Hybrid Starling Social Spider Algorithm for Energy and Load Aware Task Scheduling in Cloud Computing

Arul Xavier V M, Annadurai S

Abstract: ***The efficiency of the cloud-based systems is greatly relying on the task scheduling algorithm which affects the performance parameters such as makespan, response time, degree of imbalance and cost. In recent years, the energy efficiency is also considered as another challenging issue which affects the efficiency of cloud computing systems. This paper proposes a Hybrid Starling Social Spider Algorithm (Starling-SSA) for Energy and Load Aware Task Scheduling in cloud computing. The Starling-SSA is designed as a hybrid algorithm inspired by the intelligent behavior of social spider and the collective response behavior of starling birds. The foraging behavior of spider is implemented to identify the best VMs for the given task with minimum makespan and degree of imbalance. In addition to this, the distance factor is incorporated inspired by starling flock distance in order identify the closeness of VM pairs and avoids the VMs that are far away, thereby VMs can be limited during the searching process. This will greatly reduce energy consumption by taking only VMs that are belongs to the distance factor. The performance metrics such as makespan, degree of imbalance and energy efficiency are evaluated against the existing algorithms such as EATS, CBAT and HC-ACO. The results presents a significant improvements when comparing to the existing algorithms.***

Index Terms: ***cloud computing, virtual machine, task scheduling, makespan, load balancing, energy efficiency, social spider, starling bird's behavior, degree of imbalance***

I. INTRODUCTION

Cloud computing is a framework of interconnected web services that delivers services to consumers over the internet with certain Quality of Services (QoS) requirements[1]. The evolution of cloud computing facilitated to various business organization to meet their client's requirements in providing services such as network infrastructure, software platforms and software applications[2]. It provides a new way of offering the computational services based on "Pay-Per-Usage" policy which drastically reduces the investment of running their businesses[3]. Typically, the cloud services are delivered to consumer premise via virtualization which helps to satisfy the demands of cloud provider as well as cloud users. This innovation makes the user applications run in an isolated virtual machine environment and able to integrate the distributed resources independently[4]. This provides many benefits in computer based business solutions in terms of several factors such as time, cost, and flexible arrangement of computing resources at the consumer premises[2], [3].

Revised Manuscript Received on July 06, 2019.

Arul Xavier V M, Assistant Professor, Computer Science, Karunya Institute of Technology and Sciences, Coimbatore, India.

Dr. Annadurai S, Professor, Computer Science and Engineering, Hindusthan College of Engineering and Technology, Coimbatore, India.

Task scheduling process in cloud computing needs to be carefully planned in order to improve the efficiency of the whole cloud computing system[5]. If the user submitted tasks are not scheduled properly to the proper Virtual Machines (VMs), performance reduces in terms of cloud provider's profit which makes system not able to meet the client's requirements[6]. Thus, task scheduling plays a vital role in cloud computing since which affects the performance in great deal. In every cloud provider, a Task scheduling algorithm is utilized to manage the incoming tasks with proper scheduling mechanism in finding the best suitable VM so that it benefits the cloud provider and consumer[4]. Moreover, these scheduling algorithms must adapt the dynamic environment where the incoming tasks keep on changing based on some parameters such as its size, cost, deadlines and other QoS metrics[7]. In the same way, the cloud providers have certain constrains such as profit, resource utilization, energy consumption which is the basic requirement of any cloud provider[1]. Additionally, the cloud provider must have capability to accommodate sudden increase in client requests which leads to increase in resource utilization, cost, and energy consumption. In such dynamic environment, the static task scheduling algorithms are suitable since the users may expect utilize thousands of virtualized resources in a form of VMs at any time[7]. This complexity can be easily tackled by virtualization if there is an efficient task scheduling mechanism is employed. Thus, scheduling the user task to the best VM considered as a challenging issue in cloud computing in assigning the task to resources (VM) in an efficient manner[8]. Currently, various types of scheduling algorithms were presented by various researchers. This includes static scheduling, dynamic scheduling, heuristic scheduling and various types of hybrid algorithms[5]. Most of the research works reveals that the task scheduling problem in cloud computing is a type of NP-complete where heuristic algorithms are more suitable, due to its dynamic and heterogeneous nature[1]. There are few parameters which plays a major role in fine tuning the performance of any task scheduling mechanisms. These parameters greatly affect the efficiency of the cloud computing in both cloud provider and cloud consumer perspectives[9]. The efficient task scheduling mechanism must address the few performance metrics such as resource utilization, less computational cost, low energy consumption, scalability as part of the cloud provider's benefit[10]. As part of cloud user's concern, the scheduling algorithm must provider minimum makespan, reduced response time and low cost in getting the service. Moreover, load balancing and energy consumption is another kind of problem which needs to be addressed while scheduling the tasks to cloud resources[11]. During the schedule of user tasks, load and energy of each resource needs to be realized and decisions can be taken to assign a task to the best VM so that the overall makespan, degree of imbalance and energy consumption is minimized[12].

Thus, makespan optimization with load balancing is a critical issue in achieving high performance in cloud computing. Additionally, reducing the energy consumption is also an important parameter that needs to be considered while assigning the task to the resources since it sustains the whole cloud computing system with green ambience[13]. The task scheduling policies plays an important role on reducing energy consumption in order to allocate tasks on distributed VMs. In recent studies carried out by [14] have estimated in average of 55% of energy utilized by the computing system in cloud computing. Thus, green computing is also an important issue for ensuring the sustainability of cloud computing system in terms of cost and environment safety.

This paper proposes a hybrid algorithm for energy efficient load balance aware task scheduling in cloud computing inspired by social spider's foraging behavior and starling birds flock behavior. The social spider algorithm is an algorithm inspired by foraging behavior of social spiders has been adopted by [15] for global optimization. It is an intelligent exploitation of foraging strategies present in the spider's web. This swarm intelligence helps the spiders in the population to find the food sources which is hybridized with the intelligent flocking behavior[16] of starling birds as major component for scheduling the cloud user tasks in energy efficient way. The flocks of starling birds are an interesting group behavior where simple rules of interaction among the birds sufficient to produce the collective behavior. The starling flock intelligence is incorporated to the social spider foraging in order to achieve energy efficient task scheduling in cloud computing with balanced load on VMs[17]. The remainder of this paper is organized as follows. Section II describes the various research study related to improve the energy efficiency of task scheduling cloud computing and Section III presents the background concepts of proposed system. Section IV describes the task scheduling model and detailed procedure of proposed system. Section V provides the simulation results and performance evaluation with existing techniques.

II. RELATED WORKS

There are several ways to optimize the energy consumption of user tasks when executed by the cloud computing system. Some of the researchers adopted re-designing the algorithm and changing the scheduling polices in order to improve the energy efficiency. A shadow price guided Genetic Algorithm (GA) has been presented by [18] in order to minimize the energy consumption where the mutation operation is modified with determination of shadow prices. The simulation results show few improvements in minimizing the energy consumption. A PSO based algorithm was adopted by [19] in an attempt to maximize the resource utilization with minimum energy consumption. The load balancing based task scheduling problem has been studied by taking account on energy consumption by [20] and presented an approach called EATS, where the energy consumption is periodically measured across the resources for different load conditions. Their results shown few improvements in minimizing the overall energy consumption. The same problem have been studied in [11] and proposed an energy efficient approach based on PSO for task scheduling in cloud computing. The energy consumption taken into an account with fitness in order to guide the scheduling and VM allocation policies in order to tackle the problem of local convergence also. The comparison results show that the proposed algorithm better than the traditional approaches but it is found that the algorithm fails to treat the scalability. As mentioned in[21] more research is focusing on "Green Cloud" by reducing the energy consumption in cloud computing environment. The authors have presented a DENS algorithm for energy efficient task scheduling. They have taken the network awareness as a key part to take decisions in assigning the tasks and utilized the feedback channel information to limit the computing servers. Their results shown few improvements in minimizing the energy consumption but fails to tackle the problem of load balancing. Another research work [22] have presented an approach called e-STAB for task scheduling with reduced energy consumption. They have considered the traffic load balancing strategy across the cloud data centers Their results show improvements in communication delays and congestion control. The research work done by [23] have proposed CBAT algorithm for scheduling the user task in cloud computing. The authors have designed the algorithm with array concept to represent the candidate solution and formulated the fitness correlated with makespan and energy consumption. The simulation results show significant improvement in reducing the makespan and few improvements were noted in reducing the energy consumption. A dynamic resource allocation strategy was presented in [24] for scheduling independent tasks where the priority order is used to perform pre emptiable task execution and adjust the scheduling policies dynamically based the feedback information. The proposed algorithm has shown significant improvements in energy consumption and makespan optimization. A hybrid task scheduling algorithm was proposed in [25] using a combination of cultural algorithm and the intelligent behavior of ant colony. The objective of the proposed algorithm is to reduce the makespan while minimizing the energy consumption by make using the benefits of both algorithms. The proposed hybrid algorithm shows significant improvement in makespan optimization and minimization of energy consumption. However, the algorithm fails in dynamic cloud computing environments due to static computation model.

III. BACKGROUND

A. Social Spider Foraging Behavior

The social spiders and its behavior for searching the food sources and its peer spider group's collective behavior are quite interesting. Social spiders have been found in anywhere in the world and are one of the most deviated groups among all other groups of organisms. They use an intelligent searching strategy in preying the food sources with interaction among its peer groups. The most interesting strategy is its sensitive vibrations which guides the movement of spiders in order to reach the food prey. The movement decisions will be taken based on the vibration intensity and frequency. It can reach the prey when the intensity and frequency is within the range.

The peer groups are very sensitive and intelligent to differentiate the vibrations populated among the spider's web as given in [26].Based on the literature study, every spider's receive vibrations from its neighbors to have detail information about the environment which is the unique characteristics of social spiders compared to others. It helps the neighboring spiders to identify not only the location of food prey, the quality and quantity as well. The computation model of SSA has been adopted by [15]for global optimization problems. The SSA is based on the Information Sharing (IS) model which has been dealt in[27] where it is belongs to the group foraging. The basic operations of computation model of SSA are as follows.

- Set of social spiders were generated as initial population with initial parameters and the initial fitness value is determined.
- Generation and propagation of vibrations at each spider's position in the spider web.
- Selection of best received vibrations and updates the initial target vibration.
- Perform random walk on the spider's web towards to target position.

B. Starling Birds Flocking Behavior

The starling birds are small to medium sized birds which gather in flocks in the cloud space in order to search for food sources. The movement of each birds in the flock quite interesting because of its communication and coordination. This collective behavior helps them to moves synchronously and takes a swift when there are any obstacles or predators. The change in movement starts from one bird and rippled to other neighbor birds through the flock. The information which decides the orientation and movement is not only the individual bird's self-observation and also the collective responses from its nearby birds. Every individual in the large flocks consistently coordinate their movements with their nearest neighbors. This behavior helps each and every bird update their original location, velocity and orientation there by it can reach the food sources without much efforts.

The computational model of starling flock behavior has been dealt in [28],[16] for optimization problems. The study reveals that the group formation of starling birds having limitations in terms of number of starlings involved in interactions but each starling ensures that it is synchronized with the neighboring starling birds. The basic procedure is as follows

- Choose an individual and pick update partner within a range.
- Update the position and velocity based on its partner's distance under three constraints such as small distance, medium distance, long distance.

IV. PROPOSED ALGORITHM

An extended version of SSA with Starling Flock collective response behavior is presented to improve the performance of task scheduling process in cloud computing. The objective of the proposed algorithm is to efficiently allocate the user tasks to the best possible VMs with realization of energy consumption and load balancing. The intelligence features in SSA and Starling Flocking is incorporated in order to achieve global exploration and exploitation.

A. Task Scheduling Problem and Mathematical Model

The cloud computing framework is considered as spider web where resources are interconnected and configured as VMSet= {VM1, VM2, VM3…VMm}. Each VM contains memory, computing elements, and communication bandwidth and location information. The location of each VM is randomly assigned in the solution space of spider web. The user submitted tasks are considered as TaskSet = {T1, T2, T3, …Tn} which needs to be scheduled on the VMSet. This scheduling problem focused on finding the best task schedule, which minimizes makespan, energy, degree of imbalance in order to improve the performance. Hence, the proposed work designed with the following three-fold objectives.

- To find an efficient schedule of tasks and VM set that minimizes the total makespan of the task scheduling process so that the cloud user can satisfy with reduced execution time and response time.
- To realize the load on each VM before assigning the task that avoids the resources getting overloaded so that it benefits the cloud provider and cloud user in terms of resource utilization and makespan.
- To optimize the energy consumption by finding the best VM pair that are close to distance factor for the given user tasks.

a. Makespan Model

Makespan is a measure in cloud computing which represents the time taken to complete the whole tasks submitted by the user. It includes the processing time, waiting time and communication delay if any. The Processing Time of each task is determined based on the Expected Time to Complete (ETC) matrix provided by the cloud broker. The Waiting Time (WT) is calculated from the arrival time and execution start time metrics. Sometimes, if there is any delay due to communication overhead it is considered as Delay Time (DT).

Let us consider CT_i is considered as a completion time of a Task Ti , $i \in \{1,2,3,..N\}$on resource VMj, $j \in \{1,2,3,..M\}$ is calculated by adding up the PT_i, WT_i and DT_i of task Ti which is mathematically modelled as in Equation (1).

$$CT_i = PT_i + WT_i + DT_i \qquad (1)$$

As stated above, the makespan is described as total completion of all the tasks in the TaskSet which is represented as CT_{max} and it is determined as given in Equation (2).

$$CT_{max} = \sum CT_i \qquad (2)$$

The Equation (2) defines a first part of the objective which needs to be minimized so that user feels more satisfaction with cloud provider in terms of faster completion of the service. Moreover, as a cloud provider it can achieve user satisfaction which improves the scalability of the system.

b. Energy Model

The energy consumption is not only calculated by the various hardware components used in the framework, but also dependent on the resource management system used on the infrastructure and the efficiency of running each user tasks Beloglazov et al. (2011). Energy efficiency directly impacts cloud users in terms of cost, which is determined by the usage of resources. Additionally, as a cloud provider perspective it affects the resource utilization and scalability. Apart from the energy consumed by the hardware resources it is mandatory to deal with the energy consumed by each VM since it depends on the way in which the user tasks are getting assigned to the VM. The energy ware scheduling model has been adopted by Beloglazov et al. (2012), which states that the CPU consumes more energy than other hardware equipment's in cloud computing framework. The energy consumed by CPU is mostly relies upon the type of the task assigned and the utilization of resources by the assigned task. The total energy consumption in a cloud for the given user tasks is depends on the energy consumed during the communication time and the processing time. The energy consumed during the communication time is represented as E_{CE} which depends on the energy required to setup the task for execution such as copying the necessary data that is required by the task to run. Then, the energy required to execute the task by the allocated VM is considered as a E_{PE} which directly deals with the amount of energy consumed by CPU to execute the task in specified VM. Hence, the overall energy consumption E_{total} by the user tasks in the TaskSet is determined using the Equation 3.

$$E_{total} = E_{CE} + E_{PE} \qquad (3)$$

The mathematical model described in Equation (3) is the second part of the objective that needs to be realized as minimum while allocating the tasks to the VM. This will greatly reduce the VM usage cost per user which leads to again user satisfaction.

c. Load Balance Constraint Model

Load balancing is a critical problem in cloud computing which affects the scalability and throughput of the cloud computing based systems which needs to be taken in to account the effective performance improvement. The VM resources can be rapidly overloaded during the task scheduling process which needs to be monitored and such VMs should be avoid for particular period of time. This will help the cloud task scheduling to improve the resource utilization. The load on each VM is depends on the amount of resources utilized by the available tasks which is currently scheduled and running. In dynamic environment, it is mandatory to check the load of VM where the total tasks are being executed and tasks that are waiting to execute. The load of each task (T_{load}) can be calculated based on the resources consumed by the user task with certain parameters such as CPU utilization(T_{cpu}) RAM utilization (T_{ram}), Storage utilization (T_s) which is described as in Equation (4).

$$T_{load} = T_{cpu} + T_{ram} + T_s \qquad (4)$$

Moreover, the total load on the VM (VM_{load}) is calculated based on the total tasks that are scheduled on a particular VM which is described as summation of load for all the tasks that are scheduled during the specified time. This is determined as given by the Equation (5).

$$VM_{load} = \sum_{i=1}^{n} T_{load_i} \qquad (5)$$

In order to realize the load balancing during the scheduling process, the average load (σ_{load}) is used which is the ratio of total VM load and the total number of VMs (N) which is dynamically calculate during schedule and has been modelled in Equation (6).

$$\sigma_{load} = \frac{VM_{load}}{N} \qquad (6)$$

In order to take load balancing decision, the degree of imbalance (DI) is calculated, which implies the load distribution factor between VMs with respect to their ability of execution. The small value of DI shows that the load of the system is more balanced. It is determined by Equation (7).

$$DI = \frac{Load_{max} - Load_{min}}{\sigma_{load}} \qquad (7)$$

Where, $Load_{max}$, $Load_{min}$ are the maximum and minimum load of VM among the set of VMs. Here, the Equation (7) specifies the third part of the objective function that is needs to be minimized.

d. The Objective Function

The proposed algorithm is aims to minimize the makespan with balanced load distribution with energy efficiency. This objective is considered as a multi-objective scheduling process where the total makespan needs to minimized, the total energy consumption needs to be minimized and the load of all VMs needs to be balanced by ensuring the DI is kept minimal. However, there is a trade-off between the minimization of makespan, energy and the degree of imbalance. Sometimes, when the algorithm tries to minimize the makespan it may increase the energy consumption and vice versa. Similarly, when the solution tries to reduce the energy consumption it may leads to improper load variations among the VMs. Hence, the proposed scheduling algorithm tries to find a good trade-off between those mentioned objectives. The energy efficient task scheduling must consider energy model as one of the major parameters in defining the fitness function. In order to achieve the overall objectives, a multi objective function is proposed which combines the makespan(CT_{max}), energy efficiency (E_{total}) and degree of imbalance (DI) which is mathematical modelled as given in Equation (8).

$$Fitness(f) = minimize\ \Big(\Big(\gamma \frac{CT_{max}}{nf1} \times DI\Big) + 1{-}\gamma Etotalnf2 \qquad (8)$$

Where $\gamma \in [0,1]$ weight factor user-controlled parameter based on weighted sum method as given in [29] to adjust the importance between makespan and energy. The $nf1$, $nf2$ are normalization factors used for normalizing the different type of metrics and makes them as comparable.

B. Starling Social Spider Algorithm for Energy Efficient Task Scheduling

The intelligence of social spider's and starling flock collective behaviour has been incorporated in the proposed algorithm called as Hybrid Starling Social Spider Algorithm (Starling-SSA). According to the proposed work, the cloud computing framework is considered as a spider web where the resources are virtually connected and available in the form of VMs. The food sources are modelled as VM in cloud computing environment and the social spider are considered as Search Agent (SA) moves freely in the cloud computing environment to find the best VM for the user tasks. Each SA is configured with inbuilt memory and mobility support to move in the solution space. The memory of SA used to remember the location of VM and its fitness value. Moreover, each SA is designed to be able to send their fitness to the neighbors via broadcasting in a format called Broadcast Message (BM). Moreover, each SA is defined with movement capability in the web in order to search the target VM as per the requirement.

a. Initialization Phase

The proposed algorithm starts with creating a collection of VMs $\{VMs\}$ and configuration of SAs location randomly with its initial fitness calculated as per the Equation (8) and prepare the BM for the location. The BM contains the information related to the capability of VM available in the specified location. After, the preparation of BM, each SA send their BM to all of its neighbors via in built broadcast facility. The following initial parameters setting are introduced to configure the BM and its quality.

- Initial quality of BM target is defined as $({BM_q}^{tar})$
- The location of VM is represented as L_{vm}.
- Initial (BM_q) is set to 0, which contains the location of VM
- Each SA is defined with certain velocity SA_{V_i}

b. Propagation of BM phase

In each VM location the fitness is determined with its location information and velocity. After that, the algorithm proceeds to the next phase called BM propagation where the prepared BMs of each VM location is broadcasted to all of its neighbors. For example, the location of i^{th} VM is considered as $L_i(t)$ and each SA generates the BM_q and it is represented as $BM_q(t)$ which is considered as a fitness of each SA locations. The calculation of $BM_q(t)$ for each SA location is done by the Equation 8 and it is represented as given in Equation 9

$$BM_q(t) = log\left(\frac{1}{Fitness}\right) \qquad (9)$$

c. Iteration Phase

In this phase, the received BMs will be processed by each SA from its neighboring VM locations and arranges in ascending order in to find the best BM which is represented as BM_q^{best} and necessary updation is initiated in the target BM as per the following.

```
if ($BM_q^{best} > {BM_q}^{tar}$)
  {
     ${BM_q}^{tar} = BM_q^{best}$
     $S_g = 1$
   }
else{
     // No updation
     $S_g = 0$
   }
```

Here, the S_g is introduced to manage the iteration levels and monitor the updation frequency which helps to stop the iteration phase if it is not updated frequently.

d. Random VM Selection based on Starling Flock for Energy Efficiency

In this phase, a group of SAs (SA_{group}) are selected based on quality of received BM_q^{best} and algorithm proceeds based on starling flock collective behavior. The update velocity for the selected SA_{group} assumed as its initial velocity then the distance between each SA is calculated in the SA_{group}. Then the algorithm proceeds to an iteration process as follows. For each selected SA in SA_{group}, select one SA_i and one partner SA_j which has Euclidean distance and the fitness of both locations is compared. If the BM_q^{best} of SA_i is greater than the SA_j, change the location of SA_j to SA_j. Then the distance of each SA and its partner SA is compared with respect three constraints such as Repulsion Rate (RR), Cohesion Rate (CR), and Attraction Rate (AR). If the distance is below RR then the velocity of SA is changed according to Equation (10). When the distance is between the RR and CR then the velocity is updated based on Equation (11). Finally, if the velocity is between the range of CR and AR then the velocity changes according to Equation (12).

$$SA_{V_{update}} = SA_{V_{update}} + \alpha \times V_{max} \times (SA_i - SA_j)/SA_{total} + (1-\alpha) \times SA_{best} \qquad (10)$$

$$SA_{V_{update}} = SA_{V_{update}} + \alpha \times V_{max} \times (SA_j)/SA_{total} + (1-\alpha) \times SA_{best} \qquad (11)$$

$$SA_{V_{update}} = SA_{V_{update}} + \alpha \times V_{max} \times (SA_j - SA_i)/SA_{total} + (1-\alpha) \times SA_{best} \qquad (12)$$

Where SA_{total} is a total number of search agents and the α is the starling weight factor introduced in order to guide the random VM selection. In addition, the V_{max} is a velocity that needs to be used while changing the direction and random VM location. Finally, for each SA the velocity is updated based on the outcomes of the above equations. The process flow of the proposed algorithm is depicted in Figure (1).

Algorithm: Hybrid Starling Social Spider Algorithm (Starling - SSA) Search Procedure

Input: User Tasks

Output: Optimized VM allocation of all user tasks submitted

```
Create VMList ← $[VM_1, VM_2, VM_3, \ldots VM_n]$
Create TASKList← $[T_1, T_2, T_3, \ldots T_n]$
Set unique $L_{id}$(Location id) to each VMList items.
Generate SA with memory and configure the BM.
Randomly set the VM's $L_{id}$ to each SA
Initialize the maximum number of iterations $(Iter^{max})$
Initialize $BM_q^{tar}$ $and$ $S_g$
Set al.l the user controlled parameters
for each Task(t) in TASKList do
for$i \leftarrow 0$ $to$ $Iter^{max}$
for each SA
compute the fitness of each VM in each SA
store the fitness in $BM$ of SA
propagate the $BM$
end for
for each SA
Store the received $BM$ in a $BM$List
  Arrange the items in $BM$ List in fitness $(BM_q)$ ascending order
```

```
    Select the best $BM_q$
    Select N number of SA ($SA_{group}$) based on the $BM_q$
    Initialize the $SA_{V_{update}}$ to its initial velocity
    For each SA in $SA_{group}$
        Select a partner SA
        Compare the distance
        If ( distance < RR)
            Update the velocity based on Equation (10)
        Else if (RR< distance < CR)
            Update the velocity based on Equation (11)
        Else if( CR < distance < AR)
            Update the velocity based on Equation (12)
        End for
    End for
End for
Output the VM allocation schedule.
```

Figure 1 Hybrid Starling Social Spider Algorithm (Starling-SSA) for Task Scheduling

V. SIMULATION RESULTS AND ANALYSIS

The simulation for the proposed system is carried out in CloudSim Framework and the results are compared with existing algorithms. The performance is evaluated based on the several parameters such as energy efficiency, makespan, degree of imbalance and compared with existing techniques EATS, CBAT HC-ACO. We have made the performance comparisons for different size of tasks which varies from 200 to 1000 and the results are represented in Table 1 – Table 3.

A. Makespan Analysis

The makespan evaluation is carried out for number simulations for the different set of tasks ranging from 200 to 1000. The simulation results for makespan are shown in Table 1 along with the values of existing algorithms.

Table 1 Makespan Values of Staring-SSA with Existing Algorithms

Number of Tasks	EATS (Secs)	CBAT (Secs)	HC-ACO (Secs)	Starling-SSA (Secs)
200	1255.3	903.33	1083.21	892.28
400	2555.1	1401.6	1801.71	1234.5
600	4843.4	2986.1	3414.51	2018.33
800	5818.8	5533.3	4355.21	2572.29
1000	6712.1	5212.1	5855.13	4118.45

The makespan results are compared with existing algorithms for different task group ranging from 200 to 1000 and represented in graphical manner as shown in Figure 2. The results comparison reveals that, the Starling-SSA outperforms when comparing to the existing algorithms such as EATS, HC-ACO and CBAT with the help of multi objective fitness function introduced in the proposed algorithm which helps to select best VM based on the overall makespan of the tasks on the VMs.

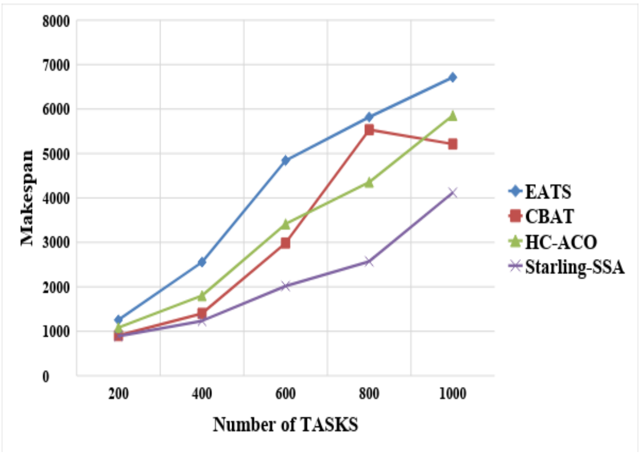

Figure 2 Makespan Improvements in Starling-SSA Degree Imbalance Analysis

The degree of imbalance helps to identify the load on the VMs and enable the system to take load balancing decisions. The degree imbalance evaluation is carried out and the results are shown in Table 2 for number simulations in different set of tasks ranging from 200 to 1000. The task scheduling technique with minimum degree of imbalance will improve the resource utilization.

Table 2 Degree of Imbalance Results of Starling-SSA and Existing Algorithms

Number of Tasks	EATS	CBAT	HC-ACO	Starling-SSA
200	1.1	1.34	1.25	0.67
400	1.5	1.2	1.45	1.1
600	2.3	2	1.9	1.3
800	3.3	3.1	2.65	1.9
1000	3.9	3.7	3.45	2.05

The degrees of imbalance results are compared with existing algorithms as shown in Figure 3. It shows significant improvements in minimizing the degree of imbalance with the help of load balancing model in the fitness function where the load of each VMs are calculated and incorporated while choosing VM for the given user tasks. In Starling-SSA, major improvements are shown as the tasks size increases.

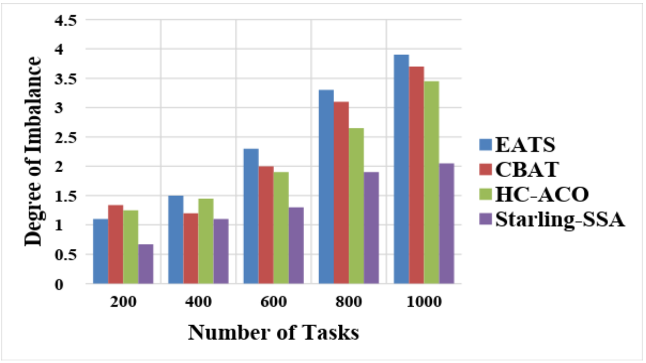

Figure 3 Minimization of Degree of Imbalance in Starling-SSA

Energy Efficiency Analysis

The energy consumption is calculated based on the energy model used in fitness function that helps to keep the cloud computing environment green and improves the profit of cloud providers. The energy consumption of proposed Starling-SSA is calculated and converted as percentage of improvements in energy savings are determined. The energy efficiency percentage levels are shown in Table 3 for different number of simulations in different set of tasks ranging from 200 to 1000.

Table 3 Energy Efficiency Percentage of Starling-SSA and Existing Algorithms

Number of Tasks	EATS (%)	CBAT (%)	HC-ACO (%)	Starling-SSA (%)
200	45	65	67	78
400	68	62	75	87
600	75	80	78	85
800	72	75	70	83
1000	55	67	65	88

The energy efficiency percentages are compared with existing algorithms as shown in Figure 4. It shows significant improvements in maximizing the energy efficiency with the help of energy model introduced in the fitness function where the energy consumption of each VMs are calculated and incorporated while choosing VM for the given user tasks. In addition, an intelligent search pattern is incorporated using starling birds flocking behavior which helps to filter the VMs that are not feasible based on the distance measure. With this starling flock, the VMs that not belongs to the distance measure are eliminated during the VM selection process. In Starling-SSA, major improvements are shown as the tasks size increases from 200 to 1000.

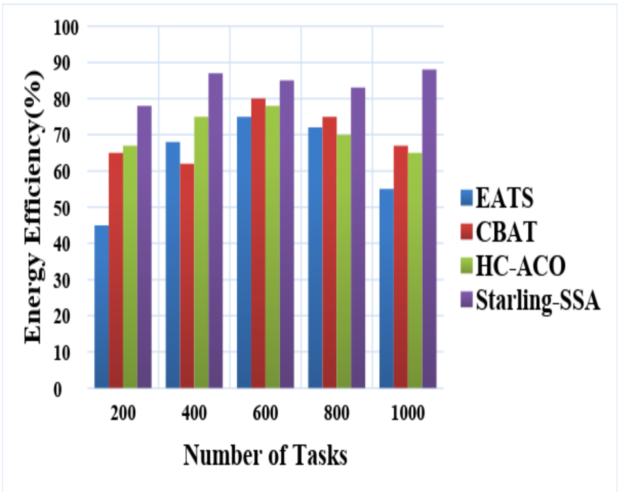

Figure 4 Energy Efficiency Comparison of Starling-SSA and Existing Algorithms

VI. CONCLUSION

A hybrid Starling Social Spider Algorithm (Starling-SSA) is proposed for energy efficient task scheduling with load balance awareness for the cloud computing environments. It is modelled using CloudSim framework as per the cloud computing environment requirement in order to find the best VM for the given user task. Random VM selection has been incorporated with the help of starling birds flock behavior in order to limit the VMs so that it helps to reduce the overall energy consumption. The main objective is minimizing the makespan with energy savings and resource utilization in cloud scheduling process thereby improving the throughput of the cloud system. In this regard, various parameters such as makespan, energy efficiency, degree of imbalance are evaluated and compared to existing algorithms such as EATS, HC-ACO and CBAT. The experiments results are analyzed which shows that, the Staring-SSA produce the makespan minimization with minimum degree of balance when compared to EATS, HC-ACO, CBAT and also average energy efficiency 19.02% has been improved when compared to the existing algorithms.

REFERENCES

1. M. Kalra and S. Singh, “A review of metaheuristic scheduling techniques in cloud computing,” *Egypt. Informatics J.*, vol. 16, no. 3, pp. 275–295, 2015.
2. S. Srichandan, T. Ashok, and S. Bibhudatta, “ScienceDirect Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm,” *Futur. Comput. Informatics J.*, vol. 3, no. 2, pp. 210–230, 2018.
3. S. Kumar, S. Surachita, and S. Kumar, “A pair-based task scheduling algorithm for cloud computing environment,” *J. King Saud Univ. - Comput. Inf. Sci.*, 2018.
4. “A genetic algorithm-based task scheduling for cloud.pdf.” .
5. A. R. Arunarani, D. Manjula, and V. Sugumaran, “Task scheduling techniques in cloud computing : A literature survey,” *Futur. Gener. Comput. Syst.*, vol. 91, pp. 407–415, 2019.
6. S. Basu *et al.*, “An Intelligent / Cognitive Model of Task Scheduling for IoT Applications in Cloud Computing Environment,” *Futur. Gener. Comput. Syst.*, 2018.
7. P. Zhang and M. Zhou, “Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy,” pp. 1–12, 2017.
8. L. F. Bittencourt, A. Goldman, E. R. M. Madeira, L. S. Nelson, and R. Sakellariou, “Scheduling in distributed systems : A cloud computing perspective ☆,” *Comput. Sci. Rev.*, vol. 30, pp. 31–54, 2018.
9. W. Lin, W. Wang, W. Wu, X. Pang, B. Liu, and Y. Zhang, “Sustainable Computing : Informatics and Systems,” *Sustain. Comput. Informatics Syst.*, 2017.
10. E. N. Alkhanak and S. P. Lee, “A hyper-heuristic cost optimisation approach for Scientific Workflow Scheduling in cloud computing,” *Futur. Gener. Comput. Syst.*, no. ii, 2018.
11. A. Xiong and C. Xu, “Energy Efficient Multiresource Allocation of Virtual Machine Based on PSO in Cloud Data Center,” vol. 2014, 2014.
12. F. Zhang, J. Ge, Z. Li, C. Li, and C. Wong, “A load-aware resource allocation and task scheduling for the emerging cloudlet system,” *Futur. Gener. Comput. Syst.*, 2018.
13. K. Gai, M. Qiu, and H. Zhao, “Energy-aware task assignment for mobile cyber-enabled applications in heterogeneous cloud computing,” *J. Parallel Distrib. Comput.*, 2017.
14. M. Dayarathna, Y. Wen, S. Member, and R. Fan, “Data Center Energy Consumption Modeling : A Survey,” *IEEE Commun. Surv. Tutorials*, vol. 18, no. 1, pp. 732–794, 2016.
15. J. J. Q. Yu and V. O. K. Li, “Ac ce p te d cr t,” *Appl. Soft Comput. J.*, 2015.
16. J. Hereford and C. Blum, “FlockOpt : A new swarm optimization algorithm based on collective behavior of Starling birds,” pp. 17–22, 2011.
17. N. Netjinda, T. Achalakul, and B. Sirinaovakul, “Particle Swarm Optimization inspired by starling flock behavior,” *Appl. Soft Comput. J.*, vol. 35, pp. 411–422, 2015.
18. G. Shen and Y. Zhang, “A Shadow Price Guided Genetic Algorithm for Energy Aware Task Scheduling on Cloud Computers,” pp. 522–529, 2011.
19. Z. Wang, K. Shuang, L. Yang, and F. Yang, “Energy-aware and revenue-enhancing Combinatorial Scheduling in Virtualized of Cloud Datacenter,” vol. 7, no. January, pp. 62–70, 2012.
20. L. Ismail and A. Fardoun, “EATS : Energy-Aware Tasks Scheduling in Cloud Computing Systems,” *Procedia - Procedia Comput. Sci.*, vol. 83, no. Seit, pp. 870–877, 2016.
21. D. Kliazovich, P. Bouvry, and

S. Ullah, “DENS : data center energy-efficient network-aware scheduling,” no. April, 2011.

22. D. Kliazovich, S. T. Arzo, F. Granelli, P. Bouvry, and S. U. Khan, “e-STAB : Energy-Efficient Scheduling for Cloud Computing Applications with Traffic Load Balancing,” 2013.
23. F. E. Farkar, “Bi-Objective Task Scheduling in Cloud Computing using Chaotic Bat Algorithm,” vol. 8, no. 10, pp. 223–230, 2017.
24. J. P. D. Comput *et al.*, “Author ’ s personal copy Online optimization for scheduling preemptable tasks on IaaS cloud systems.”
25. P. Azad, T. Branch, N. J. Navimipour, and T. Branch, “An Energy-Aware Task Scheduling in the Cloud Computing Using a Hybrid Cultural and Ant Colony Optimization Algorithm,” vol. 7, no. 4, pp. 20–40, 2017.
26. C. F. Schaber, S. N. Gorb, and F. G. Barth, “Force transformation in spider strain sensors : white light interferometry,” no. October 2011, pp. 1254–1264, 2012.
27. C. W. Clark, “Foraging and Flocking Strategies : Information in an Uncertain Environment,” no. May, 1984.
28. [28] N. W. F. Bode, D. W. Franks, A. J. Wood, N. W. F. Bode, W. Franks, and A. J. Wood, “Limited interactions in flocks : relating model simulations to empirical data Limited interactions in flocks : relating model simulations to empirical data,” no. September 2010, 2011.
29. I. Y. Kim and O. L. De Weck, “Adaptive weighted-sum method for bi-objective optimization : Pareto front generation,” no. February, 2005.

AUTHORS PROFILE

Arul Xavier V.M has completed B.Tech degree in Information Technology from Velammal Engineering College, Madras University, Chennai and received the M.E. Degree in Computer Science and Engineering from Noorul Islam College of Engineering, Anna University, Chennai. He is currently working as Assistant Professor in Department Computer Science and Engineering at Karunya Institute of Technology and Sciences, Coimbatore. Presently, he is doing Ph.D under Anna University, Chennai. His research interest includes Cloud Computing, Data Science and Machine Learning.

Dr.S.Annadurai received B.E degree in Electronics and Communication Engineering and M.E degree in Power System from the University of Madras, Chennai and M.E degree in Computer Science and Engineering from Bharathiyar University, Coimbatore. He completed his Ph.D degree from Anna University, Chennai. He is currently working as a Professor and Advisor at Hindusthan Group of Technical Institutions, Coimbatore. He has contributed more than hundred research papers in different areas of computer science and engineering such as Automatic Speech Recognition, Neural and Fuzzy Neural Networks, Image Processing and Computer Networks. He is a life member of Professional bodies such as CSI and ISTE. He was a chairperson and organizing committee of many national and international conferences held in India and abroad. His research interests include Image Processing, Computer Networks, and Fuzzy Neural Networks, Cloud Computing and Automatic Speech Recognition.