Received: October 18, 2016

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Energy-Aware Fruitfly Optimisation Algorithm for Load Balancing in Cloud Computing Environments

M. LawanyaShri[1*] **S. Subha[2]** **Balamurugan Balusamy[3]**

[1]*SITE, VIT University, India*
[2]*VIT University, Vellore, Tamil Nadu, India*
[3]*VIT University, Vellore, India*
Corresponding author's Email : *lawanyaraj@gmail.com*

Abstract: An effective task scheduling is one of the vital aspects for effectually hitching the potential of cloud computing. The most important aspect of task scheduling focuses on balancing the load of tasks among virtual machines, which is independent in nature. Energy conservation is one of the major key issues in cloud environment which in turn reduces operation costs in cloud datacenter. Meanwhile, Energy-aware load balancing optimisation technique is a promising way to attain the goal. To ensure fast processing time and optimum utilization of the cloud resources, we propose an energy-aware Fruit fly optimisation algorithm (EFOA-LB) for balancing the load among virtual machines in the cloud system. The energy-aware EFOA-LB is a modern swarm intelligence algorithm inspired by the foraging behavior of fruit flies, aims to attain well-balanced load on virtual machines and reduces energy consumption accordingly. Based on results obtained from our simulations, the proposed algorithms minimizes makespan and reduces the energy consumption of the datacenter, while meeting the task performance. The experiment results indicate that the energy-aware EFOA-LB algorithm is more efficient than the existing load balancing algorithms.

Keywords: Cloud computing, Load balancing, Swarm intelligence, Fruitfly optimisation algorithm (FOA), Foraging behavior, Performance evaluation.

1. Introduction

Cloud computing is thriving in the recent era; an internet based computing offers shared resources such as hardware, software, and information on demand basis. The service dynamism, flexibility, robustness and elasticity afforded by this scalable technology make cloud computing an integral part of enterprise computing environment. Cloud computing provides many services like platform as a service, infrastructure as a service, software as a service, data as a service and so on to the customers via the internet. Efficient handling of the cloud environment is essential to obtain maximum aids out of it [1]. Cloud environment provides computing resources dynamically in a virtualized manner to the end user which enables elastic scaling of resources [2]. To satisfy the user needs, provisioning of cloud resources in the datacenter (DC) is an essential criterion. One of the key factors in resource management is how well the cloud resources allocated, migrated and managed [3]. End users share the resources in cloud computing through the notion of virtualization. The virtual machine is an imperative component of a cloud datacenter. Cloud data is restored on different servers, and each server is interconnected and accessed through virtual machines. The end users are interested in reducing the overall execution time of their tasks on virtual machines (VMs). Such a way that the virtual machines should complete the execution of each task which is assigned to it as early as possible. The issues mentioned above lead to the problems in the scheduling of the tasks with the available VMs [4]. The scheduler in the cloud should perform the scheduling process efficiently, to utilize the resource which is available in a significant way. This type of systems should ensure whether the loads are well balanced in all VMs or not. The key role of the scheduler is to ensure whether the particular virtual machine is neither overloaded nor under-utilized. The primary target of the load balancing techniques is to speed up the execution of tasks to the resources during unpredictable workloads assigned dynamically. Load balancing in cloud computing is a crucial problem to ensure fast processing time and optimum utilization of cloud resources [5]. The workload of each resource is calculated as the sum of the expected time to compute of individual independent tasks. The decision of the load balancing is carried out when the imbalance factor is greater than balancing overhead at a particular period [6]. The work of the scheduler or the scheduling algorithm is to assign tasks to various virtual machines. When various tasks overload a virtual machine, then the tasks should be removed and assigned to a different virtual machine which is under-utilized in the same datacenter. If more than one VM are available, then the removed task should be submitted to the virtual machine such that none of the tasks should wait to execute for a long period. In this context, load balancing in cloud computing environment is based upon on virtual machine level.

In our approach, the load balancing in a cloud environment can be achieved by the foraging behavior of fruit flies [7]. FOA is appropriate to the load balancing problem in a cloud environment for the ensuing reasons: first, the number of parameters in fruit fly optimization algorithm is very few, which makes easy to implement. Second, the applications of FOA in various problems has verified that this algorithm is applicable for scheduling and load balancing problems, and it is also competitive with other intelligent optimization algorithms. Third, FOA search framework incorporates exploitation ability enhancement, which can be achieved using local search approaches. Local Searching process is one of the evolutionary algorithms for searching optimal solution in which heuristics, meta- heuristics and operators are included in it [8]. By nature, each fruit fly is the primary part of the fly population. It has two phases to search. The first phases are the smell based searching process and the second one is a vision based searching process, each one of the fruit flies can reach the optimal solution space in an adequate computational time. When a particular virtual machine is overloaded, some set of tasks will be removed, the task which is removed from the overloaded virtual machine is named as fruit flies. The fruit flys searches for the underutilized virtual machine (food) to allocate the removed task. If more than one under-utilized virtual machine is available in the same datacenter, then based on the priority of the tasks, and load on the virtual machine, the decision is carried out, i.e., finding a best optimal solution.After submission to an under-utilized VM, the task will also update the load of selected VM to all the tasks which are in a wait state. Updating status of the virtual machine will be helpful for the other fruit flies to search their food. Load balancing in cloud provisions high utilization of resources and user satisfaction ratio. Balancing the load in cloud data center is a challenging issue where diverse applications and data are mounting so that the cloud system should tackle and process the workloads fast enough within a required period.The poor load balancing strategy leads to the reduction in response time and resource wastage. Low utilization of cloud resources will result in poor performance, QOS and high energy consumption in cloud data centers which in turn increase the operational cost of the data center.

To address these issues, energy-aware EFOA-LB is proposed to achieve not only the efficient processing and utilization of a cloud computing resources but also to minimize energy consumption.

The main contributions of the paper are:

- Design and implementation of EFOA-LB for load balancing of non-pre-emptive tasks in a cloud environment.
- EFOA-LB approach is evaluated using response time, makespan, degree of imbalance and energy consumption as a performance metrics among virtual machines.
- Extensive performance analysis and comparisons are carried out to determine the effectiveness of the proposed EFOA-LB for load balancing and striving to save energy and cost.

The proposed approach optimally balances the load among virtual machines and reduces the energy consumption . We aim to simultaneously minimise the makespan and cost of the data center while constructing the solution for load balancing. From the simulation results conducted, we could show that our proposed approach outperforms Honeybee load balancing algorithm(HBB-LB), Particle Swarm Optimization (PSO) and weighted round algorithm (WRR). The result obtained in our proposed model assuredly bounces optimum solutions and apparently indicates the triumph of an efficient load balancing of tasks on minimal energy consumption and cost.

Rest of this paper is structured as follows: Section 2 discuss the related work on existing techniques. Section 3 describes the Fruit fly optimization algorithm and problem formation. Section 4 emphases on our proposed work with detailed algorithm and section 5 illustrates the simulation results by the comparison of proposed algorithm with existing algorithms. Finally, we gave our conclusion and future enhancement in section 6.

2. Related Work

Load balancing in cloud computing is an NP-hard problem. Many researchers have recently addressed load balancing issues in cloud computing. The amount of computation essential to find the best optimum solution is based on the size of the problem. In this section, we have discussed some current load balancing mechanism in cloud computing. Abdullah et al. [9] designed and implemented Symbiotic Organism search optimisation algorithm for task scheduling in a cloud environment to reduce makespan, the degree of imbalance and the response time. Liang Hong et al. [10] proposed a typical cloud model based FOA (CMFOA) to increase the performance. Ling et al. [11] proposed a novel binary FOA (bFOA) to solve multidimensional knapsack problem (MKP). Smell-based, local vision based and global vision based searching process are designed to accomplish evolutionary search. bFOA uses binary string representation to solve MKP. Hong et al. [12] developed a hybrid annual power load forecasting model combined with fruit fly algorithm (FOA) and Generalized Regression Neural Network (GRNN), FOA is used to choose spread parameter value automatically to improve the forecasting accuracy of GRNN in power load annual forecasting. Peng et al. [13] applied an improvement in the fruit fly optimisation algorithm to find solutions for lot-streaming flow shop scheduling problem; they used neighborhood search and global cooperation search process to determine the jobs splitting and order of sub-lots instantaneously. Zheng et al. [14] used a novel optimisation algorithm named fruit fly to solve semiconductor final testing scheduling. Kennedy et al. [15] have discussed Particle Swarm Optimisation algorithm and their application in various fields. Dasgupta et al. [16] proposed a novel load balancing technique using GA. They used a Genetic algorithm for natural selection strategy to balance the workload among cloud virtual machines. Their work is compared with traditional algorithms like FCFS and Round Robin technique. Goyal et al. [17] proficiently proposed a dynamic load balancing based on Ant -colony in computational grid. The proposed approach rather than associating pheromone with the path, it is associated with the resources. The major goal of their proposed load balancing algorithm is to map the jobs with the resources to balance the load and improve resource utilisation. Alakeel et al. [18] proposed a technique for dynamic approaches in which each task moves dynamically from overload machine to the underutilised machine by changing dynamically and continuously based on the current state of the system which improves the performance compared to static load balancing strategies. Lu et al. [19] discussed load balancing method in which only the extra tasks are migrated from overloaded virtual machine to under-loaded virtual machine in the different host using particle swarm optimisation algorithm. Anton, Jemal, and Buyya et al. [20] presented energy efficient cloud architecture and principles for resource allocation and provisioning. They proposed an algorithm for energy efficiency in which it provision resources in the datacenter for client applications that improves datacenter energy efficiently with QOS and developed an autonomic mechanism for self-managing the resource state to achieve energy efficiency. Youwei et al. [21] discussed energy efficient Scheduling algorithm EEVS with deadline constraint in the cloud environment. The EEVS approach divides similar schedule periods; VMs are allocated appropriately to PMs, which reduce 20% of energy and increases 8% of processing capacity. Gregory et al.[22] presented a service energy-aware framework for managing Virtual Machines in Clouds. The solution consents the collection of various metrics the virtual and physical infrastructure for attaining multi-source monitoring in a cloud environment. Chase et al [23] proposed a policy based hosting center for allocating resources with less energy consumption. The proposed method moderates energy by switching off idle servers in an efficient way using resource management architecture. Zikos et al. [24] Proposed an efficient performance and energy aware algorithm for scheduling the tasks in clusters to improve performance and reduce power consumption. Karakoyunlu et al. [25] proposed energy efficient allocation method based on metadata heterogeneity for cloud storage. Inactive nodes are switched to low energy mode to reduce the power consumption.

From the above analysis, There are only few works, which addressed the performance issues for load balancing in cloud environment. In most of the research approaches, the availability of the service and the energy efficiency in the cloud computing was not focused. The proposed approach deals with the three dimensional facet like makespan, energy and cost optimisation, when balancing the workload among the virtual machines in the cloud environment.

.

3. Foraging Behaviour of Fruitfly

Fruit fly optimisation algorithm is an evolutionary intelligent search algorithm which impressionists the food search procedure based on the drosophila's fruit fly behaviour as illustrated in Figure 1 and Figure 2. The osphresis organs of fruit fly can locate various kinds of scents floating, and can smell food sources even which is far away from it. Fruit fly sends and collects information from its neighbours, compares and find the location which is best using its acute vision and fitness by taste, if the taste is not good, then it discards and goes to another location until it finds the best optimal solution [26]. The fruit fly searching for food process consists of two steps:

1) It uses Osphresis organ to smell food sources and start to fly in that direction.
2) It moves forward the food location, fruit flies uses sensitive vision for food finding and flocking location.

Fruit Fly optimisation Algorithm

Step 1: Initialization of parameters, Assign the population size, initial fruit fly location, and a maximum number of generations.

Step 2: With Olfactory organ behavior, assign each fruit fly with the direction and distance to move for the food search randomly

$$X_k = X-axis + RandomValue$$

$$Y_k = Y-axis + RandomValue$$

Step 3: Evaluate the smell concentration fitness value of each food location to estimate the distance of food source.

$$Dis_k = \sqrt{X_{k^2} + Y_{k^2}}$$

$$SmellConcentration(S_k) = 1/Dis_k$$

Step 4 : Find smell concentration (smell) of every fruit fly by substituting smell concentration judge values to smell concentration fitness function .

$$Smell_k = Smell\_Function(S_k)$$

Step 5 : Identify the best smell concentration which has minimum value fruit fly from the group.

$$[Best\_Smell, Best\_index] = \min(Smell_k)$$

Step 6 : Using vision-based search the fruit fly swarm flies in the direction of location with the maximum smell concentration based on the X and Y coordinates.

$$Smell\_Best = Best\_Smell$$

$$X\_axis = X(Best\_index)$$

$$Y\_axis = Y(Best\_index)$$

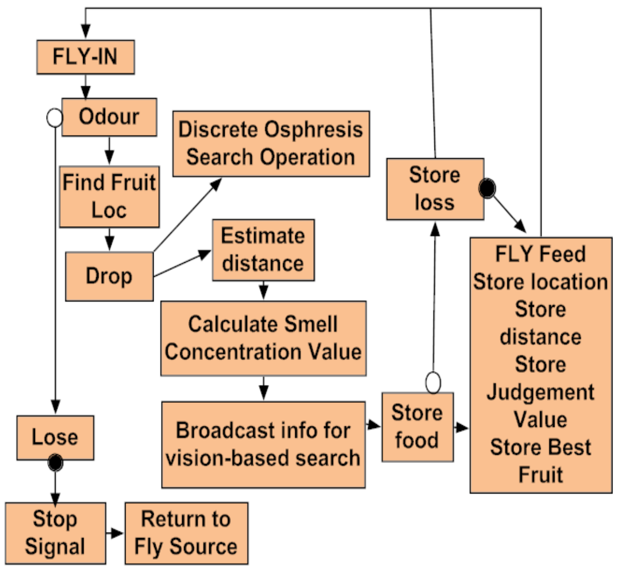
FLY-IN
Odour
Discrete Osphresis Search Operation
Find Fruit Loc
Drop
Estimate distance
Calculate Smell Concentration Value
Broadcast info for vision-based search
Store loss
Store food
FLY Feed Store location Store distance Store Judgement Value Store Best Fruit
Lose
Stop Signal
Return to Fly Source

Figure. 1 Flow diagram of Fruitfly behaviour

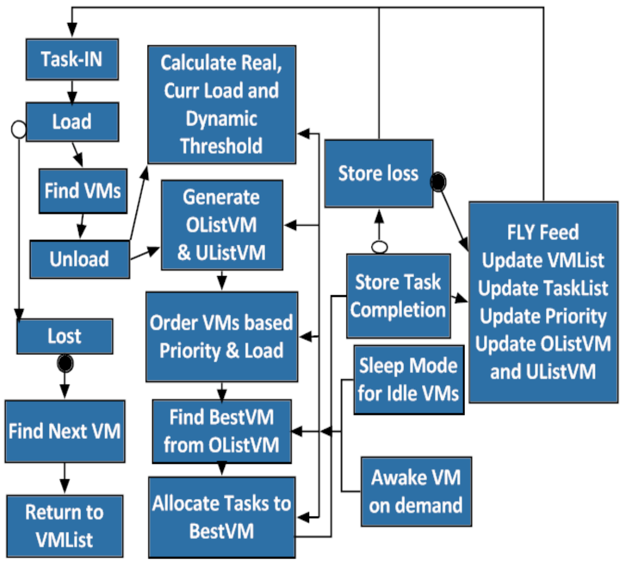
Task-IN
Load
Find VMs
Unload
Lost
Find Next VM
Return to VMList
Calculate Real, Curr Load and Dynamic Threshold
Generate OListVM & UListVM
Order VMs based Priority & Load
Find BestVM from OListVM
Allocate Tasks to BestVM
Store loss
Store Task Completion
Sleep Mode for Idle VMs
Awake VM on demand
FLY Feed Update VMList Update TaskList Update Priority Update OListVM and UListVM

Figure.2 Flow diagram of EFOA-LB

4. Energy-aware Fruit Fly Optimisation for Load Balancing (EFOA-LB)

Cloud computing is a technology trend modelled to provide resources on demand for the computational power generated by diverse computing-intensive and scientific applications. Each PM in a cloud environment can host many virtual machines, and more complex application can run on it. The virtual machine load can be determined by the size and type of application running on demand and can change dynamically. The dynamic workload nature needs an efficient scheduling and load balancing technique to reduce response time, completion time and energy consumption. So we proposed energy-aware EFOA-LB to balance the loads among virtual machines efficiently and reduce the energy consumption accordingly. Figure. 1 illustrates the flow diagram of FOA and our proposed method EFOA-LB.

Modelling:

The cloud datacenter comprise of l physical machines indexed by the set P= {p1, p2, ...pl } hosting of m virtual machines represented by the set VM= {v1, v2,………vm} and n tasks {tk1,tk2,……tk n }.The datacenter broker receives the request to execute the task; the broker maps the task j to appropriate virtual machine in the datacenter, where each task j takes $p_{ij,}$ units of time with allotted virtual machine i. The total number of tasks which is submitted to the datacenter broker in cloud environment follows a Poisson distribution; hence the arrival process of tasks to the cloud datacenter broker is a Poisson process. The service time of each cloud resources are determined as an exponential distribution, and the tasks are submitted to the datacenter broker with fixed arrival rate. The proposed approach follows *M/M/k* queuing model with k identical virtual machines with k queues not with a unique queue [28].

Average utilization of the system:

$$r = \frac{\lambda}{k\mu} \tag{1}$$

Tasks arrive at the system according to the Poisson input λ with *k* $(1 \le k \le \infty)$ parallel service channels and the service time of the virtual machine is distributed according to an exponential distribution with μ as the service rate.

Probability of no task in the system:

$$R_0 = \left[\sum_{x=0}^{k-1} \frac{(\lambda/\mu)^x}{x!} + \frac{(\lambda/\mu)^k}{k!} \left(\frac{1}{1-r} \right) \right]^{-1} \tag{2}$$

Average customers waiting in the queue:

$$L_Q = \frac{R^k (\lambda/\mu)^k r}{k!(1-r)^2} \tag{3}$$

Where L_Q is a random variable signifying the number of tasks waiting in the queue.

Average time of task spent waiting in the queue:

$$W_{TQ} = \frac{L_Q}{\lambda} \tag{4}$$

Where Wr_Q is a random variable signifying the time of the task waits in the queue

Average time of task in the system including service:

$$W_A = W_{TQ} + \frac{1}{\mu} \tag{5}$$

Where W_A is a random variable signifying the time of the task waits in the queue including service time.

Average number of tasks in service system:

$$L = \lambda W_A \tag{6}$$

Where L is the number of tasks being serviced.

Makespan in the cloud can be well-defined as the completion time of the overall task. We represent the completion time of tasks T_i on the virtual machine VM_j as Completion Time $(T)_{ij.}$ The following defines the makespan function [29].

$$\begin{aligned} &Makespan = \max\{CompletionTime(T_{ij})\\ &i \varepsilon T, i = 1,2,.....,n\\ &j \varepsilon VM, j = 1,2,....,m \end{aligned} \tag{7}$$

Our proposed energy-aware EFOA-LB technique reduces the overall completion time of task T_i (makespan) along with the response time by balancing a load of each virtual machine by task migration from the overloaded to the under-loaded virtual machine.

Real Load of VM:

$$RL_j = num_j + mips_j + bw_j \quad (8)$$

Where num_j is the number of the processors in virtual machine j, $mips_j$ is the processing elements MIPS (million instructions per second) of all the processors in virtual machine j, and bw_j is the bandwidth for a VM_j.

Real Load of all virtual machines:

$$RL = \sum_{j=0}^{m} RL_j \quad (9)$$

RL_j is the real load or capacity of a virtual machine VM_j; RL is the overall load of all virtual machines.

Current Load of VM_j:

$$CurrL(VM_j) = \frac{num(t)}{\mu_j(t)} \quad (10)$$

Where *num(t)* is the number of tasks at a time t,
$\mu_j(t)$ is the service rate of a virtual machine at time t.

Current Load of all Virtual machines in the datacenter is

$$CurrL = \sum_{j=1}^{m} CurrL(VM_j) \quad (11)$$

The Current load of all virtual machines is calculated by the summation of all loads of VM.

Standard deviation of workload:

$$\sigma = \sqrt{\frac{1}{m}\sum_{i=0}^{m}(CurrL(VM_j) - CurrL)} \quad (12)$$

Execution time of Task T_i:

$$exec(T_i) = \frac{l_{T_i}}{c(VM_j)} \quad (13)$$

Where l_{Ti} is the length/size of the task T_i and c (VM_j) is the fractions of CPU performance [30].

Load Balancing Decision:

After determining the Real load and current load of Virtual machine VM, the cloud system should take a decision that whether load balancing of the virtual machine can be done or not. The following cases are considered for decision-making

i) If the current load of all Virtual Machine VM is greater than Upper Dynamic threshold value, then load balancing is not possible, so the system should reset the load and redeploy the virtual machines.
ii) If the current load is greater low Dynamic threshold value and current load is less than the upper dynamic threshold value, then System is balanced.
iii) If the current load is less than the low Dynamic threshold value and if current load is not equal to null then find the best virtual machine from Under-loaded virtual machine list and assign the removed task to a virtual machine. Such that the system is balanced.
iv) if the current load is lesser than the low dynamic threshold value and null then remove the particular virtual machine from VMList in the host.

Smell-based and Vision based search:

The smell based search is the basic searching procedure, where S fruit flies are generated and for the sub-population. In EFOA-LB technique the smell based searching process find the list of Overloaded and Under-loaded virtual machine which is suitable for the removed task with priority consideration. The Vision based searching process evaluates the fruit flies to get smell concentration value. In EFOA-LB technique the vision based search evaluates the BestVM, which suits for the task to be allocated.

Grouping Virtual Machines:

The grouping of the virtual machines is carried out based their current loads. In the proposed work there are two categories of List, overloaded virtual machines in OListVM and underloaded virtual machines in UListVM. The task in the overloaded virtual machine from OListVM is removed and decides an appropriate virtual machine among the virtual machines in the under-loaded virtual machine list. The removed task is considered as a swarm, it searches for the best virtual machine from UListVM considering the priority of task so that no more high priority tasks in the same virtual machine and allocate the removed task into it, updates the task list, virtual machine list, OListVM, and UListVM accordingly. The procedure continues till there is no virtual machine in OListVM list.

Grouping Tasks:

Each task in virtual machines is classified in one of the three cadres such as HPT (High Priority Task), LPT (Low Priority Task) and MPT (Medium priority Task). The priority of the task is imperative when the task migration happens from overloaded virtual machine to under-loaded virtual machine. The removed task from an overloaded virtual machine called swarm can choose the best virtual machine based on the priority that is if the task is HPT, and then the swarm has to select the virtual machine having less number of HPT tasks. If the task is MPT, then the swarm has to select the virtual machine having less number of HPT and MPT tasks.

HPT: $Best_p VM = Min(Count(HPT)\varepsilon VMj)$

MPT:
$Best_p VM = Min(Count(HPT) + Count(MPT)\varepsilon VMj)$

LPT: $Best_p VM = Min(Count(TaskT)\varepsilon VMj)$

Energy Consumption and Cost:

The proposed work not only focused on load balancing, but it also concentrates on energy saving to reduce the cost of the datacenter. The primary function of energy conservation is to make a resource to on to off state which is not in use. This strategy can be attained by identifying the appropriate virtual machine and switching off it in the datacenter that is not used to minimize the energy consumption (Sleep mode). The possibility of identifying the underutilized virtual machine and turning the sleep mode off and on of the resources entirely depends on the threshold value.

The proposed approach uses dynamic threshold value which is categorized as a High dynamic threshold and Low dynamic threshold value based on the load of the virtual machines in the datacenter. If the load of VM_j goes less than LDT, then that particular virtual machine is put into sleep mode, and if the load of virtual machine VM_j goes greater than UDT, then awake the virtual machines which are in sleep mode. If the current load of a virtual machine is null and there is no Overloaded virtual machine to balance the load, then the particular virtual machine is removed from VMList to save energy.

```
If current load of $j^{th}$ virtual machine $==\varphi$ then
        LFit $\leftarrow$ $VM_j$
        Move LFit to Sleep Mode
```

It is a temporary situation where at the current state, if there are many numbers of virtual machines in the UListVMp, having current load = φ, but the system is not balanced, then, in this case, some LFit virtual machines are transferred to sleep mode to reduce energy consumption.

```
If overloaded OListVMp$!=\varphi$ and
UnderloadedUListVMp $==\varphi$ then
Awake the virtual machine from sleep mode.
```

There is a need for virtual machines to migrate the removed task from overloaded virtual machines, so the proposed system will awake the virtual machines which are in a sleep state so that the awaken virtual machines can be used to allocate the removed task from overloaded virtual machine to make the system balance the load.

```
If overloaded OListVMp $==\varphi$ and
UnderloadedUListVMp $!=\varphi$ then
VMList.remove( $VM_j$)
```

The system is balanced and still there are underloaded virtual machines in the datacenter, which consumes energy so in such case, the proposed system removes the underloaded virtual machine from the VMList. The cost for the datacenter will be decreased with the resultant of a decrease in energy consumption with a dynamic threshold value and live migration [31]. The energy consumption produced by the task Ti running on virtual machine VM_j is $econ_{ij,}$ and $econ\_rate_j$ is the energy consumption rate of VM_j, and execution time for task T_i is $exec(T_i)$ from equation (13) Energy consumption is calculated as

$$econ_{ij} = econ\_rate_j \times exec(T_i)$$
(14)

Total energy consumption is calculated from equation (14)

$$E = \sum_{j=1}^{m} \sum_{i=1}^{n} econ_{ij} \quad (15)$$

The cost for the datacenter is calculated from equation (15)

$$Cost\_D = c \times E \quad (16)$$

Where c is the cost of 1kW power

Nomenclature:

Symbol	Meaning
VM	Virtual Machine
CurrL	Current Load of Virtual Machine
LDT	Lower Dynamic Threshold value
UDT	Upper Dynamic Threshold value
$Real_L$	Real Load of VM (Capacity)
LPT	Low Priority Task
NPT	Normal Priority Task
HPT	High Priority Task
Num_s	Number of Fruit Swarms
$OListVM_p$	Overloaded Virtual Machine List
$UListVM_p$	Underloaded Virtual Machine List
BestVM	Best Virtual Machine to Migrate
LFit	Low Fit Virtual Machine
$BestVM_p$	Best VM based on Priority
FS_k	kth fruit fly swarm
VM_p	virtual machine list of FS_k
Ff_{kp}	Fruit Flies in S

EFOA-LB algorithm:

```
For all VM in DC do
    Determine the RealLoad of all VM in the datacenter
    from equation (9).
End For
For all {VMj} ε VM do
    Determine current tasks allocated to VMj
    (Current load) from equation (10)
End For
For all { $VM_j$} ε VM do
if  CurrL> UDT then
     Load Balancing is not possible
     Reset the load.
 Else if CurrL> LDT &&CurrL<UDT then
         System is balanced
         Exit
End For
For all k in $nums_S$//Smell based search
 Generate S fruit-flies $Ff_{kp}$(p=1,2,…….S) on $FS_k$
 For all { $VM_j$} ε VM do
     Determine $VM_j$ which is over-loaded
     $OListVM_p$<- $VM_j$
     Determine VMj which is underloaded
     $UListVM_p$<- VMj
End For
End For
 For all k in $nums_S$  [Vision based search]
    For all l in S
      Evaluate the generated fruit-flies $Ff_{kp}$
    End For
  For all {VMj} ε VM do
   If CurrL< LDT then
         If CurrL != φ
             Sort all VMj ε VM in ascending order
             Sort all tasks based on priority LPT, NPT,
              and HPT
             For each task in $OListVM_p$
                Find BestVM from $UListVM_p$ such that
                CurrL + TL >= LDT &&CurrL+TL<=
                UDT && $Best_P$VM
              [Best fruit fly in the sub-population – vision]
             End For
             Migrate (T, BestVM)
         Else
     LFit← $VM_j$
     Move LFit to Sleep Mode
  Else
     Sort all VMs in descending order based on Load
     Sort the task based on priority in each VMList
             For each task t in OListVMp,
               Find appropriate bestVM such that
               CurrL + TaskLength<= UDT
             End For
             Migrate (T,BestVM)
        Update TList in VM
        Update VMList in Host
        Update Load of all VMs
 If OListVMp != φ&&UListVMp ==NULL then
 Awake the virtual machine from sleep mode.
 Endif
    If OListVMp == NULL and UListVMp !=NULL
     then
        For all {VMj} in UListVMp
         VMList.remove( $VM_j$)
        End For
   Update VMList and UListVMp
End For
End For
```

5. Experimental Results

The Efficiency of our proposed Fruit Fly Optimisation algorithm for load balancing is evaluated using Cloud Sim tool [32]. It is a comprehensive simulation framework for modeling, investigating and simulating the cloud environment. We have evaluated the efficiency and performance of our proposed optimisation algorithm based on the simulation results. In this section, we have compared FOA based load balancing algorithm with the existing optimisation algorithms like HBB-LB (HoneyBee Behaviour) , PSO (Particle Swarm Optimisation) and WRR (Weigthed Round Robin Algorithm). Figure .3 illustrates the makespan comparisons before and after load balancing using EFOA-LB. The X-axis in the graph specifies the number of cloudlets (tasks) and Y-axis specifies the Completion Time (Makespan) in secs. The makespan is reduced significantly using our proposed EFOA-LB Algorithm. Figure.4 indicates the makespan comparisons of EFOA-LB, HBB-LB, PSO, and WRR. The X-axis in the graph illustrates the number of tasks and Y-axis specifies the Completion Time (makespan) in secs. Figure.5 shows the response time of virtual machines (VMs) in secs for EFOA-LB, HBB-LB ,PSO, and WRR. The X-axis indicates the number of cloudlets (tasks) and Y-axis shows the response time of secs. Figure.6 illustrates the degree of imbalance between virtual machines (VMs) before and after EFOA-LB algorithm. Degree of imbalance can be determined by the following [33].

$$Deg\_Imbalance = Max\ (T_i) - Min\ (T_i) / Avg(T_i)$$

Where Max (T_i) and Min (T_i) are the maximum and minimum task (T_i) of virtual machines (VMs) and $Avg(T_i)$ is the average of task (T_i). The proposed EFOA-LB approach reduces the degree of imbalance considerably. Figure .7 depicts the total energy consumption (Joule) based on the number of virtual machines before and after EFOA-LB algorithm. The result shows that our proposed algorithm reduces the energy consumption drastically.

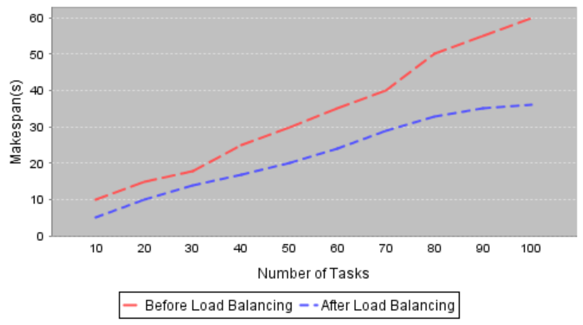

Figure.3. Comparison of Task Completion time

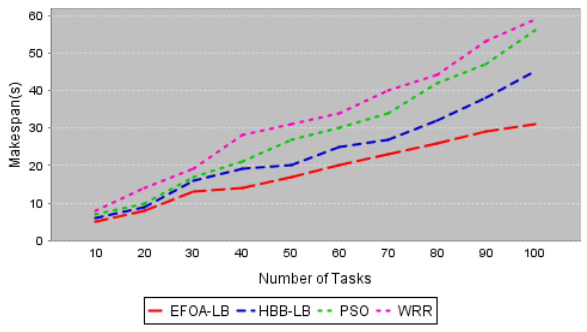

Figure.4. Comparison of Task Completion time (Makespan) for EFOA-LB, HBB-LB, PSO and WRR Algorithms

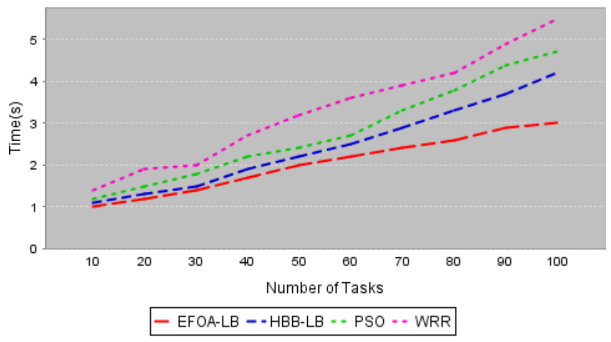

Figure.5. Comparison of Response time (seconds) of VMs for EFOA-LB, HBB-LB, PSO and WRR Algorithms

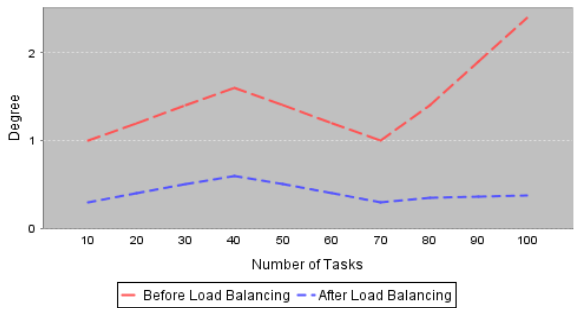

Figure.6. Degree of Imbalance before and after load balancing EFOA-LB

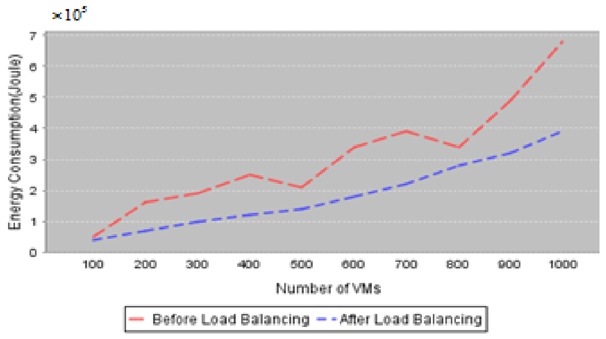

Figure.7. Total Energy Consumption of the Cloud before and after EFOA-LB

6. Conclusion and Future work:

In this work, we have proposed an energy-aware load balancing technique using Fruit fly optimisation in a cloud environment. The proposed work not only balances the load among virtual machines in the datacenter, but energy saving of datacenter in the cloud environment is also emphasized. The energy-aware load balancing strategy uses Fruit fly optimisation approach to balancing the load based on the Dynamic Threshold value along with the sleeping strategies to reduce the energy consumption. The energy-aware EFOA-LB algorithm improves the overall throughput of processing, reduces completion time and response time. Thus, it reduces the cost of the datacenter. The simulation results reveal that our proposed approach achieves greater performance compared to the existing method such as HBB-LB, PSO and WRR algorithm. The result also exposes that the proposed method is efficient, rational and address optimize usage of the virtual machines with apt load balancing and energy saving strategy from datacenter point of view. In future, we intend to extend the work with the other QOS factors such as network traffic information in a cloud environment. The network traffic in the cloud also consumes a non-trivial amount of energy which increases the cost of the datacenter.

References

[1] I. T. Foster, Y. Zhao, I. Raicu and S. Lu, “Cloud Computing and Grid Computing 360-Degree Compared ” , *In: Grid Computing Environments Workshop*, pp. 1–10, 2008.

[2] R. Buyya, R. Ranjan, and R N. Calheiros. "Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services", In *International Conference on Algorithms and Architectures for Parallel Processing*, pp. 13-31 , 2010.

[3] B. Balusamy, J. Sridhar, D. Dhamodaran and P. Venkata Krishna, “Bio-inspired algorithms for cloud computing: a review”, *International Journal of Innovative Computing and Applications*, Vol. 6, No. 3-4, pp. 181-192, 2015.

[4] R. A. Haidri, C. P. Katti, and P. C. Saxena, "A load balancing strategy for Cloud Computing environment ", In: *Proc. Of International Conf. on Signal Propagation and Computer Technology*, IEEE, pp. 636-641, 2014.

[5] B. Balusamy, "Extensive survey on usage of attribute based encryption in the cloud ", *Journal of Emerging Technologies in Web Intelligence*, Vol. 6 , No. 3 , pp. 263-272, 2014.

[6] B. Balusamy, and P. V. Krishna, “Collective advancements on access control scheme for the multi-authority cloud storage system”, *International Journal of Grid and Utility Computing* , Vol. 6, No. 3-4 , pp. 133-142 , 2015.

[7] J.Niu, W.Zhong, Y.Liang, N.Luo, F.Qian, "Fruit fly optimization algorithm based on differential evolution and its application on gasification process operation optimization ", *Knowledge-Based Systems* 88 ,pp. 253-263, 2015.

[8] B.Xing and W.J. Gao ,"Fruit Fly Optimization Algorithm", In *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*, pp. 167-170. Springer International Publishing, 2014.

[9] M.Abdullahi and M.A. Ngadi , "Symbiotic Organism Search optimization based task scheduling in cloud computing environment", *Future Generation Computer Systems* 56 ,pp. 640-650, 2016.

[10] L.Wu, C.Zuo and H.Zhang,"A cloud model based fruit fly optimization algorithm", *Knowledge-Based Systems* 89, pp. 603-617, 2015.

[11] L.Wang, X.Zheng and S.Wang , "A novel binary fruit fly optimization algorithm for so,kpkkiilving the multidimensional knapsack problem ", *Knowledge-Based Systems* 48 , pp. 17-23, 2013.

[12] H.Z. Li, S.Guo, C.J. Li and J.Q. Sun , "A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm." *Knowledge-Based Systems* 37, pp. 378-387, 2013.

[13] W.Kepu, X.Peng and H.Quanzhen, "Improved fruit fly optimization algorithm for TSP problems." *Comput Eng Des* 35, pp. 2789-2821, 2014.

[14] X.L. Zheng, L. Wang, and S.Y. Wang, "A novel fruit fly optimization algorithm for the semiconductor final testing scheduling problem", *Knowledge-Based Systems* 57, pp. 95-103, 2014.

[15] J. Kennedy, "Particle swarm optimization", In *Encyclopedia of machine learning*, pp. 760-766. Springer US, 2011.

[16] K. Dasgupta, B. Mandal, P. Dutta, J.K. Mandal, and S. Dam, "A genetic algorithm (ga) based load balancing strategy for cloud computing", *Procedia Technology* 10 , pp.340-347, 2013.

[17] S.K. Goyal and M.Singh, "Adaptive and dynamic load balancing in grid using ant colony optimization", *International Journal of Engineering and Technology* 4, no. 9 ,pp. 167,2012.

[18] A.M. Alakeel, "A guide to dynamic load balancing in distributed computer systems ", *International Journal of Computer Science and Information Security* 10, no. 6 ,pp. 153-160,2010.

[19] Y.Lu, Q.Xie, G.Kliot, A.Geller, and J.R . Larus, "Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services ", *Performance Evaluation* 68, no. 11 ,pp. 1056-1071, 2011.

[20] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing", *Future generation computer systems* 28, no. 5 ,pp. 755-768, 2012.

[21] Y.Ding, X. Qin, L.Liu, and T.Wang, "Energy efficient scheduling of virtual machines in cloud with deadline constraint ", *Future Generation Computer Systems* 50 ,pp. 62-74, 2015.

[22] G. Katsaros, J.Subirats, J.O. Fitó, J. Guitart, and P. Gilet, "A service framework for energy-aware monitoring and VM management in Clouds", *Future Generation Computer Systems* 29, no. 8 , pp. 2077-2091, 2013.

[23] J.S. Chase, D.C. Anderson, and P.N. Thakar, "Managing energy and server resources in hosting centers ", *ACM SIGOPS Operating Systems Review* 35, no. 5 ,pp. 103-116, 2001.

[24] S. Zikos and H.D. Karatza , "Performance and energy aware cluster-level scheduling of compute-intensive jobs with unknown service times", *Simulation Modelling Practice and Theory* 19, no. 1 ,pp. 239-250, 2011.

[25] C. Karakoyunlu and J.A. Chandy, "Exploiting user metadata for energy-aware node allocation in a cloud storage system", *Journal of Computer and System Sciences* 82, no. 2, pp. 282-309, 2016.

[26] H.Z. Li, S. Guo, C.J. Li and J.Q. Sun, "A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm", *Knowledge-Based Systems* 37, pp.378-387, 2013.

[27] W.T. Pan, "A new fruit fly optimization algorithm: taking the financial distress model as an example ", *Knowledge-Based Systems* 26 , pp. 69-74 , 2012.

[28] F. Spies, "Modeling of optimal load balancing strategy using queueing theory." *Micro processing and microprogramming* 41, no. 8 , pp. 555-570 , 1996.

[29] P. Brucker, "Parallel Machines." In *Scheduling Algorithms*, pp. 101-144. Springer Berlin Heidelberg, 1998.

[30] X. Zhu, L.T. Yang, H. Chen and J. Wang, "Real-time tasks oriented energy-aware scheduling in virtualized clouds." *IEEE Transactions on Cloud Computing* 2, no. 2 , pp.168-180, 2014.

[31] R. Sinha, N. Purohit, and H. Diwanji, "Energy efficient dynamic integration of thresholds for migration at cloud data centers" , *IJCA Special Issue on Communication and Networks* 1, pp. 44-49. 2011.

[32] R. Buyya, and R. Ranjan, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities." In *High Performance Computing & Simulation,. HPCS'09. International Conference on*, pp. 1-11. IEEE, 2009.

[33] L.D. Babu and P.V. Krishna," Honey bee behavior inspired load balancing of tasks in cloud computing environments",. *Applied Soft Computing*, *13*(5), pp.2292-2303, 2013.