Arabian Journal for Science and Engineering (2022) 47:1821–1830
https://doi.org/10.1007/s13369-021-06076-7

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

Multi Objective Task Scheduling in Cloud Computing Using Cat Swarm Optimization Algorithm

Sudheer Mangalampalli[1] · Sangram Keshari Swain[2] · Vamsi Krishna Mangalampalli[3]

Received: 7 June 2021 / Accepted: 12 August 2021 / Published online: 1 September 2021

Abstract
Efficient Scheduling of tasks is essential in cloud computing to provision the virtual resources to the tasks, effectively by minimizing makespan and maximizing resource utilization in cloud computing. Existing scheduling algorithms talks about makespan and resource utilization, but very few authors addressed the issues named as migration time, energy consumption, total power cost in datacenters. These three mentioned metrics are essential in the view of cloud provider, as by minimizing migration time, energy consumption and total power cost in datacenters cloud provider will be directly benefited. This paper introduces task scheduling by using Cat Swarm Optimization algorithm, which addresses the parameters makespan, migration time, Energy Consumption and Total Power Cost at Datacenters. Scheduling of tasks were done by calculating priorities of tasks at task level, and calculating priorities of VMs at VM level to schedule appropriate mapping of tasks onto VMs. It is implemented by using cloudsim simulator and input to the algorithm is generated randomly from the cloudsim for total power cost, we have used HPC2N and NASA workloads, which are parallel workload archives, which were given as an input to the algorithm. Proposed algorithm is compared against existing algorithms like PSO and CS. From the simulation results, we have observed that we got a significant improvement in different parameters when we used HPC2N and NASA workloads. Makespan is improved by 16%, 10%, 27%, 20% by using HPC2N and NASA workload over PSO and CS algorithms, respectively. Energy Consumption is minimized by 22%, 12%, 31%, 21% by using HPC2N and NASA workload over PSO and CS algorithms, respectively. Total Power cost is minimized by 31% and 39% over PSO and CS algorithms, respectively. Migration time is minimized by 34%, 29%, 20%, 14% by using HPC2N and NASA workloads over PSO and CS algorithms, respectively.

Keywords Cloud computing · Task scheduling · Cat swarm optimization · Total power cost at datacenters · Energy consumption · PSO-Particle Swarm Optimization · CSO-Cat Swarm Optimization · CS-Cuckoo Search

1 Introduction

Cloud computing is a paradigm which recently revolutionized in the IT industry, as the most of the IT companies are moving their on premises infrastructure to the cloud through which all the companies render their services to the customers in terms of compute, storage, and network. This model can be helpful to the users to get access to the different resources on demand by using different service models. To get on demand access to the resources in the cloud and to provision the resources requested by the users on demand, there is a scheduling mechanism needed in cloud paradigm. In this model i.e. in cloud computing paradigm, many of the users submit the requests simultaneously and all these requests may or may not have the same processing capacity. To handle these complex requests, there is a definite need of appropriate scheduling mechanism, which can efficiently map the tasks of the users onto the corresponding virtual machines. In the cloud paradigm, many of the users will submit requests simultaneously and there will be a broker which needs to take care of all these requests and submits it to Task manager, and it will validate requests and submits these

✉ Sudheer Mangalampalli
mssudheer2015@gmail.com

1 Centurion University of Technology and Management, Village Alluri Nagar, R.Sitapur 761211, Odisha, India

2 Department of CSE, Centurion University of Technology and Management, Village Alluri Nagar, R.Sitapur 761211, Odisha, India

3 Department of CSE, Chaitanya Institute of Science and Technology, Hanamkonda, Andhra Pradesh, India

requests to scheduler which maps these incoming requests on to the virtual machines which were resided in physical machines. Mapping of these huge numbers of requests to VMs and mapping appropriate request based on processing capacity of the request and assigning it to a VM, which can process the request, efficiently by minimizing makespan and other parameters. For task scheduling in cloud computing, many of the authors used nature inspired algorithms to solve task-scheduling problem in cloud computing, as it is a NP hard problem. Many of the existing works carried out to solve task scheduling in cloud computing using PSO, ACO,GA,Cuckoo search and Crow search algorithms in [1–5] respectively. In [1–5] as mentioned, they have used the above mentioned nature inspired algorithms to solve task scheduling problem. All the existing algorithms addressed the metrics named as makespan as it is the primary metric, but many of the authors also addresses execution time, Total execution cost and energy consumption.

In this paper, we have designed our algorithm in such a way that we have used a nature inspired algorithm named as cat swarm optimization, which addresses primary parameter named as makespan, while minimizing energy consumption and total power cost at datacenters.

2 Objectives and Highlights of the Paper

The main highlights of the paper are mentioned below.

- Multi objective task scheduling algorithm is designed by using cat swarm optimization.
- This algorithm calculates priorities of tasks and VMs and there by appropriately maps tasks onto the corresponding VMs.
- Makespan, Migration time, Energy consumption and Total power cost at datacenters were evaluated as metrics in this algorithm.
- Entire simulation is carried out in cloudsim [6] simulator and input to the algorithm is generated from HPC2N [7] and NASA [8] parallel workloads from super computing logs.

3 Literature Survey

In [1] proposed a task scheduling algorithm which concentrates on makespan and resource utilization. It was implemented by using the improved PSO algorithm, input to the algorithm is generated randomly and is evaluated by using cloudsim tool, results were compared with the existing algorithm PSO, and it proves it creates huge impact over the existing algorithm in view of specified parameters. In [2], an algorithm which focuses on makespan and preprocessing time. It was modeled by using ACO algorithm, by inculcating diversification and reinforcement technique. Input to this algorithm generated as randomly, and it was simulated on cloudsim and evaluated over ACO algorithm variants and outperforms existing algorithms in view of above mentioned metrics. In [3], proposed an algorithm focused on minimization of makespan and throughput. It was developed using GA algorithm and workload for this generated randomly and it was implemented on cloudsim simulator. It was compared with existing greedy and simple allocation approaches, and it is outperformed in the view of specified parameters. In [4], a multi objective scheduling algorithm is proposed to address parameters makespan, Total Execution cost and execution time. It is modeled by using hybrid CSPSO. It was simulated on cloudsim and compared with PSO and CS algorithms, it was outperformed over existing algorithms in terms of above mentioned metrics. In [5], which concentrates on makespan with heterogeneous workloads. It is modeled by using crow search algorithm, and it is evaluated against PSO, CS algorithms and it outperforms the mentioned algorithms in view of specified metrics.In [9], which minimizes energy consumption, memory. Hybridization of CS and HS algorithms are used for modeling algorithm. It is simulated against existing CS, HS algorithms and it outperforms existing algorithms with above mentioned parameters. In [10], minimization of makespan and maximization for resource utilization was concentrated by the authors. The methodology used is hybridization of Iterative local search and cuckoo search. It was implemented on cloudsim and it is evaluated against existing CS and variants of CS algorithms, and it shows that it is outperforms over compared algorithms with specified parameters. In [11], minimization of makespan and maximization of resource utilization was proposed by authors. Cuckoo search algorithm used as a methodology and input to this algorithm is generated from parallel work logs named as HPC2N [7] and NASA [8]. It was compared over ACO, GA, PSO multi objective variants and it is outperformed over compared algorithms with mentioned paramters. In [12], authors focuses on minimization of makespan. CSO and LDIW algorithm are used as a methodology. It was implemented on cloudsim and compared with existing CSO and PSO- LDIW variant, and it outperforms existing algorithms in view of makespan. In [13], minimizing makespan and degree of imbalance were concentrated by authors. CSO algorithm used as a methodology for this algorithm. It was implemented on cloudsim and compared with CSO, PSO-LDIW algorithms, and it outperforms compared approaches in terms of makespan and degree of imbalance. In [19], focused on developing a workflow scheduling algorithm which focuses on minimization of makespan. DGWO algorithm is used as a methodology in this approach. It was simulated against PSO, BPSO and it shown better improvement over compared algorithms in terms of makespan. In

[20], aimed at the parameters named as makespan, execution time and degree of imbalance. FMPSO was used as a methodology for this approach implemented on cloudsim and compared with the existing algorithms named as PSO variants, and it shows better improvement in the mentioned parameters. In [21], focused on quality of service, execution time and execution cost. MPSO was used as a methodology for this approach, and used artificial neural network to predict spot instances in cloud for future purpose. It was compared with existing algorithms like RR, FCFS, ACO and GA approaches, and it shows improvement over these approaches for the mentioned parameters.

In the above section, various task scheduling algorithms were discussed and many of the authors addressed single metrics like makespan, Execution time and Energy consumption, but none of the authors addressed the metrics in combination of migration time, Energy consumption with Total power cost at Datacenters. So, it needs to be addressed and we have developed a scheduler which addresses migration time, Energy and Total power cost at datacenters by using Cat Swarm Optimization.

The below section discuss about Cat Swarm Optimization algorithm [16–18] which is used as a methodology in our proposed approach.

4 Cat Swarm Optimization Algorithm

Cat Swam optimization is an algorithm, which is based on real time behavior of cats. It was developed by Chu and Tsai [16]. The behavior of cats in active mode and taking rest at some point of times. In this algorithm, cats are to be represented as solutions and strive to achieve target by iterating over different cats, and tend to strive for the optimized solution by use of cats. Generally cats were in two modes. i.e. seeking mode and tracing mode. In seeking mode, cats will be in rest but they will be alerted all the time, and the other mode known as tracing mode in which cats will chasing for the targets. This process can be continuously done, until all the iterations are completed. In this algorithm, initially all cats were randomly initialized over the number of dimensions we are specifying in the problem. In this case, we have taken as n dimensions because tasks were may be generated randomly and these were divided into seeking and tracing modes [16]randomly as per the population is given randomly. For every iteration, solutions are calculated for cats and then fitness is calculated for all the cats, until the best solution is achieved. In seeking mode, all cats are in the rest mode but when cats are into tracing mode, the following steps need to be done.

In tracing mode, initially cats have initialized with specific velocity with specified dimension, so new velocity can be calculated as follows.

$$Vel_i^d(t+1) = p * Vel_i^d(t) + c * r * \left(x_d^{best} - x_i^d\right) \tag{1}$$

where $Vel_i^d(t)$ is the velocity of *ith* cat at *tth* iteration, c is a constant and r is the random number to be in between 0 and 1.x_d^{best} is the cat's global best position.x_i^d is the position of the cat at *ith* iteration.

After this cat's position can be changed as follows by equation mentioned below.

$$x_i^d(t+1) = x_i^d + Vel_i^d(t+1) \tag{2}$$

These calculations needs to be done until cats should reach targets with an optimized value and after calculation of fitness then update solutions with non-dominated cats [16–18].

5 Proposed System Architecture

The below table represents the used notations in the proposed system architecture.

The above Fig. 1 indicates the proposed system architecture for task scheduling in cloud computing. Initially we have considered tasks as $T_k = \{T_1, T_2, T_3 \ldots ..T_k\}$. Initially these tasks are to be submitted onto cloud console and these in turn submitted to task manager. Here in this architecture, task manager module need to calculate priorities of Tasks, and then also need to calculate priorities of VMs based on electricity unit cost. It is evaluated because tasks which were coming onto cloud platform are with different variations and these tasks are to be appropriately mapped on to suitable virtual resources in the cloud. This process needs to be carried out by the Task scheduler. Task

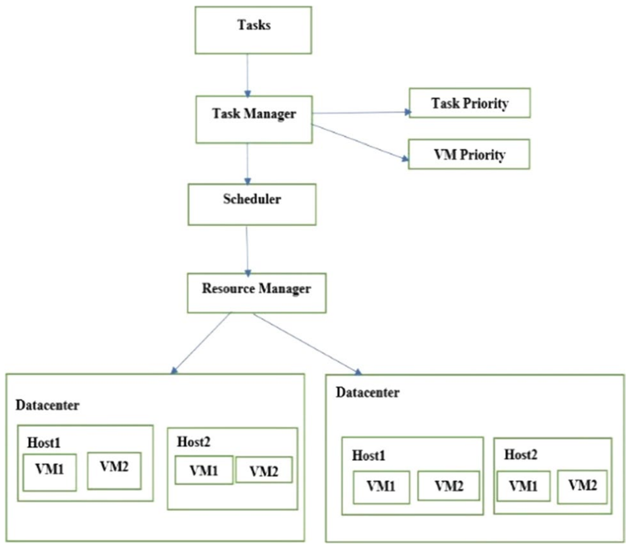

Fig. 1 Proposed System Architecture

scheduler connected to a module named as resource manager, which keeps track of the requested resources by task manager and the allocated resources to users and availability of the resources in corresponding Physical hosts which were resided in the datacenters. In this algorithm, we have assumed n VMs named as $V_n = \{V_1, V_2, V_3, \dots, V_n\}$ and these VMs have to be resided in Physical hosts which are represented as $H_i = \{H_1, H_2, H_3, \dots ..H_i\}$ and these hosts are in turn resided in Datacenters named as $d_j = \{d_1, d_2, d_3, \dots \dots ., d_j\}$. In the above architecture which is represented in Fig. 1, tasks are scheduling to the appropriate VMs based on calculation of Task Priority and VM Priority based on electricity of cost at datacenters. The reason to calculate priority of tasks are as task processing capacities varies as and these need to be mapped to an appropriate VM. VM priority is also need to be calculated based on electricity price per unit cost varies from place to place, as cloud is distributed around the globe [14]. So our approach calculates both priorities named as Task and VM, and then maps tasks onto VMs which can run tasks with lower electricity unit cost, and thereby minimizing energy consumption and Total Power cost at Datacenters. From the above considered tasks, VMs, Physical hosts and Datacenters, we can define the problem in such a way that scheduler need to map appropriate tasks onto VMs based on the priorities considered, and thereby minimizing energy consumption and Total power cost at datacenters. We have assumed that d_j datacenters, H_i Physical hosts, V_n Virtual machines and T_k tasks to schedule tasks onto VMs. In this architecture, we have to calculate priorities for both tasks and VMs. These priorities are calculated once when we submit tasks to task manager. To calculate priorities, we need to calculate the current load on VMs, as these VMs have to run the tasks which were to be assigned by scheduler. The load on all the VMs can be evaluated as represented in the below equation.

$$L_V = \sum_{n=1}^{q} L_n \tag{3}$$

where L_n represents current load on all VMs.

After calculating load on all VMs, we need to calculate capacity of Physical hosts as all VMs are to be resided inside of the physical hosts.

$$L_h = \frac{L_V}{\sum_{k=1}^{m} H_k} \tag{4}$$

where L_h is a load on the physical host which consists of VMs.

In cloud computing paradigm, Load balancing module is needed, as if more number of tasks comes onto cloud interface and if VMs cannot handle to process requests or tasks, then it have to migrate task to the next VM in the same physical host by creating a new instance, or it have to migrate to the next existing VM. This can be achieved by Load balancer and for that, a specific threshold value have to be created whether to identify the system is balanced or imbalanced. Threshold value for system is identified as below

$$TH_k = L_h * H_k \tag{5}$$

After identifying threshold value, we are calculating the system load balance whether it is overloaded or underloaded or balanced.

If system is overloaded then

$$V > TH_k - \sum_{i=1}^{n} L_V \tag{6}$$

If system is in underload condition then

$$V < TH_k - \sum_{i=1}^{n} L_V \tag{7}$$

If system is balanced then

$$V = TH_k - \sum_{i=1}^{n} L_V \tag{8}$$

For appropriate scheduling of tasks onto suitable VMs, there is a need to know the capacity of VM i.e. processing capacity. It is calculated as follows.

$$\text{procapacity}_V = pr^{no} * pr^{mips} \tag{9}$$

Each VMs processing capacity is calculated by using the above equation now all the VMs processing capacity is identified as follows.

$$\text{Totprocapacity}_V = \sum_{i=1}^{n} \text{procapacity}_V \tag{10}$$

We have identified and calculated the prerequisites for to schedule tasks onto VMs. Now we have calculated the priorities of tasks and VMs. Priority of Task is always depends on size and processing capacity of a task. Size of task can be calculated as

$$T_k^{size} = T^{mips} * T^p \tag{11}$$

From the above equation, we have identified the size of the task and then priority of a task can be defined as ratio of size of task and processing capacity of a VM. It is calculated as follows.

$$T^{\text{priority}} = \frac{T_k^{size}}{\text{proccapacity}_V} \tag{12}$$

In this work, we are scheduling tasks by checking at two levels. Initially, we are calculating priority of tasks and then calculating priorities of VMs based on electricity unit cost to map tasks precisely onto suitable VMs. VM priority based on electricity unit cost is calculated as follows.

$$V^{\text{Priority}} = \frac{\text{elec unit cost}^{high}}{\text{elect unit cost}_j^{d}} \tag{13}$$

After calculation of priorities, we have to calculate the metrics which we are addressing in this work. For any task scheduling mechanism in cloud computing, it is important to address makespan, as it is a very important metric to address in this paradigm and more over that minimization of response time helps cloud provider and cloud users in a great manner, thereby effecting different parameters like energy consumption and total power cost indirectly. It is calculated as follows

$$m^k = \text{availability}^n + e^k \tag{14}$$

In above equation, availability^n represents availability of a VM and e^k represents execution time of a task.

After calculation of makespan, there are two another parameters we need to calculate, they are energy consumption and total power cost. Energy consumption is one of the important parameter in cloud computing, because due to high consumption of energy which incurs a huge power cost to the cloud provider, and which also to be impacted on cloud user. In this work, we are striving to minimize energy consumption and total power cost at datacenters and it is calculated as follows.

$$\text{energy}^{\text{con}}\left(V_k\right) = \int_0^k \text{energy}^{\text{con}}_{\text{comp}}\left(V_k\, t\right) + \text{energy}^{\text{con}}_{\text{idle}}\left(V_k,\, t\right)\text{d}t \tag{15}$$

Overall energy consumption can be calculated as follows.

$$\text{energy}^{con} = \sum \text{energy}^{con}\left(V_k\right) \tag{16}$$

Energy consumption indirectly effects the total power cost at data centers and Total power cost in datacenters typically depends on energy generated from Grids, diesel generators and green resources [15] and power consumption is calculated per hour in datacenters and it is calculated as follows.

$$Total_{Powcost} = grid_{cost} + \left(dg_{cost} * \alpha_{dg}\right) + green_{cost} \tag{17}$$

where $grid_{cost}$ is the cost incurred for the power from grid, dg_{cost} is the cost incurred for the power due to diesel generator, which is an alternative source of energy in the absence of other sources. α_{dg} is a parameter which is to be kept as ON or OFF state, as when generator is on it is in ON state, otherwise it is in OFF state. $green_{cost}$ is cost incurred for the power from green resources.

$$\text{grid}_{cost} = energy^{con}\left(d_j, h\right) * p\left(d_j, h\right) \tag{18}$$

Migration time is also one of the important parameter included in this work, and it is required in cloud paradigm when huge number of tasks comes onto VM load balancer will automatically migrate these tasks to the next VM in the same physical host, or to a VM which is in another physical host. Migration time indirectly impacts the makespan because if migration time increases automatically, the execution time increases which impacts makespan. Migration time is calculated as below

$$Mig_t = \sum_{i=1}^{k} T_k mov/t \tag{19}$$

After detailed mathematical modeling for the proposed system architecture, we have written Multiobjective Scheduling Optimization model.

5.1 Multi Objective Scheduling Optimization

This Multiobjective Scheduling Optimization model helps to optimize parameters named as makespan, Energy consumption and Total power cost at datacenters. It is calculated as follows

$$f(x) = \min \sum_x m^k(x), Mig_t(x), \text{energy}^{con}(x), Total_{\text{Powcost}}(x) \tag{20}$$

In the above equation, we have mentioned three parameters, as we are trying to get optimized values for makespan, migration time, energy consumption and total power cost.

Now we have defined the multi objective scheduling approach and in our work, we are working with cat swarm optimization algorithm [16–18]. In the next section, we have precisely discusses about cat swarm optimization algorithm [16–18] in a detailed way.

5.2 Proposed Algorithm

In this section, we have discussed about proposed task scheduling algorithm by using cat Swarm optimization in which, we assumed each cat is used as a schedule to map tasks and VMs i.e. virtual resources which are resided in physical hosts and get optimized solution for considered multiple objectives mentioned in the Eq. 17 and those objectives are makespan, Energy consumption and total power cost at data centers. The below algorithm clearly discusses about the algorithm.

In the above algorithm, cat population is generated randomly and considered all the parameters tasks,VMs. After generation of cat population, we have calculated both task and VM priorities. After calculation of priorities, we have evaluated parameters named as makespan, energy consumption and Total Power cost at datacenters, then evaluated the fitness function mentioned in multiobjective optimization for the cat population. Identified solutions are arranged in ascending order and apply either seek or tracing mode on the population of cats. If the cats were in the seeking mode, store them in the non dominant solutions, if not tracing mode have to be applied then by calculating velocity of cats and positions of the cats, until all the iterations are completed. From all the solutions, identify the present best solution of corresponding cat and global best solution. If current solution of any cat is better than the global best solution, replace global solution with current solution of corresponding cat. This have to be done, until all the iterations are completed with all the population of cats. The flow of the algorithm is depicted in the below Fig. 2.

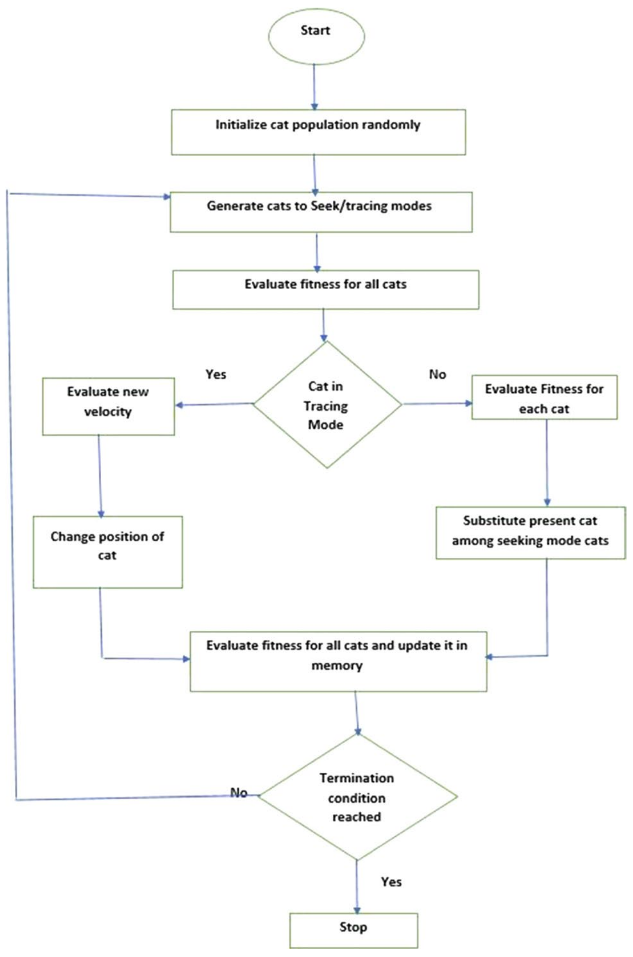
Start
Initialize cat population randomly
Generate cats to Seek/tracing modes
Evaluate fitness for all cats
Yes
Cat in Tracing Mode
No
Evaluate new velocity
Evaluate Fitness for each cat
Change position of cat
Substitute present cat among seeking mode cats
Evaluate fitness for all cats and update it in memory
No
Termination condition reached
Yes
Stop

Fig. 2 Flow of the Algorithm

6 Simulation and Results

This section precisely discusses about simulation of the algorithm, which is implemented on cloudsim [6] toolkit through which we have simulated the cloud environment. It was written in java programming language and we can simulate the cloud environment through simple java programs. In this paper, we have simulated cat swarm optimization and input to this algorithm is generated from the HPC2N[7] and NASA[8] which are supercomputing sites which maintain parallel workload archives (Figs. 3, 4, 5, 6, 7, 8, 9).

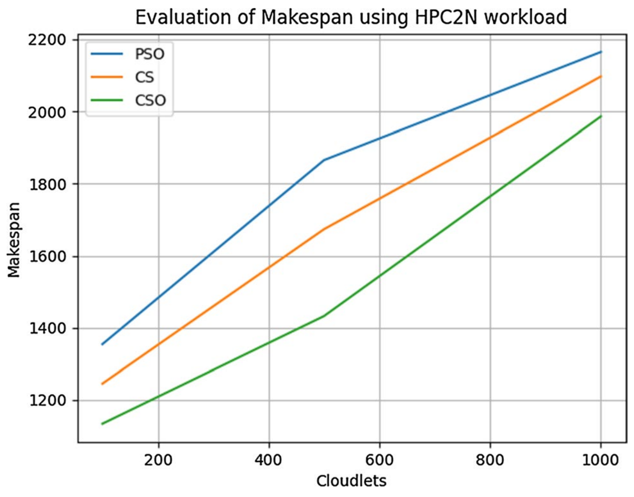

Fig. 3 Evaluation of makespan using HPC2N workload

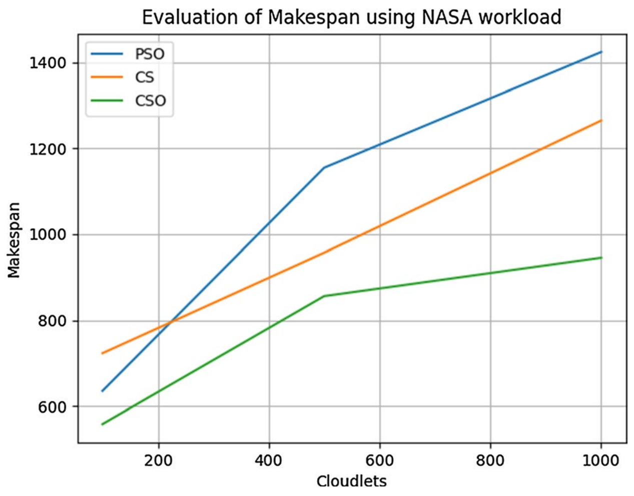

Fig. 4 Evaluation of makespan using NASA workload

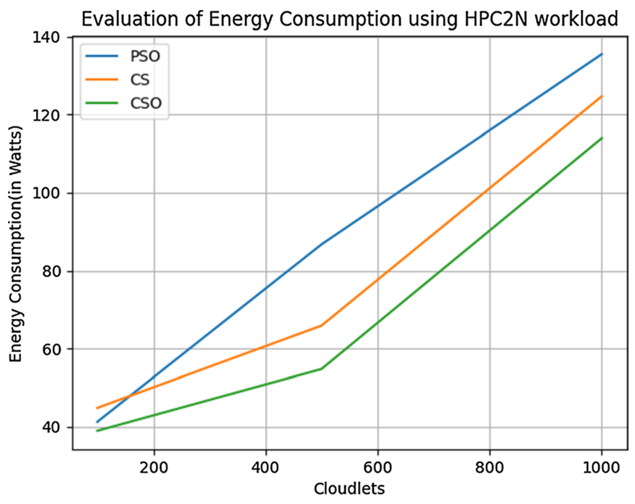

Fig. 5 Evaluation of Energy consumption using HPC2N workload

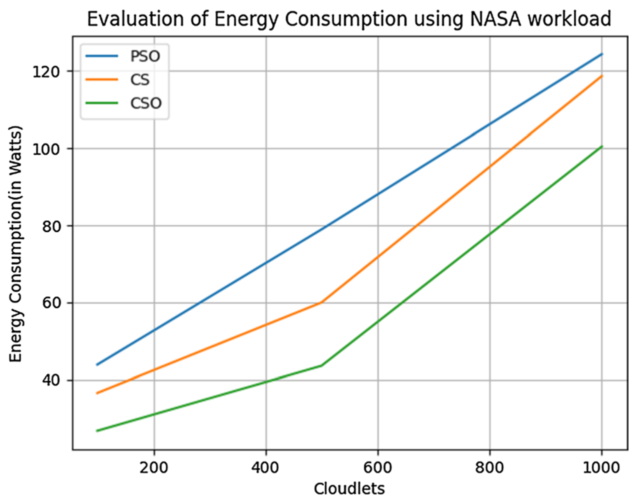

Fig. 6 Evaluation of Energy consumption using NASA workload

6.1 Simulation Setup

For carrying out simulation, we have taken the settings. Those details were mentioned in the below Table 1. After configuration of the simulation in cloudsim [6]

6.2 Evaluation of Makespan

We have evaluated makespan as a primary parameter in this simulation and we have used HPC2N [7] workload initially and evaluated makespan and it is compared with existing PSO and CS algorithms, and the proposed CSO scheduler reveals that it is outperformed over existing algorithms in terms of makespan.

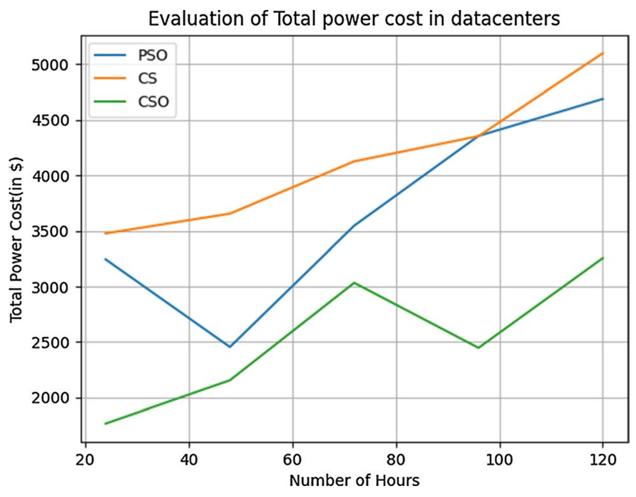

Fig. 7 Evaluation of Total Power cost in datacenters

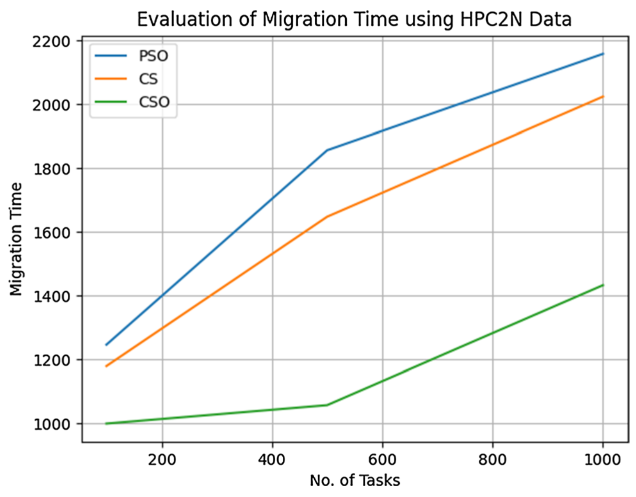

Fig. 8 Evaluation of Migration Time using HPC2N Workload

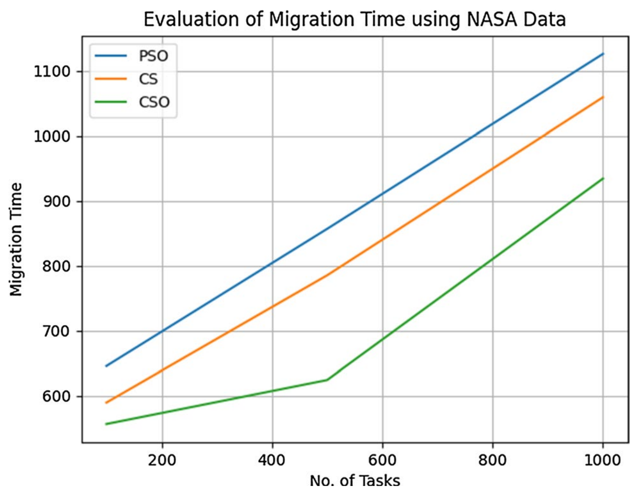

Fig. 9 Evaluation of Migration Time using NASA Workload

Table 1 Notations used in Proposed Architecture

Notation	Meaning of the Entity
T_k	Tasks 1 < i < k
V_n	VMs 1 < j < n
n, k	Number of Virtual Resources and Tasks
d_j	Datacenters
H_i	Physical Hosts
L_V	Load on all VMs
L_h	Load on all Physical Hosts
TH_k	Threshold value for the VM
procapacity_V	Processing capacity of a VM
T^{priority}	Priority of Tasks
V^{Priority}	Priority of VM based on Electricity price per unit cost
m^k	Makespan of Tasks
energy^{con}	Energy Consumption
$\text{Total}_{\text{Powcost}}$	Total Power cost at Datacenters
Mig_t	Migration time

For the simulation we have considered a total of 1000 tasks and workload is taken from HPC2N [7] workload archives. For 100 tasks a makespan of 1354.6, 1245.45, 1134.32 for PSO, CS and CSO Scheduler, respectively. For 500 tasks a makespan of 1865.24, 1673.45, 1432.34 for PSO, CS and CSO Scheduler, respectively. For 1000 tasks, a makespan of 1134.32, 1432.34, 1985.98 for PSO,CS and CSO scheduler, respectively. We have compared our proposed CSO scheduler with existing PSO and CS algorithms and there is an improvement of makespan of 16% over PSO and improvement of makespan of 10% over CS algorithm.

For the evaluation of makespan using NASA [8] workload we have considered 1000 tasks to run simulation and for 100 tasks a makespan of 635.89, 723.49 and 558.87 for PSO, CS and CSO Scheduler, respectively. For 500 tasks, a makespan of 1154.78, 957.26 and 856.35 for PSO, CS and CSO Scheduler, respectively. For 1000, tasks a makespan of 558.87, 856.35 and 945.67 for PSO, CS and CSO Scheduler, respectively. We have compared our proposed CSO scheduler with existing PSO and CS algorithms and there is an improvement of makespan of 27% over PSO and 20% of CS algorithms (Tables 2, 3, 4, 5, 6, 7, 8, 9).

6.3 Evaluation of Energy Consumption

Energy consumption is one of the important parameter as it effects both the cloud provider and cloud user. Minimization of energy consumption is one of the challenge to address in cloud computing paradigm through which with little amount of consumption of energy cloud provider, can successfully run tasks on virtual resources in cloud and cloud user can also be benefited in view of availability of resources to the less cost. We have evaluated energy consumption by taking input from HPC2N [7] and NASA [8] workload archives.

For the evaluation of Energy consumption using HPC2N [7] workload, we have considered 1000 tasks to run simulation, an amount of energy is consumed for 100 tasks 41.28, 44.84 and 38.98 for PSO, CS and CSO algorithms

Table 2 Simulation Settings

S. No	Name of the Entity	Quantity
1	Number of tasks	100–1000
2	Length of the tasks	500,000 to 200,000,000
3	Number of Datacenters	10
4	Capacity of VM	4096 MB
5	Name of Hypervisor	Xen
6	Number of Virtual machines	200
7	Number of processing elements to VM	1
8	Number of Physical hosts	500
9	Physical host RAM Capacity	64 GB

Table 3 Evaluation of Makespan using HPC2N Workload

Cloudlets	PSO	CS	CSO Scheduler
100	1354.56	1245.45	1134.32
500	1865.24	1673.45	1432.34
1000	2164.78	2096.75	1985.98

Table 4 Evaluation of Makespan using NASA workload

Cloudlets	PSO	CS	CSO Scheduler
100	635.89	723.49	558.87
500	1154.78	957.26	856.35
1000	1423.56	1264.56	945.67

Table 5 Evaluation of Energy Consumption using HPC2N workload

Cloudlets	PSO	CS	CSO Scheduler
100	41.28	44.84	38.98
500	86.75	65.89	54.78
1000	135.48	124.67	113.98

Table 6 Evaluation of Energy Consumption using NASA workload

Cloudlets	PSO	CS	CSO Scheduler
100	43.88	36.56	26.76
500	78.95	59.88	43.57
1000	124.36	118.67	100.45

Table 7 Evaluation of Total Power Cost in Datacenters

No. of Hours	PSO	CS	CSO Scheduler
24	3245	3476	1768
48	2456	3654	2156
72	3546	4125	3034
96	4354	4352	2448
120	4687	5096	3256

Table 8 Evaluation of Migration Time using HPC2N Workload

Cloudlets	PSO	CS	CSO Scheduler
100	1245.9	1178.8	998.9
500	1856.7	1647.9	1056.7
1000	2158.8	2024.5	1432.8

Table 9 Evaluation of Migration Time using NASA Workload

Cloudlets	PSO	CS	CSO Scheduler
100	645.9	589.9	556.9
500	856.8	785.3	624.5
1000	1125.6	1058.9	934.2

respectively. For 500 tasks 86.75, 65.89 and 54.78 for PSO, CS and CSO algorithms, respectively. For 1000 tasks 135.48, 124.67 and 113.98 for PSO, CS and CSO algorithms, respectively. We have compared our proposed CSO scheduler with PSO and CS algorithms, and results revealed that consumption of energy is greatly minimized over PSO by 22% and CS by 12%, respectively.

We have evaluated Energy Consumption by using NASA [8] workload archives. We have considered 1000 tasks to run simulation, an amount of energy is consumed for 100 tasks 43.88, 36.56 and 26.76 for PSO,CS and CSO algorithms, respectively. For 500 tasks 78.95, 59.88 and 43.57 for PSO, CS and CSO algorithms, respectively. For 1000 tasks 124.36, 118.67 and 100.45 for PSO, CS and CSO algorithms, respectively. We have compared our proposed approach with existing PSO and CS algorithms, and energy consumption is greatly minimized over PSO by 31% and CS by 21%.

6.4 Evaluation of Total Power Cost at Datacenters

For evaluation of Total Power cost in datacenters, we have used the random generated workload as an input to the algorithm and we have considered total no hours as 120 h as Total power cost is evaluated in terms of number of hours in datacenters and then for 24 h 3245, 3476 and 1768, for 48 h 2456, 3654 and 2156, for 72 h 3546, 4125 and 3034, for 96 h 4354, 4352 and 2448, for 120 h 4687, 5096 and 3256 for PSO, CS and CSO algorithms, respectively. We have compared our proposed scheduler with existing PSO and CS algorithms, and Total power cost in terms of dollars greatly minimized over PSO by 31% and CS by 39%.

6.5 Evaluation of Migration Time

We have evaluated Migration time using HPC2N [7] workload. A total of 1000 tasks were used for simulation. For 100 tasks 1245.9, 1178.8 and 998.9, for 500 tasks 1856.7, 1647.9 and 1056.7, for 1000 tasks 2158.8, 2024.5 and 1432.8 for PSO, CS and CSO Scheduling algorithms, respectively. We have compared our proposed algorithm with PSO and CS algorithms, and migration time is greatly minimized by 34% over PSO and 29% by CS algorithms.

We have evaluated Migration time using NASA [8] workload. We have considered 1000 tasks for this simulation. For 100 tasks, a migration time of 645.9, 589.9 and 556.9, for 500 tasks 856.8, 785.3 and 624.5, for 1000 tasks 1125.6, 1058.9 and 934.2 for PSO, CS and CSO algorithms. We have evaluated migration time by using NASA workload against PSO and CS algorithms, and Migration time is greatly minimized by 20% over PSO and 14% over CS algorithms.

7 Conclusion and Future Work

Task scheduling is one of huge challenge in cloud computing, and it is difficult to schedule appropriate tasks onto Virtual resources in cloud. Many of the existing authors used several nature inspired algorithms to solve task scheduling problem in cloud computing, but in existing works, authors were not able to schedule tasks based on their priorities. To appropriately map tasks onto VMs, we come up with a solution by calculation of Task and VM priorities to precisely schedule tasks onto VMs. The proposed algorithm is modeled by the cat swarm optimization. We have evaluated the metrics, such as makespan, migration time, Energy consumption and Total power cost at the datacenters and we have evaluated proposed scheduler against existing algorithms named as PSO, CS algorithms and from the results, we can clearly observe that our proposed scheduler is outperformed over existing algorithm in view of specified parameters. In future, we want to evaluate our algorithm in real time environment to test the efficiency of our algorithm.

Acknowledgements I would like to thank my Research Supervisors and University members and I also would like to thank the reviewers who gave the necessary comments for my manuscript.

Author Contributions New Authors were contributed in Revision of Abstract and adding a new metric mentioned in the revised manuscript.

Funding There is no disclosures and no financial interests and we have not taken any fund from any agencies.

References

1. Ebadifard, F.; Babamir, S.M.: A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment. Concur. Comput. Practice Exp **30**(12), e4368 (2018)
2. Moon, Y.J.; HeonChang, Y.; Gil, J.-M.; Lim, J.B.: A slave ants based ant colony optimization algorithm for task scheduling in cloud computing environments. HCIS **7**(1), 1–10 (2017)
3. Rekha, P.M.; Dakshayini, M.: Efficient task allocation approach using genetic algorithm for cloud environment. Clust. Comput. **22**(4), 1241–1251 (2019)
4. Prem Jacob, T.; Pradeep, K.: A multi-objective optimal task scheduling in cloud environment using cuckoo particle swarm optimization. Wireless Personal Commun. **109**(1), 315–331 (2019)
5. Prasanna Kumar, K.R.; Kousalya, K.: Amelioration of task scheduling in cloud computing using crow search algorithm. Neural Comput. Appl. **32**(10), 5901–5907 (2020)
6. Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; De Rose, C.A.; Buyya, R.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software Practice Exp. **41**, 23–50 (2011)
7. HPC2N: The HPC2N Seth log; 2016. http://www.cs.huji.ac.il/labs/parallel/workload/l_hpc2n/.0
8. NASA:https://www.cse.huji.ac.il/labs/parallel/workload/l_nasa_ipsc/
9. Pradeep, K.; Prem Jacob, T.: A hybrid approach for task scheduling using the cuckoo and harmony search in cloud computing environment. Wireless Personal Commun. **101**(4), 2287–2311 (2018)
10. Loheswaran, K.; Daniya, T.; Karthick, K.: Hybrid cuckoo search algorithm with iterative local search for optimized task scheduling on cloud computing environment. J. Comput. Theor. Nanosci. **16**(5–6), 2065–2071 (2019)
11. Madni, S.H.; Hussain, M.S.; Latiff, A.; Ali, J.: Multi-objective-oriented cuckoo search optimization-based resource scheduling algorithm for clouds. Arab. J. Sci. Eng. **44**(4), 3585–3602 (2019)
12. Gabi, D., Ismail A.S., and Dankolo N.M: Minimized makespan based improved cat swarm optimization for efficient task scheduling in cloud datacenter. *Proceedings of the 2019 3rd High Performance Computing and Cluster Technologies Conference*. 2019
13. Gabi, D., et al.: Orthogonal Taguchi-based cat algorithm for solving task scheduling problem in cloud computing. Neural Comput. Appl. **30**(6), 1845–1863 (2018)
14. Sudheer, M.S.; Vamsi Krishna, M.: Dynamic PSO for task scheduling optimization in cloud computing. Int J Recent Technol Eng **8**(2), 332–338 (2019)
15. Aslam, S. et al: Towards energy efficiency and power trading exploiting renewable energy in Cloud data centers. *International Conference on Advances in the Emerging Computing Technologies (AECT)*. IEEE, 2019.
16. Chu, S-C; Tsai, P-W; Pan J-S: Cat swarm optimization. *Pacific Rim international conference on artificial intelligence*. Springer, Berlin, Heidelberg, 2006
17. Sharafi, Y.; Khanesar, M.A.; Teshnehlab, M.: Discrete binary cat swarm optimization algorithm. In: 3rd IEEE International Conference on Computer, Control and Communication (IC4), pp. 1–6 (2013)
18. Chu, S.C.; Tsai, P.W.: Computational intelligence based on the behavior of cats. Int. J. Innov. Comput. Inf. Control **3**, 163–173 (2007)
19. Abed-alguni, B.H.; Alawad, N.A.: Distributed Grey Wolf Optimizer for scheduling of workflow applications in cloud environments. Appl. Soft Comput. **102**, 107113 (2021)
20. Mansouri, N.; Zade, B.M.H.; Javidi, M.M.: Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory. Comput. Ind. Eng. **130**, 597–633 (2019)
21. Domanal, S.G.; Reddy, G.R.M.: An efficient cost optimized scheduling for spot instances in heterogeneous cloud environment. Future Generation Comput. Syst. **84**, 11–21 (2018)