Energy-aware Hybrid Fruitfly Optimization for load balancing in cloud environments for EHR applications

M. Lawanyashri, Balamurugan Balusamy, S. Subha

www.elsevier.com/locate/imu

PII: S2352-9148(17)30018-7
DOI: http://dx.doi.org/10.1016/j.imu.2017.02.005
Reference: IMU32

To appear in: *Informatics in Medicine Unlocked*

Received date: 2 January 2017
Revised date: 20 February 2017
Accepted date: 22 February 2017

Energy-aware Hybrid Fruitfly Optimization for load balancing in cloud environments for EHR applications.

M.Lawanyashri[*], Dr. Balamurugan Balusamy, Dr. S.Subha
School of Information Technology and Engineering ,VIT University.

[*]Corresponding Author's. lawanyaraj@gmail.com

Abstract:
Cloud computing has gained precise attention from the research community and management of IT, due to its scalable and dynamic capabilities. It is evolving as a vibrant technology to modernize and restructure healthcare organization to provide best services to the consumers. The rising demand for healthcare services and applications in cloud computing leads to the imbalance in resource usage and drastically increases the power consumption resulting in high operating cost. To achieve fast execution time and optimum utilization of the virtual machines, we propose a multi-objective hybrid fruitfly optimization technique based on simulated annealing to improve the convergence rate and optimization accuracy. The proposed approach is used to achieve the optimal resource utilization and reduces the energy consumption and cost in cloud computing environment. The result attained in our proposed technique provides an improved solution. The experimental results show that the proposed algorithm efficiently outperforms compared to the existing load balancing algorithms.

Keywords : Cloud computing, Electronic Health Records (EHR),Load balancing, Fruitfly optimization Algorithm(FOA), Simulated annealing (SA), Energy consumption

Introduction

Cloud Computing is a technology trend that offers utility oriented services to a wide range of users [1]. It is a thriving paradigm, enables to host pervasive applications from the convergence of scientific and business domains [2]. The distributed and dynamic access to the virtualized datacenter makes the users access the IT resources anywhere and anytime using pay-as-you-go model. Recently, cloud computing has been extensively used and adopted by healthcare industries due to scalable and cost effective nature [3]. This hastening migration of health care to the cloud evidently signifies a step-change model for the healthcare industry due to the intrinsic features like rapid provisioning, fast deployment, elasticity, greater resiliency, lower costs, and data storage solutions. It affords an ultimate platform for the healthcare industry and thereby providing efficient medical services to the users [4]. Cloud computing can act as a pervasive and enduring game-changer in all the operations of the healthcare industry such as service provisioning, collaborative abilities, operating models and end-user services. The primary purpose of the cloud health care services is to offer healthcare users quick and easy access to the resources and provide a variety of efficient distributed services. The key objective of the cloud based health care applications is to improve the availability, scalability and increase the performance of the healthcare applications [5].

Many cloud healthcare users can get connected to the services offered by several datacenters, server and storage through an efficient load balancing mechanism. Balancing workloads among multiple servers, application scaling, routing traffic to the nearest server, scrutinising and minimising heavy traffic are the vital features of load balancer in cloud computing [6]. The datacenters in cloud computing are typically comprised of powerful heterogeneous servers, hosting several virtual machines with possibly different specifications and resource usages; this may lead to the imbalance in resource usage among virtual machines, which results in performance degradation and SLA violations

[7]. The workload on the datacenter resources is hoarded massively with the progression of EHR (Electronic Health Record) applications. However , the availability of healthcare data to the patients and consumers without causing significant delay is the most promising role in EHR aplications. To improve the utilization of the cloud resources and increase the performance , a proficient load balancing approach is predominantly crucial. The healthcare data are real-time data, with varied size over time. Allocating resources and resizing the resource usage based on critical data is an important research issue. Thus balancing the load among virtual machines for cloud based health care services is very essential [8] [9]. Load balancing in the datacenter is to assign the workloads evenly. Workload Migration from overloaded resources to under-utilized resources is the primary solution [10] [11].

Natural Computation has fascinated extensively with emergent concern among researchers in the field of optimisation. Nature-inspired algorithms are meta-heuristics and computationally fast that proficiently imitates the nature. Metaheuristic approaches are used to solve optimization problems. Optimisation is the progression of hunting the best conceivable solution to a given complex issue with certain limitations [12]. It is apprehensive to find best solutions which could be maximising the efficacy of a particular system or minimising the cost of processing. Hybridization can improve the convergence speed and quality of the solution attained by the metaheuristic techniques[13].

In our proposed approach, the balancing of workloads among the resources(VMs) can be achieved by the fruitfly foraging behavior. Compared to other species, sense of smell and vision of the fruitflies sensory acuity is healthier. The algorithm is easily understandable based on the simplest structure of the Fruitfly optimization (FOA). Due to very few number of parameters in fruitfly and easy to implement the nature makes the FOA suitable for load balancing problem in cloud computing. FOA approach is competitive to other optimization algorithms. There are some defects which make the technique with low accuracy of optimization, and it easily falls into a local optimum. To solve the drawbacks, we propose a hybrid FOA approach with SA. The hybridization proves that, the Simulated annealing based Fruitfly optimization algorithm for load balancing (FOA-SA-LB). It has a greater ability of global optimum searching. The convergence rate and accuracy of optimization are enriched greatly.

Here, we propose to apply FOA-SA-LB for load balancing in a cloud environment. In this approach, the fruitflies act like a collaborative agent which is used to balance the workloads among virtual machines. The FOA has two stages such as smell and vision based search. Whenever virtual machines are overloaded, the task of the overloaded virtual machine will be removed and placed in a different virtual machine which is not overloaded or underutilised. The searching stages in FOA-SA-LB is used to find an appropriate virtual machine which is suitable for the removed task to be allocated. The multi-objective function is defined, and load balancing model can follow the constraints subjected to the objective function. The approach uses a threshold value to identify the load of a virtual machine. If a particular virtual machine is overloaded, the task is removed and assigned to the identified virtual machine based on the deadline of the task execution. The comparison of FOA and FOA-SA-LB is depicted in figure.1 . A virtual machine having minimum deadline task will be chosen to improve the performance of the datacenter. The smell based search is used to find the locations or virtual machines available, and vision based search is used to find the best location or virtual machine to migrate the task from the overloaded one.

In this paper, we propose a hybrid Fruitfly optimization algorithm with a Simulated annealing approach to improve the convergence speed and quality of the solution. The main contribution of the paper is:

1. Hybridization of FruitFly optimisation algorithm with simulated annealing to find the optimum solution.
2. Design and implementation of the Hybrid FruitFly optimisation algorithm for balancing the tasks in a cloud environment.
3. A Multi objective function is defined for efficient load balancing of tasks among virtual machines to minimise makespan, energy and cost.
4. Performance analysis of the proposed approach with existing load balancing algorithm.

Rest of this paper is structured as follows: Section 2 discuss about the background work required for understanding the concepts. Section 3 gives the basic Fruitfly optimization and Simulated annealing approach. Section 4 presents the proposed approach. Section 5 focuses on the simulation results. Lastly, Section 6 concludes our work and delineates the future enhancement.

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 time , needed 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. Goyal et al. [14] efficiently proposed a dynamic load balancing algorithm using ant colony optimisation in grid computing. This paper associates the pheromone with the resources. The major goal is to balance the workload and improve resource utilization. Ajit et al. [15] presented a VM level load balancing approach using weighted signature. This paper considers the analysis of three existing algorithms as the preparation phase to reduce the response time of the user. Moradi et al. [16] Proposed a New Time probabilistic optimising algorithm; it elects the cloud resources grounded on best past status and the minimum completion time. The major goal of this optimised load balancing algorithm is to reduce the overall response time. Abdullah et al. [17] proficiently designed and proposed symbiotic organism optimisation algorithm for scheduling of tasks among virtual machines in the cloud datacenter. The primary goal of their suggested approach is to schedule the task efficiently and to minimise the makespan, response time and degree of imbalance. Maguluri et al. [18] proposed a stochastic model for balancing the load in a cloud environment, where the task arrives based on stochastic process. The author designates the model that achieves any random fraction of the capacity region of the datacenter in the cloud environment. They define through user constraints and optimisation conditions that the BestFit Scheduling approach is not throughput optimal. Shanthi et al.[19] proposed firefly algorithm for load balancing in cloud computing. In their approach, the algorithm adjusts the load, and the results show that their approach increases the performance on task migration, job arrival rate and reduced computational time. Ali et al. [20] proposed a guide for dynamic methods in which the task moves dynamically from overload machine to the underutilised virtual machine, by changing dynamically and continuously based on the current state of the system. The results show that it improves the performance compared to static load balancing methods. However, implementing dynamic load balancing approach effects in the more accurate result.Ramazani et al.[21] presented a task based load balancing algorithm. They removed only the additional tasks which make the virtual machine to be overloaded and placed to the underloaded virtual machine using particle swarm optimisation. Krishna et al.[22] proposed another solution for balancing the load in a cloud environment. They used honey bee load balancing algorithm to migrate the task from one virtual machine to the other using three types of priority. The task which is removed from one virtual machine updates the status of the task in that particular virtual machine. The load balancing technique increases the throughput and reduces the waiting time and makespan.The computational approach inspired from nature represents a source for researchers to build algorithms for complex optimisation methods that are usually infeasible to solve based on traditional approaches

The computational approach inspired from nature poses a source for researchers to create algorithms for complex optimization methods that are usually infeasible to solve based on traditional approaches. The metaheuristic optimization algorithm, like Genetic Algorithm (GA) [16], artificial bee colony (ABC) [17], particle swarm optimization (PSO) [18], tabu search [19], firefly optimization (FA) [20], ant colony optimization (ACO) [21] and so on are proposed by many researchers to find multi-objective solutions,. A Significant count of them is nature inspired to find the best optimal solutions. Each will find the candidate solutions for the problem and the quality of the solution is determined by the fitness function [22]. Genetic algorithm fits the greater class of evolutionary algorithms, and chromosomal principles used to generate better solutions to the optimization problems mostly coded in the binary string. However, the genetic operations are more complicated and time-consuming [23]. PSO and ABC algorithms are a powerful optimization technique which has relatively rapid iteration time, On the other hand, they can certainly fall into local extremes [24]. The hybrid algorithms were proposed by many authors to overcome the drawbacks of single optimization techniques. The hybrid mechanism in cloud computing is receiving increasing attention in the research community. It prevails the existing static and dynamic load balancing algorithm drawbacks by coalescing them and retaining each of the algorithm's benefits. By compounding more than one metaheuristic approach, the inherited disadvantages can be circumvented. In [25], a hybrid PSO-GA (particle swarm optimization-genetic algorithm) algorithm was presented for scheduling in machine tool production. GA-ACO (Genetic algorithm - particle swarm optimization) was suggested in [26] for electricity load forecasting.

Buyya et al. [32] presented a simulation –driven energy efficient model to evaluate the heuristics using active migration approach to dynamically reallocating the virtual machines based on the current desires of CPU performance. The result showed that the energy consumption is reduced substantially with QOS (Quality of Service) metrics. Zomaya et al. [33] present an energy efficient technique for task consolidation to increase utilization of resources. This approach maps each task to the virtual machine so that the energy consumed to execute the task is reduced without the performance degradation. Wang et al. [34] presented an adaptive-model-free technique for allocating resources and minimizing power consumption based on time-varying loads with QOS metrics such as queuing state, throughput and rejection amount for designing the scheme for resource tuning. Gregory et al. [35] investigated a service energy framework for virtual machine management in the cloud. The model monitors and calculates the power consumption of the cloud infrastructure and the gathered data are evaluated to make an efficient virtual management.

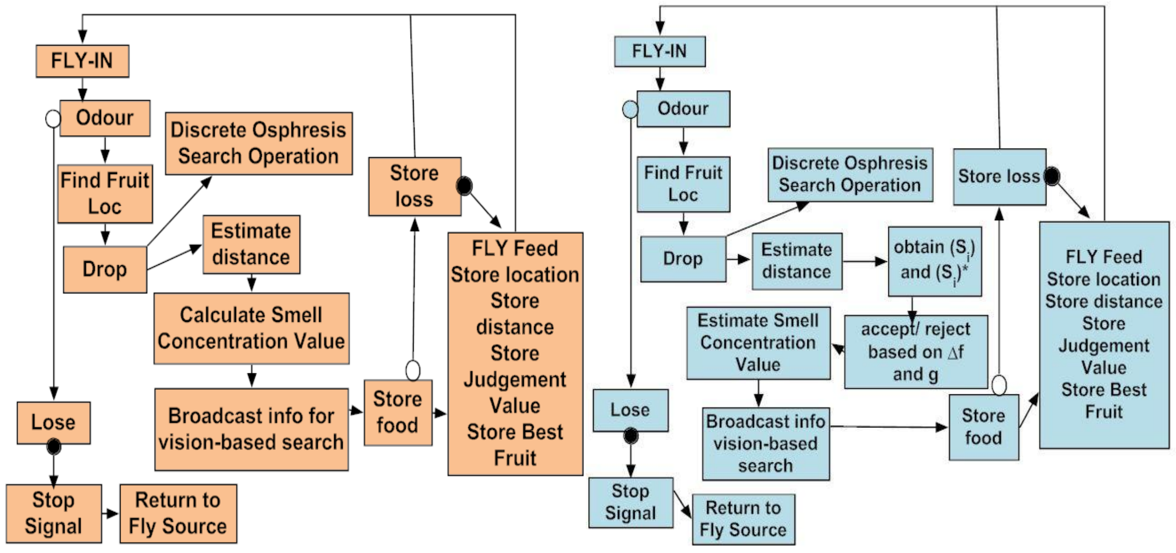

FLY-IN
Odour
Find Fruit Loc
Drop
Discrete Osphresis Search Operation
Estimate distance
Calculate Smell Concentration Value
Broadcast info for vision-based search
Store loss
Store food
FLY Feed Store location Store distance Store Judgement Value Store Best Fruit
Lose
Stop Signal
Return to Fly Source
FLY-IN
Odour
Find Fruit Loc
Drop
Discrete Osphresis Search Operation
Estimate distance
obtain (S_i) and (S_i)*
accept/ reject based on Δf and g
Estimate Smell Concentration Value
Broadcast info 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(a) FOA technique. (b) FOA-SA-LB technique

From the above analysis to the best of our knowledge, only handful work considers the availability of the service and the energy efficiency metrics for load balancing in a cloud environment. The proposed approach uses hybrid Fruitfly optimization with simulated annealing to attain the best optimum solution. The aim of our technique is to define a multi-objective function for optimal use of resources by reducing the three-dimensional aspects such as makespan, energy and cost optimization. From the simulation results conducted, we could show that our proposed approach outperforms better and overcomes the drawbacks such as the convergence rate in the FOA. Our approach outperforms the other existing algorithms like Honeybee load balancing algorithm (HBB-LB), Particle Swarm Optimization (PSO) and EFOA-LB (Energy-aware Fruitfly optimization for load balancing).

3 Fruit Fly Optimization Algorithm

Fruit fly optimisation is the new metaheuristic intelligent optimisation algorithm, proposed by Pan [36]. The simplest procedure and operative searching capability make the algorithm popular among researchers. The fruit fly optimisation algorithm is enthused by the scavenging behaviour of fruit flies. Recently it is used in many research fields , like web auction logistic services [37] , multidimensional knapsack problem [38], annual power load forecasting model [39], lot-streaming Flow-shop scheduling[40], financial distress [41], PID controller tuning [42], Travelling Salesman problem [43],steel making casting problem [44], location allocation inventory problem [45], Service composition [46]. The Fruit fly is superior, compared to other species in vision and osphresis. The following is the fruit fly searching process, and Algorithm.1 defines the procedure for FOA.

Step1: It uses Osphresis organ to smell the food sources and start to fly towards that direction.

Step2: It uses sensitive vision to find the best food and flocking location.

Algorithm1: **The basic procedure for FOA**

```
INPUT:  Population size, initial Fruit fly location, maximum number of iterations
OUTPUT: Optimal Solution
1.   Do
2.       For (All food in various location)
3.               Initialization
```

```
4.                Assign direction and distance to move
5.                Evaluate smell concentration fitness value
6.                Substitute the smell-concentration fitness value into fitness function
7.                Find the best smell by maximal smell concentration
8.                Use vision search to fly towards the best smell value location.
9.        End For
10.   While stopping condition is not exceeded.
```

It is an efficient method for finding global optimisation using Osphresis organ to smell the food and move towards that direction. It communicates information through its neighbours, equates and finds the perfect location using its desperate vision and fitness by taste.

Initialization Phase:

Initialization of the parameters, Assign the population size, Random Fruit fly location (X, Y) and a maximum number of iterations.

Assignment Phase: Give arbitrary direction and distance to fly for searching food.

$$X_k = X_{axis} + Random\_Value(dis\tan ce)$$
$$Y_k = Y_{axis} + Random\_Value(dis\tan ce) \tag{1}$$

Evaluation Phase: Evaluate the smell judgement value based on food location of each Fruit fly and estimate the distance of food source.

$$Dis_k = \sqrt{X_{k^2} + Y}_{k^2}$$
$$SmellConcentration(S_k) = 1 / Dis_k \tag{2}$$

Substitution Phase: Substitute the smell fitness value to fitness function to determine the smell concentration of each fruit fly.

$$Smell_k = Smell\_Function(S_k) \tag{3}$$

Identification Phase: Identify the best smell concentration which has a maximum value of smell.

$$[Best\_Smell, Best\_index] = \max(Smell_k) \tag{4}$$

Selection Phase: Using vision-based search, it flies towards the direction of food location based on $\max(Smell_k)$

$$Smell\_Best = Best\_Smell$$
$$X\_axis = X(Best\_index) \tag{5}$$
$$Y\_axis = Y(Best\_index)$$

Simulated Annealing:

Simulated annealing is a metaheuristic approach, inspired by the annealing process in metallurgy. It is a simple optimisation method comprising heating and cooling controller of material to intensify the size of its crystal. The energy is reduced according to the room temperature to lessen the defects in metallic structures. The simulated annealing technique uses its temperature progress as the control factor and its internal energy as the objective function. The simulated annealing process starts with a primary solution S and an updated solution created as S' . The solution for the procedure is generated if the fitness function F(S^*) value is lesser than F(S).

$$P_b = \exp\left(\frac{-(f(S^*) - f(S))}{T_m} \right) \tag{6}$$

The higher fitness value of S^* is accepted with the defined probability in equation (). This particular policy enables the searching process to avoid the entangle in local optima. Here F(S^*) is the fitness function of the neighbour solution, and F(S) is the fitness function of the current solution. Temperature Tm defines the control parameter. The equilibrium state is attained based on the succession of moves and based on the cooling rate, the temperature control parameter is determined. The control parameter T_m affects the performance of the global search. If the temperature obtains a high initial value, then the simulated anneal process will have a greater chance. After a succession of a decrease in the temperature, the SA procedure will be terminated, if there are no improvements. The possibility of locating global solution is further limited, if the initial temperature is low, and the computation time will be shorter.

$$T_m = \delta^k + T_o + T_{fn} \tag{7}$$

Where δ^k is the descending rate of the T_m, $0<\delta<1$, k is the number of stints, the neighbour solutions produced; T_o is the initial value of temperature, and T_{fn} is the final value of temperature. Algorithm. 2 defines the procedure for SA.

Algorithm 2: The basic procedure for Simulated Annealing

```
INPUT: Initial Value of Temperature, Final value of Temperature, and Cooling-rate
OUTPUT: Best optimal solution
1: Generate a primary solution so
2: Do
3: Generate a current solution s0 with the neighbour of s0
4: Calculate the probability $P_b$ based on the Equation ()
5: Accept or reject of new solution according to $P_b$
6: Update the best solution among the existing one
7: Minimize the Temperature
8: While stopping condition is not exceeded.
```

4. Problem Formation:

The physical machines m in the cloud data center is represented by the set of P={p1,p2,….pm}, with q virtual machines signified by V={v1,v2,…..vn} and k tasks T_k={t1,t2,….tk}. The user submits the task to the cloud broker. It is denoted by a group of parameters like t_i= {ar_i,ln_i,dl_i,ft_i}, where ar_i is the arrival time, s_i is the length or size of the task, dl_i is the time limit or deadline for the execution of the task, and ft_i is the finishing time of the task. The submitted task t_i is mapped to the virtual machine v_j by the cloud broker. The cloud broker maps the submitted task to the virtual machines. In this approach, we mainly focus on the utilization of the virtual machines, the completion time of the tasks (makespan), energy consumption, and cost of the datacenter.

Let P_k be the processing time of all tasks.[22]

$$P_n = \sum_{i=1}^{k} P_{ij} \quad j = 1, \ldots\ldots, m \tag{8}$$

Virtual Machine Capacity

$$cap_j = pe_{num\_p_j} \times pe_{mips\_p_j} \times v_{bw\_v_j} \tag{9}$$

Where pe is the processing element, $pe_{num\_p_j}$ is the processor count, $pe_{mips\_p_j}$ is the mips of all processors, $v_{bw\_v_j}$ is the bandwidth of V_j.

$$L_{VM(t)} = \frac{Num(T_k, t)}{Service\_rate(V_j, t)} \tag{10}$$

Where L_{VM} is a load of a particular virtual machine, $Num(T_k, t)$ is the total number of tasks at time t, $Service\_rate(V_j, t)$ is the service rate of virtual machine V_j at t.

A load of all virtual machines:

$$Load(VM) = \sum_{j=1}^{m} L_{VM(t)} \tag{11}$$

Processing time of virtual machine $PT_{(VM)_j} = \dfrac{L_{VM(t)}}{cap_j}$ (12)

Processing time of all virtual machines $PT_{(VM)} = \dfrac{Load \text{ of all } V_j}{Capacity \text{ of all } V_j}$ (13)

Execution time of Task T_k: $execu(T_k) = \dfrac{l_{T_k}}{c(V_j)}$ (14)

Where l_{T_k} is the length /size of the task T_k and the fractions of CPU performance is determined by $c(V_j)$.[47]

Let stt_{ij} be the start time of the task t_i on the virtual machine v_j, and ft_i be the finishing time of the task t_i on the virtual machine v_j. ft_i can be calculated by the following

$$ft_i = stt_{ij} + execu(T_k) \tag{15}$$

χ_{ij} is a decision variable used because, each task should be assigned to only one virtual machine. P_{ij} is defined as the processing time of the task T_i allocated to a virtual machine V_j.

$$\chi_{ij} = \begin{cases} 1 \text{ if } T_i \text{ is assigned to the virtual machine } V_j \text{ and } ft_i < dl_i, \\ 0 \text{ otherwise, if } ft_i > dl_i \end{cases} \tag{16}$$

Therefore, the objective function of the load balancing model is as follows:

(i) **Makespan**: Makespan in the cloud defines the overall completion time of tasks Tk in virtual machine V_j. Thus, This objective function is used to reduce the makespan of tasks.

Objective function :

$$\mathbf{F_1(Y)}= \text{Minimize} \left\{ \max_{t_i \in T_k, v_j \in V} ft_{ij} \right\} . \tag{17}$$

Where ft_{ij} is the finishing time of task t_i on the virtual machine v_j .

(ii)**Energy consumption**:

Let $econs_{ij}$ is the energy consumption produced by the task t_i running on the virtual machine v_j , econs_ratej represents the energy consumption rate of the virtual machine, and exect(T_k) is the execution time of the task.

The energy consumption is calculated by

$$econs_{ij} = econs\_rate_j \times exec(T_k) \tag{18}$$

The total energy consumption is calculated by

$$E(X) = \sum_{i=1}^{k} \sum_{j=1}^{n} econs_{ij} \tag{19}$$

Therefore, the objective function is defined as

$$\mathbf{F_2(Y)} = \text{Minimize } \{E(X)\} \tag{20}$$

(iii) Cost of the datacenter

The cost of the datacenter is calculated using the equation ()

$$C(X) = c \times E(X) \tag{21}$$

Where c is the cost of 1kW power.

Therefore, the objective function for cost is defined as

$$\mathbf{F_3(Y)} = \text{Minimize } \{C(X)\} \tag{22}$$

The objective function for the load balancing model are subject to

$$\sum_{j=1}^{m} \chi_{ij} = 1 \;\; (\mathrm{t}_i \in T, v_j \in V) \tag{23}$$

$$\sum_{i=1}^{k} T_i \le \mathrm{dl}_i \; (\mathrm{t}_i \in T,\; \mathrm{v}_j \in V) \tag{24}$$

$$\sqrt{\frac{1}{k}\sum_{i=1}^{k}(PT_{(VM)_j} - PT_{(VM)})^2} \le Th_{upper} \tag{25}$$

Constraint (23) represents that only one task should be allocated to $v_{j,}$ constraint (24) denotes that the execution of each task should be less that the deadline, constraint (25) reveal that the standard deviation of load should be less that the upper threshold value Th_{upper}.

Response time:

Response time is the time taken from the task enters the system and the time the task being scheduled. Response time is calculated as follows

$$\mathrm{Re}\, s_{time} = ft_i - ar_i \tag{26}$$

Degree of imbalance:

$$Deg\_imb = Max(T_k) – Min(T_k)/Avg(T_k) \tag{27}$$

Where $Max(T_k)$ is the maximum number of task, $Min(T_k)$ is the minimum number of task and $Avg(T_k)$ is the average of task (T_k) [22].

FOA-SA-LB Algorithm:

In this paper , a hybrid Fruitfly optimization technique is used to balance the load among virtual machines in cloud datacenter. The low-optimization accuracy and easy to fall into local optimum are the drawbacks in fruit fly optimization. The foremost motive for developing the FOA-SA-LB is to overcome the defects of the original Fruitfly optimization algorithm. The procedure of the proposed approach consist of two stages. FOA is employed first stage, where each swarm of flies moves in different directions to follow a uniform distribution. The second stage integrates simulated annealing to update the current locations and solutions to force the hurdle of FOA out of premature convergence, due to its exploration and exploitation ability. The proposed approach improves the convergence rate and optimization accuracy accordingly. The Algorithm which uses FOA in Simulated Annealing is defined in Algorithm. 3.

Algorithm3: Simulated Annealing Procedure based on FOA search solution

```
INPUT: Initial Value of Temperature, Final value of Temperature, and Cooling-rate
OUTPUT: Best optimal solution
  1. Do
  2. For (All food in various location)
  3.         Initialization
  4.         Assign direction and distance to move
  5.         Evaluate smell concentration fitness value
  6.         Substitute the smell-concentration fitness value into fitness function
  7.         Find the best smell by maximal smell concentration
```

```
8.              Use vision search to fly towards the best smell value location.
9.              $\Delta f = f(S_i^*) - f(S_i)$
10.             $g = \exp\left(-\frac{\Delta f}{T_m}\right)$
11.             $if\ \Delta f \le 0$ or g > R(0,1)
12.                 $S_i \leftarrow S_i^*$
13.             Endif
14.           R(0,1) is random number generated uniformly between 0 and 1
15.       End For
16.  While stopping condition is not exceeded.
```

Procedure for FOA-SA-LB solution construction for load balancing:

Every fruitfly initiates the process of solution construction with a set of all Virtual machines V_j and the task T_k allocated randomly. The procedure first identifies whether the loads on all virtual machines are balanced based on threshold value. If any virtual machine is overloaded , the task from the overloaded virtual machine is removed. Here the removed task is termed as fruitfly. Fruitfly searches the location using smell based searching process. Each fruitfly identifies the location that is virtual machine to allocate the removed task. In other words the location of the virtual machine (food) among all eligible virtual machines to assign the removed task. The process continues till a particular datacenter reaches to the state that all the virtual machines have balanced load. For updating the neighbour swarm location and finding the best optimal solution , the simulated annealing approach is used in this work. The fitness value is determined by the multi-objective function.

Smell-based and Vision based search:

Smell-based searching process is the primary procedure, in which N fruitflies are produced for sub-population. In FOA-SA-LB approach, a list of overloaded and underloaded virtual machines which is appropriate to remove task with deadline constraint is determined. The vision based searching process is used to evaluate the best virtual machine suitable to allocate the removed task.

Load Balancing Decision:

The fruitfly searches for the food and finds the location of food based on smell search and best smell based on vision search. The SA approach is used to find the best optimum solution based on the energy and temperature. In our proposed technique, the removed task is considered as fly, which searches for the suitable virtual machine based on the multi-objective function. The basic constraints are followed such as the load of the virtual machine, after assigning the task should not to be greater than the upper threshold value to choose a suitable virtual machine for the removed task.. If there is more number of virtual machine is available, then deadline constraint is considered. The deadline of task is imperative to migrate the task from heavy loaded VM to low loaded VM. If the deadline dl_i of the removed task is high, then the virtual machine having minimum of higher deadline task is selected. If the deadline of task is medium, then the virtual machine having less number of higher and medium deadline task is selected. The virtual machine grouping is based on the current load $L_{VM(t)}$ of the virtual machine. We consider two types of group such as overloaded VM group ($findVMList_{ol}$) and underloaded VM group ($findVMList_{ul}$). The task is removed from the $findVMList_{ol}$ and allocated to the virtual machine in $findVMList_{ul}$ based on the objective function. The process of removing task from findVMList$_{ol}$ is continued, till the findVMList$_{ol}$ is NULL. This work not only focuses on load balancing, it also distillates on saving the energy consumed in the datacenter to reduce cost. The radical process of energy conservation is based on making the virtual machines to ON and OFF state, which is not in use. It identifies the applicable virtual machines in the datacenter, which is underutilized and changing the state from active to sleep. The threshold value is used to make the virtual machine to sleep and awake mode in the datacenter. If the load of the particular virtual machine is lesser than the lower threshold value Th$_{lower}$, then that virtual machine is put into sleep mode, and if the virtual machine load is greater than Th$_{upper}$, then awake the virtual machine from the sleep mode. If the load of virtual machine is NULL, then the virtual machine is removed from VMList to save energy.

Algorithm 4: FOA-SA-LB for Energy-aware -Load balancing (FOA-SA-LB):

```
Step 1:    $Ff\_S_l$ is the $l$ th Fruitfly swarm
Step 2:    $findVMList_{(ul)} \leftarrow Null$ , $findVMList_{(ol)} \leftarrow Null$
Step 3:    for each VM $v_j$ in the datacenter do
Step 4:    Calculate the capacity from equation (9) and load from equation (10)
Step 5:    if Load(VM) > max(capacity) then
               Load Balancing is not possible. Reset the load.
           Else if $L_{VM(t)}$ > $Th_{lower}$ && $L_{VM(t)}$<$Th_{upper}$ , then
               Balanced load for all virtual machines. Exit.
Step 6:    Endif
Step 7:    For each $l$ in number of Fruitfly swarms // Smell-based search
Step 8:      Generate S Fruitflies $Ff_{l\,p}$ (p=1,2,……..S) on $l$ Fruitfly swarm
Step 9:      For each $v_j$ inVM do
Step 10:         If $L_{VM(t)}$ > $Th_{upper}$ , then
                     $findVMList_{(ol)} \leftarrow V_j$
                 Else $findVMList_{(ul)} \leftarrow V_j$
Step 11:         Endif
Step 12:     Endfor
Step 13:   Endfor
Step 14:   For each $l$ in number of Fruitfly swarms do //Vision based search
             For each in S do
               Evaluate the fruitflies generated $Ff_{lp}$
               Generate new solution $s_o$ in SA based on neighbour swarms using Algorithm3.
             Endfor
Step 15:   Endfor
Step 16:   For each $v_j$ in VM do
Step 17:     If $L_{VM(t)}$ !=$\phi$ ,then
Step 18:       Sort all VM in ascending order.
Step 19:       Sort all task based on the deadline ($dl_i$)
Step 20:       For each $T_k$ in $FindVMList_{(ol)}$ do
                 Find BestVM in $findVMList_{(ul)}$ such that $L_{VM(t)}$ +$ln_i$ >=$Th_{lower}$ && $L_{VM(t)}$ +$ln_i$<$Th_{upper}$
                 [best Fruitfly swarm location (sub-population) – vision]
Step 21:       Endfor
Step 22:       Migrate(T,BestVM)
Step 23:     Else Move $V_j$ to sleep mode.
             Endif
            Update Tasklist, VMlist and $L_{VM(t)}$
Step 24:    Endfor
Step 25:    If $FindVMList_{(ol)} != \phi$ and $findVMList_{(ul)} == \phi$ , then
Step 26:      Awake virtual machine in sleep mode.
Step 27:    Endif
Step 28:    If If $FindVMList_{(ol)} == \phi$ and $findVMList_{(ul)} != \phi$ , then
              VMlist.remove($V_j$)
              Update VMlist and $findVMList_{(ul)}$
Step 29:    Endif
Step 30:   Endfor
Step 31:   Update Best fruitfly location based on SA in equation (7) using Algorithm 3.
Step 32:  Endfor
```

5. Experiment Results:

In this section we represent two different experiment, the first we show the comparison of FOA with Hybrid FOA (FOA-SA-LB). The experiment uses four well-known benchmark optimization functions which is shown in Table 1. to compare the original FOA with the FOA-SA-LB [48]. We compare the optimization precision and the convergence speed using fixed iterations and population size. We use fly size as 30 and the number of iterations as 100 in our experiment. The experimental results is illustrated in Table 2. The order of magnitude is better for FOA-SA-LB compared to FOA for all the four functions. FOA-SA-LB has optimization effect 2 to 3 times better than FOA. From table 1 , we can see that the mean value of FOA-SA-LB is lower than FOA, closer to the minimum theoretical value and the iteration times of FOA-SA-LB is faster than FOA. So from the above analysis, SOA-FA has the better stability compared to FOA.

Table 1. BenchMark Functions

Function Name	Equation	Boundary	Optimum	Peak(s)
Sphere f1	$f1(x)=\sum_{i=1}^{n} x_i^2$	[-100,100]	0	Single
Rastrigin f2	$f2(x)=\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$	[-5.12,5.12]	0	multiple
Griewank f3	$f3(x)=1/4000\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600,600]	0	multiple
Ackley f4	$f4(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right)+20+e$	[-30,30]	0	multiple

Table 2. Test Results

Optimization Function	Method	Worst-case	Optimal-value	Average-value	Variance
$f_1(x)$	FOA	9.294E-005	7.342E-005	8.258E-005	3.311E-011
	FOA-SA-LB	1.201E-005	7.268E-006	8.402E-006	1.467E-012
$f_2(x)$	FOA	4.319E-005	3.589E-005	4.011E-005	9.336E-012
	FOA-SA-LB	4.343E-006	3.551E-006	3.848E-006	8.768E-014
$f_3(x)$	FOA	4.311E-002	3.779E-002	4.019E-002	2.421E-006
	FOA-SA-LB	1.339E-002	1.089E-002	1.399E-002	5.179E-007
$f_4(x)$	FOA	1.852E-002	1.389E-002	1.613E-002	1.193E-006
	FOA-SA-LB	3.126E-003	1.275E-003	1.629E-003	7.512E-008

The second experiment deals the efficiency of FOA-SA-LB for load balancing in cloud environment for EHR applications using cloudsim tool. Cloud computing scenarios are modelled and simulated using a powerful tool called cloud sim.[49]. We here evaluated the performance of the proposed approach based on the simulation results. The proposed approach compares the FOA-SA-LB load balancing algorithm with the existing intelligent optimization methods like PSO, HBB-LB and EFOA-LB. The simulation of the proposed algorithm is based on the multi-objective functions defined along the constraints specified. The makespan comparison is made before and after load balancing, which is depicted in figure. 2. The proposed approach not only focuses on load balancing issues, but reducing the energy and cost of the datacenter is also emphasized. Based on the Threshold value, the workloads assigned to each virtual machine are balanced in the datacenter along with the sleeping strategy to make the virtual machine in sleep state, if there is no load assigned to the virtual machine. If the load is greater than the lower threshold value, then that particular virtual machine is used to assign the task which is removed from the overloaded virtual machine. Thus the sleeping strategy incorporated in the proposed approach reduces the energy and cost subsequently. Figure. 3 depict the comparison of makespan with other existing optimization algorithms like HBB-LB, PSO and EFOA-LB. The x-axis in the figure illustrates the number of tasks and the y-axis illustrates the overall execution time based on deadline constraint (makespan). The result shows that our proposed approach FOA-SA-LB has minimum makespan time compared to the existing algorithms. Figure.4 indicates the total energy consumption used according to the number of virtual machines before and after FOA-SA-LB. The result shows that our proposed approach drastically reduces the energy consumption. Table 3 shows the simulation comparison result for energy consumption based on FOA-SA-LB, HBB-LB, PSO and EFOA-LB. The result obtained shows that FOA-SA-LB approach is having less energy consumption compared to other load balancing techniques which is depicted in figure 5. The x-axis in figure. 5 illustrates the number of tasks and the y-axis illustrates the energy consumed using kWh.

Table 3. Comparison of energy consumption of various load balancing algorithms

No of tasks	FOA-SA-LB(kWh)	EFOA-LB(kWh)	HBB-LB(kWh)	PSO(kWh)
100	1.13	1.17	2.44	1.78
200	1.20	1.28	3.66	3.12
300	2.01	2.21	4.44	4.34
400	3.08	3.20	5.5	7.21

Figure 6 indicates the degree-of-imbalance of virtual machines before and after FOA-SA-LB using equation (27).

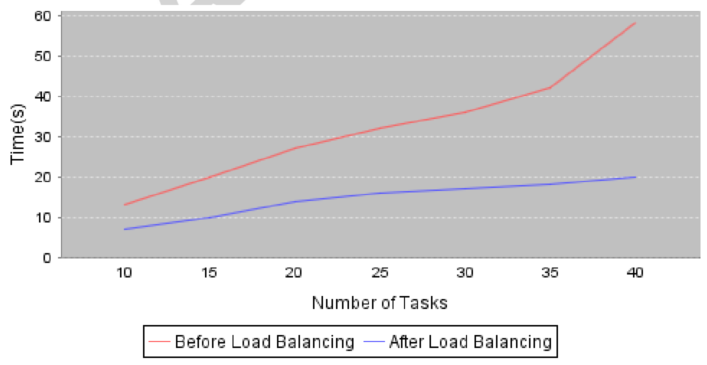


Figure.2 Comparison of makespan before and after FOA-SA-LB

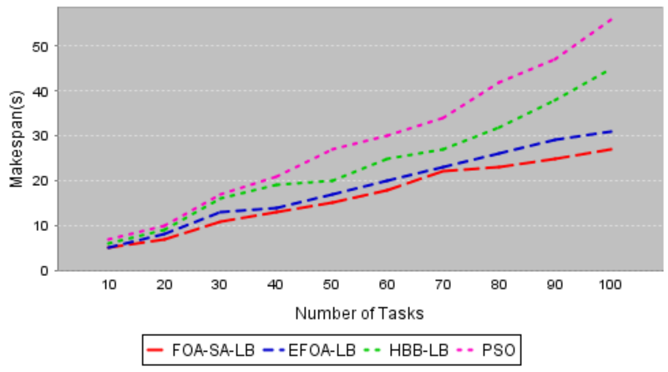


Figure.3 Comparison of makespan for FOA-SA-LB, EFOA-LB,HBB-LB and PSO Algorithms

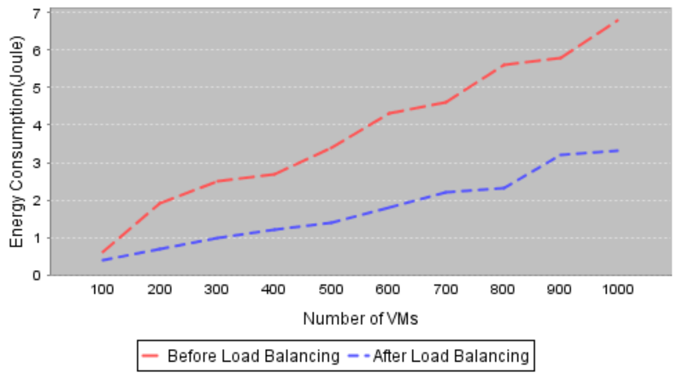


Figure.4 Total Energy Consumption before and after load balancing.

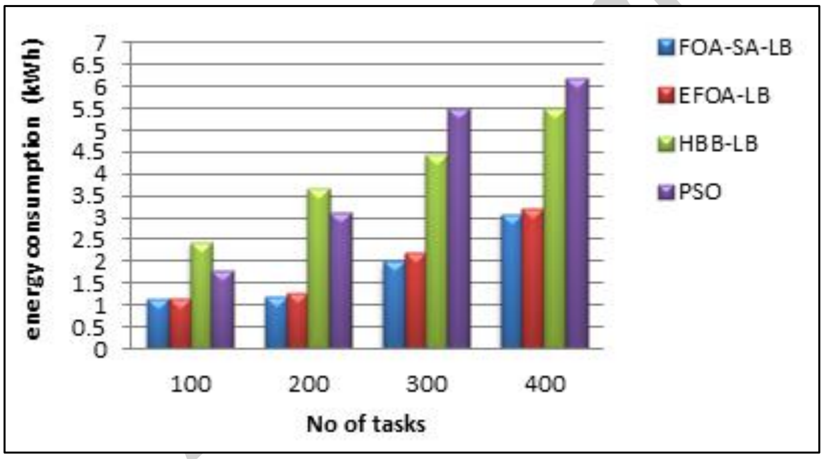
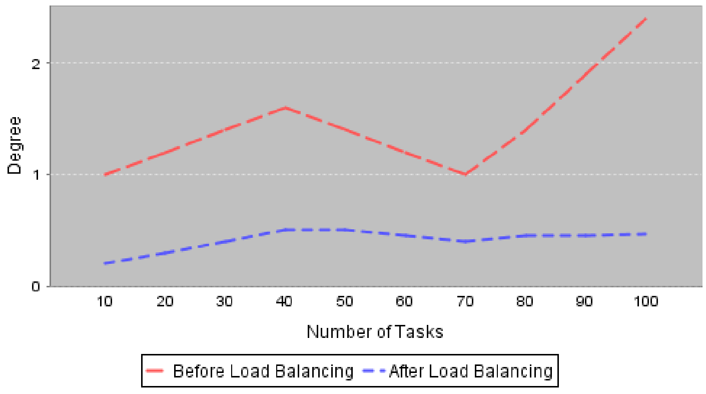


Figure 5 . Comparison of energy consumption for FOA-SA-LB, EFOA-LB, HBB-LB and PSO.



6 Conclusion and Future work:

In this paper , we propose a multi-objective hybrid Fruitfly optimization technique based on SA for load balancing in cloud computing environments. We have carried out two experiments. The first experiments compares the original FOA with the Hybrid FOA approach and the second part simulates the FOA-SA-LB and compares the result with the existing metaheuristic approaches such as PSO, HBB-LB and EFOA-LB. The proposed FOA-SA-LB overcomes the original FOA drawbacks and attain optimum solution for balancing the workloads among virtual machines effectively. The FOA-SA-LB uses simulated annealing approach which is best suit for local search ability to increase the convergence rate and performance of the datacenter. FOA-SA-LB outperforms FOA with respect to convergence rate. The proposed work balances the workload using dynamic threshold values and reduces the makespan ,energy consumption and cost of the datacenter. The FOA-SA-LB uses the sleeping strategy to reduce the energy consumption. The simulation results expose that our proposed method achieves greater performance compared to other existing methods like HBB-LB, EFOA-LB and PSO. In future , we intend to extend this work for load balancing workflows with QOS factors like network traffic information in cloud computing. The network traffic is one of the important research issue in cloud computing, which consumes more energy and increases the cost of the data center. We will also try to improve the utilization of resources in mobile cloud computing. The suitability of the proposed algorithm will also be tested for fog and mobile edge computing.

References

Marston S, Li Z, Bandyopadhyay S, Zhang J, Ghalsasi A. Cloud computing—The business perspective. Decision support systems. 2011 Apr 30;51(1):176-89.

Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation computer systems. 2009 Jun 30;25(6):599-616.

Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M. A view of cloud computing. Communications of the ACM. 2010 Apr 1;53(4):50-8.

Li M, Yu S, Zheng Y, Ren K, Lou W. Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption. IEEE transactions on parallel and distributed systems. 2013 Jan;24(1):131-43.

Sultan N. Making use of cloud computing for healthcare provision: Opportunities and challenges. International Journal of Information Management. 2014 Apr 30;34(2):177-84.

Gholami MF, Daneshgar F, Low G, Beydoun G. Cloud migration process—A survey, evaluation framework, and open challenges. Journal of Systems and Software. 2016 Oct 31;120:31-69.

Beloglazov A, Buyya R. Managing overloaded hosts for dynamic consolidation of virtual machines in cloud datacenters under quality of service constraints. IEEE Transactions on Parallel and Distributed Systems. 2013 Jul;24(7):1366-79.

Kaur T, Chana I. Energy aware scheduling of deadline-constrained tasks in cloud computing. Cluster Computing. 2016:1-20.
Mondal B, Dasgupta K, Dutta P. Load balancing in cloud computing using stochastic hill climbing-a soft computing approach. Procedia Technology. 2012 Dec 31;4:783-9.
Xu G, Pang J, Fu X. A load balancing model based on cloud partitioning for the public cloud. Tsinghua Science and Technology. 2013 Feb;18(1):34-9.
Beloglazov A, Abawajy J, Buyya R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future generation computer systems. 2012 May 31;28(5):755-68.
Črepinšek M, Liu SH, Mernik L. A note on teaching–learning-based optimization algorithm. Information Sciences. 2012 Dec 1;212:79-93.
Mezmaz M, Melab N, Kessaci Y, Lee YC, Talbi EG, Zomaya AY, Tuyttens D. A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. Journal of Parallel and Distributed Computing. 2011 Nov 30;71(11):1497-508.
Goyal SK, Singh M. Adaptive and dynamic load balancing in grid using ant colony optimization. International Journal of Engineering and Technology. 2012 Aug;4(9):167.
Ajit M, Vidya G. VM level load balancing in cloud environment. InComputing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on 2013 Jul 4 (pp. 1-5). IEEE.
Moradi M, Dezfuli MA, Safavi MH. A new time optimizing probabilistic load balancing algorithm in grid computing. InComputer Engineering and Technology (ICCET), 2010 2nd International Conference on 2010 Apr 16 (Vol. 1, pp. V1-232). IEEE.
Abdullahi M, Ngadi MA. Symbiotic Organism Search optimization based task scheduling in cloud computing environment. Future Generation Computer Systems. 2016 Mar 31;56:640-50.
Maguluri ST, Srikant R, Ying L. Stochastic models of load balancing and scheduling in cloud computing clusters. InINFOCOM, 2012 Proceedings IEEE 2012 Mar 25 (pp. 702-710). IEEE.
Florence AP, Shanthi V. A load balancing model using firefly algorithm in cloud computing. Journal of Computer Science. 2014 Jul 1;10(7):1156.
Alakeel AM. A guide to dynamic load balancing in distributed computer systems. International Journal of Computer Science and Information Security. 2010 Jun;10(6):153-60.
Ramezani F, Lu J, Hussain FK. Task-based system load balancing in cloud computing using particle swarm optimization. International Journal of Parallel Programming. 2014 Oct 1;42(5):739-54.
Krishna PV. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Applied Soft Computing. 2013 May 31;13(5):2292-303.
Zhao C, Zhang S, Liu Q, Xie J, Hu J. Independent tasks scheduling based on genetic algorithm in cloud computing. In2009 5th International Conference on Wireless Communications, Networking and Mobile Computing 2009 Sep 24 (pp. 1-4). IEEE.
Balusamy B, Sridhar J, Dhamodaran D, Krishna PV. Bio-inspired algorithms for cloud computing: a review. International Journal of Innovative Computing and Applications. 2015;6(3-4):181-202.
Shi XH, Liang YC, Lee HP, Lu C, Wang LM. An improved GA and a novel PSO-GA-based hybrid algorithm. Information Processing Letters. 2005 Mar 16;93(5):255-61.
Silva CA, Sousa JM, Runkler TA. Rescheduling and optimization of logistic processes using GA and ACO. Engineering Applications of Artificial Intelligence. 2008 Apr 30;21(3):343-52.
Bahga A, Madisetti VK. A cloud-based approach for interoperable electronic health records (EHRs). IEEE Journal of Biomedical and Health Informatics. 2013 Sep;17(5):894-906.
Schweitzer EJ. Reconciliation of the cloud computing model with US federal electronic health record regulations. Journal of the American Medical Informatics Association. 2012 Mar 1;19(2):161-5.
Lin CW, Abdul SS, Clinciu DL, Scholl J, Jin X, Lu H, Chen SS, Iqbal U, Heineck MJ, Li YC. Empowering village doctors and enhancing rural healthcare using cloud computing in a rural area of mainland China. Computer methods and programs in biomedicine. 2014 Feb 28;113(2):585-92.

Miah SJ, Hasan J, Gammack JG. On-Cloud Healthcare Clinic: An e-health consultancy approach for remote communities in a developing country. Telematics and Informatics. 2017 Feb 28;34(1):311-22.
Rolim CO, Koch FL, Westphall CB, Werner J, Fracalossi A, Salvador GS. A cloud computing solution for patient's data collection in health care institutions. IneHealth, Telemedicine, and Social Medicine, 2010. ETELEMED'10. Second International Conference on 2010 Feb 10 (pp. 95-99). IEEE.
Beloglazov A, Buyya R. Energy efficient resource management in virtualized cloud data centers. InProceedings of the 2010 10th IEEE/ACM international conference on cluster, cloud and grid computing 2010 May 17 (pp. 826-831). IEEE Computer Society.
Lee YC, Zomaya AY. Energy efficient utilization of resources in cloud computing systems. The Journal of Supercomputing. 2012 May 1;60(2):268-80.
Wang X, Du Z, Chen Y. An adaptive model-free resource and power management approach for multi-tier cloud environments. Journal of Systems and Software. 2012 May 31;85(5):1135-46.
Katsaros G, Subirats J, Fitó JO, Guitart J, Gilet P, Espling D. A service framework for energy-aware monitoring and VM management in Clouds. Future Generation Computer Systems. 2013 Oct 31;29(8):2077-91.
Pan WT. A new evolutionary computation approach: fruit fly optimization algorithm. InProceedings of the Conference on Digital Technology and Innovation Management 2011.
Lin SM. Analysis of service satisfaction in web auction logistics service using a combination of fruit fly optimization algorithm and general regression neural network. Neural Computing and Applications. 2013 Mar 1;22(3-4):783-91.
Wang L, Zheng XL, Wang SY. A novel binary fruit fly optimization algorithm for solving the multidimensional knapsack problem. Knowledge-Based Systems. 2013 Aug 31;48:17-23.
Li HZ, Guo S, Li CJ, Sun JQ. A hybrid annual power load forecasting model based on generalized regression neural network with fruit fly optimization algorithm. Knowledge-Based Systems. 2013 Jan 31;37:378-87.
Zhang P, Wang L. Grouped Fruit-Fly Optimization Algorithm for the No-Wait Lot Streaming Flow Shop Scheduling. InInternational Conference on Intelligent Computing 2014 Aug 3 (pp. 664-674). Springer International Publishing.
Pan WT. A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowledge-Based Systems. 2012 Feb 29;26:69-74.
Han J, Wang P, Yang X. Tuning of PID controller based on fruit fly optimization algorithm. In2012 IEEE International Conference on Mechatronics and Automation 2012 Aug 5 (pp. 409-413). IEEE.
Choubey NS. Fruit Fly Optimization Algorithm for Travelling Salesperson Problem. International Journal of Computer Applications (0975–8887). 2014 Dec;107(18):22-7.
Li JQ, Pan QK, Mao K, Suganthan PN. Solving the steelmaking casting problem using an effective fruit fly optimisation algorithm. Knowledge-Based Systems. 2014 Dec 31;72:28-36.
Mousavi SM, Alikar N, Niaki ST, Bahreininejad A. Optimizing a location allocation-inventory problem in a two-echelon supply chain network: A modified fruit fly optimization algorithm. Computers & Industrial Engineering. 2015 Sep 30;87:543-60.
Zhang Y, Cui G, Wang Y, Guo X, Zhao S. An optimization algorithm for service composition based on an improved FOA. Tsinghua Science and Technology. 2015 Feb;20(1):90-9.
Zhu X, Yang LT, Chen H, Wang J, Yin S, Liu X. Real-time tasks oriented energy-aware scheduling in virtualized clouds. IEEE Transactions on Cloud Computing. 2014 Apr;2(2):168-80.
Abdullahi M, Ngadi MA. Hybrid symbiotic organisms search optimization algorithm for scheduling of tasks on cloud computing environment. PloS one. 2016 Jun 27;11(6):e0158229.
Calheiros RN, Ranjan R, Beloglazov A, De Rose CA, Buyya R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience. 2011 Jan 1;41(1):23-50.