IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 3, Ver. VI (May – Jun. 2015), PP 96-106
www.iosrjournals.org

Task Scheduling using Hybrid Algorithm in Cloud Computing Environments

Ali Al-maamari[1], Fatma A. Omara[2]
[1](Department of Computer Science, Cairo University, Egypt)
[2](Professor, Department of Computer Science, Faculty of Computers & Information, Cairo University, Egypt)

Abstract: *The cloud computing is considered the latest network infrastructure that supports the decentralization of computing. The main features of the Cloud are the possibilities for building applications and providing various services to the end user by virtualization on the internet. One of the main challenges in the field of the cloud computing is the task scheduling problem. Task scheduling problem concerns about the dynamic distribution of the tasks over the Cloud resources to achieve the best results. Many of the algorithms have been existed to resolve the task scheduling problem such as a Particle Swarm Optimization algorithm (PSO). The PSO is a simple parallel algorithm that can be applied in different ways to resolve the task scheduling problems. In this paper, a task scheduling algorithm has been proposed to the independent task over the Cloud Computing. The proposed algorithm is considered an amalgamation of the PSO algorithm and the Cuckoo search (CS) algorithm; called PSOCS. To evaluate the proposed algorithm, the cloudsim simulator has been used. The experimental results show the reduction of the makespan and increase the utilization ratio of the proposed PSOCS algorithm compared with PSO algorithms and Random Allocation (RA).*

Keywords: *Cloud Computing, PSO algorithm, CS algorithm, Task Scheduling, simulation*

I. Introduction

Cloud Computing has become the fast spread in the field of computing and industry in the last few years. As part of the services offered by the Cloud Computing is the new possibilities for building applications and providing various services to the end user by virtualization on the internet.

On the other hand, many challenges need to be solved in the cloud computing. The two important issues used in this paper are load balancing and Task scheduling. Load Balancing; is the process of distributing the load statically or dynamically, among various nodes of a distributed system to improve both resource utilization and job response time while avoiding a situation where some of the nodes are heavily loaded and other nodes are idle or doing very little work [1]. According to task scheduling; an appropriate number of tasks would be scheduled over the virtual machines. Task scheduling is the most important issue in the cloud computing, because the user will have to pay for resource using on the basis of time. The goal of task scheduling is to spread the load evenly between the system by maximizing utilization and reducing task execution Time [2].

Moreover, there are many load balanced algorithms have been proposed in different environments. However, with the cloud environment, some additional challenges have been existed and must be addressed. In the Cloud Computing, efficient allocation of tasks, and the contract of such a request will be processed as efficiently as possible[3].

The work in this paper, an algorithm based on particle swarm optimization algorithm and cuckoo search algorithm has been presented to optimize the task scheduling problem in the Cloud environment to minimize the completion time and increase the utilization ratio using open source Cloudsim3.0.3 simulator. The cloudsim is a cloud computing simulation software developed by Gridbus project team and the grid Laboratory of the University of Melbourne in Australia. The Cloudsim can run on Linux and Windows systems[4].

Kennedy and Eberhart[5] have presented a as a self-adaptive global search based optimization technique. The algorithm is similar to other population based algorithms as Genetic algorithms (GA) without direct re-combination of individuals of the population. Instead, it relies on the social behavior of the particles. In every generation, each particle adjusts its trajectory based on its best position (local best) and the position of the best particle (global best) of the entire population. These concepts increase the stochastic nature of the particle and converge quickly to a global minimum with a reasonable good solution. Particle Swarm Optimization has become popular because of its simplicity and its effectiveness in a wide range of application. Some of the applications that have used PSO to solve NP-Hard problems like Scheduling problem[6], task allocation problem[7], the data mining problem[8], environmental engineering problem[9]. On the other hand, Yang, X.-S. And S. Deb.[10] Have introduced Cuckoo Search algorithm, which is considered a meta-heuristic optimization algorithm based on the behavior of the cuckoo bird. There are some similarities between CS algorithm and hill-climbing algorithms in respect with some large scale randomization. But, these two algorithms are in essence very different. Firstly, CS algorithm is population-based algorithm in a way similar to GA algorithm and PSO algorithm, but it uses some kind of selection similar to that used in harmony search. Secondly, the randomization is more efficient as the step length is heavy-tailed, and any large step is possible. Finally, the number of tuning parameters is less than in GA and PSO, and thus CS can be much easier adapted to a wider class of optimization problems.

The proposed task scheduling is considered an amalgamation of tow algorithms PSO and CS.

The rest of this paper is organized as follows;Section 2 provides related work. Section 3 describes the main components task scheduling. Sections 4 describe the intelligent PSO algorithm. Sections 5 describe the intelligent CS algorithms, and sections 6 describe our proposed algorithm in detail. Section 7 presents the comparative study between our proposed algorithms with PSO and RA algorithms. Finally presents the conclusions and future research directions.

II. Related Works

Task scheduling has been a significant research topic whose objective is to ensure that every computing resource is distributed efficiently and fairly and in the end improves resource utility. In traditional computing environments of distributed computing, parallel computing and grid computing, a set of static and dynamic and mixed scheduling strategies have proposed.

Suraj Pandey et al. [11]have proposed a particle swarm optimization (PSO) based heuristic to schedule applications to cloud resources that optimizes computation cost and data transmission cost. The PSO based mapping algorithm has much lower cost as compared to another algorithm called (Best Resource Selection) based mapping. The PSO algorithm is used for a workflow application through the difference of its computation and communication costs. The results show that PSO can achieve cost savings and good distribution of the workload onto resources.

B. Radojevic[12] has considered to fix the load balancing algorithms and overcome the defects round robin algorithm, Which is the very famous algorithm and it is working on the basis of a conversion session in the application layer. The main feature of the algorithm is to improve communication time between the customer and the node in the cloud computing. If the connection time has exceeded the threshold standard, then the relation between the customer and the node will be finished, the task is to provide some other node by using round robin rules. The approach sends a request to a node with less number of communications.

Lee[13], has proposed a dynamic load balancing algorithm on the basis of an existing algorithm called WLS (weighted least connection).According to this algorithm is that assigning tasks to node according to the number of connections that exist for this node. It is comparing between a set of connections from each node in the cloud and the task assigned to the node with the lowest number of connections. Nonetheless, weighted least connection would not take into account the capabilities of each node, such as processing speed and storage and capacity and bandwidth.

Ren[14], An improved has been in reduced to the algorithm of the WLC [13], with taking into account the time series and trials, called Exponential Smooth Forecast based on Weighted Least Connection (ESWLC).The ESWLC steps are; 1) builds the conclusion of the assignment of a specific task to the node after having a number of tasks assigned to that node and identify node capabilities. 2) Builds a decision on the basis of experience of the node’s memory, CUP memory, number of connections and the amount of disk space currently being used. 3) Then predicts which node is to be chosen on the basis on exponential smoothing.

Hai Zhong1 et al.[15]have proposed algorithm to present an optimum, scheduling algorithm to achieve the optimization for cloud scheduling. In this algorithm an Improved Genetic Algorithm (IGA) is used for the automatic schedule policy. It is used to increment the utilization ratio of resources and speed.

Xin Lu, ZilongGu.[16], Dong, Wang, D. [17] and Song, X., L. [18] are proposed a Load balancing task scheduler balance the entire system load while trying to minimizing the makespan of a given task set. They are used two different load balancing scheduling algorithms based on the solution of ant-colony optimization (ACO) technique, which aims to minimize the completion time based on pheromone.

Pooranian et al.[19], have proposed a task-scheduling technique for grid computing based on a merge PSO with the gravitational emulation local search (GELS) algorithm, which aims to minimize makespan and the number of tasks that fail to meet their deadlines.

III. The Principle And Components Algorithm

Figure 1 shows an overview of the algorithm model, the model consists of three modules, the first module is the scheduler system, which consists of a scheduler, and the second module is the application which represents the set of the cloudlets and a set of the virtual machines. The third module is the Mapping Algorithms (MA) which estimate the expected time for each cloudlet to allocate on each virtual machine, and it is assumed that these values are available to the scheduler. The cloudlets exepected time have been stored in an m × n matrix, where m is the number of virtual machines and n is the number of cloudlets. Obviously, n/m will generally be greater than 1, with more cloudlets than virtual machines, so that some machines will need to be assigned multiple cloudlets. The Estimated Running Time (ERT) is defined as the time the of executing task j on resource r [20]. Each column i of the expected running times (ERT) matrix contains estimates of the expected running time (ERT) of each cloudlet j on machine i.

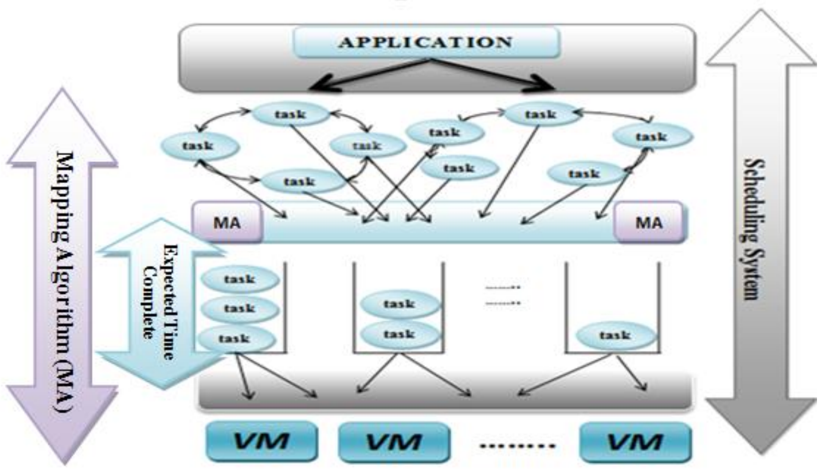


Figure 1: The Component of PSOCS algorithm

The main target of allocating on a virtual machine is to reduce Makespan, Makespan can be defined as the overall task completion time. We denote completion time of task T_i on VM_j as CT_{ij}. Hence, the makespan is defined as the following equation[21]:

$$\text{Makespan} = CT_{max}\,[i,j] \mid i \in T, \quad i = 1,2,\dots n \text{ and } j \in VM, j = 1,2,\dots m \qquad (1)$$

Where $CT_{max}\,[i,j]$ is the maximum which can be defined as the time for completing cloudlet i on a virtual machinej.

3.1 Mathematical Modules

Let $VM = VM_1, VM_2, \dots. VM_m$ be the number of m virtual machines that must be process n tasks represented by the group $T = T_1, T_2, \dots T_n$. Each virtual machines are parallel and independent, the schedule independent tasks to theseVM_s, the Processing of that task on a virtual machine cannot interrupt (i.e) Non-preemption, we denote end time of a task T_i byCT_{ij}. Our aim is to reduce the Makespan which can be denoted asCT_{max}, the run time of each task for each virtual machine must be calculated for the purpose of scheduling, If the processing speed of virtual machine VM_j is PS_j, then the processing time for cloudlet P_i can be calculated by equation.(2)[22]:

$$P_{ij} = C_i / PS_j \qquad (2)$$

Where P_{ij} the processing is time of task P_i by virtual machineVM_j and C_i is the computational complexity of the task P_i[22]. The values obtained from equation (2), are stored in the runtime matrix.
Th processing time of each task in the virtual machine can be calculated by equation (3)

$$P_j = \sum_{i=1}^{n} P_{ij} = 1, \dots, m \qquad (3)$$

By minimizingCT_{max}, Eqs. (1), (2) and (3),we can calculated that (4).

$$\sum_{i=1}^{n} P_{ij} \leq CT_{max} \qquad (4)$$

By considering of the load balancing, the tasks will be transferred from one VM to other in order to reduce CT_{max} ,as well as, response time. The processing time of a task varies from one VM to another based on the virtual machines speed. In case of transferring, the completion time of a task may vary because of load balancing, optimally.

$$CT_{max} = \{\max_{i=1}^{n} CTi\,, \max_{j=1}^{n} \sum_{i=1}^{n} Pij\ \} \qquad (5)$$

The main target our proposed algorithm is that tasks should be allocated virtual machine and minimized makespan and maximizing the resource utilization. Our proposed tasks scheduling algorithm is considered an extension of an existing PSO algorithm and the Cuckoo Search algorithm merged with the concept of Cuckoo Search.

IV. The Particle Swarm Optimization Algorithm

In this section, the algorithm for task scheduling based on PSO will be explained.

According to the PSO system, the population is individuals of particles. Each particle forms a candidate solution to problem space. Particles are initialized randomly with a fitness value. This value is calculated by a fitness function to be best solution in each generation. Each particle knows best position pbest and the best position so far among the complete set of particles gbest.The pbest of a particle is the best result that is calculated by the fitness value, while gbest is the best particle in terms of fitness for all population. The evaluation is implemented to number of repetitions[23, 24]. When iteration optimization process is bag an, each of the iteration of the velocity and the positions of all particles are updated according to the equation. (6, 7)[23].

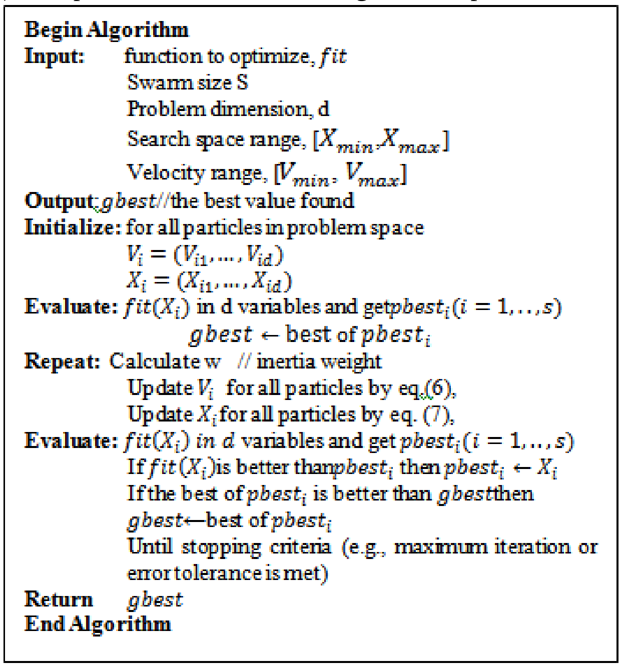
$$\mathrm{v_i(k+1) = w * v_i(k) + c_1 * r_1 * \big(pbest_i(k) - x_i(k)\big) + c_2 * r_2 * \big(gbest_i(k) - x_i(k)\big)} \qquad (6)$$

$$\mathrm{x_i(k+1) = k_i(k) + v_i(k+1)} \qquad (7)$$

Where,$\mathrm{v_i(k)}$ and$\mathrm{x_i(k)}$are the velocity of particle$\mathbf{i}$ at iteration k, $\mathrm{r_1}$and $\mathrm{r_2}$are random numbers with a regular distribution in the period between 0 and 1.$\mathrm{c_1}$and$\mathrm{c_2}$ are learning factors called the cognition and the social parameter and w is inertia weight can be dynamically different through applying an annealing scheme for the w setting of the PSO. w is decreased and contributed to convergence. In general the inertia weight w is set according to equation. (8)[25].

$$\mathrm{w = w_{max} - \frac{w_{max} - w_{min}}{itra_{max}} \times itra} \qquad (8)$$

Where$\mathrm{w_{max}}$ is the initial value, $\mathrm{w_{min}}$ is the final value of the weight coefficient, $\mathrm{itra_{max}}$ is the largest possible number of repetitions. The performance of the PSO algorithm is improved heavily by changing inertia using the updated equation (6) and (7). The pseudo code of the PSO algorithm is presented in figure (2)[23].



```
Begin Algorithm
Input:      function to optimize, $fit$
            Swarm size S
            Problem dimension, d
            Search space range, $[X_{min}, X_{max}]$
            Velocity range, $[V_{min}, V_{max}]$
Output: $gbest$//the best value found
Initialize: for all particles in problem space
            $V_i = (V_{i1}, \dots, V_{id})$
            $X_i = (X_{i1}, \dots, X_{id})$
Evaluate: $fit(X_i)$ in d variables and get$pbest_i (i = 1,..,s)$
                $gbest \leftarrow$ best of $pbest_i$
Repeat: Calculate w   // inertia weight
            Update $V_i$ for all particles by eq.(6),
            Update $X_i$ for all particles by eq. (7),
Evaluate: $fit(X_i)$ $in$ $d$ variables and get $pbest_i (i = 1,..,s)$
            If $fit(X_i)$is better than$pbest_i$ then $pbest_i \leftarrow X_i$
            If the best of $pbest_i$ is better than $gbest$then
            $gbest \leftarrow$best of $pbest_i$
            Until stopping criteria (e.g., maximum iteration or
            error tolerance is met)
Return      $gbest$
End Algorithm
```

Figure2: The PSO algorithm Pseudo code

V. Cuckoo Search Algorithm

The cuckoo is considered special bird because it has many of the characteristics that distinguish it from other birds.

It is characterized by aggressive breeding strategy. Cuckoo lays their eggs in the nest of another species, sometimes the cuckoo's egg in the host nest is discovered may lead to the removal of other eggs or abandons the nest and builds their own brood somewhere else in [26].

5.1 Cuckoo Behavior

Some cuckoo species have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry incolor and pattern of the eggs of a few chosen host species. This reduces the probability of eggs being abandoned and increases their reproductively.Parasitic cuckoos often choose a nest where the host bird just laid its own eggs. In general, the cuckoo eggs hatch slightly earlier than their host eggs.Once the first cuckoo chick is hatched, the first instinct action will be taken is to evict the host eggs by blindly propelling the eggs out of the nest, which increases the cuckoo chick's share of food provided by its host bird[26].Cuckoo characteristics could be described, as a model for good behavior other animals have extensive use in computing Intelligence Systems[27].

5.2 Cuckoo Rules & Parameters

To simplify the principle of the novel Cuckoo Search (CS) algorithm, three exemplary rules can be used [26].

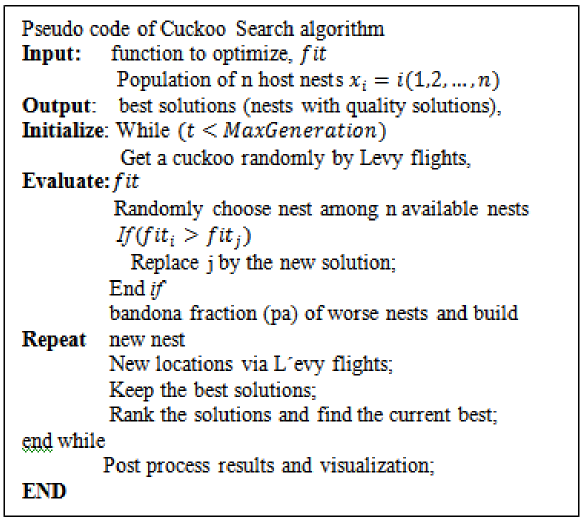
1. Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest.
2. The best nests with high quality of eggs (solutions) will carry over to the next generations.
3. The number of available host nests is fixed, and a host can discover an alien egg with a probability **pa ∈ [0, 1].** In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location.

As a further approximation, this last assumption can be approximated by a fraction **pa** of the **n** nests being replaced by new nests with new random solutions at new locations. For a maximization problem, the quality or fitness of a solution can simply be proportional to thevalue of its objective function. Other forms of fitness can be defined in a similar way ofthe fitness function in genetic algorithms and other evolutionarycomputation algorithms. For simplicity, the following Simple protests is used that each egg in the nest is a solution, the cuckoo egg represents the new solution, and the goal is to use a new and potentially better solution (Cuckoo) to replace worse solutions that are in the nests. Of course, this algorithm can be extended to more complex situation where each nest has multiple eggs represent a set of solutions. When generating new solutions $X^{(t+1)}$ for a cuckoo i, a Lévy flight is performed using the equation (10) [26].

$$X^{(t+1)} = X^{(t)} + \alpha \oplus \text{Lévy}(\lambda) \qquad (9)$$

Where $X^{(t)}$ represents the current location, $\alpha > 0$ is the step size, which should be related to the scales of the problem that the algorithm is trying to solve. In most cases, $\alpha = 1$, and λ €(0,3) are used. The equation(9) is in core stochastic equation for a random walk which is similarly of a markov chain who's next location (status) depends on two parameters; current location (the first term in equation. 9) and the possibility of transmission (the second term in Eq. 10).The product $\oplus$ represents entry-wise multiplication[10]. Something it is similar to entry-wise product as in PSO algorithm, but random walk through a Lévy flight is a random walk in which the step-lengths are distributed according to a heavy-tailed probability distribution. After a large number of steps, the distance from the origin of the random walk tends to be a stable distribution[28]. Here, the consecutive jumps (steps) of a cuckoo essentially form a random walk process which obeys a power-law step length distribution with a heavy tail. Taking into account basic three rules described above.

The first proposed algorithm is called the Cuckoo Search (CS).The Pseudo code for the CS algorithm is shownin Figure 3 [10]. According to the CS algorithm, an initial set of nests, which represent the solutions, are randomly generated. These solutions are then updated over multiple generations. The process of updating an individual solution is as follows; a random nest is chosen, and a new solution is generated by random-walking from this previous solution. This new solution can then replace a different randomly chosen solution if it has a fitness value better than the original. After this possible replacement of a solution, all of the nests are ranked by fitness and the worst fraction of the nests is replaced with random solutions. This combination of mechanisms allows the solutions to search locally and globally at the same time for the optimal solution[10].



```
Pseudo code of Cuckoo Search algorithm
Input:      function to optimize, $fit$
            Population of n host nests $x_i = i(1,2,\ldots,n)$
Output:     best solutions (nests with quality solutions),
Initialize: While $(t < MaxGeneration)$
              Get a cuckoo randomly by Levy flights,
Evaluate: $fit$
            Randomly choose nest among n available nests
            $If(fit_i > fit_j)$
              Replace j by the new solution;
            End if
            bandona fraction (pa) of worse nests and build
Repeat      new nest
            New locations via L´evy flights;
            Keep the best solutions;
            Rank the solutions and find the current best;
end while
          Post process results and visualization;
END
```

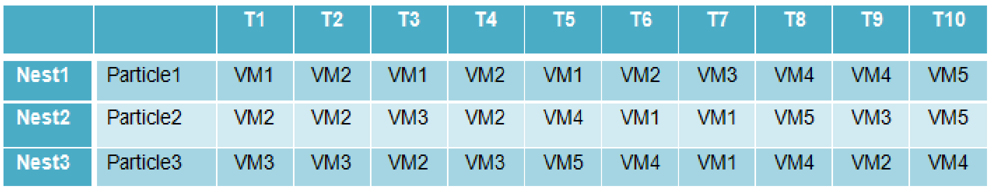
Figure2: Pseudo code for The CS algorithm

VI. The Proposed Task Scheduling ALGORITHM

There various search stages in the PSO algorithm. These stages are close to the optimum stage with their pbestandgbest values. The main drawback of the PSO algorithm is its weakness of local searches because there is a possibility of becoming trapping in a local search in the last repetition. The problem is not taken place in the global search, because the PSO algorithm is always trying to reach for the solutions that have better fitness functions in the search problem space. Therefore, the PSO algorithm is unable to recognize and avoid local optima. As a result, the PSO algorithm may become trapped in local optima and have a reduction convergence ratio in the late repetition process. Then, Cuckoo Search (CS) algorithm has been used to overcome the local optima problem by amalgamating with the PSO algorithm.

According to the proposed task scheduling algorithm, the PSO algorithm has been used as the main search algorithm, while the CS algorithm is used to improve the population. There are two reasons for using both algorithms. First, it needs an algorithm based on a population to search the entire cloud space for this problem. Second, the cloud environment is dynamic, so the scheduling algorithm must be fast enough to adapt with the natural cloud environment and must be able to converge faster than other algorithms. Moreover, although the PSO algorithm is weak for local searches, by combining the PSO algorithm with CS algorithm is considered powerful in searches addresses this problem. Because CS actions have a high computational cost for each particle in the PSO search and the cloud scheduler should execute quickly, CS is run on the global result of the last iteration of the PSO. That is, an initial solution for CS which is provided by PSO during the mix search process.

One of the most major challenges to apply CS and PSO algorithms in the task scheduling problem is that how to enter a schedule as a search solution, find appropriate maps among problem solutions, and how the CS nests with the PSO particles. In our proposed PSOCS algorithm, each particle represents a possible solution for the task assigned using an array of n elements, where all elements randomly produce integer values between 1 and m. Figure 4 shows the assignment of ten tasks to five virtual machines. For example, in Particle 1 or nest 1, tasks T_1, T_5 and T_3 are assigned to VM_1 and tasks T_2, T_6 and T_4 are assigned to VM_2 and T_7 is assigned to VM_3 and T_8 and T_9 are assigned to VM_4, and T_{10} is assigned to VM_5.



		T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
Nest1	Particle1	VM1	VM2	VM1	VM2	VM1	VM2	VM3	VM4	VM4	VM5
Nest2	Particle2	VM2	VM2	VM3	VM2	VM4	VM1	VM1	VM5	VM3	VM5
Nest3	Particle3	VM3	VM3	VM2	VM3	VM5	VM4	VM1	VM4	VM2	VM4

Figure 4: Particle Representation

6.1 The PSOCS Task Scheduling Algorithm

The methodology of the PSOCS process is illustrated by the flowchart shown in Figure 5.The main flowchart steps are:

1- Initialization by dividing the application into set of cloudlet.
2- Create cloudlet and virtaul machine.
3- Schedule the cloudlet using the proposed algorithm.

4- Assign the cloudlte T to the scheduler, using parallel computation to compute the cloudlets with the help of the workers
5- Find out the cloudlet execution time T_{et} and Total timeT_t.
6-Termination check when the entire cloudlet T_t has been assigned to the scheduler, the algorithm terminates
Step 3 is the main process of the algorithm.
Figure 5 presents the flowchart of the proposed PSOCS task scheduling algorithm.

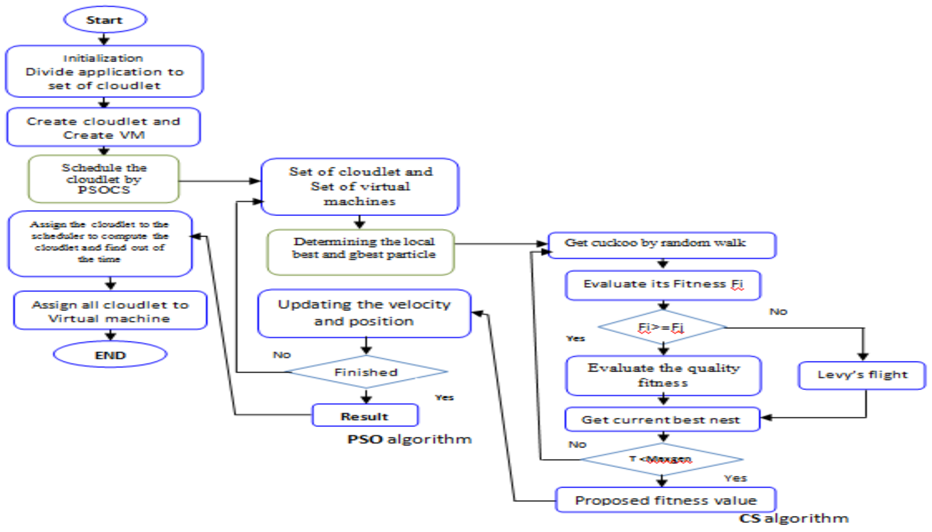

Start
Initialization Divide application to set of cloudlet
Create cloudlet and Create VM
Schedule the cloudlet by PSOCS
Assign the cloudlet to the scheduler to compute the cloudlet and find out of the time
Assign all cloudlet to Virtual machine
END
Set of cloudlet and Set of virtual machines
Determining the local best and gbest particle
Updating the velocity and position
No
Finished
Yes
Result
PSO algorithm
Get cuckoo by random walk
Evaluate its Fitness Fi
Fi>=Fj
No
Yes
Evaluate the quality fitness
Levy's flight
Get current best nest
No
T <Maxgen
Yes
Proposed fitness value
CS algorithm

Figure 5: Flowchart of the Proposed PSOCS Scheduling Algorithm

VII. Performance Evaluation

Cloudsim is cloud computing, simulation software developed by the Gridbus project team and the grid Laboratory of the University of Melbourne in Australia. CloudSim has been used to implement the proposed PSO-CS task scheduling algorithm. Also, a comparative study has been done to evaluate the performance of the proposed PSO-CS algorithm with respect to the existed Random Allocation algorithm(RA) [29], and the original PSO algorithm. This simulation mainly validates the advantage of the makespan and the resource utilization among these scheduling algorithms in the Cloud Computing environment.

7.1. The Implementation Results

By using equation (1), the simulation results to evaluate the makespan of the three algorithms; Random Allocation (RA), original PSO, and PSO-CS; using 5, and 10 Virtual machines, and 10, 20, 30, and 40 tasks are described in Figures (4, 5) and Table (1,2) respectively. According to the simulation results, it is found that the proposed approach (PSO-CS) outperforms the RA and the original PSO algorithms.

Table 1 represents the makespan of RA, PSO and PSO-CS task scheduling algorithms using 5 virtual machines. Figure 4 illustrates the comparison of Makespan using the algorithms. The X-axis represents the number of cloudlets (tasks) and Y-axis represents the execution time (Makespan) in seconds. According to the results in figure 4, it is found that makespan of our proposed (PSOCS) is significant decreased by considering more number of cloudlets. The proposed PSO-CS algorithm provides best results relative to RA and PSO algorithms. By RA and PSO vs. PSO_CS improvement; 21.45% and 26.65% respectively.

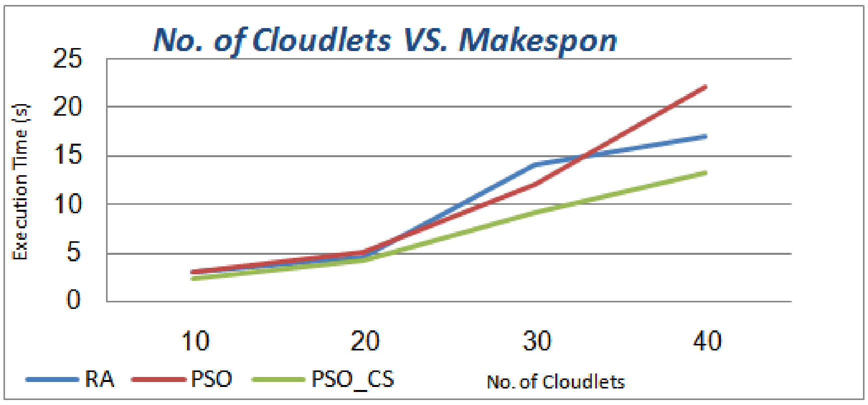


Figure 4: Execution time of all Cloudlet when No. VMs (5)

Table1: Compared scheduling algorithm with Execution Time (Sec)

RA	PSO	PSO_CS	VM	Cloudlet
3	3.11069307	**2.391129032**		10
4.6	5.230736842	**4.268631579**	**5**	20
14.1	12.07002398	**9.076738609**		30
17.1	22.13103448	**13.21939655**		40

Table 2 represents the makespan of RA, PSO and PSO-CS task scheduling algorithms using 10 virtual machines. By RA and PSO vs. PSO_CS improvement; 27.02% and 22.48% respectively.

Table2: Compared scheduling algorithm with Execution Time (Sec)

RA	PSO	PSOCS	VM	Cloudlet
2.2	2.21908549	**1.48072562**		10
5.9	4.48764805	**3.57886497**	**10**	20
7	7.15238095	**6.09310345**		30
9.9	9.70897704	**7.61503132**		40

Figures (4, 5) show comparisons of makespan when number of VMs is varied of 5 and 10 for RA, PSO and PSOCS. In all the cases algorithms it is clearly from the graph that our algorithm performs better other algorithm.

The simulation result utilization is described as in Tables (3, 4) and Figure. (6, 7), Tables (3, 4) show comparison of utilization with RA, PSO and PSOCS when number of virtual machines is varying once when No.VM is five other once when NO. VM is ten. Fig (6, 7) illustrates the comparison of utilization using algorithms discussed previously. The X-axis represents the number of cloudlets and Y-axis represents the utilization ratio. It is clearly from the graph that PSOCS is more efficient when compared with other algorithms. We used around 40 tasks for our comparisons.

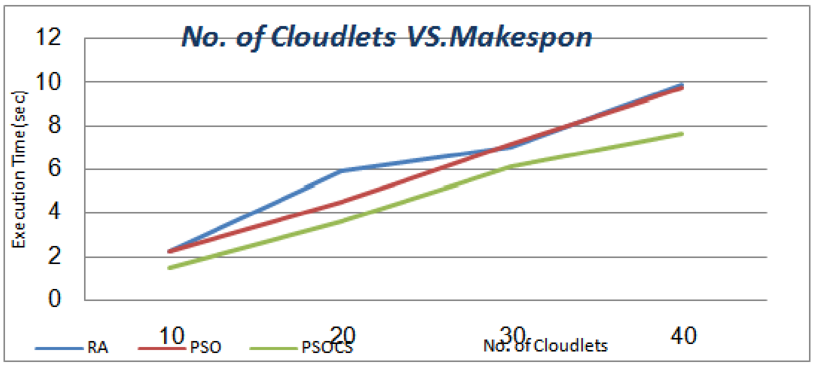


Figure 5: Execution time of all Cloudlet when No. VMs (10)

Figures (6, 7) show comparisons of utilization when number of VMs is varied of 5 and 10 for RA, PSO and PSOCS. In all the cases algorithms it is clearly from the graph that our algorithm performs better other algorithm.

Table3: Comparison three algorithms with Utilization vs. NO. VMs(5)

RA	PSO	PSOCS	VM	Cloudlet
0.62	0.50629	**0.82404**		10
0.7521739	0.724139489	**0.888377435**		20
0.5858156	0.644754177	**0.871109968**	**5**	30
0.49239767	0.388064025	**0.639837228**		40

Table4: Comparison three algorithms with Utilization vs. NO. VMs(10)

RA	PSO	PSOCS	VM	Cloudlet
0.368182	0.38077399	**0.58296606**		10
0.355932	0.41487024	**0.49601614**		20
0.457143	0.36903696	**0.49878417**	**10**	30
0.39596	0.3348205	**0.45243068**		40

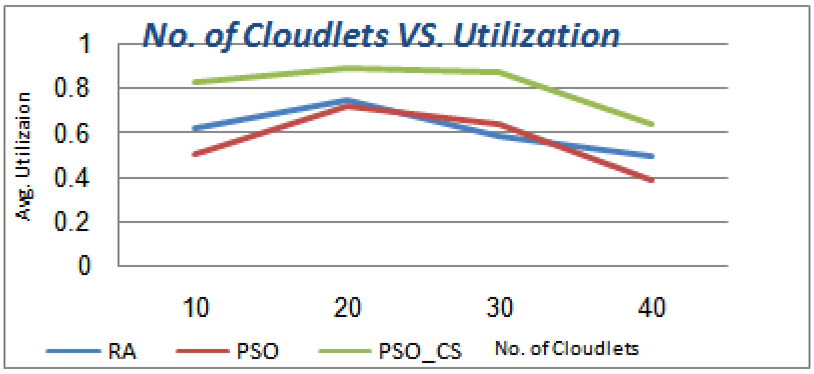


Figure 6: Calculate Utilization of all Cloudlet when No.VMs (5)

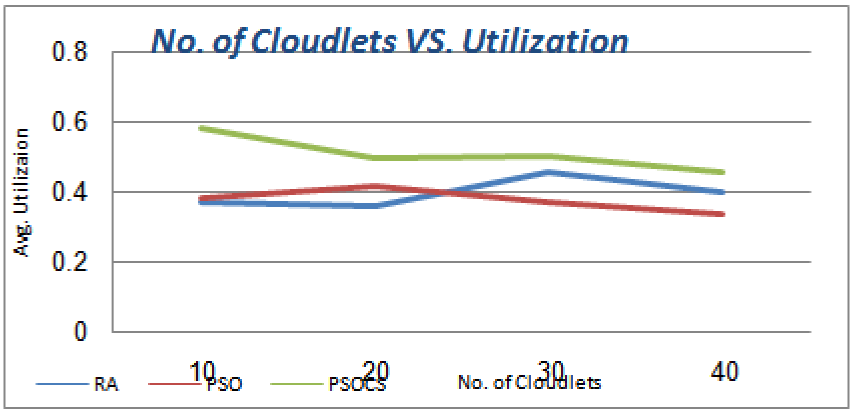


Figure 7: Calculate Utilization of all Cloudlet when No.VMs (10)

In the experiment, the number of the virtual machine is five and the number of the Cloudlets is varying. And show the experimental results with clarification of the process of distribution of tasks on virtual machine in Table 5 and Table 6. In the table shows the first column is the number of cloudlet. The second column is the number of virtual machines. The third column is the cloudlets running time, the fourth column is the start time, and the fifth column is the end time.

From table 5 It can be seen that because of scheduling PSO four tasks are Assigned to the virtual machine No. 4, is not assigned to any task to the virtual machine No. 1.These lead to load imbalance and the total execution time of ten cloudlets is the longest.

Table5: Distributed cloudlet on VM in PSO algorithm

PSO				
Cloudlet NO	VM NO	Run Time	Start Time	Finish Time
7	4	0.688118812	0.1	0.788118812
2	4	0.706534654	0.788118812	1.494653466
5	4	0.732079208	1.494653466	2.226732674
1	2	0.784835165	0.1	0.884835165
0	4	0.883960396	2.226732674	**3.11069307**
4	2	0.910549451	0.884835165	1.795384616
3	2	0.962637363	1.795384606	2.758021969
9	3	0.990554415	0.1	1.090554415
6	0	1.007526882	0.1	1.107526882
8	0	1.309408602	1.107526882	2.416935484

In Table 6, it can be seen that because of the PSOCS, each virtual machine has cloudlets to run. The load is very balanced. The execution time of the ten cloudlet is less, In Table 5 maxtime is **3.11069307**either table 6 maxtime is **2.391129032.**

Table5: Distributed cloudlet on VM in PSOCS algorithm

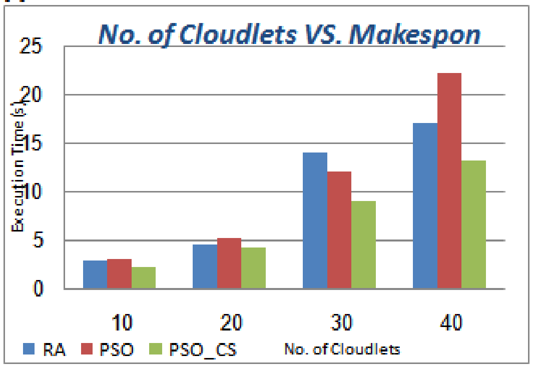
PSOCS				
Cloudlet NO	VM NO	Run Time	Start Time	Finish Time
7	4	0.688118812	0.1	0.788118812
1	4	0.707128713	0.788118812	1.495247525
6	3	0.769609856	0.1	0.869609856
2	2	0.784175824	0.1	0.884175824

5	2	0.812527473	0.884175824	1.696703297
0	4	0.883960396	1.495247525	2.379207921
8	3	1.000205339	0.869609856	1.869815195
4	0	1.113709677	0.1	1.213709677
3	0	1.177419355	1.213709677	**2.391129032**
9	1	1.296774194	0.1	1.396774194

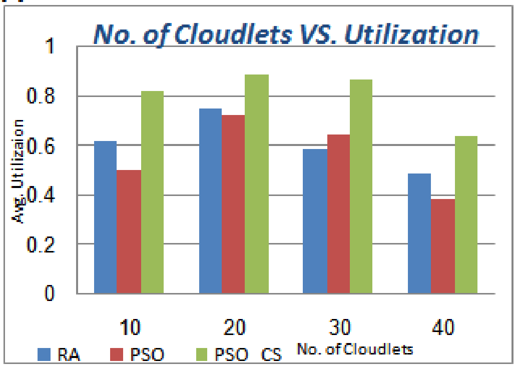
Figure (a, b) shows the comparison of makespan and utilization to number of virtual machines when number of VMs are 5 vs. a set varied of cloudlets. Also in Figure 8 (c, d) illustrates comparison of makespan and utilization when number VMs are 10. Results illustrate that PSOCS is more performance when compared with RA and PSO algorithms.

Figure 9 shows the makespan with different numbers of virtual machine (i.e., VM). As we can see, performance is also affected by the number of virtual machines .In specifically, the more virtual machine available, the easier to find an optimal solution and reduce makespan.

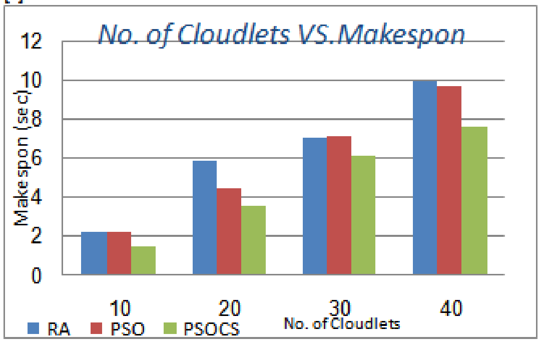
[a]



[b]



[c]



[d]

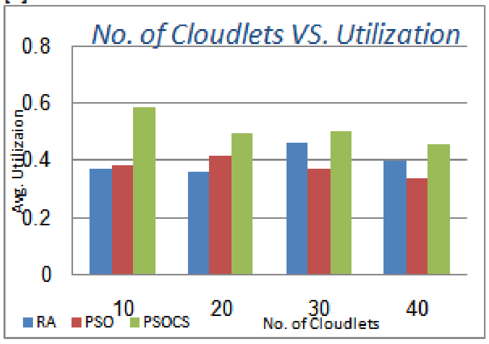


Figure 8: (a, c) Comparison between Makespan of numbers of Cloudlets vs. number of the VMs.
(b, d) Comparison between Utilization numbers Cloudlets vs. Number of the VMs.

Figure 10 shows the utilization with different numbers of virtual machine (i.e., VM). As we can see, utilization is also affected by the number of virtual machines. In specifically, the more virtual machine available, the utilization ratio increases.

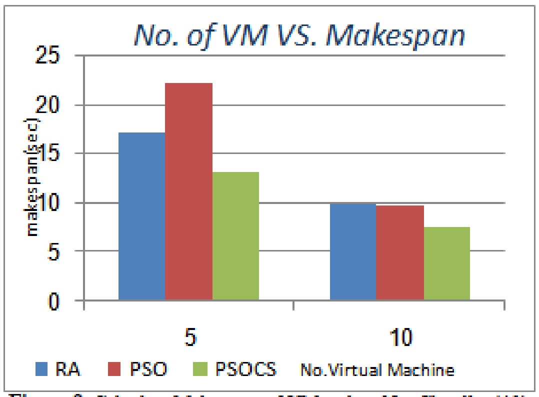


Figure 9 Calculate Makespan of VMs when No. Cloudlet (40)

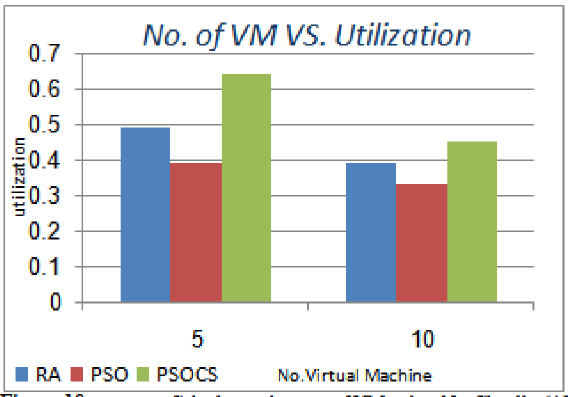


Figure 10 Calculate utilization of VMs when No. Cloudlet (40)

VIII. Conclusion And Future Work

The work in this paper present a combine task scheduling algorithm based on the PSO and CS. This algorithm is called PSOCS. The target of the combine is to minimize makespan and increase utilization ratio of the application workflows in the Cloud computing. Where increases utilization ratio is obtained through decreases the completion time of the executed tasks on a virtual machine. The core principle of the proposed PSOCS algorithm is based on a combine of PSO and CS algorithms. The combine PSO with CS algorithms performs better for local searches. Because the CS used in the local search instead other local search algorithms such as hill-climbing. The combine algorithm finds better solutions than other algorithms. Generally, a comparison of the performance of PSO-CS with existed RA and the original PSO algorithms through a simulation experiment using cloudsim shows that PSO-CS performs better than that RA and the original PSO algorithms. In future work, we plan to improve PSO and CS algorithms in order minimum execute time, and maximum of utilization resource.

References

[1]. Mishra, R. and A. Jaiswal, Ant colony optimization: A solution of load balancing in cloud. International Journal of Web & Semantic Technology (IJWesT), 2012. 3(2): p. 33-50.
[2]. Uma, J., V. Ramasamy, and A. Kaleeswaran, Load Balancing Algorithms in Cloud Computing Environment-A Methodical Comparison. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume. 3.
[3]. Nishant, K., et al. Load balancing of nodes in cloud using ant colony optimization. in Computer Modelling and Simulation (UKSim), 2012 UKSim 14th International Conference on. 2012. IEEE.
[4]. Calheiros, R.N., et al., CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 2011. 41(1): p. 23-50.
[5]. Kennedy, J. and R. Eberhart. Particle swarm optimization. in Neural Networks, 1995. Proceedings., IEEE International Conference on. 1995.
[6]. Yu, B., X. Yuan, and J. Wang, Short-term hydro-thermal scheduling using particle swarm optimization method. Energy Conversion and Management, 2007. 48(7): p. 1902-1908.
[7]. Yin, P.-Y., et al., A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems. Computer Standards & Interfaces, 2006. 28(4): p. 441-450.
[8]. Sousa, T., A. Silva, and A. Neves, Particle swarm based data mining algorithms for classification tasks. Parallel Computing, 2004. 30(5): p. 767-783.
[9]. Lu, W., et al., Analysis of pollutant levels in central Hong Kong applying neural network method with particle swarm optimization. Environmental Monitoring and Assessment, 2002. 79(3): p. 217-230.
[10]. Yang, X.-S. and S. Deb, Engineering optimisation by cuckoo search. International Journal of Mathematical Modelling and Numerical Optimisation, 2010. 1(4): p. 330-343.
[11]. Suraj, P., et al., A Particle Swarm Optimization (PSO)-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environments. 2010.
[12]. Radojevic, B. and M. Zagar. Analysis of issues with load balancing algorithms in hosted (cloud) environments. in MIPRO, 2011 Proceedings of the 34th International Convention. 2011. IEEE.
[13]. Lee, R. and B. Jeng. Load-balancing tactics in cloud. in Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on. 2011. IEEE.
[14]. Ren, X., R. Lin, and H. Zou. A dynamic load balancing strategy for cloud computing platform based on exponential smoothing forecast. in Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on. 2011. IEEE.
[15]. Zhong, H., K. Tao, and X. Zhang. An approach to optimized resource scheduling algorithm for open-source cloud systems. in ChinaGrid Conference (ChinaGrid), 2010 Fifth Annual. 2010. IEEE.
[16]. Lu, X. and Z. Gu. A load-adapative cloud resource scheduling model based on ant colony algorithm. in Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on. 2011. IEEE.
[17]. Li, K., et al. Cloud task scheduling based on load balancing ant colony optimization. in Chinagrid Conference (ChinaGrid), 2011 Sixth Annual. 2011. IEEE.
[18]. Song, X., L. Gao, and J. Wang. Job scheduling based on ant colony optimization in cloud computing. in Computer Science and Service System (CSSS), 2011 International Conference on. 2011. IEEE.
[19]. Pooranian, Z., et al., An efficient meta-heuristic algorithm for grid computing. Journal of Combinatorial Optimization, 2013: p. 1-22.
[20]. Blythe, J., et al. Task scheduling strategies for workflow-based applications in grids. in Cluster Computing and the Grid, 2005. CCGrid 2005. IEEE International Symposium on. 2005. IEEE.
[21]. Brucker, P. and P. Brucker, Scheduling algorithms. Vol. 3. 2007: Springer.
[22]. Kruekaew, B. and W. Kimpan. Virtual Machine Scheduling Management on Cloud Computing Using Artificial Bee Colony. in Proceedings of the International MultiConference of Engineers and Computer Scientists. 2014.
[23]. Visalakshi, P. and S. Sivanandam, Dynamic task scheduling with load balancing using hybrid particle swarm optimization. Int. J. Open Problems Compt. Math, 2009. 2(3): p. 475-488.
[24]. Selvarani, S. and G.S. Sadhasivam. Improved cost-based algorithm for task scheduling in cloud computing. in Computational intelligence and computing research (iccic), 2010 ieee international conference on. 2010. IEEE.
[25]. Uma, S., et al., A hybrid PSO with dynamic inertia weight and GA approach for discovering classification rule in data mining. International Journal of Computer Applications, 2012. 40(17).
[26]. Yang, X.-S. and S. Deb. Cuckoo search via Lévy flights. in Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on. 2009. IEEE.
[27]. Rambharose, T. and A. Nikov, Computational intelligence-based personalization of interactive web systems. 2011.
[28]. Brown, C.T., L.S. Liebovitch, and R. Glendon, Lévy flights in Dobe Ju/'hoansi foraging patterns. Human Ecology, 2007. 35(1): p. 129-138.
[29]. Xu, X., et al., Cloud task and virtual machine allocation strategy in cloud computing environment, in Network Computing and Information Security. 2012, Springer. p. 113-120.