Received: April 14, 2024. Revised: June 16, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

HSSAGWO Scheduler for Efficient Task Scheduling in an IaaS Cloud Computing Environment

Javid Ali Liakath[1] Pradeep Krishnadoss[2]* Manikandan Nanjappan[3] Bhavana Sivadas[2]

[1]St. Joseph's Institute of Technology, Chennai, Tamil Nadu, India
[2]Vellore Institute of Technology, Chennai, Tamil Nadu, India
[3]SRM Institute of Science and Technology, Chennai, Tamil Nadu, India
* Corresponding author's Email: pradeep.k@vit.ac.in

Abstract: Innovations in cloud technology over the recent years have shown tremendous growth. Apart from the need to possess a basic internet connection, another major hiccup for researchers in cloud computing is load balancing. It refers to the way resources are distributed and tasks are performed to achieve the most optimal utilization. Effective load balancing provides more user satisfaction. There are many algorithms developed to tackle the challenge of Load Balancing. An attempt is made in this paper to provide solution to this issue by developing an optimization technique that efficiently regulates the scheduler in assigning tasks to cloud resources such that optimal results are obtained. A hybridized Sparrow Search Algorithm – Grey Wolf Optimizer (HSSAGWO) had been proposed to optimize the task scheduling activity in cloud. The exploration and exploitation activities that are a part of original algorithms have been refined to achieve better performance in the proposed HSSAGWO algorithm. The performance efficiency of HSSAGWO algorithm had been ascertained by comparing it with Sparrow Search Algorithm (SSA), Grey Wolf Optimizer (GWO), Gravitational Search Algorithm (GSA), and Particle Swarm Optimization (PSO). Simulated experiments had been conducted using Cloudsim 3.0 tool for obtaining the results. The performance comparison had been carried out by considering the makespan, cost and response time parameters. The proposed HSSAGWO technique had produced an improvement of 9.31%,12.23%,15.55% and 17.95% for makespan when compared with SSA, GWO, GSA and PSO algorithms respectively when arrival rate is 10.

Keywords: Cloud computing, Scheduling, Makespan, Cost, Response time.

1. Introduction

One of the most active fields right now is Cloud computing. The Internet is the major factor that it depends on. Prime companies like Google, Amazon, and Microsoft support this model. Both hardware and software are used as computer resources (internet-based provision). Users pay for the services; they receive the information and services via mobile devices or computers very easily [1-4].

To assign all the processes and to handle the tasks efficiently, load balancing comes into play. This is done to maximize the rate of resource productivity. E.g.: processor load, the used memory, delays, or network load [5-8]. The efficiency is dependent on the load of the virtual machines. The system's performance is based on how well it distributes the demands among its resources i.e., the target is to deliver optimal quality of service (QoS).

NP-complete problems constitute those that involve tasks to be scheduled in real time on distributed platforms. Meta-heuristics come handy when abundant tasks get executed in the cloud avoiding any untoward happenings and to achieve better cloud performance. [9-11].

An efficient scheduling technique had been proposed in this work that is based on the GWO and SSA algorithms. These two algorithms, as individuals perform well under minimum requirements for fitness equation formulation and optimization. Both have high probability of getting stagnated under local optima and could not balance between the exploration and exploitation process to

Received: April 14, 2024. Revised: June 16, 2024.

identify the global optima. Hence hybridization is a preferred solution resulting in the formation of a potent optimization technique, especially for task scheduling activities in cloud.

SSA imitates the hunting behaviour of sparrows in arriving solutions to optimization type of problems. Though SSA is arguably the best for its search accuracy, faster convergence, higher stability and robustness, at farther stages of convergence it becomes flat and gets stagnated in local optima owing to its weaker exploitation technique. Thus, there are more chances of finding a non-ideal optimal global solution.

The original GWO had been widely practised by researchers for optimization problems. GWO emulates the hunting and dominating characteristics of grey wolves. Though widely preferred for its simplicity and ease of deployment, it too can become a victim of local optima due its poor exploration technique employed. This fact could be easily identified when GWO is applied for complex problems with multidimensional and unimodal attributes.

Hence, in this work we had integrated the original SSA and GWO to attain faster convergence while churning out optimal solution and maintain balance between the exploration and exploitation activities. The objectives of this research had been listed out below.

A Hybrid Sparrow Search Algorithm – Grey Wolf Optimizer (HSSAGWO) had been proposed to enhance the task scheduling performance in cloud.

A fitness function has been formalized by considering makespan, cost and response time QoS parameters.

Simulated experiments had been carried out using the Cloudsim 3.0 tool. The superiority of the proposed HSSAGWO had been ascertained by comparing its performance with SSA, GWO, GSA and PSO algorithms.

This research paper has been structured as follows: Section 2 handles the related literature. Section 3 throws light upon the proposed model and QoS parameters considered. Sections 4, 5 and 6 describe the features of Sparrows, Wolves and their mathematical modelling, and subsequent hybridization. Section 7 depicts the simulation experiment set-up and results obtained. Section 8 concludes the paper by providing tips for future expansion.

2. Related work

Researchers over the years had carried out numerous enhancements that accounted for the performance efficiency of the cloud environment. All such enhancements focussed on optimizing various QoS parameters. Cloud environment is evolutionary, where improvisation can be attained continuously by integrating existing multi-heuristic algorithms in such a way that the disadvantage of one algorithm is overturned by the advantage present in the other.

[12] Authors had come up with a new Particle Swarm Optimization-Bandwidth Aware divisible Task (PSO-BATS) that is scheduled with Multi-Layered Regression Host Employment (MLRHE) for reducing the complexity involved in the scheduling operation and balances the load effectively. However, while this method addresses several core issues, it still requires further validation in real-world scenarios to fully assess its scalability and adaptability to unpredictable cloud behaviour.

[13] Optimal resource consumption can be enforced through a variety of algorithms like Artificial Bee Colony, Genetic Algorithms, Bacteria Foraging, etc. Though they work well, there are certain limitations to each one of them. An EBGO algorithm is proposed to address these limitations by considering QoS parameters like cost, energy consumed, processing time, and response time. Despite their effectiveness, these algorithms tend to struggle with complex and dynamic load variations, often leading to inefficient resource utilization and potential increases in operational costs.

[14] A scheme named Resource and Deadline Aware Dynamic Load-Balancer (RADL) for Cloud had been proposed. The proposed technique works to make sure that all the tasks are executed within the deadline and that the new tasks added are accommodated. Task dismissal is minimized by implementing RADL. However, this method lacks support for dynamic, SLA-aware scheduling, crucial for linked tasks or those needing quick adjustments. RADL can also increase overhead for urgent new tasks, reducing system efficiency.

[15] A new artificial algorithm is brought about which helps in solving the problem of energy consumption. This is called deep reinforcement Q-learning. Terms like time, cost analysis, make-span, load balance, and deadline overflow are taken into consideration for comparative analysis. However, this technique is good but the training time for these models can be lengthy, impacting their practical deployment.

[16] A novel algorithm called Dual Conditional Moth Flame Algorithm (DC-MFA) has been introduced to achieve optimization and load balance. This concept considers multi-objective functions and hence looks into factors like make-span, CPU utilization, cost of resources, migration, security, and

consumption of energy. However, DC-MFA's effectiveness in environments with rapidly changing conditions may be limited due to its dependence on predetermined multi-objective functions.

[17] The Priority Based Load Balancing algorithm can be optimized by splitting the entire task prioritizing activity into four sub-activities. These include the allocation of tasks without starvation, moving the tasks to the dispatcher, re-arranging the order of tasks inside the queues and mapping them to appropriate virtual machines (VMs). While the effectiveness of this algorithm is contingent upon the precise coordination of these sub-activities and their adaptive responses to dynamic cloud environments.

[18] Load balancing aims to even out the overloaded and under-loaded nodes. An algorithm called Paired-Tree Algorithm is introduced to improve load balancing by making the performance of make-span and migration more efficient. However, further research is required to evaluate the scalability of the Paired-Tree Algorithm across diverse cloud computing environments and larger datasets.

[19] The authors had minimized the energy consumption and scheduled tasks with maximum computational load. The proposed Dynamic Replica model is found to have a more balanced storage space compared to other algorithms. However- the efficacy of the Dynamic Replica model is constrained by the initial configuration of the system and may not adapt well to rapid changes in task priorities or network conditions.

[20] The proposed Capacity Allocation algorithm lowers the total execution cost using joint load balancing. This algorithm had produced optimal results with minimal QoS. Despite its efficiency, the algorithm's performance may degrade under dynamic or unpredictable QoS demands.

[21] The authors had introduced a new algorithm namely Multi-Objective Service Brokering with Availability-Based Load Balancing (MOSB-ALB) to reduce cost and response time parameters. There are two ways of selection in MOSB-ALB. MOSB-ALB proves to surpass a few other cloud services in terms of minimized monetary cost and response time. However, the MOSB-ALB algorithm, while effective in improving cost efficiency and response times, may exhibit limitations in scalability and performance under varying cloud conditions and workloads not extensively discussed in this study.

[22] The authors had introduced an algorithm based on machine learning for balancing the load on the host machines (HMs). A learning agent receives a reward depending on the optimizing factor of its solution after execution. This helps the learning agent to keep training and testing till it reaches the most optimal solution. This algorithm has boosted the inter-HM load balance. Despite the algorithm's success in improving load balance, its dependency on continuous rewards and potentially high computational overhead could limit its practical application in environments with dynamic or unpredictable workloads.

[23] HDCBS is a task-scheduling model that looks into task sequencing and other factors like cost and load balancing. In HDCBS, queuing theory is implemented to get minimal the waiting time and response time. However, the practical implementation of HDCBS may face challenges in diverse cloud environments where dynamic factors and real-time changes in task characteristics can impact overall effectiveness. Authors [28-35] had suggested various optimized solutions for local search operations that were based on metaheuristics.

3. The proposed model

The performance of cloud depends on how it handles a large number of user requests. Resource sharing must be fair and efficient. Load balancing is one major challenge that cloud computing faces. There are various algorithms developed for solving this problem of load balancing. Our goal is to provide a good QoS using a suitable algorithm that also targets a minimum cost and response time. The objective function of the proposed algorithm is shown below:

$$\text{Objective function} =$$

$$\sum_{i=1}^{m} T_i \begin{pmatrix} \alpha \cdot \text{Makespan} + \\ \beta \cdot \text{Cost} + \gamma.\ \text{Response Time} \end{pmatrix} \quad (1)$$

Statement of the problem

Let n be the number of physical machines or any one machine having M number of virtual machines. Then cloud C will be:

$$C = \{\text{Physical}^{\text{Machine 1}}, \text{Physical}^{\text{Machine 2}} \text{Physical}^{\text{Machine N}}\} \quad (2)$$

A physical machine will have multiple virtual machines and that is represented as follows:$PMn = \{\text{Virtual}^{\text{Machine1}}, \text{Virtual}^{\text{Machine2}}, \text{Virtual}^{\text{Machine M}}\}$

The user function can be represented as follows:

$$Ui = \{T1, T2\ \ldots\} \quad (4)$$

We need to lower the costs and energy and increase resource exploitation and Quality of Service (QoS) for achieving optimal load balancing, failing which the cloud performance degrades. We hence propose a solution developed using the SSA & GWO algorithms. This system is represented in Fig. 1.

The parameters of service quality

The major factors that are vital in ensuring good QoS are make-span, cost and response time. The tasks are represented as {T1, T2, T3, …, Tn} where the items are dependent. The VMs are represented as {VM1, VM2, VM3, …, VMm} where the items are independent.

Execution Time: Let the time taken to process a task be:

$$A_1^T = \frac{1}{P_{max}^C \times N} \sum_{j=1}^{q} d_{jl \times e_{jl}} \quad (5)$$

In the equation given above, P_{max}^C stands for the maximum processing capacity, N represents the number of tasks, d_{jl} represents the distribution matrix and e_{jl} stands for the execution time.

Table 1. Notation used in HSSAGWO algorithm

Notations	**Description**
E_i	Time taken by a user per VM.
Ru_k	Number of requests collected from the user
PM_i	Physical machine i $1 \le i \le n$
VM	Virtual machine
P_{max}^C	Maximum processing capacity
N	Represents the number of tasks,
d_{jl}	Distribution matrix
e_{jl}	Execution time
V_i	Vi represents the cost of each virtual machine

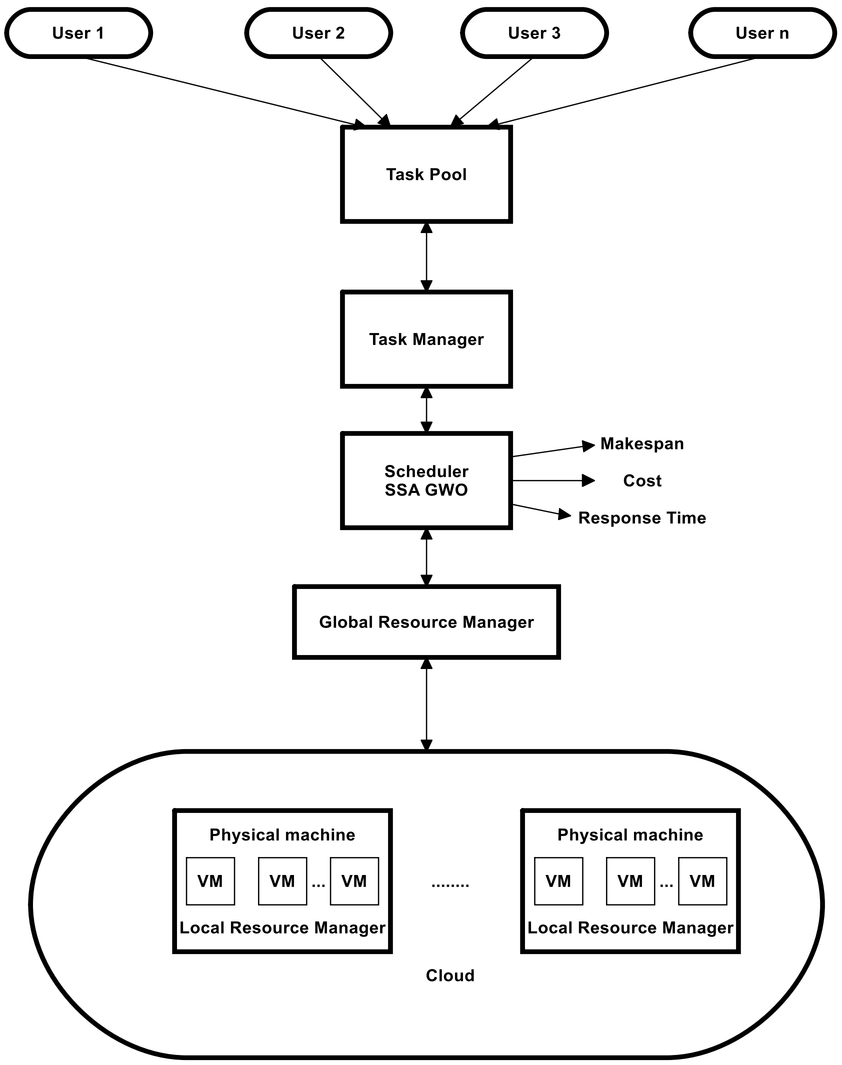

Figure. 1 Task Scheduling Architecture of HSSAGWO

Received: April 14, 2024. Revised: June 16, 2024.

Cost

It is defined as the money that a user pays for a virtual machine for each request sent. It is computed based on the factors of memory, processes, the bandwidth and the VMs that have been utilized. The equation is as follows:

$$\mathrm{Cost} = \sum_{i=1}^{N} V_i \times E_i \tag{6}$$

In this equation, N stands for the number of VMs, Vi represents the cost of each virtual machine and Ei is the time taken by a user per VM.

Response Time

It is defined as the scope of a service to be able to fulfil its requirement for a particular time period, under any circumstances. The equation is as follows:

$$RA_{RK} = \frac{WR_k}{Ru_k} \tag{7}$$

In this equation, Ru_k represents the number of requests collected from the user, and WR_k represents the work sent to Rk. This work is reallocated and is computed in milliseconds.

4. Proposed mathematical model and algorithm (SSA)

The Sparrow Search Algorithm is inspired from the behaviour of sparrows and hence its name. The sparrows possess high levels of energy reserves, identifying and spotting areas with plentiful food sources [24]. This efficiency is dependent on the fitness of each individual.

(i) When a sparrow sees its predator, it will start chirping to warn its flock. If the warning is more than the safety threshold, then they will follow the producers to a safe location.

(ii) The two levels in a flock are scroungers and producers. Each of the scroungers can turn into a producer depending on the food sources that it finds. Yet, the percentage of producers and scroungers will remain constant.

(iii) Producers have higher levels of energy than the scroungers. The scroungers fly to different locations for food because of starvation.

(iv) Producers are the leaders in a flock and the ones that provide the most efficient food source are followed by the scroungers.

(v) The sparrows in the middle walk closer to others in the flock, whereas the ones at the edge will move to the safe area to protect themselves from danger.

For our computation, we'll take virtual sparrows that search and find food. The matrix given below describes the location of the sparrows.

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,b} \\ a_{2,1} & a_{2,2} & a_{2,b} \\ a_{n,1} & a_{n,2} & a_{n,b} \end{bmatrix} \tag{8}$$

In this equation, b indicates the variables' dimension and n indicates the sparrow count. The value of b has to be optimized.

Efficiency is determined by assessing the fitness measure of the sparrows, as shown below:

$$Ga = \begin{bmatrix} g([a_{1,1} & a_{1,2} & a_{1,b}]) \\ g([a_{2,1} & a_{2,2} & a_{2,b}]) \\ g([a_{n,1} & a_{n,2} & a_{n,b}]) \end{bmatrix} \tag{9}$$

In this equation, n stands for the number of sparrows, each row in Ga gives the fitness value of each individual sparrow. We need to look for those producers that have better fitness according to SSA. They will have more priority to get food, and also, they are in charge of the entire flock. Hence, the producers have a larger scope of searching.

The producer's position is updated from rules (i) and (ii). The equation is as given below:

$$a_{i,j}^{t+1} = \begin{cases} a_{i,j}^{t} . \exp\left(\frac{-i}{\alpha . i_{max}}\right) & \text{if } Al_2 < Th \\ a_{i,j}^{t} + R.M & \text{if } \quad Al_2 \geq Th \end{cases} \tag{10}$$

In this equation, t stands for the current iteration, the value of j is 1, 2, …, d. $a_{i,j}^{t}$ is the measure in the jth dimension of the ith sparrow while carrying out the tth iteration and imax denotes a constant value holding the maximum number of iterations. α is a random number in the range of [0, 1]. Al_2 and Th indicate the alarm value and the safety threshold, where Al_2 has a value in between 0 and 1and Th is assigned a value in between 0.5 and 1.0. R denotes a random value that follows a normal distribution. M represents a matrix of dimension, 1 x d which has elements '1'.

When there aren't any predators, that is $Al_2 < Th$, the producer goes into wide search mode. If a sparrow detects the predator, that is $Al_2 \geq Th$, then the flock will fly to other safe location.

Some scroungers fight for food. If they win, they get the producer's food and continue with rule (v). The scrounger updates its location with the equation given as follows:

$$a_{i,j}^{t+1} = \begin{cases} R.\exp\left(\frac{a_{worst}^{t} - a_{i,j}^{t}}{i^2}\right) \text{ if } i > n/2 \\ a_p^{t+1} + \left|a_{i,j}^{t} - a_p^{t+1}\right|.Z^{+}.M \text{ otherwise} \end{cases} \quad (11)$$

In this equation, ap denotes the optimal position occupied by the producer. aworst stands for the present global worst location. Z represents a matrix of dimension 1 x d that has elements of either '1' or sometimes '-1', and Z+ equals ZT (ZZT)-1.

When i value is greater than n/2, then it indicates that the ith scrounger is starving because of very bad fitness value and needs to fly to other location. 'otherwise' case indicates that the sparrow seeking for food is closer to the best location.

We are assuming that the sparrows that are knowledgeable of the danger constitute 10-20 % of the flock. The mathematical equation for their initial positions based on rule (v) is as follows:

$$a_{i,j}^{t+1} = \begin{cases} a_{best}^{t} + \beta.\left|a_{i,j}^{t} - a_{best}^{t}\right| \quad \text{if } g_i > g_f \\ a_{i,j}^{t} + S.\left(\frac{\left|a_{i,j}^{t} - a_{worst}^{t}\right|}{(g_i - g_w) + \varepsilon}\right) \quad \text{if} g_i = g_f \end{cases} \quad (12)$$

In this equation, a_{best} is the present global optimal location and β denotes the step size control factor having normal distribution of randomized numbers, with an average value of 0 and a variance measure of 1. S is a just a random number in the range of −1 and 1. g_i denotes the fitness measure of the present sparrow. g_f and g_w represent the current best and worst measures of fitness globally. To overcome zero division error, the constant ε is included.

When $g_i > g_f$, the sparrow is positioned at the edge amongst others. abest is the safest position, usually at the centre. If $g_i = g_f$, then it indicates that the sparrow at the middle needs to move closer to the others to protect itself. Step size control coefficient and direction-determining factor of the sparrow is S.

The mutation technique implemented in the sparrow search algorithm impacts the speed and convergence accuracy. Even though this algorithm performs well to resolve intricate optimization problems, it has shortcomings in the form of reduced population range and fails in achieving accurate convergence. There are more chances of it to falling into the local optima, thereby the solution produced may not be always optimal.

5. Grey wolf optimization

Grey wolf, referred by the zoological name Canis Lupus are generally apex type of predators for being on top of their food chain [25]. They live in a packed habitat, with groups ranging from 5 to 12 on an average. They are led by both a male and female denoted by the name alphas that are responsible for making important decisions pertaining to hunting, leisure location, waking up time and other day to day happenings. The decision of alphas is upheld in the group with every other wolf showing its acceptance by keeping their tail down.

Beta variants are the second level in the group hierarchy. They assist their leaders, the alphas in making better decisions and other group management activities. They pass on the commands of the alpha to the group and provide advice to their superior alpha.

The lowest in the hierarchy are the omega variants. They always remain non-dominant amidst their gatherings and usually made as scapegoat. Delta types are the ones that obey the alpha and beta but remain dominant against omegas. The scout, sentinel, elder, hunter and care taker are a few varieties falling under this category.

Apart from the hierarchy they possess, prey hunting behaviour of the grey wolves can be categorized into three phases namely - tracking, chasing and approaching. The wolves continuously pursue, encircle and harass their prey and bring it to stationary mode, before launching their attack. In our proposed work, the hunting technique of these grey wolves and their social hierarchy had been mathematically modelled for designing the GWO operation and optimization purposes.

5.1 Mathematical modelling and proposed algorithm

The characteristic features of grey wolves like their social hierarchy, tracking, encircling and attack on prey had been mathematically modelled in this subsection.

5.1.1. Social hierarchy

The social hierarchy exhibited by the grey wolves had been mathematically modelled by fixing the solution provided by alpha (a) as the fittest one. The solutions produced by beta (b) and delta (d) are regarded as second and third best, respectively. Other candidate solutions produced are categorized under the omega (x). While optimizing the hunting section of the GWO algorithm a, b and d are taken into account. The x category follows a, b and d type of wolves.

5.1.2. Prey encircling

While hunting, grey wolves encircle their prey and the same can be mathematically modelled using the equations shown below:

$$\vec{A} = |\vec{P}.\vec{Y}_p(t) - \vec{Y}(t)| \quad (13)$$

$$\vec{Y}(t+1) = Y_p(t) - \vec{B}.\vec{A} \quad (14)$$

Where t indicates the current iteration and $\vec{B}$, $\vec{P}$ are coefficient vectors, the position vector of the prey is indicated by $\vec{Y}_p$ and $\vec{Y}$ indicates the grey wolf's position vector. Values of $\vec{B}$, $\vec{P}$ could be determined through equations shown below:

$$\vec{B} = 2\vec{b}.\vec{x_1} - \vec{b} \quad (15)$$

$$\vec{P} = 2.\vec{x_2} \quad (16)$$

Where x1 and x2 are random vectors, whose values are in the range [0, 1] and the value of $\vec{b}$ gets linearly decreased from 2 to 0 during each iteration.

A 2D position vector, along with a few neighbouring positions could be deduced from Eqs. (13) and (14). It can be seen that a grey wolf updates its initial position (X, Y) to a new location with respect to the prey's position (X*, Y*). The values of $\vec{B}$ and $\vec{P}$ can be adjusted to obtain different locations that are closer to the best agent, that are easily reachable from the present location. Like for instance, by fixing the values of $\vec{B}$ to (0, 1) and $\vec{P}$ to (1, 1), a new location (X*–X, Y*) can be reached. Eqs. (15) and (16) depict the various likely (updated) locations of the grey wolf in a 3D space. The wolves can change their present location to any random position arrived between the points that could be deduced using Eqs. (13) and (14) with the help of random vectors x1 and x2. Hence the grey wolves update their present position to any random position around the prey using Eqs. (13) and (14). An identical ideology could be applied to a search space of n dimensions, where the grey wolves' movement could likely be hyper-cubical or hyper-spherical type, towards the best solution that has been deduced till now.

5.1.3. Hunting

Alpha variety of grey wolves lead the hunting activity which could also involve the participation of beta and delta varieties occasionally. When the search space is deemed to be abstract, the optimum location of the prey is usually unknown. The hunting behaviour of grey wolves could be mapped mathematically, alpha (top candidate solution), beta and delta are expected to be familiar with the prey's location. The first three best solutions obtained till now are marked as reference, that are in turn used for instructing the other search agents (including omega variety) to adjust their present location in accordance to the best search agent's location. To model these activities mathematically, following equations have been proposed:

$$\overrightarrow{A_a} = |\overrightarrow{B_1}.\overrightarrow{Y_a} - \vec{Y}|, \overrightarrow{A_b} = |\overrightarrow{B_2}.\overrightarrow{Y_b} - \vec{Y}|, \overrightarrow{A_c} = |\overrightarrow{B_3}.\overrightarrow{Y_c} - \vec{Y}| \quad (17)$$

$$\overrightarrow{Y_1} = \overrightarrow{Y_a} - \overrightarrow{P_1}.(\overrightarrow{A_a}), \overrightarrow{Y_2} = \overrightarrow{Y_b} - \overrightarrow{P_2}.(\overrightarrow{A_b}), \overrightarrow{Y_3} = \overrightarrow{Y_c} - \overrightarrow{P_3}.(\overrightarrow{A_c}) \quad (18)$$

$$\vec{Y}(t+1) = \frac{\overrightarrow{Y_1} + \overrightarrow{Y_2} + \overrightarrow{Y_3}}{3} \quad (19)$$

The above equations convey how a search agent's location gets updated with respect to alpha, beta and delta varieties in the 2D search space. The final location is found at a random location within a circle that gets set according to the locations of alpha, beta and delta wolf varieties in the search space. That is, the prey's position is deduced by the alpha, beta and delta categories and the other varieties update their present position randomly around the prey.

5.1.4. Prey attacking (Exploitation)

As soon the prey ceases its movement, the grey wolves launch their attack on it. Mathematically this behaviour could be modelled by decreasing the measure of $\vec{b}$. It is to be noted at this juncture that the variation range of $\vec{B}$ too gets decreased by $\vec{b}$. Or more precisely, when the random value of $\vec{B}$ is in the range [-2b, 2b], $\vec{b}$ gets decreased from 2 to 0 for each iteration performed. When the random values of $\vec{B}$ are at [1, 1], the new position taken up by the search agent could be anywhere between its present and the position of the prey. When $|B| < 1$, the wolves launch their attack on the prey. The operators defined in the GWO algorithm permits the search agents to adjust their respective positions in accordance to the positions of alpha, beta and delta wolf varieties and carry out their attack on the prey. In spite of these operators present, the GWO algorithm gets stagnated in local solutions. Though the encircling process described promisingly describes the exploration task to few extents, more operators are still required in GWO to describe the exploration task in a more comprehensive manner.

Received: April 14, 2024. Revised: June 16, 2024.

5.1.5. Search for prey (exploration)

Searching of prey by the grey wolves is coordinated mainly in accordance to the positions of alpha, beta and delta wolf varieties. Even though they get swerved during searching, while attacking they get together. This swerving activity can be mathematically modelled by assigning random values to $\vec{B}$ which could be either greater than 1 or lesser than -1 to describe the swerving feature of the search agent. This adjustment leads to accentuating the exploration activity thereby favouring GWO to carry out a global search without getting stagnated locally. By fixing $|B| > 1$, the grey wolves get swerved from the prey in order to locate a healthier prey. $\vec{P}$ is the next component in GWO that assists in exploration. It could be witnessed from Eq. (14), that the $\vec{P}$ vector has random values in the range [0, 2]. This results in assigning random weights to the prey to randomly accentuate $P > 1$ or de-accentuate $P < 1$ to highlight the impact of prey in describing the distance in Eq. (11). This makes GWO to exhibit random behaviour throughout the optimization process, supporting the exploration task and significantly avoiding local optima. It has to be noted that the measure of P is not linearly decreased as that of B. P is required to deliver random values for accentuating the exploration task throughout i.e., randomization is carried out right from the initial iteration till the last iteration. Hence the values of P are of much more important to overcome the local optima stagnation, particularly during the final iterations.

The position of sparrows indicated in Eq. (9) needs to be integrated with the GWO Eq. (15) that describes the distance, by a new weighing factor Φ for upholding the final deduced values to near optimal measures. This has been described below:

$$\overrightarrow{A_a} = \left|\overrightarrow{B_1}.\overrightarrow{Y_a} - \Phi\vec{Y}\right|,$$
$$\overrightarrow{A_b} = \left|\overrightarrow{B_2}.\overrightarrow{Y_b} - \Phi\vec{Y}\right|, \overrightarrow{A_c} = \left|\overrightarrow{B_3}.\overrightarrow{Y_c} - \Phi\vec{Y}\right| \quad (20)$$

The probability factor for adjusting the location of every agent could then be deduced using the below equation:

$$\Omega\,[Y(t+1) - Y(t)] = \left|\frac{[Y(t+1)-Y(t)]}{\sqrt{[Y(t+1)-Y(t)]^2+1}}\right| \quad (21)$$

Where Ω is the probability factor. If the value of $[Y(t+1) - Y(t)]$ turns out to be positive, the sparrows succeed in getting the food. On the contrary, if negative, the sparrows start to swerve far away from the predators. If the search space is properly constructed, convergence is quickly achieved.

6. Proposed algorithm:

The specific steps involved in HSSAGWO are herewith enlisted:

Step 1: Initialize the sparrow search population and its parameters (total sparrow count n, maximum iteration imax and the total number of variables, d).

Step 2: Till the condition (t < imax) holds, sparrows are ranked based on their fitness values. The current best value that has been assessed is assigned to the minimum fitness value and the maximum fitness value will be the current worst value assessed.

Step 3: Update the discoverer sparrow's position using Eq. (10).

Step 4: When the value of ith individual in the ongoing iteration becomes lesser than or equal to the half of sparrow population, the position of the follower is adjusted using Eq. (11). Then proceed to step 9 else, the GWO can be executed.

Step 5: The values of b, B and P are initialized.

Step 6: Deduce the first best, second best and third best values of alpha, beta and delta wolf varieties.

Table 2. Simulation Environment

Entity Type	Parameters	Value
Task	Total Number of Tasks	100-500
	Length	[400,1000]MIPS
	File Capacity	[200,1000]MB
	Output File Capacity	[20,40]MB
Host	Available Memory (RAM)	1860 MIPs, 2660 MIPS
	Available Storage	10 GB
	Available Bandwidth	100 M/s
Virtual Machine	Total Number of VMs	50
	Policy Type	Time-shared
	VM RAM Capacity	512MB
	VMM Type	Xen
	OS	Linux
Data center	Total Number of CPUs	1 on each
	Total Number of Data centers	10
	Total Number of Hosts	10

Received: April 14, 2024. Revised: June 16, 2024.

Step7: The distance between the wolves and prey is then deduced by applying Eq. (17). Subsequently the new position is then deduced by applying Eqs. (18) and (19).

Step 8: Trade the positions of 3 best wolves with current sparrows.

Step 9: The position of follower is then adjusted using Eq. (11). Subsequently, the position of the investigator too is adjusted using Eq. (12).

7. Result and simulation experiment

A data center is created by implementing Cloudsim toolkit 3.0 and simulating the experiment. The table given represents the VM (Virtual Machine) in the data center and task [26].

7.1 Make-span evaluation (Arrival rate=10 &40)

In this simulation, 100 to 500 tasks are considered with an arrival time value of 10 in Fig. 3 and an arrival time value of 40 in Fig. 4 for evaluating the performance. The performance of the proposed HSSAGWO had been compared with SSA [24], GSA [27], PSO [12] and GWO [25] algorithms.

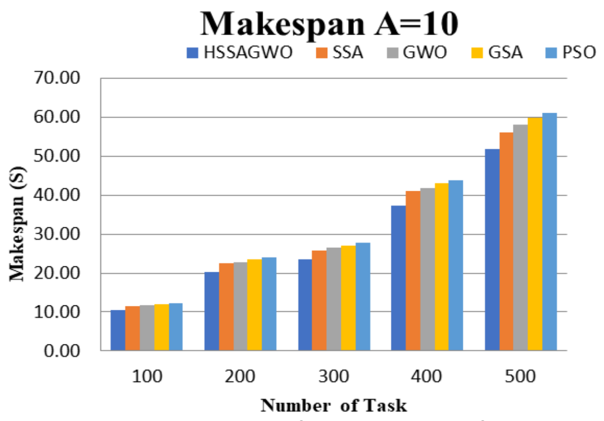

Figure. 3 Makespan for arrival time of 10

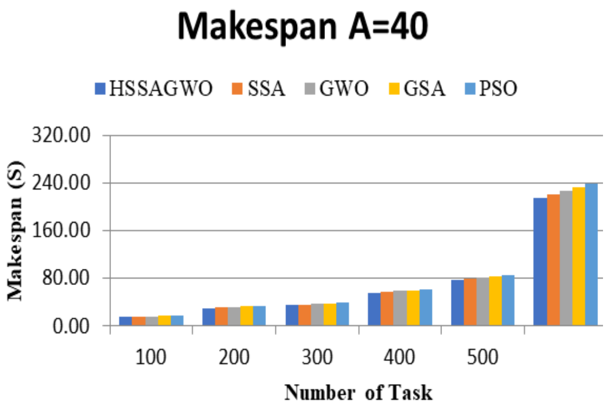

Figure. 4 Makespan for arrival time of 40

It could be inferred from the below Figs. 3 and 4 that the proposed HSSAGWO yielded better results for makespan for both the cases. The proposed HSSAGWO technique had produced an average improvement of 9.31%, 12.23%,15.55% and 17.95% for makespan when the arrival rate is 10 and for arrival rate of 40 the average makespan improvement obtained is 2.66%, 5.62%, 7.84% and 10.70% compared to SSA, GWO, GSA and PSO algorithms respectively.

7.2 Cost evaluation 200 & 500 tasks

In this simulation, the cost incurred for executing 200 and 500 tasks with varying deadlines ranging from 10 to 100 had been evaluated. It could be seen from the below Figs. 5 and 6 that the proposed HSSAGWO yielded minimum cost for both cases when compared to SSA, GWO, GSA and PSO algorithms

7.3 Response time 500 tasks

The below Fig 7 shows the response time comparison for the proposed HSSAGWO and SSA, GWO, GSA and PSO algorithms. For the 500 tasks with varying deadlines that were submitted, the proposed HSSAGWO had produced an improvement of 10%, 12.2%, 15.56% and 17.87% for response time when compared to other algorithms.

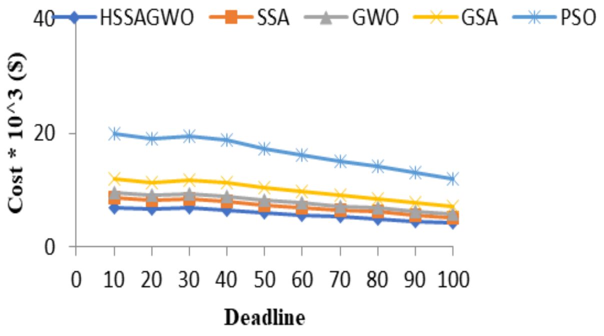

Figure. 5 Cost Comparison for 200 tasks

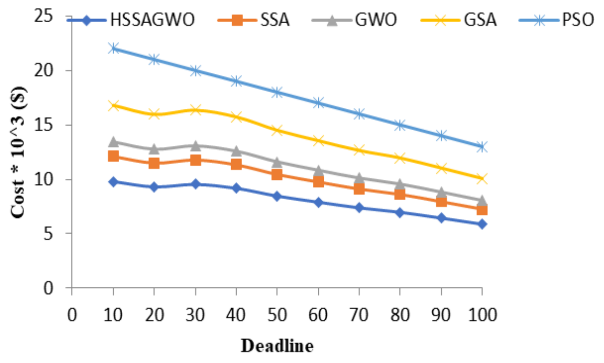

Figure. 6 Cost Comparison for 500 tasks

Received: April 14, 2024. Revised: June 16, 2024.

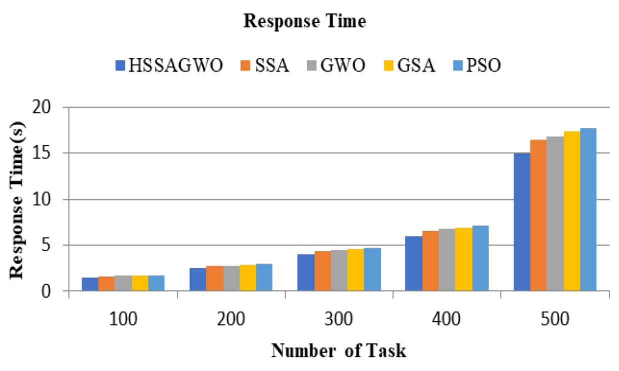

Figure. 7 Response time comparison

8. Conclusion

The Sparrow Search algorithm and Grey Wolf Optimization algorithm had been integrated and a Hybridized Sparrow Search Algorithm – Grey Wolf Optimizer (HSSAGWO) had been proposed. The HSSAGWO does not gets stagnated in local optima, thus exhibiting higher convergence speed and greater accuracy. The proposed algorithm exhibits optimal load balancing and excellent performance features with respect to QoS parameters makespan, cost and response time when compared to SSA, GWO, GSA and PSO algorithms, thus making it a viable solution for task scheduling in cloud environment. The proposed HSSAGWO technique had produced an improvement of 2.66%, 5.62%, 7.84% and 10.70% for makespan when compared with SSA, GWO, GSA and PSO algorithms respectively when arrival rate is 40. In future, additional QoS parameters can be included in the fitness equation while deriving the global optimal solution and the entire experimental domain can be shifted from simulated environment to the real cloud environment for evaluating the real time performance of HSSAGWO.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

"Conceptualization, Pradeep Krishnadoss and Javid Ali; methodology, Javid Ali; software, Manikandan Nanjappan; validation, Bhavana Sivadas; formal analysis, Javid Ali; investigation, Pradeep Krishnadoss; resources, Manikandan Nanjappan; data curation, Javid Ali; writing—original draft preparation, Javid Ali; writing—review and editing, Bhavana Sivadas; formal analysis, Javid Ali; investigation, Pradeep Krishnadoss; visualization, Manikandan Nanjappan; supervision, Javid Ali"

References

[1] X. Zuo, G. Zhang, and W. Tan, "Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud", *IEEE Transactions on Automation Science and Engineering*, Vol. 11, No. 2, pp.564-573, 2013.

[2] P. Krishnadoss, N. Pradeep, J. Ali, M. Nanjappan, P. Krishnamoorthy, and V. Kedalu Poornachary, "CCSA: Hybrid cuckoo crow search algorithm for task scheduling in cloud computing", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp.241-250, 2021, doi: 10.22266/ijies2021.0831.22.

[3] K. Pradeep, and TP. Jacob, "CGSA scheduler: A multi-objective-based hybrid approach for task scheduling in cloud environment", *Information Security Journal: A Global Perspective*, Vol. 27, No. 2, pp. 77-91, 2018.

[4] K. Pradeep, LJ. Ali, N. Gobalakrishnan, CJ. Raman, and N. Manikandan, "CWOA: hybrid approach for task scheduling in cloud environment", *The Computer Journal*, Vol. 65, No. 7, pp.1860-1873, 2022.

[5] P. Krishnadoss, C. Chandrashekar, and VK. Poornachary, "RCOA Scheduler: Rider Cuckoo Optimization Algorithm for Task Scheduling in Cloud Computing", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 5, pp.1-10, 2022, doi: 10.22266/ijies2022.1031.44.

[6] FS. Prity, MH. Gazi, and KMA. Uddin, "A review of task scheduling in cloud computing based on nature-inspired optimization algorithm", *Cluster computing*, Vol. 26, No. 5, pp. 3037-3067, 2023.

[7] P. Krishnadoss, V. K. Poornachary, P. Krishnamoorthy and L. Shanmugam, "Improvised Seagull Optimization Algorithm for Scheduling Tasks in Heterogeneous Cloud Environment", *Computers, Materials & Continua*, Vol. 74, No. 2, 2023.

[8] S. Chakraborty, A. K. Saha and A. Chhabra, "Improving whale optimization algorithm with elite strategy and its application to engineering-design and cloud task scheduling problems", *Cognitive Computation*, Vol. 15, No. 5, pp.1497-1525, 2023.

[9] S. Mangalampalli, G. R Karri and U. Kose, "Multi Objective Trust aware task scheduling algorithm in cloud computing using Whale Optimization", *Journal of King Saud University-Computer and Information Sciences*, Vol. 35, No. 2, pp.791-809, 2023.

[10] A. Y. Hamed, M. K. Elnahary, F. S. Alsubaei and H. H. El-Sayed, "Optimization Task Scheduling Using Cooperation Search Algorithm for Heterogeneous Cloud Computing Systems", *Computers, Materials & Continua*, Vol. 74, No. 1, 2023.

[11] P. Pirozmand, H. Jalalinejad, A. A. R. Hosseinabadi, S. Mirkamali and Y. Li, An improved particle swarm optimization algorithm for task scheduling in cloud computing, *Journal of Ambient Intelligence and Humanized Computing*, Vol. 14, No. 4, pp.4313-4327, 2023.

[12] AR. Shaheen, and SS. Kumar, "Tasks Scheduling in Cloud Environment Using PSO-BATS with MLRHE", *Intelligent Automation & Soft Computing*, Vol. 35, No. 3, pp.1-16, 2023.

[13] P. Velpula, R. Pamula, "EBGO: an optimal load balancing algorithm, a solution for existing tribulation to balance the load efficiently on cloud servers", *Multimedia Tools and Applications*, Vol. 81, No. 24, pp.34653-34675, 2022.

[14] S. Nabi, M. Aleem, M. Ahmed, MA. Islam, MA. Iqbal, "RADL: A resource and deadline-aware dynamic load-balancer for cloud tasks", *The Journal of Supercomputing*, Vol. 78, No. 12, pp.14231-14265, 2022

[15] J. Uma, P. Vivekanandan, S. Shankar, "Optimized intellectual resource scheduling using deep reinforcement Q-learning in cloud computing", *Transactions on Emerging Telecommunications*, Vol. 33, No. 5: e4463, 2022.

[16] G. Verma, "Secure VM migration in cloud: Multi-criteria perspective with improved optimization model", *Wireless Personal Communications*, pp. 1-28, 2022.

[17] G.Sharma, N. Miglani, and A. Kumar, "PLB: a resilient and adaptive task scheduling scheme based on multi-queues for cloud environment", *Cluster Computing*, Vol. 24, No. 3, pp.2615-2637, 2021.

[18] UK. Sonangeri Pushpavati, DA. D'Mello, "A tree based mechanism for the load balancing of virtual machines in cloud environments", *Journal of Information Technology*, Vol. 13, pp.911-920, 2021.

[19] C. Li, J. Liu, B. Lu, and Y. Luo, "Cost-aware automatic scaling and workload-aware replica management for edge-cloud environment", *Journal of Network and Computer Applications*, Vol. 180, 2021.

[20] D. Ardagna, M. Ciavotta, R. Lancellotti, M. Guerriero, "A hierarchical receding horizon algorithm for QoS-driven control of multi-IaaS applications", *IEEE Transactions on Cloud Computing*, Vol. 9, No. 2, pp.418-434, 2018.

[21] M. A. Khan, "An Effective Low-Cost Cloud Service Brokering Approach for Cloud Platforms", *Arabian Journal for Science and Engineering*, Vol. 45, No. 12, pp. 10653-10668, 2020.

[22] A. Ghasemi, A. Toroghi Haghighat, "A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning", *Computing*, Vol. 102, pp.2049-2072, 2020.

[23] W. Cai, J. Zhu, W. Bai, W. Lin, N. Zhou, and K. Li, "A cost saving and load balancing task scheduling model for computational biology in heterogeneous cloud datacenters", The *Journal of Supercomputing*, Vol. 76, pp.6113-6139, 2020.

[24] J. Xue, and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm", *Systems Science & Control Engineering*, Vol. 8, No. 1, pp. 22-34, 2020.

[25] S.Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer", *Advances in Engineering Softwar*, Vol. 69, pp.46-61, 2014.

[26] RN. Calheiros, R. Ranjan, A. Beloglazov, CAF. De Rose, R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms", *Software: Practice and Experience*, Vol. 41, No. 1, pp. 23-50, 2011.

[27] E. Rashedi, H. Nezamabadi-Pour and S. Saryazdi, "GSA: a gravitational search algorithm", *Information Sciences*, Vol. 179, No. 13, pp.2232-2248, 2009.

[28] PD. Kusuma, and A. Dinimaharawati, "Extended stochastic coati optimizer", *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 3, pp.482-494, 2023, doi: 10.22266/ijies2023.0630.38.

[29] PD. Kusuma, and A. Dinimaharawati, "Swarm Bipolar Algorithm: A Metaheuristic Based on Polarization of Two Equal Size Sub Swarms", *International Journal of Intelligent Engineering & Systems*, Vol. 17, No. 2, pp.1-13, 2024, doi: 10.22266/ijies2024.0430.31.

[30] P. D. Kusuma and M. Kallista, "Swarm Space Hopping Algorithm: A Swarm-based Stochastic Optimizer Enriched with Half Space Hopping Search", *International Journal of Intelligent Engineering & Systems*, Vol. 17, No. 2, 2024, doi: 10.22266/ijies2024.0430.54.

[31] P. D. Kusuma and M. Kallista, "Migration-Crossover Algorithm: A Swarm-based Metaheuristic Enriched with Crossover

Technique and Unbalanced Neighbourhood Search", International *Journal of Intelligent Engineering & Systems*, Vol. 17, No. 1, 2024, doi: 10.22266/ijies2024.0229.59.

[32] P. D. Kusuma, and A. Dinimaharawati, "Four Directed Search Algorithm: A New Optimization Method and Its Hyper Strategy Investigation", *International Journal of Intelligent Engineering & Systems*, Vol. 16, No. 5, 2023, doi: 10.22266/ijies2023.1031.51.

[33] P. D. Kusuma and A. Novianty, "Total Interaction Algorithm: A Metaheuristic in which Each Agent Interacts with All Other Agents", *International Journal of Intelligent Engineering & Systems*, Vol. 16, No. 1, 2023, doi: 10.22266/ijies2023.0228.20.

[34] P. D. Kusuma and A. L. Prasasti, "Walk-Spread Algorithm: A Fast and Superior Stochastic Optimization", *International Journal of Intelligent Engineering & Systems*, Vol. 16, No. 5, 2023, doi: 10.22266/ijies2023.1031.24.

[35] P. D. Kusuma, P. D. and F. C Hasibuan, "Attack-Leave Optimizer: A New Metaheuristic that Focuses on The Guided Search and Performs Random Search as Alternative", *International Journal of Intelligent Engineering & Systems*, Vol. 16, No. 3, 2023, doi: 10.22266/ijies2023.0630.19.