing tasks and resources: The simulation process should include generating task entities. Initially, a cloud task list was produced. Subsequently, the operation task was added to the task list [30]. **Table 1** tabulates the experimental parameter configurations for the simulation [10].

Table 1.Experimental Parameter Configurations For The Simulation

Cloud entity	Variable	Value
User	"Number of users"	50–100
Datacenter	"Number of data centers"	1
Host	"Number of hosts Capacity for storing data''	2
	Storage	1 T
	RAM	16 GB
	Bandwidth	10 Gb/s
	Architecture	X86
	Type of policy	Time shared
VM	No. of VMs	25
	MIPS	5000
	Size	10 GB
	RAM	0.5 GB
	VMM	Xen
	Number of central processing units (CPUs)	1
	Type of policy	Time shared

7.2 The Datasets

The efficiency of the proposed HGHHM algorithm was examined in the test experiments using a real dataset. These tasks were regarded as independent and non-preemptive. The workload traces from HPC2N were used in this study [1]. The workloads are accessible through the URL (http://www.cs.huji.ac.il/labs/parallel/workload/) or (https://github.com/fernandodeperto/swf/blob/master/HPC2N-2002-2.1-cln.swf.gz).
Approximately 30 cycles were performed independently in each experiment for each approach, which improved the accuracy of outcome predictions. Subsequently, the outcomes were computed as an average across these iterations. **Table 2** lists the parameters for the proposed algorithm and various meta-heuristics examined in this study .These parameters of HGSWC and HGHHM were chosen according to previous studies [1,10]. Moreover, HGHHM factors were employed during the implementation phase, changes were carried out, and the results were noticed. The parameters utilized in the proposed algorithm were chosen based on the resulting

outputs, as well as informed by previous research works.

Table 2.Setting parameters for the compared algorithm

Algorithm	HGSO	HHO	HGSWC	HGHHM

Parameter	a = 2
	b = 1
	l= 5E−2
		E0 [−1, 1]	a = 2, b = $^{\alpha}$ = 1 , $^{\beta}$ = 1, l=5E−2	a = 1
				b = 1
				l=5E-2

7.3 Metrics for Assessing Performance

This subsection elucidates the performance of the algorithm by employing specific metrics that are utilized to evaluate and compare the proposed HGHHM algorithm. Similar task scheduling approaches in other existing studies accompanied the process.

MKS: The MKS is a frequently employed metric for evaluating the effectiveness of scheduling strategies and regulations [25].This metric exhibits the completion time of the most recent task. Hence, a short MKS indicates an effective task allocation to VMs. This study aimed to reduce MKS, which the fitness function was formalized in Equation (4) as follows:

$$Average\ value\ (Avg_F): \frac{1}{N_r}\sum_{i=1}^{N_r} F_b^i \qquad (14)$$

$$Worst\ value\ (Wrst_F): \underset{1\le i\le N_r}{Max}\ F_b^i \qquad (15)$$

$$Best\ value\ (Bst_F): \underset{1\le i\le N_r}{Min}\ F_b^i \qquad (16)$$

where F_b^i and N_r are the optimal fitness values of the ith run and the entire number of runs, respectively [19].

Resource Utilization (RU): describes the efficient use of resources that are available to achieve the desired outcomes. *As in equation (7)*

The Rate of Performance Improvement (PIR): The PIR was defined in Alkaam *et al.*'s study as a quantitative metric that assesses how much a given methodology outperforms existing scheduling methods from other studies [6]. Therefore, PIR is formulated as follows:

$$PIR = \frac{Zd-Zd}{Zd\prime} * 100 \qquad (17)$$

where *Zd'* and *Zd* are the fitness values of the suggested approach and the contrasting values from the previous studies, respectively [10, 31].

8. RESULTS AND DISCUSSION

This subsection thoroughly examines the suggested HGHHM algorithm's performance evaluation and analysis, providing insight into how effective it is in terms of makespan (MKS) and Resource usage (RU). An evaluation in comparison to the many current algorithms (HHO and HGSO) using HPC2N dataset with size 500 to 2500, this comparison offers important information about the algorithm's capabilities. Furthermore, we evaluated simulated HGSWC algorithm with our algorithm using dataset size from 500 to 4000, whereas the baseline algorithm was tested on datasets ranging from 500 to 2500 in the base paper [1]. This allowed us to assess the scalability of our proposed technique by gradually expanding the dataset size. As the dataset size grew, our approach consistently outperformed the baseline algorithm in key performance parameters which are makespan and resource usage.

This higher performance highlights our algorithm's scalability, demonstrating its capacity to effectively manage larger-scale task scheduling instances. Moreover, the baseline algorithm's average performance highlights the benefits of our technique in tackling scaling issues. These results indicate the robustness and scalability of our suggested method in real-world cloud computing environments. The outcomes show that the algorithms' hybridization successfully balanced their exploitation and exploration capacities in the proposed algorithm, preventing it from being trapped in local optima.

In our analysis, we used p-values, which were consistently below the alpha level of 0.05, derived from the studies by [28, 50, 29]. Additionally, the minimum and maximum values for the datasets were derived using methods outlined by [1, 10, 16, 34].

Figures 6 and 7 demonstrate the convergence curves of HGSWC and the proposed HGHHM algorithm for MKS with 500 and 4000 instances respectively. The convergence curves show that the suggested HGHHM approach outperforms HGSWC in terms of convergence accuracy. The increased diversity of ecosystems allows the HGHHM to avoid entrapment in local optima, which increases the exploitation ability of the

HGHHM method and adds to higher convergence accuracy.

Similarly, **Figures 8 and 9** elegantly depicts the convergence curves for both proposed and simulated algorithms with the same dataset sizes to clarify **RU**. This benchmark shows how the proposed HGHHM algorithm efficiently allocates resources for balance and optimality.

Table 3 shows the optimal makespan values achieved by the benchmark algorithms (HHO, HGSO, and HGSWC) with mention to statistics (min, max, mean) values in comparison to the proposed HGHHM method with varying dataset sizes (500, 1000, 1500, 2000, and 2500). The results clearly demonstrate that the HGHHM algorithm exhibits superior performance compared to the other algorithms in terms of makespan. The outstanding results of the HGHHM algorithm are due to its successful integration of exploitation and exploration capabilities. This integration enables more efficient task scheduling and superior overall optimization. The HGHHM algorithm improves operational efficiency and offers a more dependable solution for cloud job scheduling difficulties by obtaining reduced makespan values.

Figure 10 depicts the optimal makespan values achieved by the benchmark algorithms (HHO, HGSO, and HGSWC) in comparison to the proposed HGHHM method with varying dataset sizes (500, 1000, 1500, 2000, and 2500). The results clarify that the HGHHM algorithm exhibits superior performance compared to the other algorithms in terms of makespan.
The findings of a t-test with two samples are presented in **tables 4 and 5**.

The purpose of the test was to determine if there were significant variations in the makespans and resource utilization obtained with the HGHHM and HGSWC while utilizing the same stopping criterion for all task instances. The p-value must be smaller than the predetermined significance level of alpha, which is set at 0.5. The tables demonstrate that P-values for practically all of the datasets were lower than this alpha value, indicating a considerable enhancement in the suggested HGHHM strategy compared to the HGSWC strategy. Thus, it can be concluded that HGHHM outperforms HGSWC in performance as the search space increases. **Tables 4 and 5** present the results for the HGHHM and HGSWC algorithms using larger datasets. Additionally, **Table 7** provides the p-values for comparisons with other algorithms [1, 10], the p-values < .00001, the result is significant at $p < .05$. If the p-value < 0.05, the null hypothesis is rejected, indicating a significant difference between the groups. While if the p-value > 0.05, the null hypothesis fail to rejected, suggesting no significant difference between the groups. Hence, the null hypothesis is rejected.

Furthermore, the proposed against the simulated HGSWC algorithm, the **RU** effectiveness of the HGHHM algorithm was thoroughly assessed in a range of scenarios and a certain number of iterations. The comparison data is summarized in **Tables 6**, which shows that the suggested HGHHM algorithm achieved a higher average of resource usage than the compared algorithms. This important discovery highlights the algorithm's outstanding performance under various testing scenarios, leading to increase the use of the resources.

Table 8 and **Fig 11** display the percentage improvements for (MKS and RU), summarizing the simulation findings. The results clearly and definitively show that the HGHHM algorithm performs nearly optimally and is superior to benchmark scheduling alternatives. The findings emphasize the algorithm's skillful distribution of resources and substantial decrease in makespan (MKS), hence enhancing the overall efficiency of cloud computing systems. The HGHHM algorithm has shown considerable improvements in performance, surpassing the benchmark algorithms and showcasing its potential effectiveness in addressing the intricate challenges of cloud job scheduling.

To sum up, the suggested HGHHM algorithm is stronger and more effective than previous algorithms because it can consistently produce lower makespan values with optimizing resource use, as shown by the t-test, p-values, and PIR. For this reasons, the HGHHM algorithm represents a significant development in cloud computing, providing enhanced task scheduling capabilities that can efficiently manage a range of workloads.

9. CONCLUSION AND FUTURE DIRECTION

This study successfully presented a novel HGHHM algorithm to address the shortcomings of a recently established meta-heuristic algorithms [1, 10]. The suggested method applied the HHO method as a local search technique to strengthen the capabilities of HGSO while increasing the quality of generated solutions. A modified MOBL version was also utilized for the least ideal solutions, which efficiently allocated the tasks to the computing resources of the cloud. This hybridization of the algorithms successfully balance the exploitation and exploration skills, preventing it from being trapped in local optima. Considering that the HHO, HGSO, and simulated HGSWC algorithms validated the proposed HGHHM algorithm, the HGHHM outperformed benchmark algorithms for all test functions. This outcome highlighted that when tackling the cloud task scheduling issue, the proposed HGHHM algorithm was more efficient than the different algorithms while providing near-optimal results.

In addition, our proposed HGHHM algorithm exhibits promising scalability characteristics. As we expanded the dataset size, HGHHM's scalability became evident. As the algorithm continued to function effectively and efficiently in handling instances of larger-scale task scheduling. The HGHHM algorithm's practical relevance and agility in tackling changing issues in cloud computing systems are further highlighted by these scalability findings.

Therefore, the proposed HGHHM algorithm was ideal and possessed a shorter MKS and higher RU for all the test conditions. However, one of our work's limitations is that it does not take into account other objectives such as energy usage and cost. Furthermore, testing the suggested technique in real-world cloud environments with dynamic workloads could provide more information about its practical applicability.

Future research is necessary since the number of activities is increasing and the data environment is getting more complex. The most important aspect is that artificial intelligence (AI) is developing at a rapid pace to increase search intelligence. Therefore, AI should be used to improve the suggested method while also enhancing other objective functions such as throughput, efficiency, and load balancing. Finally, extending the algorithm's application to other fields, such as edge computing and IOT, could be valuable.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia.

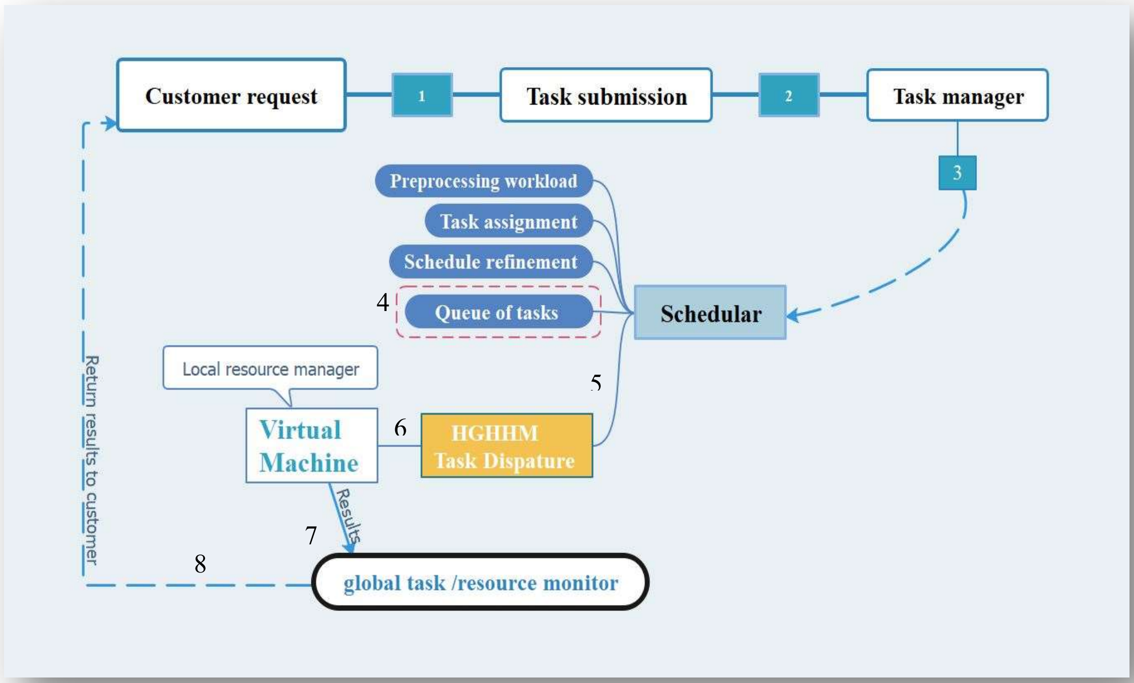


Figure 5: The Model of the Proposed HGHHM Task Scheduling Strategy

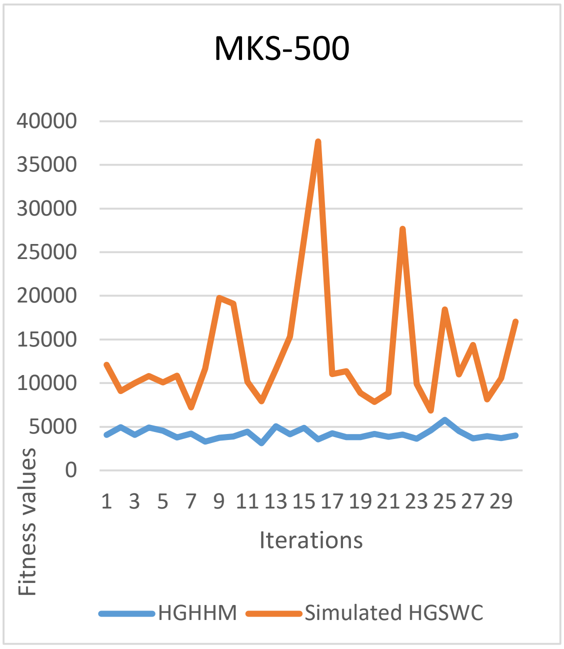


Figure 6: The convergence curves with 500 instances of tasks.

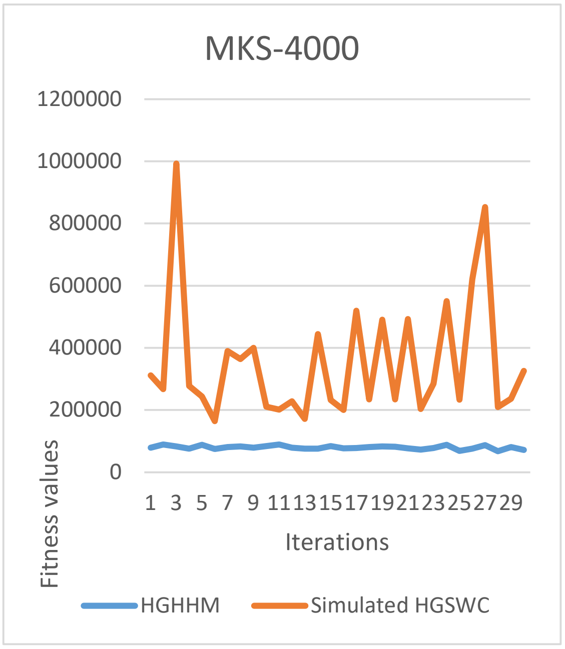
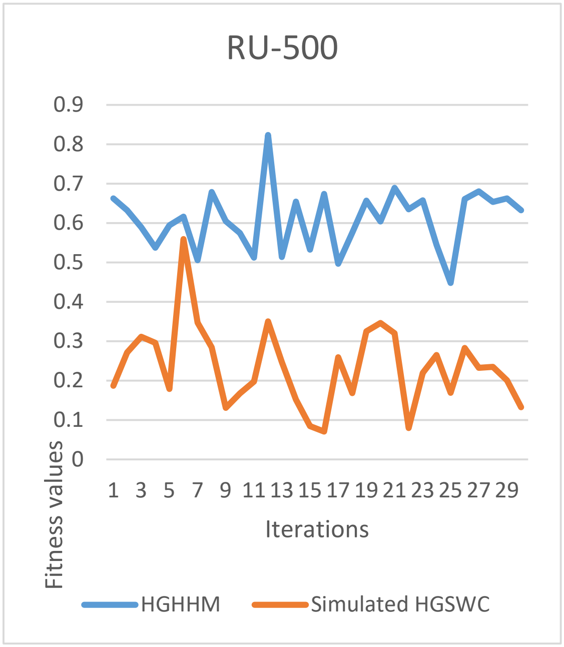


Figure 7: The convergence curves with 4000 instances of tasks.



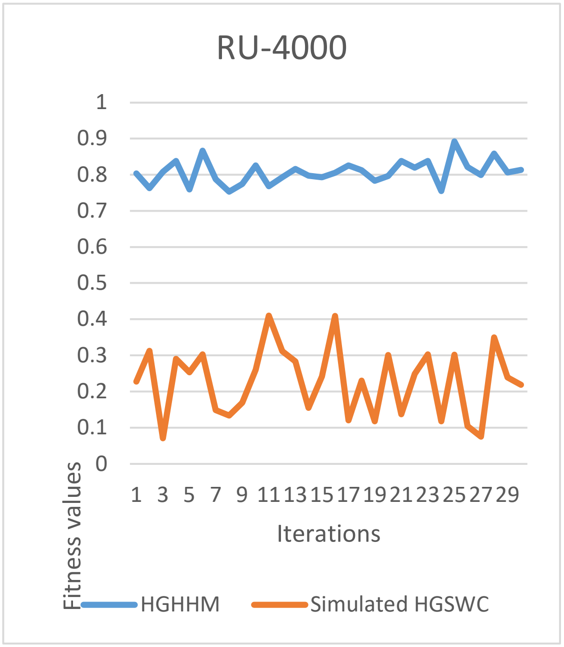


Figure 8: The convergence curves with 500 instances of tasks.

Figure 9: The convergence curves with 500 instances of tasks.

Table 3: comparison of makespan for the HGHHM algorithm and compared (HHO and HGSO) algorithms.

Instances	**Measure**	**HHO**	**HGSO**	**HGHHM**
500	Min/Best	5572.43	21934.34	3123.7668
	Max/Worst	6935.65	45687.86	5825.0994
	Mean/Avg.	6116.11	33385.90	4151.0541
1000	Min/Best	12253.81	41325.23	8669.9292
	Max/Worst	14824.45	124985.89	13837.2814
	Mean/Avg.	13591.03	86629.71	10862.9093
1500	Min/Best	19732.12	99027.29	15479.7548
	Max/Worst	23411.63	257541.14	22676.8434
	Mean/Avg.	21383.15	177520.92	17847.4402
2000	Min/Best	28076.55	132057.90	22311.0126
	Max/Worst	35356.93	437714.51	30165.8898
	Mean/Avg.	30552.34	255899.54	25746.0362
2500	Min/Best	36228.46	159871.20	35593.44
	Max/Worst	44466.31	554332.77	43318.5486
	Mean/Avg.	39698.23	367746.20	36789.3450

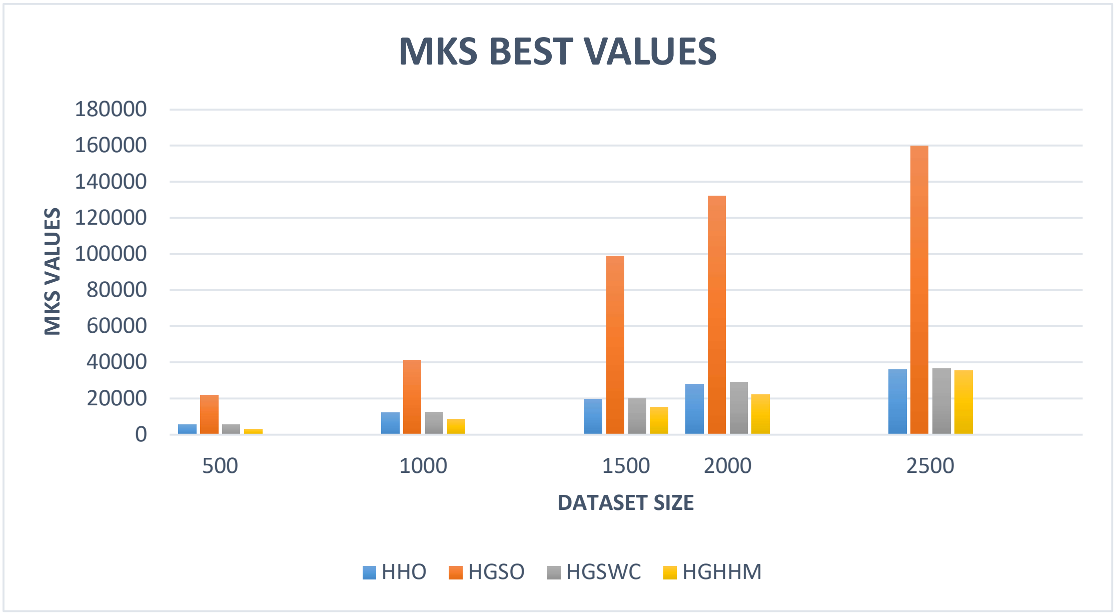


Figure 10: Comparison of best MKS values for the benchmarks algorithm with proposed HGHHM algorithm

Table 4. Comparison of best MKS values obtained by HGHHM and HGSWC for HPC2N workload

Instances	**HGSWC**	**HGHHM**	***T-test***	***p-value and hypothesis***
500	5843.56	3123.7668	-7.25722	The result is significant at $p < .05$ If the p-value < 0.05, The null hypothesis is rejected, indicating a significant difference between the groups. If the p-value > 0.05, The null hypothesis fail to rejected, suggesting no significant difference between the groups. For that, the our null hypothesis $\boldsymbol{H_0}$ is rejected
1000	12498.23	8669.9292	-6.94502	
1500	20151.27	15479.7548	-7.64343	
2000	29316.27	22311.0126	-7.44	
2500	81222.6368	35593.44	-11.82	
3000	89068.2896	45428.0338	-9.398874	
4000	164763.6	68148.461	-7.588634	

Table 5. Comparison of best RU values obtained by HGHHM and HGSWC for HPC2N workload

Instances	HGSWC	HGHHM	*T-test*	*p-value and hypothesis*
500	0.559934	0.823476	17.09534	The result is significant at p < .05 If the p-value < 0.05, The null hypothesis is rejected, indicating a significant difference between the groups. If the p-value > 0.05, The null hypothesis fail to rejected, suggesting no significant difference between the groups. For that, the our null hypothesis $\boldsymbol{H_0}$ is rejected
1000	0.516571	0.782442	22.377468	
1500	0.384578	0.799621	29.212048	
2000	0.419005	0.843817	29.117433	
2500	0.524741	0.868942	29.330736.	
3000	0.558811	0.853322	29.092466	
4000	0.410727	0.892281	34.17829.	

Table 6. Comparison of RU Obtained by HGHHM and HGSWC for HPC2N Workload

	Proposed algorithm HGHHM						
	500	**1000**	**1500**	**2000**	**2500**	**3000**	**4000**
BEST	0.8234763	0.7824422	0.7996212	0.8438166	0.868942	0.8533222	0.8922811
WORST	0.4478971	0.5306938	0.6303068	0.6815652	0.709755	0.7092268	0.7533871
AVG	0.6101444	0.6889814	0.7309229	0.7791990	0.78442	0.7847194	0.8069406
	Simulated algorithm HGSWC						
	500	**1000**	**1500**	**2000**	**2500**	**3000**	**4000**
BEST	0.5599343	0.5165711	0.3845777	0.4190054	0.524741	0.5588106	0.4107270
WORST	0.0711222	0.0768226	0.0645975	0.0684602	0.093425	0.0807665	0.0705065
AVG.	0.2355435	0.2690702	0.1959118	0.2438856	0.215867	0.2255931	0.2279506

Table 7. Detail Findings of Wilcoxon Signed Test in Term of MKS

Detail	HHO	HGSO	HGSWC
p-value	.02535	.025015	< .00001
The result is significant at $p < .05$			
conclusion	Reject H_0	Reject H_0	Reject H_0

Table 8.The PIR (%) For The HPC2N Workload over Benchmark Algorithms

Size of tasks	**OVER HGSWC**	
	PIR% improvement MKS	**PIR% improvement RU**
500	87.06774142	32.00360423
1000	44.15607915	33.97964322
1500	30.17822479	51.90496498
2000	31.39820467	50.3440912
2500	120.7776104	39.61150457
3000	96.06459305	34.51346619
4000	141.771564	53.96887303

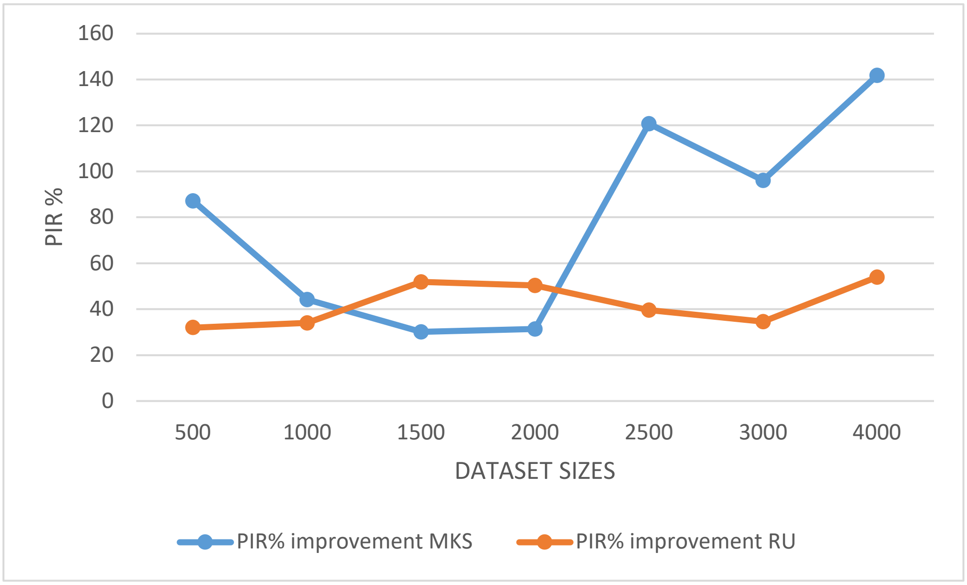


Figure 11: The Improvement of the Proposed HGHHM Algorithm over HGSWC algorithm for MKS & RU

REFERENCES

[1]. Abd Elaziz, M., Attiya, I., 2021. An improved Henry gas solubility optimization algorithm for task scheduling in cloud. Artif. Intell. Rev. 54, 3599–3637.

[2]. Abd Elaziz, M., Xiong, S., Jayasena, K.P.N., Li, L., 2019. Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution. Knowledge-Based Syst. 169, 39–52.

[3]. Abdulredha, M.N., Bara'a, A.A., Jabir, A.J., 2020. Heuristic and meta-heuristic optimization models for task scheduling in cloud-fog systems: A review. Iraqi J. Electr. Electron. Eng. 16, 103–112.

[4]. Abualigah, L., Diabat, A., 2021. A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. Cluster Comput. 24, 205–223. https://doi.org/10.1007/s10586-020-03075-5

[5]. Alboaneen, D., Tianfield, H., Zhang, Y., Pranggono, B., 2021. A metaheuristic method for joint task scheduling and virtual machine placement in cloud data centers. Futur. Gener. Comput. Syst. 115, 201–212.

[6]. Alkaam, N.O., Sultan, A.B., Hussin, M., Sharif, K.Y., 2023. The methods used for solving Task Scheduling Problem In Cloud Computing Environment Section : Research Paper The methods used for solving Task Scheduling Problem In Cloud Computing Environment The methods used for solving Task Scheduling Problem In Cloud 12, 12538–12544.

[7]. Alsaidy, S.A., Abbood, A.D., Sahib, M.A., 2022. Heuristic initialization of PSO task scheduling algorithm in cloud computing. J. King Saud Univ. Inf. Sci. 34, 2370–2382.

[8]. Amer, D.A., Attiya, G., Zeidan, I., Nasr, A.A., 2022. Elite learning Harris hawks optimizer for multi-objective task scheduling in cloud computing. J. Supercomput. 1–26.

[9]. Annie Poornima Princess, G., Radhamani, A.S., 2021. A hybrid meta-heuristic for optimal load balancing in cloud computing. J. grid Comput. 19, 21.

[10]. Attiya, I., Abd Elaziz, M., Xiong, S., 2020. Job scheduling in cloud computing using a modified harris hawks optimization and simulated annealing algorithm. Comput. Intell. Neurosci. 2020.

[11]. Al-Arasi, R., & Saif, A. (2020). Task scheduling in cloud computing based on metaheuristic techniques: A review paper. EAI Endorsed Transactions on Cloud Systems, 6(17), 162829. https://doi.org/10.4108/eai.13-7-2018.162829

[12]. Belgacem, A., Beghdad-Bey, K., 2022. Multi-objective workflow scheduling in cloud computing: trade-off between makespan and cost. Cluster Comput. 25, 579–595.

[13]. Banerjee, P., Roy, S., Sinha, A., Hassan, M. M., Burje, S., Agrawal, A., Bairagi, A. K., Alshathri, S., & El-Shafai, W. (2023). MTD-DHJS: Makespan-Optimized Task Scheduling Algorithm for Cloud Computing with Dynamic Computational Time Prediction. IEEE Access, 11(September), 105578–105618. https://doi.org/10.1109/ACCESS.2023.3318553

[14]. Chhabra, A., Singh, G., Kahlon, K.S., 2021. Multi-criteria HPC task scheduling on IaaS cloud infrastructures using meta-heuristics. Cluster Comput. 24, 885–918.

[15]. Choi, T.J., Togelius, J., Cheong, Y.-G., 2021. A fast and efficient stochastic opposition-based learning for differential evolution in numerical optimization. Swarm Evol. Comput. 60, 100768.

[16]. Chen, J., Gao, Y., Kasihmuddin, M. S. M., Zheng, C., Romli, N. A., Mansor, M. A., Zamri, N. E., & When, C. (2024). MTS-PRO2SAT: Hybrid Mutation Tabu Search Algorithm in Optimizing Probabilistic 2 Satisfiability in Discrete Hopfield Neural Network. *Mathematics*, *12*(5). https://doi.org/10.3390/math12050721

[17]. Das, S., 2023. Application of henry gas solubility optimization algorithm for short-term hydrothermal scheduling considering variable water transportation delay and penstock head loss. e-Prime-Advances Electr. Eng. Electron. Energy 6, 100287.

[18]. Dewangan, B.K., Jain, A., Choudhury, T., 2020. AP: Hybrid Task Scheduling Algorithm for Cloud. Rev. d'Intelligence Artif. 34, 479–485.

[19]. Duan, Q., Wang, L., Kang, H., Shen, Y., Sun, X., Chen, Q., 2021. Improved salp swarm algorithm with simulated annealing for solving engineering optimization problems. Symmetry (Basel). 13, 1092.

[20]. Dubey, K., Sharma, S.C., 2021. A novel multi-objective CR-PSO task scheduling

algorithm with deadline constraint in cloud computing. Sustain. Comput. Informatics Syst. 32, 100605.

[21]. Fu, X., Sun, Y., Wang, H., Li, H., 2023. Task scheduling of cloud computing based on hybrid particle swarm algorithm and genetic algorithm. Cluster Comput. 26, 2479–2488.

[22]. Gao, M., Zhu, Y., Sun, J., 2020. The multi-objective cloud tasks scheduling based on hybrid particle swarm optimization, in: 2020 Eighth International Conference on Advanced Cloud and Big Data (CBD). IEEE, pp. 1–5.

[23]. Haris, M., Zubair, S., 2022. Mantaray modified multi-objective Harris hawk optimization algorithm expedites optimal load balancing in cloud computing. J. King Saud Univ. Inf. Sci. 34, 9696–9709.

[24]. Heidari, A.A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H., 2019. Harris hawks optimization: Algorithm and applications. Futur. Gener. Comput. Syst. 97, 849–872.

[25]. Houssein, E.H., Gad, A.G., Wazery, Y.M., Suganthan, P.N., 2021. Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends. Swarm Evol. Comput. 62, 100841.

[26]. Hosseini Shirvani, M. (2020). A hybrid meta-heuristic algorithm for scientific workflow scheduling in heterogeneous distributed computing systems. Engineering Applications of Artificial Intelligence, 90(January), 103501. https://doi.org/10.1016/j.engappai.2020.103501

[27]. Hosseini Shirvani, M., & Noorian Talouki, R. (2022). Bi-objective scheduling algorithm for scientific workflows on cloud computing platform with makespan and monetary cost minimization approach. Complex and Intelligent Systems, 8(2), 1085–1114. https://doi.org/10.1007/s40747-021-00528-1

[28]. Jamaludin, S. Z. M., Romli, N. A., Kasihmuddin, M. S. M., Baharum, A., Mansor, M. A., & Marsani, M. F. (2022). Novel logic mining incorporating log linear approach. *Journal of King Saud University - Computer and Information Sciences*, *34*(10), 9011–9027. https://doi.org/10.1016/j.jksuci.2022.08.026

[29]. Kasihmuddin, M. S. M., Jamaludin, S. Z. M., Mansor, M. A., Wahab, H. A., & Ghadzi, S. M. S. (2022). Supervised Learning Perspective in Logic Mining. *Mathematics*, *10*(6), 1–35. https://doi.org/10.3390/math10060915

[30]. Kruekaew, B., Kimpan, W., 2020. Enhancing of artificial bee colony algorithm for virtual machine scheduling and load balancing problem in cloud computing. Int. J. Comput. Intell. Syst. 13, 496–510.

[31]. Kumar, M., 2022. Hybrid Cuckoo Search Algorithm for Scheduling in Cloud Computing. Comput. Mater. Contin. 71.

[32]. Liu, H., 2022. Research on cloud computing adaptive task scheduling based on ant colony algorithm. Optik (Stuttg). 258, 168677.

[33]. Lv, Z., Zhao, Y., Kang, H., Gao, Z., & Qin, Y. (2024). An Improved Harris Hawk Optimization Algorithm for Flexible Job Shop Scheduling Problem. Computers, Materials & Continua, 78(2), 2337–2360. https://doi.org/10.32604/cmc.2023.045826

[34]. Manoharam, G., Kassim, A. M., Abdeen, S., Kasihmuddin, M. S. M., Rusdi, N. A., Romli, N. A., Zamri, N. E., & Mansor, M. A. (2024). Special major 1,3 satisfiability logic in discrete Hopfield neural networks. *AIMS Mathematics*, *9*(5), 12090–12127. https://doi.org/10.3934/math.2024591

[35]. Mansouri, N., Ghafari, R., Zade, B.M.H., 2020. Cloud computing simulators: A comprehensive review. Simul. Model. Pract. Theory 104, 102144.

[36]. Mashaleh, A.S., Ibrahim, N.F.B., Al-Betar, M.A., Mustafa, H.M.J., Yaseen, Q.M., 2022. Detecting spam email with machine learning optimized with harris hawks optimizer (hho) algorithm. Procedia Comput. Sci. 201, 659–664.

[37]. Muthsamy, G., Ravi Chandran, S., 2020. Task scheduling using artificial bee foraging optimization for load balancing in cloud data centers. Comput. Appl. Eng. Educ. 28, 769–778.

[38]. Murad, S. A., Azmi, Z. R. M., Muzahid, A. J. M., Bhuiyan, M. K. B., Saib, M., Rahimi, N., Prottasha, N. J., & Bairagi, A. K. (2024). SG-PBFS: Shortest Gap-Priority Based Fair Scheduling technique for job scheduling in cloud environment. Future Generation Computer Systems, 150, 232–242. https://doi.org/10.1016/j.future.2023.09.005

[39]. Murad, S. A., Azmi, Z. R. M., Muzahid, A. J. M., Sarker, M. M. H., Miah, M. S. U., Bhuiyan, M. K. B., Rahimi, N., & Bairagi, A. K. (2024). Priority based job scheduling technique that utilizes gaps to increase the efficiency of job distribution in cloud

computing. Sustainable Computing: Informatics and Systems, 41(November 2023), 100942. https://doi.org/10.1016/j.suscom.2023.100942

[40]. Noorian Talouki, R., Hosseini Shirvani, M., & Motameni, H. (2022). A hybrid meta-heuristic scheduler algorithm for optimization of workflow scheduling in cloud heterogeneous computing environment. Journal of Engineering, Design and Technology, 20(6), 1581–1605. https://doi.org/10.1108/JEDT-11-2020-0474

[41]. Pradhan, A., Bisoy, S.K., Das, A., 2022. A survey on PSO based meta-heuristic scheduling mechanism in cloud computing environment. J. King Saud Univ. Inf. Sci. 34, 4888–4901.

[42]. Sa, S., Muhammed, A., Abdullahi, M., & Abdullah, A. (2021). An Enhanced Discrete Symbiotic Organism Search Algorithm for Optimal Task Scheduling in the Cloud. 1–24.

[43]. Sharma, K., Hassan, M. M., Pandey, S. K., Saini, M., Doriya, R., Ojha, M. K., Sinha, A., Kaushal, C., Bairagi, A. K., & Soliman, N. F. (2023). Cloud Based Multi-Robot Task Scheduling Using PMW Algorithm. IEEE Access, 11(January), 146003–146013. https://doi.org/10.1109/ACCESS.2023.3344459

[44]. Shukri, S.E., Al-Sayyed, R., Hudaib, A., Mirjalili, S., 2021. Enhanced multi-verse optimizer for task scheduling in cloud computing environments. Expert Syst. Appl. 168, 114230.

[45]. Si, T., Bhattacharya, D., Nayak, S., Miran da, P.B.C., Nandi, U., Mallik, S., Maulik, U., Qin, H., 2023. PCOBL: A Novel Opposition-based Learning Strategy to Improve Metaheuristics Exploration and Exploitation for Solving Global Optimization Problems. IEEE Access.

[46]. Singh, H., Tyagi, S., Kumar, P., 2020. Crow–penguin optimizer for multiobjective task scheduling strategy in cloud computing. Int. J. Commun. Syst. 33, e4467.

[47]. Tanha, M., Hosseini Shirvani, M., & Rahmani, A. M. (2021). A hybrid meta-heuristic task scheduling algorithm based on genetic and thermodynamic simulated annealing algorithms in cloud computing environments. In Neural Computing and Applications (Vol. 33, Issue 24). Springer London. https://doi.org/10.1007/s00521-021-06289-9

[48]. Yang, T., Fang, J., Jia, C., Liu, Z., & Liu, Y. (2023). An improved harris hawks optimization algorithm based on chaotic sequence and opposite elite learning mechanism. PLoS ONE, 18(2 February), 1–23. https://doi.org/10.1371/journal.pone.0281636

[49]. Velliangiri, S., Karthikeyan, P., Xavier, V.M.A., Baswaraj, D., 2021. Hybrid electro search with genetic algorithm for task scheduling in cloud computing. Ain Shams Eng. J. 12, 631–639.

[50]. Zamri, N. E., Mansor, M. A., Kasihmuddin, M. S. M., Sidik, S. S., Alway, A., Romli, N. A., Guo, Y., & Jamaludin, S. Z. M. (2024). A modified reverse-based analysis logic mining model with Weighted Random 2 Satisfiability logic in Discrete Hopfield Neural Network and multi-objective training of Modified Niched Genetic Algorithm. Expert Systems with Applications, 240(October 2023), 122307. https://doi.org/10.1016/j.eswa.2023.122307

[51]. Zakaria, B., Abouelmehdi, K., Beni-Hssane, A., & Khaloufi, H. (2021). NEW HYBRID ALGORITHM for TASK SCHEDULING in CLOUD COMPUTING. Journal of Theoretical and Applied Information Technology, 99(24), 6272–6279.

[52]. Manikandan, N., Gobalakrishnan, N., & Pradeep, K. (2022). Bee optimization based random double adaptive whale optimization model for task scheduling in cloud computing environment. Computer Communications, 187(January), 35–44. https://doi.org/10.1016/j.comcom.2022.01.016