March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

A Hybrid Slap Swarm and Harris Hawks Optimization for Secure Task Scheduling in Cloud Environment based on Multi-Objective Function

[*1]Srinivas Mudepalli, [2]Dr.V. SrinivasaRao and [3]Dr.Reddi Kiran Kumar
[*1]*Research Scholar, Department of Computer science,Krishna University, Machilipatnam, India.*[2]*Professor and Head, Department of Computer Science and Engineering,VELTECH RangarajanDr.Sagunthala R&D Institute of Science & Technology, Chennai.*
[3]*Department of Computer Science, Krishna University,Machilipatnam, India.*

Article Info
Volume 83
Page Number: 2743 – 2762
Publication Issue:
May - June 2020

Article History
Article Received*: 11August 2019*
Revised*: 18November 2019*
Accepted*: 23January 2020*
Publication*: 10May2020*

***Abstract*:**
Task scheduling is the prime significant parameter in cloud computing (CC) which plays a major role in the effectiveness of the entire CC facilities. In the cloud environment (CE), the submitted tasks are executed on proper time using the accessible resources in order to accomplish appropriate resource utilization, efficiency and low makes pan which involves effective task scheduling (TS) procedure for accurate task distribution. This paper develops a hybrid slap swarm and Harris hawks optimization for secure Task Scheduling in Cloud Environment (TSCE) based on multi-objective (MO) function. For improving the scheduling process, Slap swarm (SS) and Harris hawks' optimization (HHO) procedure is hybridized as SSHHO to solve the optimization difficult. These two algorithms are successfully merged to perform the task allocation process. The proposed SSHHO algorithm considers the multi-objective function like makes pan, resource utilization, energy consumption, final task weight and round trip time latency for the scheduling process. Using the Clouds environment, the proposed SSHHO algorithm is evaluated and the performance of the proposed system is related to the existing algorithms of CSPO, HGAACO, FMPSO, MSDE and HGPSO with different multi-objective constraints. By evaluating the outcome, the developed SSHHO process achieves maximum efficiency, less energy consumption, low makes pan, and maximum resource utilization compared to existing algorithms.

Keywords: *Cloud computing, makes pan, task scheduling, Virtual machine, Slap swarm, and Harris hawks optimization.*

1. INTRODUCTION

Cloud computing turns out to be increasingly proliferating in the business, the scholarly community, and society via the pervasive development of the Internet access and enormous information in their capacity, speed, and assortment. A collection of autonomic computing, grid computing, disseminated computing, and utility computing is termed as cloud computing [1]. For the provision supplier, the cloud term is utilized in the CC that holds all kinds of resources for computing, storage, etc. CC has

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

numerous difficulties out of the different concerns in cloud planning that assumes a significant job in deciding the effective execution [2]. Administrations are managed in the types of IaaS (Infrastructure-as-a-Service), SaaS (Software-as-a-Service) and PaaS (Platform-as-a-Service). In the cloud, service provider host applications are available to users is termed as SaaS. A hardware and software apparatus are available to clients in PaaS. Highly accessible virtualized computing resources are delivered by IaaS [3]. Distributed computing can be characterized as a sort of dispersed framework which includes the virtualized PCs that are effectively managed.

Cloud platforms enable initiatives to lease computing power with the support of the virtualization method. Since hundreds of thousands of VMs are utilized, it is challenging to manually assign tasks to computing resources in clouds. Accordingly, the complications of CC is how to perform the planning of task and resource distribution [4]. The presentation of scheduling procedure to progress QoS (quality of service) in cloud surroundings. In the distributed system, the objective of scheduling is distribution the load on mainframes and improving their application by reducing the entire task execution time. To enhance flexible and reliable systems, task scheduling performs a major part in cloud [5]. The objective is to assign tasks to the flexible assets with a short time. Tasks can be performed beneath business logic constraints. Two types of scheduling procedure classified such as static and dynamic scheduling algorithms. The dynamic scheduling method has high enactment than static procedure but has a lot of overhead parallel to it [6].

The task planning problem is a NP-complete issue. The aim of task planning is to lessen the waiting time of task and expand the utilization of resources. By preserving the balance between fairness and productivity, dispatchers should schedule tasks to develop the QoS [7]. The incredible improvement of virtualization and the CC method reflects the increasing need for VM services. Based on the QoS necessities of CC centers and users, the key question is how to allocate user's tasks effectively and sensibly to different users [8]. The goal of task scheduling is to attain high system throughput, enhance the load balance and decrease the completion period with obtainable virtualized resources [9]. Based on the task scheduling, a set of tasks to be allocated to the VMs. Tasks planning over the CC resources are the most significant part because the user will have to pay for resource use on the basis of the time.

Scheduling denotes the distribution of the tasks to the VMs. For the proper consumption of the supporting resources, an effective task planning is always needed in CC surroundings [10]. Scheduling of tasks supports to preserve the make span of the VMs. In addition, by providing the QoS, it conserving the situations of SLA (service level agreement) to the clients [11]. In CC, the

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

requests have to meet the numerous requirements of resources and planning is the main challenging because of the dynamic environment of CS. Improper task scheduling methods can decrease the throughput of the entire CS and improve the task's execution time [12]. The meta-heuristic algorithm characteristic implements superior to the heuristic and scientific schemes. In clouds, some of the familiar approaches for resolving the problems of TS are Genetic Algorithm (GA) [13], Water drop Algorithm (WDA) [14], League championship Algorithm (LCA) [15], Simulated Annealing (SA) [16], Particle Swarm Optimization (PSO) [17], Firefly Algorithm (FA) [18], Ant Colony Optimization (ACO) [19] and Whale Optimization algorithm (WOA) [20]. The existing methods are not ready to obtain ideal task distribution as far as various QoS constraints due to low convergence rate, poor search capability, and inability. To solve these difficulties, the hybrid structure has been developed to accomplish effective task distribution outcomes.

In this paper, presented a MO based task allocation using a grouping of slap swarm algorithm (SSA) [26] and Harris hawks' optimization (HHO) [27] algorithm. The motivation of SSA is the crawling nature of slaps once steering and hunting in oceans. It effectively improves the random initial solutions and easily converged to the optimal value. The motivation of HHO is the supportive conduct and hunting stylishness of Harris hawks in the environment termed surprise pounce. Energy consumption, make span, round trip time latency, final task weight, and resource utilization are considered as a multi-objective function. According to these multi-objective functions, task scheduling is performed.

The major contributions of the paper as follow:

This paper presents the combination of slap swarm and Harris hawks optimization algorithm hybridized as SSHHO for task scheduling depends on the multi-objective utility. This scheduling technique can obtain the ideal solutions and it has the benefit of easily converged compared to the existing algorithms. In cloud computing, the task scheduling considers the multi-objective problems like make span, energy consumption, round trip time latency, final task weight and utilization of resources. The performance of the SSHHO algorithm is equated to the earlier algorithms with different multi-objective parameters.

The rest of the paper is prepared as trails: The related works are stated in Section 2. Section 3 delivers the system model, problem construction and hybrid SSHHO method for multi-objective task scheduling. Section 4 shows the simulation results. As a final point, conclusion and future scope are defined in section 5.

2. RELATED WORKS

Recent related works are given below:

Using HGPSO procedure, a task scheduling was presented by Senthil and Venkatesan [21] in cloud computing surroundings. Optimization methods are

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

mostly applied to solving NP-hard issues. In the file director, consumer jobs are stored in the developed scheme. Once the task is repeated the priority is computed and proper resources are allotted for the task. In the on-demand queue, new tasks are examined and stored. The input of HGPSO process is the output of on-demand queue. GA and PSO methods are combined to execute HGPSO method. In the on-demand queue, it estimates the available resources for the user tasks.

According to the hybrid moth hunt and differential evolution, task scheduling was developed by Mohamed et al [22] in cloud computing. In VMs, with balanced task delivery, moth search based DE method defined as MSDE is utilized for task planning. DE procedure is employed as local search method and the exploration capability of MSA could be enhanced. Using the functions of GA and ES algorithm, the DE method has been performed. These functions are high populace of GA and self-adapting transformation of ES. According to the functions, the performance is enhanced than other MH methods.

The MO based TS in cloud surrounding was introduced by Senthil and Venkatesan [23] using HGA-ACO. For task distribution, the efficient multi-objective HGA-ACO is utilized to manage the huge cloud consumer's demands. Service based scheduler finds the order of task and available resources to be planned. Based on the throughput, completion period and response period, the HGA-ACO deliberates the function based scheduler yield and discovers the better job distribution scheme. The HGA-ACO procedure merges GA and ACO processes. GA method initializes the efficient pheromone for ACO. For crossover operation of GA, ACO is utilized to improve the GA solution.

Using CPSO methods, a multi-objective ideal task planning was presented by Perm and Pradeep [24] in cloud surroundings. CPSO algorithm combines the Cuckoo search (CS) and PSO. The process of CPSO is to decrease the cost, violation rate, make span, and deadline. Using the Clouds toolkit, the enactment of the developed CPSO process is estimated. The proposed systems reduce the violation rate, cost, deadline, and make span than ACO, FCFS, MIN-MIN, and PBACO.

A hybrid task scheduling method using a modified PSO and fuzzy approach was described Najme et al [25] for cloud computing. The proposed FMPSO combines modified PSO and fuzzy concepts to improve cloud throughput and load balancing. FMPSO method considers the four improved velocity developing schemes and roulette wheel assortment method to improve the overall exploration ability. To solve the local bests in PSO, it uses mutation and crossover operators. For fitness calculation, this method using fuzzy inference system. Speed of CPU, RAM size, task length and entire finishing time are the input factors for the developed fuzzy system. Through the fuzzy concept, FMPSO is utilized to lessen the resource usage and the completing time.

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

After investigation of the works in literature, it has been seen that exiting methods are well prepared for task planning. However, every work has a certain drawback, for example, high energy consumption, make span, low QoS constraints, and complexity. To resolve these difficulties, in this paper, a hybrid algorithm for multi-objective based TS is developed.

3. SYSTEM MODEL

The task scheduling aims to allocate the task to equivalent VMs of cloud service supplier with minimum fitness function. Figure 1 displays the structure of task scheduling in cloud. At first, the user interface received the request from the user. By the user interface element, the application request is expected from the cloud user. Then the user interface sends the tasks to the request manager. All user's request are accomplished by the request manager. Through the resource observer, the cloud resource pool like memory, storage and CPU is monitored. Then the task is given to the scheduler. Based on the performance of VMs, the process of scheduling happens through the constrained task. The scheduler creates a rule to allocate every task to the equivalent VM based on the data obtained from the resource monitor and request manager. The aim of the scheduler is to allow the task to the resultant VM by minimizing fitness function based on available resources. This is a significant issue if task scheduling. During the task scheduling, the routing problem occurs and sequence the task on the VMs to reduce make span, energy consumption and improve resource utilization. To solve this difficulty, this paper develops a hybrid task scheduling algorithm.

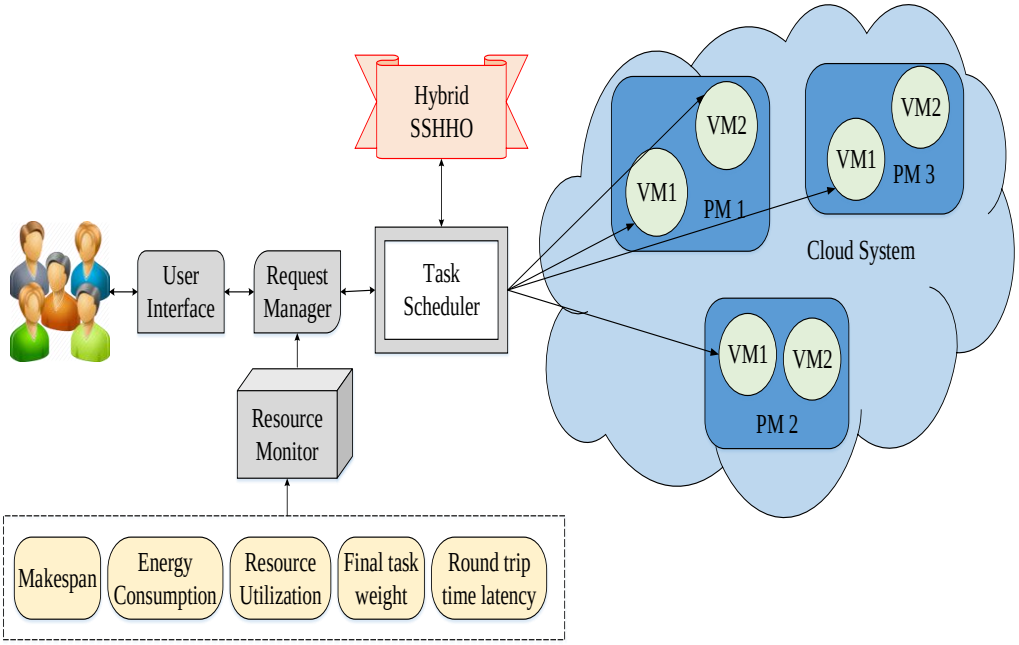

Fig. 1. Scheduling of task in CE

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

3.1 Task Scheduling Problem

In the cloud environment, the problem of scheduling is how to assign the multiple tasks to different VMs with minimum execution time. Consider the cloud system (CS) that contains a number of PMs (physical machines).

$$CS = \{PM_1, ..., PM_i, ..., PM_{N_{pm}}\} \quad (1)$$

Where $PM_i (i = 1, ..., N_{pm})$ represented the PMs available in the cloud. Every PM contains the number of VMs.

$$PM_i = \{VM_1, ..., VM_k, ..., VM_{N_{vm}}\} \quad (2)$$

Where $VM_k (k = 1,2, ..., N_{vm})$ denotes the k^{th} VM. The number of VMs are defined by N_{vm}. The component VM_k is defined as:

$$VM_k = \{SIDV_k, MIPS_k\} \quad (3)$$

Where the serial number of VMs is defined by $SIDV_k$ and $MIPS_k$ (millions of instructions per second) is the data processing speed of VMs.

$$T = \{Task_1, ..., Task_l, ..., Task_{N_{tsk}}\} \quad (4)$$

Where the number of tasks is denoted by N_{tsk}. In the task series, the l^{th} task is represented by $Task_l$. The factor $Task_l$ is represented as:

$$Task_l = \{SIDT_l, task\_length_l, ECT_l, PI_l\} \quad (5)$$

Where the sequential amount of tasks are denoted by $SIDT_l$, the length of task is defined by $task\_length_l$ and ECT_l is the expected completion time for them $Task_l$. PI_l Defines the task priority of the number of tasks. ECT matrix of size $N_{tsk} \times N_{vm}$ defines the execution time to run the task on each VM.

$$ECT = \begin{bmatrix} ECT_{1,1} & ECT_{1,2} & \cdots & ECT_{1,N_{vm}} \\ ECT_{2,1} & ECT_{2,2} & \cdots & ECT_{2,N_{vm}} \\ \vdots & \vdots & \ddots & \vdots \\ ECT_{N_{tsk},1} & ECT_{N_{tsk},2} & \cdots & ECT_{N_{tsk},N_{vm}} \end{bmatrix} \quad (6)$$

$$ECT_{lk} = \frac{task\_length_l}{MIPS_k}, k = 1,2, ..., N_{vm}, l = 1,2, ..., N_{tsk} \quad (7)$$

Where ECT_{lk} denotes to the finishing period of l^{th} task on k^{th} VM.

3.2 Multi-Objective Function For Task Scheduling

Scheduling of tasks is necessary when many clients demand identical resource concurrently. With different computing speeds, the number of tasks to allocate on several VMs. Each and every task can be scheduled on any VM. Every VM can implement numerous tasks. The proposed SSHHO algorithm reduces the fitness value by considering the multi-objective functions such as the energy consumption (EC), make span (M), final task weight (FTw), round trip time latency (RTTL) and resource utilization (RU). The key objective of the proposed method is to assign all tasks to all VMs by considering the energy consumption, make span, round trip time latency, resource utilization, and final task weight.

3.2.1 Make Span (M)

It is a significant element for the performance of the scheduling schemes. Using the ECT matrix, the makespan is

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

estimated on VM by finding the maximum ECT from allocated tasks. Hence, the low makespan value provides the best and efficient task scheduling of VMs.

$$makespan = \max_{j \in M} \{ \max_{x \in assigned\_tasks} ECT(x, j) \} \quad (8)$$

3.2.2 Resource Utilization (RU)

In a cloud, certain tasks need several CPU resources while other tasks demand fewer CPU resources and more storage. RU can be obtained by the following equation:

$$RU = \frac{1}{4}[CPU_{cost} + Memory_{cost} + Storage_{cost} + Bandwidth_{cost}] \quad (9)$$

$$CPU_{cost} = CPU_{base} * CPU\ of\ VM * runtime_{cost} + CPU_{Tran}\ cost \quad (10)$$

Here, the base cost of CPU is CPU_{base} when a resource is employed by the lowermost utilization, and the cost of CPU transmission defines the $CPU_{Tran}\ cost$.

$$Memory_{cost} = memory_{base} * Memory\ of\ VM * runtime_{cost} + memory_{Tran}\ cost \quad (11)$$

Likewise, $memory_{base}$ is the base cost of memory, and the cost of transmission memory is defined by $memory_{Tran}\ cost$.

$$Storage_{cost} = Storage_{base} * Storage\ of\ VM * runtime_{cost} + Storage_{Tran}\ cost \quad (12)$$

The base cost of storage is denoted by $Storage_{base}$, the cost of storage transmission is represented by $Storage_{Tran}\ cost$.

$$Bandwidth_{cost} = Bandwidth_{base} * Bandwidth\ of\ VM * runtime_{cost} + Bandwidth_{Tran}\ cost \quad (13)$$

The cost of transmission bandwidth (BW) is denoted by $Bandwidth_{Tran}$ and $Bandwidth_{base}$ is the base cost of BW.

3.2.3 Energy Consumption (Ec)

The amount of energy utilized by a task T_i in VM_j can be computed as:

$$E_{cost}(j) = ECT_{ij} * PO_{exe}^{j} \quad (14)$$

Where ECT_{ij} indicates expected completion time and PO_{exe}^{j} is an energy consumption during task execution. The total energy utilized to finish all tasks by VM_j can be given as:

$$Total\ Energy(j) = \sum_{j=1}^{|VM|} E_{cost}(j) \quad (15)$$

3.2.4 Final Task Weight (Ftw)

It is the main component that is affected by prioritization of tasks. It contains two constraints like task weight and client type. The privilege classes of clients are defined by client types. It consists of a number of classes, for instance, class A, class B, class C, and class D. The client should participate as a member if they decide to join previous classes by paying a fees and they using the monthly or weekly offer which is depending on the tasks size and amount of resources. Every task has a priority weight that provides the high, medium and low priority as shown in table 1. The priority weights are 0.4, 0.3, 0.2, and 0.1. The summation of these values are equal to one.

$$FTw = client type * Task\ weight \quad (16)$$

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

Table 1: Weights for four tasks

Classes	Class A	Class B	Class C	Class D
Priority	Urgent	High	Medium	Low
Weight	0.4	0.3	0.2	0.1

3.2.5 Round Trip Time Latency (RTTL)

It defines the latency time of the whole process which includes the start of transmission, reception and waiting time for a response. RTTL is calculated by:

$$RTTL = EET + ETT + 2 * delay \quad (17)$$

Where EET means expected execution time and ETT specifies the expected transmission time.

EET contains the four constraints like CPU speed, size of task, RAM speed and BW. The task is performed with a minimum time when the RAM speed and CPU speed are maximum. The execution time is low when the task size is small. EET and ETT are calculated by the following expressions:

$$EET = CPU\_Speed + RAM\_Speed + {Task\ size}/{Bandwidth} \quad (18)$$

$$ETT = {Task\ size}/{Bit\ rate} \quad (19)$$

3.3 Task Scheduling Using Hybrid Sshho

In this section, the Salp swarm-based Harris hawks optimization (SSHHO) algorithm is proposed for the problem of TSCE. To enhance the exploitation capability of SSA, the HHO is utilized. The hybrid SSHHO begins within the input constraints such as the extreme amount of iteration $t_{\max}$, the number of tasks N_{tsk}, the number of PMs N_{pm}, the number of VMs N_{vm}, and the number of solutions N. The random amount of population X is created in the SSHHO with dimension N_{tsk} and size N. The value of every solution x_j are lies between 1 to N_{vm}. For every solution, the fitness function is estimated. In the exploitation stage, the current population X will be modernized using either the SSA or HHO. This process is repeated up to meet the stopping criteria. The flowchart of SSHHO algorithm is shown in figure 2.

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

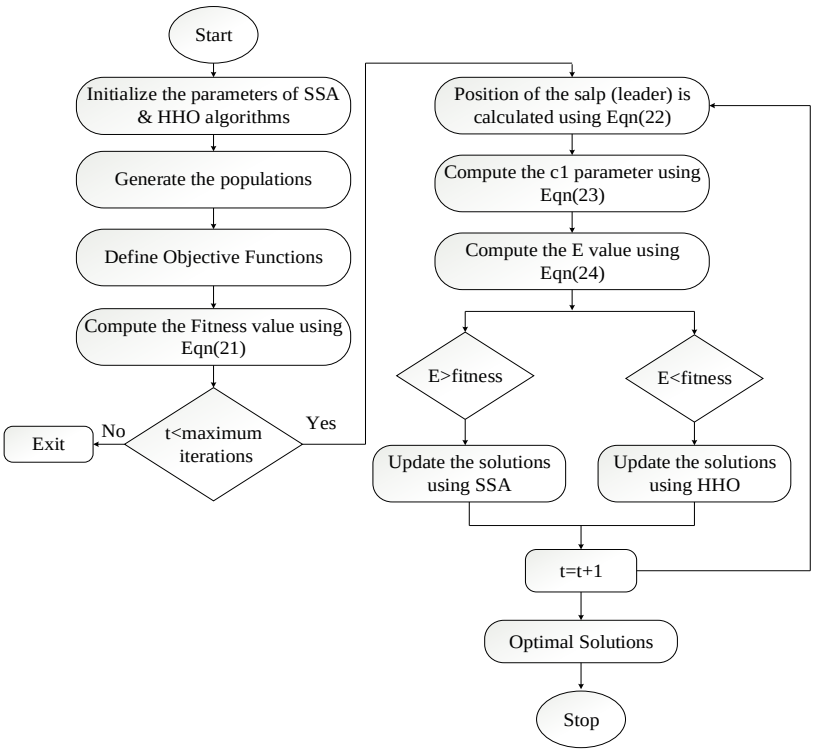

Fig. 2. Proposed SSHHO algorithm for task scheduling

3.3.1 Initial Stage

Here, the hybrid SSHHO creates a random number of population X with size N. The dimension of every solution is identical to the number of tasks (N_{tsk}) $\left(i.e, x_j = \lfloor x_{j,1}, x_{j,2}, ..., x_{j,N_{tsk}} \rfloor\right)$. In the value $x_{j,k}$, $k = 1,2,...,N_{tsk}$ is chosen from $[1, N_{vm}]$ by the equation (20).

$$x_j = round\{rand(ub_j - lb_j) + lb_j\}, \quad j = 1,2,...,N_{tsk} \tag{20}$$

Here, $round(.)$ function provides the round value to the adjacent whole number. $rand$ is a random number in the interval of 0 and 1. The lower boundary of search space is represented by lb_j and the upper boundary of search space is signified by ub_j (Here $lb_j = 1$ and $ub_j = N_{tsk}$).

An instance for making x_j, consider the 10 tasks and 5 VMs then the x_j value is $x_j = [4251321452]$. Here, the initial value of 4 in x_j states to the initial task will be assigned in the 4th VM.

3.3.2 Fitness Calculation

The fitness value of every individual is calculated and stored for future reference after the initial solution is obtained. The MO function is deliberated in this paper that includes energy consumption, final task weight, resource utilization, round trip time latency, and makespan. Therefore, the fitness function for the SSHHO is given by:

$$Fitness Function = Min\{\alpha(M) + \beta(RU) + \gamma(EC) + \eta(F \tag{21}$$

Here the values of $\alpha, \beta, \gamma, \eta$ and λ are the random numbers between 0 and1.

3.3.3 Exploration Stage

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

The salps positions are represented in an n-dimensional hunt space in the SSA. Here, n is the sum of variables of a specified issue. In a two-dimensional matrix x_i, the location of all salps are kept. The position of the leader is computed as:

$$x_i = \begin{cases} F_i + c_1((ub_i - lb_i)c_2 + lb_i) & c_3 \geq 0 \\ F_i - c_1((ub_i - lb_i)c_2 + lb_i) & c_3 < 0 \end{cases} \quad (22)$$

Where x_i and F_i denotes the site of leader and the food source in the i^{th} measurement. ub_i and lb_i designates the upper bound and the lower bound of i^{th} dimension, the parameter $c_2 and c_3$ is arbitrary numbers in the interval of [0, 1]. In SSA, the significant factor c_1 is equilibriums the exploration and exploitation stage which computed as:

$$c_1 = 2e^{-\left(\frac{4l}{L}\right)^2} \quad (23)$$

Where l is the present iteration and L is the extreme amount of iterations.

3.3.4 Exploitation Stage

In this stage, the HHO algorithm is utilized to enhance the exploitation capability of SSA. Using the c_1 parameter obtained from SSA, the exploration of SSA is combined with HHO exploitation. Therefore, the escaping energy of prey is estimated to define whether the current solution will be rationalized using SSA or HHO is given as:

$$E = 2c_1\left(1 - \frac{t}{T}\right) \quad (24)$$

Where the extreme number of iterations is signified by T and t is the present iteration.

The current solution will be rationalized using either SSA or HHO which depends on the value of E. The assessment E is compared to the fitness value. If $E > fitness$, the SSA is used.If $E < fitness$, the HHO is used. The *round* function is given the integer value after updating the solution in the present population. This iteration will repeat up to which the given criteria have been satisfied. The extreme number of iteration is at the terminal conditions and the output that gives the best solution for the optimal allocation with the VMs. Overall pseudo-code of the suggested method is exposed in figure 3

Hybrid SSHHO based Task Scheduling
Input: Number of tasks, number of PMs, number of VMs, parameters of SSA and parameters of HHO algorithm **Output:** Best optimal allocation for each task **Begin:** Initialize the parameters of the SSA and HHO algorithms Generate the random population Compute the fitness function for each solution using Eqn(21) Set cycle to 1

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

```
while (t < max iteration)
        {
            Update the position of salp using Eqn(22)
            Calculate the c1 value using Eqn(23)
            Calculate the escaping energy of prey using Eqn(24)
for (each prey in the solution)
            {
                Compare the fitness value based on the E
if( E > fitness)
                {
                    Update the solution based on the SSA
                }
else if ( E < fitness)
                {
                    Update the solution based on the HHO
                }
end
            }
end
        }
end
    Let t = t + 1
    Return optimal solution
end
```

Fig. 3. Pseudocode for SSHHO based task scheduling

4. SIMULATION RESULTS AND DISCUSSIONS

The multi-objective based task scheduling using proposed SSHHO algorithm is simulated on CloudSim toolkit. The performance of hybrid SSHHO process is assessed using the constraints such as energy consumption, makespan, efficiency, resource utilization. Degree of imbalance, and improvement ratio. The performance of the developed SSHHO procedure is related to the earlier approaches of CPSO [24], HGAACO [23], FMPSO [25], MSDE [22] and HGPSO [21]. The proposed SSHHO algorithm for scheduling of task can deliver a superior results than other meta-heuristic algorithms. SSHHO parameter is the number of population (100), and a maximum number of iteration (100). Table 2 provide the cloud environment parameters for SSHHO based task scheduling.

Table 2: Simulation factors of the developed system

Entity	Parameter	Value
Cloudlet	No of cloudlets (tasks)	100-500
	Length	30000-50000
	Input size	300
	Output size	300
Virtual Machine	No of VM	10
	MIPS	250
	RAM	512 MB
	VMM	Xen
	OS	Linux
	No of CPU	1
Host	No of host	1
	RAM	2048 MB
	Storage	1000000
	Bandwidth	10000
Data center	No of data center	5

4.1 Performance Metrics

The proposed system considers the makespan, degree of imbalance, energy consumption, efficiency, resource utilization and improvement ratio as performance metrics for systematic assessments. Here, makespan, energy

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

consumption, and resource utilization are described in section 3.2.

4.1.1 Degree of Imbalance (DI)

It denotes the imbalance between VMs is given as:

$$DI = \frac{T_{\max} - T_{\min}}{T_{avg}} \quad (25)$$

Where $T_{\max}, T_{\min}, and T_{avg}$ are the maximum, minimum, and average total execution times, individually, amongst all VMs.

4.1.2 Efficiency

The efficiency for different scheduling algorithms can be computed by the following expression:

$$Speedup = \frac{serial\_length}{makespan} \quad (26)$$

Where $serial\_length$ is achieved by allocating the entire tasks to a distinct VM and it can be computed as:

$$Serial\_length = \min_{p_j \in V} \left\{ \sum_{n_i \in T} w_{i,j} \right\} \quad (27)$$

Where $w_{i,j}$ represents the finishing time of the task n_i on the VM p_j. The set of tasks and set of VMs are defined by $T\ and\ V$, respectively.

4.1.3 Improvement Ratio (Ir)

IR shows the algorithm's effectiveness based on execution time reduction.

$$IR_j(\%) = \frac{\sum_{i=1, i \neq j}^{n} ET_i - ET_j}{\sum_{i=1, i \neq j}^{n} ET_i} \times 100 \quad (28)$$

Where ET_i represents the execution time of i^{th} method.

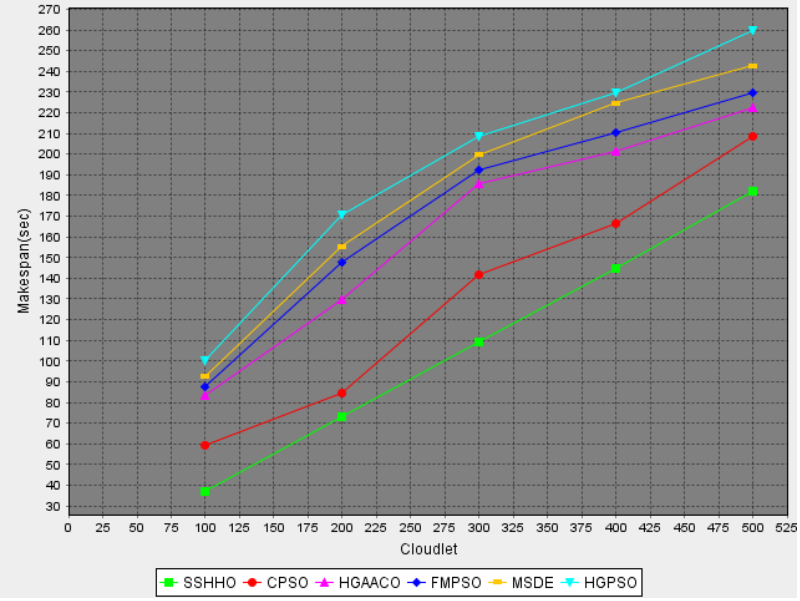

Fig. 4. Makespan for different number of tasks (Cloudlets)

Figure 4 demonstrates the makespan performance of proposed (SSHHO) and existing algorithms. Here, the makespan is computed for the different number of cloudlets. The makespan value is increased by the number of cloudlets increased. The proposed (SSHHO) scheme has a better

makespan value compared to the existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO. In 100 cloudlets, the proposed method obtained the less (39s) makespan than other algorithms. The existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO are 60s, 81s, 88s, 92s, and 100s respectively. From the graph, the SSHHO based task scheduling has significantly improved compared to the existing algorithms.

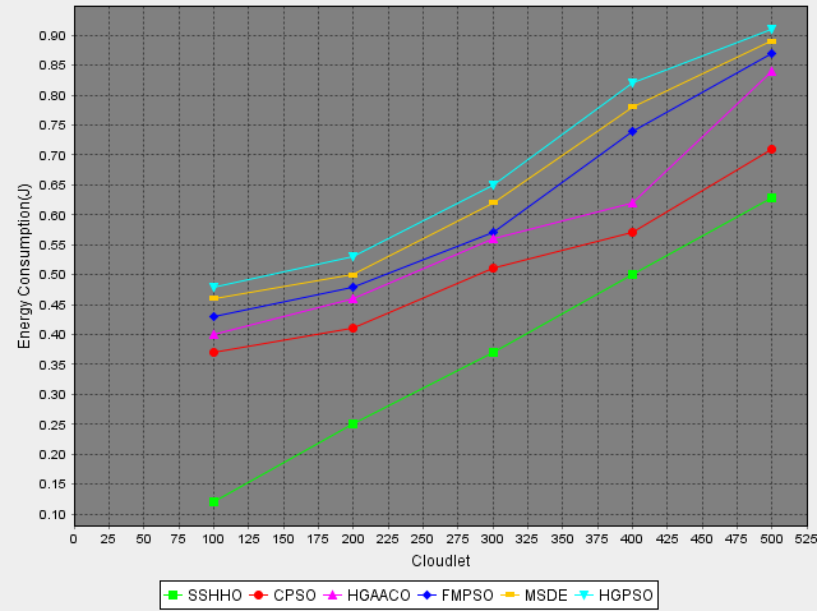

Fig. 5. Energy Consumption for different number of tasks (Cloudlets)

Figure 5 illustrates the energy consumption of proposed (SSHHO) and existing algorithms with different number of cloudlets (tasks). Energy consumption only based on how much amount of energy utilized to the scheduling of the task. The suggested method obtained the minimum energy consumption than other existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO. By the number of cloudlets increased, the energy consumption also increased. In 100 cloudlets, the proposed method obtained the low (0.12J) energy consumption than other algorithms. The existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO are 0.37J, 0.4J, 0.43J, 0.46J, and 0.48J individually. The graph clearly indicates that the SSHHO based task scheduling provides better outcomes compared to the other meta-heuristic algorithms.

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

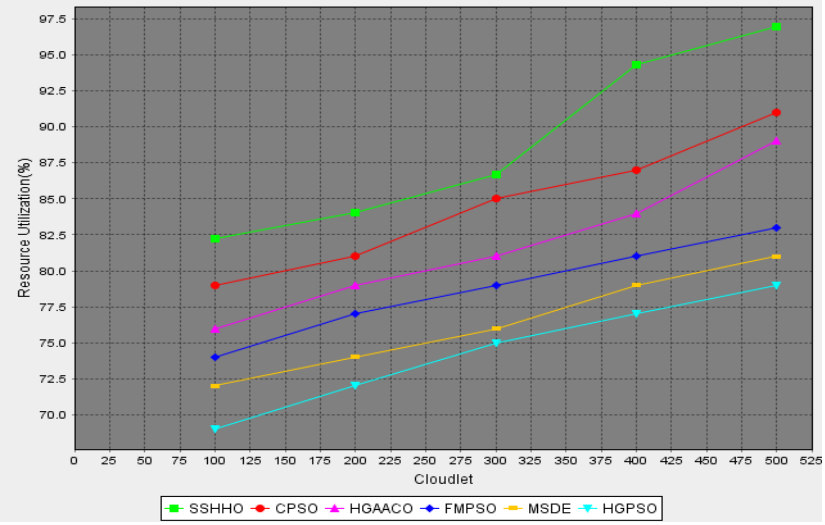

Fig. 6. Resource utilization for different number of Cloudlets

Figure 6 illustrates the resource utilization performance of proposed (SSHHO) and existing algorithms. SSHHO algorithm utilizes more resources than the other meta-heuristic algorithms. With the different number of tasks (cloudlets), the resource utilization performance is calculated. In 100 cloudlets, the proposed method obtained the high (82.5%) resource utilization than other algorithms. The existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO are 79%, 76%, 74%, 72%, and 68% individually. From the graph, the resource utilization is increased by the number of cloudlets increased. The proposed SSHHO based task scheduling obtained the maximum resource utilization compared to the other algorithms of CPSO, HGAACO, FMPSO, MSDE, and HGPSO.

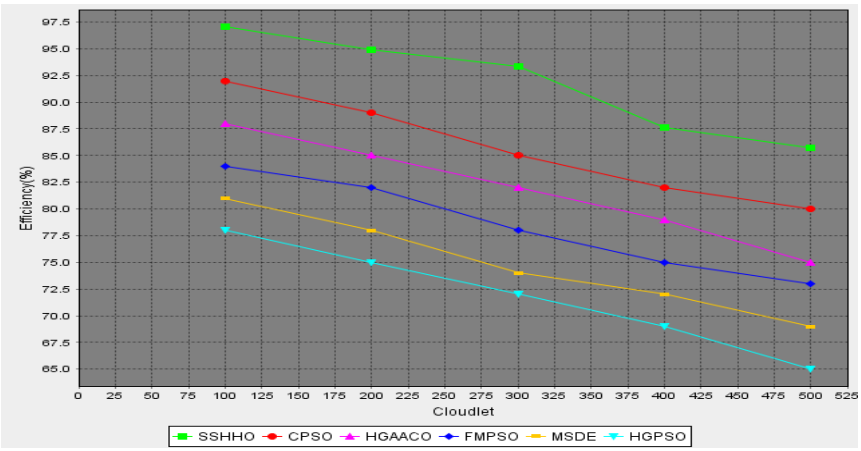

Fig. 7. Efficiency for different number of tasks (Cloudlets)

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

Figure 7 displays the efficiency of different algorithms for several number of cloudlets. The graph clearly denotes that the proposed SSHHO procedure offers superior results compared to the existing algorithms of CPSO, HGAACO, FMPSO, MSDE, and HGPSO in the parameter of efficiency. The performance of efficiency will be decreased by the number of cloudlets increased. In 100 cloudlets, the proposed method obtained the high (97.4%) efficiency than other algorithms. The existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO are 92.3%, 87.6%, 84%, 81%, and 78% individually. The SSHHO based task scheduling method obtained the maximum efficiency compared to the other meta-heuristic algorithms.

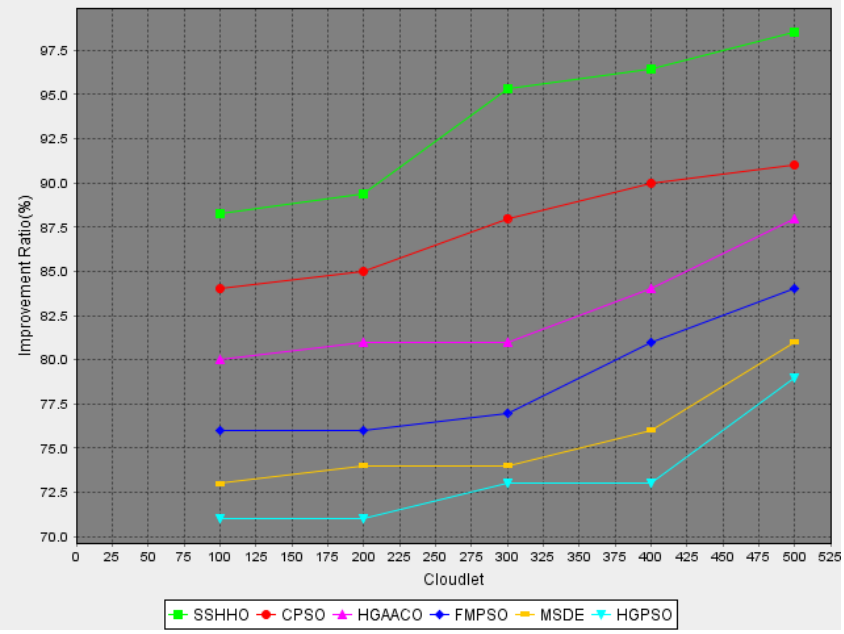

Fig. 8. Improvement ratio for different number of tasks (Cloudlets)

Figure 8 demonstrates the improvement ratio of proposed and existing algorithms for a different number of cloudlets. The improvement ratio only depends on the execution time. By the number of cloudlets increased, the improvement ratio will be increased. The proposed SSHHO algorithm obtained the maximum improvement ratio compared to the other algorithms of CPSO, HGAACO, FMPSO, MSDE, and HGPSO. In 100 cloudlets, the proposed method obtained a high (88%) improvement ratio than other algorithms. The existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO are 84%, 80%, 76%, 73%, and 71% individually. From the graph, the proposed method has significantly improved with the different number of cloudlets

.

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

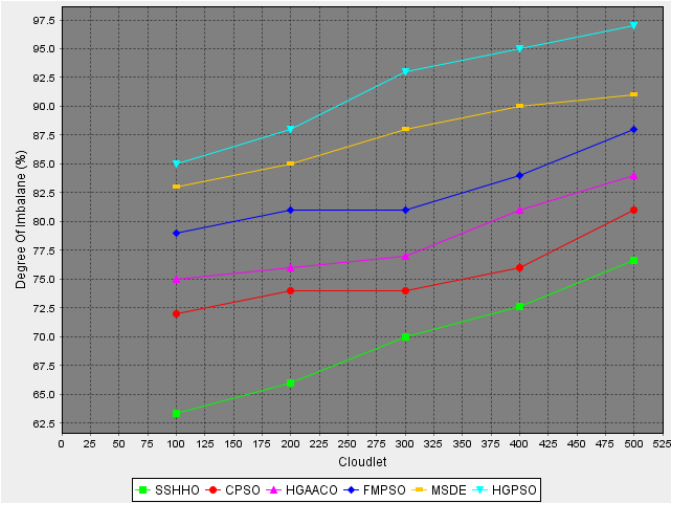

Fig. 9.DI for different number of Cloudlets

Figure 9 displays the DI performance of the proposed method and existing schemes with a different number of cloudlets. In 100 cloudlets, the proposed method obtained a low (63%) degree of imbalance than other algorithms. The DI performance will be increased by the number of cloudlets increased. The existing methods of CPSO, HGAACO, FMPSO, MSDE, and HGPSO are 72%, 75%, 79%, 83%, and 85% individually. The graph clearly indicates that the proposed SSHHO algorithm based task scheduling obtained the minimum degree of imbalance compared to other existing algorithms of CPSO, HGAACO, FMPSO, MSDE, and HGPSO.

From all the performances, it can be concluded that the suggested SSHHO procedure gives superior outcomes compared to the existing algorithms of CPSO, HGAACO, FMPSO, MSDE, and HGPSO. The reason for this enhancement in the outcomes are the merging the behavior of SSA) and HHO algorithm. Here, the HHO is utilized to develop the exploitation capability of SSA. In any case, the major drawback of the developed system is the execution time which still should be upgraded through the exact technique by considering the more tasks in the future.

5. CONCLUSION

The scheduling of tasks is the main issue in cloud environments to develop the QoS and the performance of the structure. In this paper, a hybrid salp swarm and Harris hawks optimization for secure TSCE based on multi-objective function has been presented. To expand the performance of scheduling, multi-objection function is utilized in the proposed hybrid SSHHO algorithm. The multi-objective function like energy consumption, makespan, final task weight, round trip time latency, and resource utilization were taken for better performance. The proposed SSHHO

algorithm is merging the exploration ability of SSA to the exploitation ability of HHO for improved performance of task scheduling. The SSHHO algorithm is obtained the minimum fitness function which includes the makespan, energy consumption, round trip time latency, resource utilization, and final task weight. The proposed SSHHO based multi-objective TS was simulated in CloudSim. The performance of the SSHHO is related to the existing algorithms of CPSO, HGAACO, FMPSO, MSDE, and HGPSO in terms of resource utilization, improvement ratio, makes pan, degree of imbalance, energy consumption and efficiency. The future scope of this research is trying to diminish the finishing time and consider a huge number of tasks, and introduce this method to the grid environment. In addition, it will observe the difference in time in the grid and cloud.

REFERENCES

[1] Lakra AV, and Yadav DK, et al. Multi-objective tasks scheduling algorithm for cloud computing throughput optimization. Procedia Computer Science.2015; 48: 107-113.

[2] Pham XO, and Huh EN, et al. Towards task scheduling in a cloud-fog computing system. In 2016 18th Asia-Pacific network operations and management symposium (APNOMS), IEEE. 2016; 1-4.

[3] Razaque A, Vennapusa NR, Soni N, and Janapati GS, et al. Task scheduling in cloud computing. In 2016 IEEE Long Island Systems.Applications and Technology Conference (LISAT), IEEE.2016; 1-5.

[4] Chen W, Xie G, Li R, Bai Y, Fan C, and Li K, et al. Efficient task scheduling for budget constrained parallel applications on heterogeneous cloud computing systems. Future Generation Computer Systems.2017; 74: 1-11.

[5] Arunarani AR, Manjula D, and Sugumaran V, et al. Task scheduling techniques in cloud computing: A literature survey. Future Generation Computer Systems.2091; 91: 407-415.

[6] Nayak S.C, Parida S, Tripathy C, and Pattnaik PK, et al. Modeling of Task Scheduling Algorithm Using Petri-Net in Cloud Computing. In Progress in Advanced Computing and Intelligent Engineering, Springer, Singapore. 2018; 633-643.

[7] Saleh H, Nashaat H, Saber W, and Harb HM, et al. IPSO task scheduling algorithm for large scale data in cloud computing environment. IEEE Access. 2018; 7: 5412-5420.

[8] Agarwal M, and Srivastava GMS, et al. A cuckoo search algorithm-based task scheduling in cloud computing. In Advances in Computer and Computational Sciences, Springer, Singapore.2018; 293-299.

[9] Guo S, Liu J, Yang Y, Xiao B, and Li Z. Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing. IEEE Transactions on Mobile Computing. 2018; 18(2): 319-333

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

[10] Naik KG, Gandhi M, and Patil SH, et al. Multiobjective virtual machine selection for task scheduling in cloud computing, In Computational Intelligence: Theories, Applications and Future Directions-Volume I, Springer, Singapore. 2019; 319-331.

[11] Mittal S, and Katal A, et al. An optimized task scheduling algorithm in cloud computing, In 2016 IEEE 6th International Conference on Advanced Computing (IACC), IEEE.2016; 197-202.

[12] Xavier VMA, and Annadurai S, et al. Chaotic social spider algorithm for load balance aware task scheduling in cloud computing. Cluster Computing. 2019; 22(1): 287-297.

[13] Aziza H, and Krichen S, et al. Bi-objective decision support system for task-scheduling based on genetic algorithm in cloud computing, computing. 2018; 100(2): 65-91.

[14] Niu SH, Ong SK, and Nee AYC, et al. An improved intelligent water drops algorithm for solving multi-objective job shop scheduling. Engineering Applications of Artificial Intelligence. 2013; 26(10): 2431-2442.

[15] Abdulhamid SM, and Latiff MSA, et al. League Championship Algorithm based job scheduling scheme for infrastructure as a service cloud. Ar Xiv preprint arXiv: 1410. 2208. 2014.

[16] Liu X, and Liu J, et al. A task scheduling based on simulated annealing algorithm in cloud computing, International Journal of Hybrid Information Technology. 2016; 9(6): 403-412.

[17] Jena RK, et al. Multi objective task scheduling in cloud environment using nested PSO framework, Procedia Computer Science. 2015; 57: 1219-1227.

[18] Karthikeyan S, Asokan P, Nickolas S, &PageT, et al. A hybrid discrete firefly algorithm for solving multi-objective fexible job shop scheduling problems.International Journal of Bio-Inspired Computation. 2015; 7(6): 386–401.

[19] Zuo L, Shu L, Dong S, Zhu C, and Hara T, et al. A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing. IEEE Access. 2015; 3: 2687-2699.

[20] Reddy GN, and Kumar SP, et al. Multi objective task scheduling algorithm for cloud computing using whale optimization technique. In International Conference on Next Generation Computing Technologies, Springer, Singapore.2017; 286-297.

[21] Kumar AMS, and Venkatesan M, et al. Task scheduling in a cloud computing environment using HGPSO algorithm, Cluster Computing. 2019; 22(1): 2179-2185

[22] Elaziz MA, Xiong S, Jayasena KPN, and Li L, et al. Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution, Knowledge-Based Systems. 2019; 169: 39-52.

[23] Kumar AMS, and Venkatesan M, et al. Multi-Objective Task Scheduling

March-April 2020
ISSN: 0193-4120 Page No. 2743 – 2762

Using Hybrid Genetic-Ant Colony Optimization Algorithm in Cloud Environment.Wireless Personal Communications.2019; 1-14.

[24] Jacob TP, and Pradeep K, et al.A Multi-objective Optimal Task Scheduling in Cloud Environment Using Cuckoo Particle Swarm Optimization, Wireless Personal Communications.2019; 1-17.

[25] Mansouri N, Zade BMH, and Javidi MM, et al. Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory. Computers & Industrial Engineering.2019; 130: 597-633.

[26] Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, and Mirjalili SM, et al. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Advances in Engineering Software.2017; 114: 163-191.

[27] Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, and Chen H, et al. Harris hawks optimization: Algorithm and applications. Future Generation Computer Systems.2019; 97: 849-872.