Received: March 26, 2022. Revised: July 13, 2022.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

RCOA Scheduler: Rider Cuckoo Optimization Algorithm for Task Scheduling in Cloud Computing

Pradeep Krishnadoss[1]* **Chirag Chandrashekar[1]** **Vijayakumar Kedalu Poornachary[1]**

[1]*Vellore Institute of Technology, Chennai, Tamilnadu, India*
* Corresponding author's Email: pradeep.k@vit.ac.in

Abstract: Cloud computing, a term which has become very popular on the internet platform which has evolved over time and gained more attention due to various features that are being introduced like multiplex numerous users on the identical physical structure, computing of resource on request and many more. Cloud computing is considered to be an enormous pool having essential resources which the users can make use of to complete their task through the internet. Therefore, in order to complete the tasks requested by the users, a scheduling algorithm is required making scheduling a vital part of cloud computing. Though numerous solutions with respect to scheduling techniques have been proposed in in recent literature, researcher's continuously carryout performance upgradation by considering a mixture of QoS parameters thereby enhancing the cloud performance. In this regard, we had introduced an optimal and ideal task scheduling algorithm and challenge the other prevailing algorithms in terms of QoS parameters like makespan and energy. The proposed algorithm increases the performance of the scheduling in cloud by making use of an enhanced meta-heuristic hybrid algorithm called Rider Cuckoo Optimization Algorithm (RCOA), which is an advanced version of the already existing Rider Optimization Algorithm. The proposed RCOA technique had produced an improvement of 5.6%, 4.36% and 2.27% for makespan and had reduced the energy to the tune of 16.8%, 19.18%, and 16.15% when compared with Particle Swarm Optimization (PSO), Cuckoo Search (CS), Rider Optimization Algorithm (ROA) algorithms respectively when used with 25 VMs. Also the proposed approach attains the objective of optimal scheduling swiftly and achieves convergence in a small duration of time.

Keywords: Cloud computing, Task scheduling, Makespan, Rider optimization, Cuckoo search.

1. Introduction

Cloud environment as became the most vital part of the internet, a dedicated area for computing where different task requested by the users are performed by making use of the available resources. Thus, Cloud services allow the necessity for data to be controlled by external objects or some other individual at distinct locations. This cloud platform encompasses different types of computer system technique like distributed, grid, utility and autonomic computing. But the main issue faced in cloud computing platform is the task scheduler program algorithm considering numerous characteristics like resource usage, total cost of task execution for all users, time taken for completion, energy consumption, and tolerance of faults. The task scheduling procedure looks to be an NP-complete problem as the difficulty of the problem depends on the time taken to find the required result set.

A task scheduling algorithm is required to reduce time and energy consumption while increasing the return on service providers. Therefore, such task scheduling algorithm are considered as a vital part in cloud computing as it efficiently scans and appropriate VM (Virtual Machines) are determined by using active decision making for respective tasks. In order to identify an optimal and ideal algorithm numerous parameters like the makespan, time of response, utilization of the system, network-grounded actions together with disbursement of the network communication, round trip, traffic volume and many more are taken into account. The process of optimum sequencing the

tasks present in the cloud can be characterized as hybrid, heuristic and meta-heuristics task scheduling methods. The hybrid task-scheduler is a technique where not only meta-heuristic but also heuristic approach is combined. The heuristic work scheduling program make it easy to assign the work and offer the quickest conceivable outcomes but the optimal outcome is not certain. Considering only the computational time, meta-heuristic programs can achieve huge probing space to find the optimal result. The stress now is on the scheduling of tasks, which is vital as it impacts on how the cloud computing system should operate. Users like designers/programmer are available to use shared resources due to the help of cloud and depending on the user's needs and specifications various different specialized algorithm are being mapped to help the users obtain the required results.

Therefore, an efficient and robust scheduling of task algorithms are required which can offer customers an optimal result-set.

In addition to this, enumeration is also required to construct an optimal schedule and for this it needs the procedure for constructing all task-scheduler that are available after which the obtained outcomes are related and cross-checked with one other to obtain an optimum set which is appropriate in task-scheduling as it increases the performance by reducing the time taken to handle a huge amount of tasks. Thus, the projected method of work tries to solve the task scheduling problem by making use of either meta-heuristic or heuristic grounded algorithm. An ideal solution which is optimal could be obtained by using heuristic-based techniques as these methods make use of some predefined rules apply, and the quality of the results obtained by these methods is determined by the size of the problem and the underlying rules. As a result, the results are obtained through heuristic exploration techniques that are not reasonable and operate at a very high rate.

Thus, the meta-heuristic algorithm performs much better compared to mathematical techniques or heuristic based method. Some of the recognized methods for resolving the problem of task-scheduling in cloud are Ant Colony Optimization, Genetic Algorithm, Firefly Algorithm, League championship Algorithm, Whale Optimization algorithm and Water drop Algorithm.

In this paper, introduce a new algorithm called Rider Cuckoo Optimization Algorithm (RCOA) which will optimize the task scheduling process. The main aim/objective of the proposed algorithm is to reduce the energy consumption and time taken for task execution.

To recapitulate this paper:

- Analyses the issues imposed by the various scheduling techniques and algorithms and their impact on performance of the cloud by reviewing several literatures.
- Based on the review, this paper proposes a new innovative hybrid algorithm called Rider Cuckoo Optimization Algorithm (RCOA) which improves the short-coming and challenges faced by other algorithms, and therefore increases the performance of the task scheduler by focusing on QoS parameters like energy consumption and makespan.
- Finally, we had compared the results obtained using the proposed algorithm with Cuckoo Search (CS), Particle Swarm Optimization (PSO), Rider Optimization Algorithm (ROA) algorithms to ascertain its performance supremacy.

2. Related work

In [1] Most of the literatures that are proposed to perform the task-scheduler process in the cloud. This literature tries to introduce a new algorithm to improve the scheduling method of the CS (cuckoo-search) and HS (harmony-search) program. Thus the new hybrid algorithm called CHSA is introduced to enhance the optimization issue. The above two mentioned algorithm is efficiently hybridized to perform an intelligent processing scheduler. Therefore, a new function which is multi objective in nature is introduced by joining the consumption of energy, cost for combination, penalty and credit. In the last, the performance of the introduced algorithm is related with other algorithm like the individual CS - HS program, HCGSA (Hybrid-Cuckoo-Gravitational-Search-Algorithm) with many different parameters. By studying the solution set obtained from the experiment, the introduced algorithm has reduced memory-usage, cost, penalty, consumption of energy.

In [2] this paper introduces an algorithm called PACS (power-aware cloudlet scheduling) algorithm which maps cloudlets to VMs (virtual-machine). Here goal of the program is to reduce the request processing time to a minor duration schedule and at the same time reduce the cost suffered and energy consumption. For allocating different virtual machines to cloudlets, the algorithm iteratively sets the VMs in groups by applying weights which calculated using optimization and parameters included are power consumption and resources utilization cost. At the same time increase the credit compared to the already available methods.

Table 1. Comparative table for literatures review

S.No	Algorithm	Parameter considered	Compared algorithm	Demerits	Tools
[1]	Cuckoo search and harmony search algorithm CHSA	Minimizing memory usage, energy consumption, cost, and penalty while maximizing credit	Cuckoo Search Algorithm (CSA), Harmony Search (HS)	Throughput, fault tolerance, scalability, and availability are not addressed	Cloud Sim
[2]	Power-aware cloudlet scheduling (PACS)	Energy consumption and cost	Round-robin algorithm (RRA), Greedy, genetic algorithm (GA), particle swarm optimization (PSO), and ant colony optimization (ACO)-	Exploiting parallelism attained through the clustered approach not done	Cloud Sim
[3]	Whale Optimization Algorithm and Harmony Search Algorithm	Makespan and Cost	Grey wolf optimizer (OGWO), Whale optimization algorithm (WOA) , Harmony search (HS)	Low accuracy	Cloud Sim
[4]	Adaptive Neuro-Fuzzy Inference System (ANFIS)-Black Widow Optimization (BWO) (ANFIS-BWO)	Minimize computational time, Computational cost, and Energy consumptions	LB-RC and Adaptive Neuro-Fuzzy Inference System(ANFIS), EECS	Other QoS parameters not considered.	Cloud Sim
[5]	Ant-lion particle swarm optimization ALPSO	Makespan,cost,energy	GA-PSO, particle swarm optimization (PSO), Ant-lion optimization (ALO), and Genetic Algorithm(GA)	Pricing scheme for VM leasing not done	Cloud Sim
[6]	Energy-efcient and reliability aware(EERS))	Cost, energy, makespan	Heterogeneous Earliest Finish Time (HEFT), Enhanced Energy-efficient Scheduling (EES), REEWS	Monetary cost constraints and frequency independent energy consumption not done	Workflow sim
[7]	Hybridization of the ant lion optimizer (ALO) algorithm with a Sine Cosine Algorithm (SCA)	Makespan and cost	Strength Pareto Evolutionary Algorithm 2 (SPEA2)	Multi-swarm and multi-objective cases not done.	Workflow Sim
[8]	Whale Optimization algorithm	Energy consumption and power cost	Particle swarm optimization (PSO) and cuckoo search (CS)	Other QoS parameters not considered.	Cloud Sim
[9]	Sea Lion Optimization (SLnO)	Makespan, cost, energy consumption, resources utilization and degree of imbalance	Whale Optimization Algorithm (WOA), Grey Wolf Optimization (GWO) and Round Robin (RR)	Other QoS parameters not considered.	Cloud Sim
[10]	Hybrid Oppositional Lion optimization algorithm (OLOA)	Makespan, Cost and resource utilization.	Particle Swarm Optimization (PSO) algorithm, oppositional learning based grey wolf optimizer (OGWO) and the Genetic algorithm (GA)	Other QoS paramaters not considered.	Cloud Sim
[11]	Cuckoo Crow Search Algorithm (CCSA)	Makespan, Cost	(MO-ACO), Ant colony optimization (ACO), Min-Min	Other QoS paramaters not considered	Cloud sim

This paper introduces a cloud task booking arrangement based on half Whale Harmony enhancement calculation in [3]. The primary commitment of this work is to adjust the framework stack while attempting to limit the time and cost of a given set of assignments. To replicate the new scheduling technique, the Cloudsim toolbox bundle was used. Finally, this method directs an investigation to demonstrate the execution of the proposed calculation.

In [4] the proposed algorithm makes use of an ANFIS-BWO (Adaptive-Neuro-Fuzzy-Inference-System-Black-Widow-Optimization technique to allocate a proper VM for each task in order to reduce the time delay. Scheduling of resource is another essential goal for optimum consumption of properties present in the environment of the cloud. The BWO algorithm is used to achieve an ideal solution set. The introduced technique can allocate the VMs present on the cloud by the optimum scheduler schemes. The key objective of the introduced technique is to reduce the time take for computation and cost also at the same time minimize the consumption of energy for different tasks.

In [5] this paper proposes an improved version of ant-lion optimization (ALO) algorithm which is crossbred with popular particle swarm optimization (PSO) algorithm to enhance the system scheduling precisely for cloud. A new technique for security purpose is used known as Data Encryption Standard (DES) which encodes the information present in cloud while scheduling is carried out. The goal of the research is to contribute an improved system scheduling more safely than the existing frameworks. Improvement parameters are assessed in terms of makespan, load and cost.

[6] This paper introduces a new algorithm, the energy-efficient and reliability aware workflow task scheduling in cloud environment (EERS) algorithm, which conserves energy while making the most of the system reliable. To begin, use a task-rank-calculation programme to preserve task dependencies. Following that, a task cluster-algorithm is used to reduce communication costs, thereby lowering energy consumption.

In [7] the introduced program is a crossbred of the ALO (ant-lion-optimizer) algorithm along with a Cosine- Sine Algorithm (CSA) algorithm and applied this to multi-empirical to resolve the issues of scheduling scientific systems. Originality of the introduced program was to improve the exploration performance through using arbitrary figures in-accordance to Chaos Theory which is based on the green cloud computing environment and making algorithms greedy. The aim is to increase throughput and at the same time reduce the performance-task-cost, makespan, minimize the consumption of energy in order to have a green cloud environment.

In [8] this paper introduces a new task scheduling algorithm based on the Whale optimization algorithm, whose goal is to schedule tasks on appropriate VMSs based on the calculation of Task and VM priorities, as well as to reduce data-center power costs and energy consumption. First, the priorities for tasks and VMs are calculated in order to effectively map tasks onto VMs and thus evaluate the multi-objective fitness function that addresses energy consumption and power cost at datacenters. In [9] a task scheduling technique for CC based on SLnO (Sea Lion Optimization) and a multiple-objective model is proposed in this paper. It reduces overall completion time, cost, and power consumption while maximising resource utilisation. Based on the simulation results on the tested data, the SLnO scheduler outperformed other state-of-the-art schedulers in terms of makespan, cost, energy consumption, resource utilisation, and degree of imbalance.

In [10] Services offered by the cloud computing are features that try to increase the scalability and performance at the same time, these features are efficient in terms of cost and minimum in terms of maintenance of account which makes the cloud a favoured option when allocation is done dynamic for the allocation of resources. Among the various benefits the cloud provides, the scheduling of task is an important attribute which helps to decrease the cost for operation at the same time increases the performance. In the introduced algorithm, a result is given for optimization by considering the cost and makes span as the main restraints. This is achieved by making use of 2 different algorithm OBL (Opposition-Based-Learning) and LOA (Lion-optimization-algorithm) and thus makes a OLOA (Oppositional-Lion-Optimization-Algorithm).

In [11] the environment of the cloud is made of enormous amount of resources and tasks. Identifying the correct VMs (virtual-machines) for the assignments of available resources in order to finish the requested task is done by the scheduling-of-task algorithm that plays an important role in computing process.

Scheduling of task method is enhanced in terms of makespan and also it tries to reduce the cost expenditures. An effective Cross-over task-scheduler algorithm which can duplicate the behaviour food collecting routine of the crow and that of the cuckoo, hence the name CCSA (Cuckoo-

Crow-Search-Algorithm) has been introduced to enhance the process of scheduling of task.

The preceding literature review does not provide near optimal results. for QoS parameters makespan and energy when considered together. Also the above mentioned works, though improve scheduling performance, they also impose overhead complexity. The proposed Rider Cuckoo Optimization Algorithm (RCOA) considers the makespan and cost parameters for optimizing the task scheduling and resource utilization activity amongst the virtual machines in the cloud environment by taking care of minimizing overhead complexity.

The remaining of this article was organised as follows: The solution framework and problem description are described in Section 3. We present the proposed RCOA scheduler in Section 4. In Section 5, the experimental evaluation and discussions are reported and Section 6 contains the conclusions and future work.

3. Problem with solution framework

The environment for the cloud sets up by the service providers comprise of the VMs (virtual-machines) and PMs (Physical-machines) to give an interface for the public. The users who use the cloud can give their task to the cloud by using the interface. After which the request-manager will manage the received task efficiently and aggregate them. All the available resources of the cloud are updated and maintained by the resource-monitor which contains memory, storage and CPU. The scheduler efficiently does the task-scheduling in the environment of the cloud in such a way which will express the fitness function in a reduced manner. The tasks which are constrained in nature are allocated VM (virtual-machine) in agreement to the latter's process of scheduling performance. After getting the required data from the RSM (Resource-monitor) and RQM (request-manager), the scheduler starts the task-scheduling process. After the required data is got, a choice concerning the assignment of the task to correct VMs (virtual-machine) is done. Every task requires to be assigned to correct VMs. This assignment procedure can be described as fine-tuning process which is performed once the location data of the virtual-machines. This data assists in reducing the host-parameters which include load utilization, total time consumption of energy and migration cost. Here in this, every task that was previously sent by the user are made up of different numbers of disconcertingly instantaneous and self-directed jobs. Every job requires to be performed in a single virtual-machine.

Table 2. Notation used in RCOA algorithm

Symbol	Descriptions
VMs	Virtual-Machines
PMs	Physical-Machines
N_{ij}	Properties i, l < j < K
T_{ij}	Tasks j, 1< i < K
E_m	Increase Energy consumed to maximum level when the VMs performs a given task
P_{ij}	Power consumed by the ith VMs defined under the jth PMs
α , β	Parameters used for controlling
μ_{ij}	Properties used by the ith VMs defined under the jth PMs
$Memory_{ij}$	Total Memory Capacity
CPU_{ij}	Total CPU Capacity.
PC	Capacity of the Processor
TL	Length of one particular Task
ET_{ij}	Time taken for Execution on the ith VMs defined under the jth PMs

Let a set of cloud PMs (physical-machine) be represented $PM = \{PM_1, PM_2, PM_3, \dots PM_N\}$ set of VMs (virtual-machine) be represented $VM = \{VM_1, VM_2, VM_3, \dots VM_N\}$ and set of tasks is represented by$T = \{T_1, T_2, T_3, \dots T_N\}$.

Thus, the objective function and be expressed as,

$$Objective function = \sum_{i=1}^{m} T_i \begin{pmatrix} \alpha \cdot Makespan + \\ \beta \cdot Energy \end{pmatrix} \quad (1)$$

Time for execution: The time for execution for a completion of task by the task-scheduler algorithm is dependent on the processing capacity and length of the task. Therefore, the time for execution can be expressed as:

$$Ei = Task\ Size\ /Computational\ Capacity\ Processor) \quad (2)$$

Here, CCP represents the Computation capacity Process, Ts denotes the size of the task, and E_i represents the time for execution.

Energy Consumption: For calculating the performance of the data present in the cloud, the total consumption of the energy in executing a bunch of task T is considered to be an important metric. Consumption of energy can be associated to the memory as the amount of memory that are used uniformly growths so is the usage of energy. In this literature, the consumption of energy parameters requires to be reduced and can be expressed by using the formula Eq. (14).

$$Energy\ Consumption = \frac{1}{(Physical\ machine)X\ (Virtual\ Machine)} \left[\sum_{i=1}^{PM}\ \sum_{j=1}^{VM}\ P_{ij}E_{max} + \left(i - P_{ij}\right)\mu_{ji}E_{max}\right] \quad (3)$$

Here, and μ_{ij} denotes the resource used by the ith VMs defined under the jth PMs, E_{max} denoted the highest energy consumed by the VMs during the execution of a particular task, and P_{ij} denotes the power used by the ith VMs defined under jth PMs. The value of μ_{ij} can be calculated using the formula mentioned below:

$$\mu_{ij} = \frac{1}{2}\left[\left(\frac{CPU\ Utilized}{CPU_{ij}}\right)\right] + \left[\left(\frac{Memory\ Use}{Memory_{ij}}\right)\right] \quad (4)$$

4. ROA-rider optimization algorithm

The [14] ROA program is a part of the Algorithm-for-Optimization class that was established by D Binu and this algorithm gained its inspiration from the riders riding the race. Here are the mainly 4 distinct class of riders, that is, Fr (Follower), Br (Bypass), Ar (Attacker) and Or (Over-taker). The idea used in POA is shown as follows:

Initialization: Initialization in ROA happens for all the 4 classes *Fr*, *Br*, *Ar* and *Or* which are shown as *Gi* and the scientific method used in group-initialization is shown in Eq. (5), where the quantity of riders *Ri* is similar to *Gi*, the quantity of dimensions of uth riders and their locations or coordinates at a particular instance of time are shown by making using the symbols E^t and Q^i (u,v) correspondingly. The total number of riders is calculated as the summation of riders by applying the Eq. (6). Moreover, along with the group-initialization, are parameters such as accelerator, brake and steering are also initialized. The formula used for *Sa* (Steering-Angle) at that instance of time during the rider's vehicle is represented in Eq. (7).

$$E^t = \{E^t(u,v)\}; 1 \le u \le Ri; 1 \le v \le Qi \quad (5)$$

$$Ri = Bypass + follower + overtaker + attacker \quad (6)$$

$$Sa = \left\{Sa_{u,v}^t\right\}; 1 \le u \le Ri; 1 \le v \le Qi \quad (7)$$

Success rate evaluation: The *Sr* (Success-rate) regarding the distance amid the location of the target and the rider is calculated using the Eq. (8). The location of the target and the location of the rider are denoted by To^{Sa} and E^u correspondingly. Based on the *Sr* the leading rider is selected. The highest accuracy rate is obtained by the rider where the distance between the locations is low.

$$Sr = \frac{1}{\left\|E^u - To^{Sa}\right\|} \quad (8)$$

Updating the position of the Rider: The leading rider is calculated by making use of the position update process which is dependent on the rider's position in each set.

i. **Bypass-updating the position of the Rider:** The Bypass-Riders are the 1st class set of riders who have reached the goal position without following other riders. The formula applied for the calculation for the position of the bypass of the riders is shown in Eq. (9), where δ and β are arbitrary values, whose range varies from 1 to 0. In addition to this λ and χ are also arbitrary values, whose range varies from *Ri* to 1.

$$E_{Br}^{t+1}(u,v) = \beta[E^t(x,v) \times \delta(v) + E^t(\lambda,v) \times [1 - \delta(v)]] \quad (9)$$

ii. **Update process for Follower-Rider:** The Follower-Riders trails the track followed by the *Br* in order to arrive at the goal much quicker and this is calculated using the Eq. (10). The Index-leading of the Rider, Selector-Coordinates and the location of the Leading-Rider are shown as b and E^{To}. The other terms like $Sa_{u,b}^t$, to used, are to manifest the angle of steering at the instance of time hand-off.

$$E_{Fr}^{t+1}(u,b) = E^{To}(To,b) + \left[\cos\left(Sa_{u,b}^t\right) \times E^{To}(To,b) \times di_u^t\right] \quad (10)$$

iii. **Update process for the Attacker-Riders:** The Attacker-Rider type try to dominate the position of the Leading-Rider by covering the similar path as the leading-rider does. The Attacker-Rider location is updated by applying the Eq. (12), where the location of the distance that needs to be covered by *uth* in terms of coordinates b and leading-rider position are symbolized as $E^{To}(To,v)$ and di_u^t. The rate of success for every rider is calculated once the location is updated, but ion order to calculate the efficient optimum solution, it is very important to update the rider arguments.

$$E_{Ar}^{t+1}(u,v) = E^{To}(To,v) + \left[\cos\left(Sa_{u,v}^{t+1}\right) \times E^{To}(To,v)\right] + di_u^t \quad (11)$$

Fig. 1 depicts the overall architecture of the proposed RCOA algorithm for efficient task scheduling. The Task Manager receives the tasks

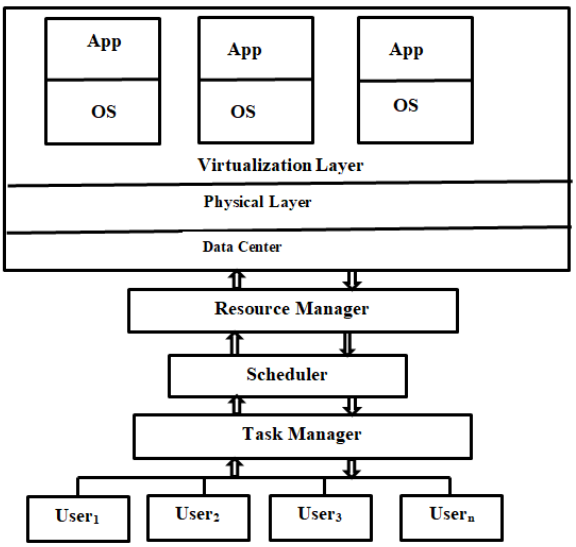

Figure. 1 Architectural structure for the introduced RCOA

submitted by the users. The Task Manager component forwards these tasks to the Scheduler component, which schedules them based on their fitness function. The Resource component monitors virtual machine utilisation in relation to the QoS parameters energy and makespan. The proposed RCOA Algorithm optimises task scheduling by reducing energy and time. [12-15]

4.1 Proposed RCOA scheduler algorithm:

Step 1: Begin
Step 2: Population Initialization: N host nest (X_i=1, 2, n)
Step 3: While t< Max^t(Maximum Iteration)
Step 4: With the help of levy flights, obtain the cuckoo (say i) randomly
Step 5: The fitness function Fit_i is evaluated
Step 6: Randomly select the nest among n(say j)
Step 7: If(ε>0.5)
Update the solution with attacker update using Eq.(12)
Else
Update the solution with over taker update using Eq.(11)
The current best solutions are found by ranking the solutions
Step 8: End if
Step 9: End While
Step 10: Return the best nest
Step 11: End

5. Results and discussion

Experiments were carried-out in an environment that was simulated using Cloudsim tool along with Java (jdk-1.6). Application arrangement is made up of a Personal-Computer having an operating system of Windows 7 and some of the other features are 64-bit windows 2007 OS, RAM of 4 GB and 2 GHz dual core. The Experiments were carried-out by changing the input-task-number from 50-300. The evaluation for the performance for the introduced RCOA algorithm has been done by comparing the solution obtained in RCOA in terms of make-span and energy parameters with other algorithms like the ROA, CS and PSO. Here CS and PSO have a goal to give optimized result for the scheduling of task action in cloud as it uninterruptedly iterates the exploration-space. In the same way ROA produces optimized result for scheduling of task action by taking into account the energy and make-span as its parameters. As the introduced RCOA tries to give a near-optimum result for the scheduling of task action, its therefore compared with ROA, CS and PSO programs. The different types of parameters, entities and the respective values are also considered during the experiment.

Table 3. In cloud simulator values for every-parameters

Object Category	Parameters	Values
Task-in- Cloudlet	File size	300-5000
	Total tasks	100,500
	Task Length	5000-100000
VMs (Virtual-Machines)	Bandwidth	500-1200
	Number of VM	25,75
	Storage	100000-800000
	MIPS	512-1024
	Memory	512-2048

The effectivities in terms of performance along with the make-span for the introduced RCOA method has been determined by changing the quantity of tasks from 100 and 500. To handle these tasks 25 to 75 quantity of VMs were used for allocation and performance of the task and the calculation is done in 25 iterations. The relative make-span bar-plots for CS, PSO, RCOA and ROA methods are shown in Fig. 2 and 3. Fig. 2 mentioned below shows the bar-plot for 100 tasks executed by making use of 25 VMs. From the graph, it is clear that during the 5th-iteration, values of make-span for CS, PSO, RCOA and ROA are 0.46(s), 0.46(s), 0.44(s) and 0.45(s). The values of make-span during the 10th-iteration for CS, PSO, RCOA and ROA are 0.46(s), 0.47(s), 0.44(s) and 0.45(s). The values of make-span during the 15th-iteration for CS, PSO, RCOA and ROA are 0.47(s), 0.47(s), 0.44(s) and 0.46(s). The values of make- span during the 20th-iteration for CS, PSO, RCOA and ROA are 0.46(s), 0.47(s), 0.44(s) and 0.45(s). At last, the values of

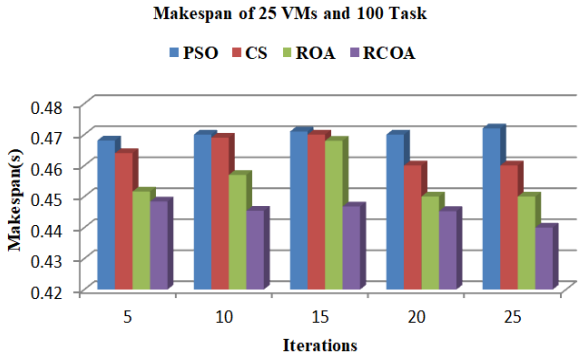

Figure. 2 Using 25 VMs calculating the makespan

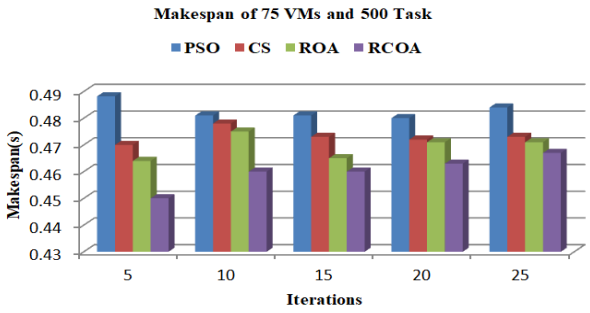

Figure. 3 Using 75 VMs calculating the Makespan

make-span during the 25th-iteration for CS, PSO, RCOA and ROA are 0.46(s), 0.47(s), 0.44(s) and 0.45(s). Thus, from the above obtained data, it is clear that the introduced RCOA method has better effectiveness in terms of make-span-values as compared to the other methods such as CS, PSO and ROA. The proposed RCOA technique had produced an improvement of 5.6%, 4.36% and 2.27% for makespan when compared with PSO, CS, ROA algorithms respectively when used with 25 VMs.

As seen in Fig. 2 and 3 also displays the comparative analysis of make-span bar-plot for CS, PSO, RCOA and ROA methods. Fig. 3 mentioned below shows the bar-plot for 500 tasks executed by making use of 25 VMs. From the graph, it is clear that during the 5th-iteration, values of make-span for CS, PSO, RCOA and ROA are 0.47(s), 0.49(s), 0.45(s) and 0.46(s). The values of make-span during the 10th-iteration for CS, PSO, RCOA and ROA are 0.48(s), 0.48(s), 0.46(s) and 0.48(s). The values of make-span during the 15th-iteration for CS, PSO, RCOA and ROA are 0.47(s), 0.48(s), 0.46(s) and 0.47(s). The values of make-span during the 20th-iteration for CS, PSO, RCOA and ROA are 0.47(s), 0.48(s), 0.46(s) and 0.47(s). At last, the values of make-span during the 25th-iteration for CS, PSO, RCOA and ROA are 0.47(s), 0.48(s), 0.47(s) and 0.47(s). Thus, from the above obtained data, it is clear that the introduced RCOA method has better effectiveness in terms of make-span-values as compared to the other methods such as CS, PSO and ROA The proposed RCOA technique had produced an improvement of 4.9%, 2.86%, and 2% for makespan when compared with PSO, CS, ROA algorithms respectively when used with 75 VMs.

5.1 Comparison of energy

The effectivities in terms of performance along with the consumption of energy for the introduced RCOA method has been determined by changing the quantity of tasks from 100 and 500. To handle these tasks 25 to 75 quantity of VMs were used for allocation and performance of the task and the calculation is done in 25 iterations. The relative make-span bar-plots for CS, PSO, RCOA and ROA methods are shown in Fig. 4 and 5. Fig. 4 mentioned below shows the bar-plot for 100 tasks executed by making use of 25 VMs. From the graph, it is clear that during the 5th-iteration, consumption of energy for CS, PSO, RCOA and ROA are 0.53(KWH), 0.49(KWH), 0.41(KWH) and 0.47(KWH). The consumption of energy during the 10th-iteration for CS, PSO, RCOA and ROA are 0.47(KWH), 0.451(KWH), 0.391(KWH) and 0.48(KWH). The consumption of energy during the 15th-iteration for CS, PSO, RCOA and ROA are 0.47(KWH), 0.45(KWH), 0.44(KWH) and 0.48(KWH). The consumption of energy during the 20th-iteration for CS, PSO, RCOA and ROA are 0.44(KWH), 0.46(KWH), 0.37(KWH) and 0.431(KWH). At last, the consumption of energy during the 25th-iteration for CS, PSO, RCOA and ROA are 0.451(s), 0.464(s), 0.37(s) and 0.44(s). Thus, from the above obtained data, it is clear that the introduced RCOA method has better effectiveness in terms of consumption of energy as compared to the other methods such as CS, PSO and ROA. RCOA had produced energy to the tune of 16.8%, 19.18%, and 16.15% when compared with PSO, CS, ROA algorithms respectively when used with 25 VMs.

As seen in Fig. 4 and 5 also displays the comparative analysis of consumption of energy bar-

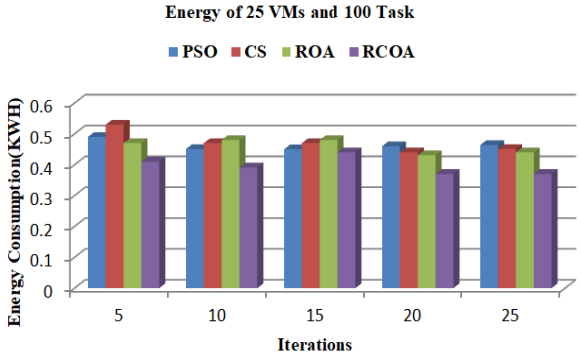

Figure. 4 Using 25 VMs calculating the consumption of energy

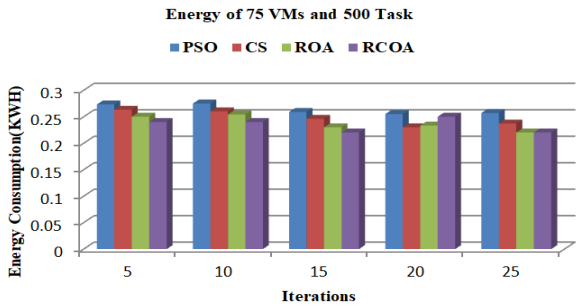

Figure. 5 Using 75 VMs calculating the consumption of energy

plot for CS, PSO, RCOA and ROA methods. Fig. 5 mentioned below shows the bar-plot for 500 tasks executed by making use of 75 VMs. From the graph, it is clear that during the 5th-iteration, consumption of energy for CS, PSO, RCOA and ROA are 0.263(KWH), 0.273(KWH), 0.24(KWH) and 0.25(KWH). The consumption of energy during the 10th-iteration for CS, PSO, RCOA and ROA are 0.26(KWH), 0.275(KWH), 0.24(KWH) and 0.2542(KWH). The consumption of energy during the 15th-iteration for CS, PSO, RCOA and ROA are 0.246(KWH), 0.259(KWH), 0.22(KWH) and 0.23(KWH). The consumption of energy during the 20th-iteration for CS, PSO, RCOA and ROA are 0.23(KWH), 0.255(KWH), 0.25(KWH) and 0.2332(KWH). At last, the consumption of energy during the 25th-iteration for CS, PSO, RCOA and ROA are 0.237(KWH), 0.257(KWH), 0.22(KWH) and 0.22(KWH). Thus, from the above obtained data, it is clear that the introduced RCOA method has better effectiveness in terms of composition of energy as compared to the other methods such as CS, PSO and ROA. RCOA had produced energy to the tune of 12.73%, 5.64%, and 1.48% when compared with PSO, CS, ROA algorithms respectively when used with 75 VMs.

6. Conclusion and future work

In this proposed work, we had designed a hybrid RCOA to address the scheduling of task problem in the cloud-environment. RCOA considers the Quality-of-Service parameters like the energy and make-span to maximize the performance of the scheduling of the task actions. Results obtained by experiments using RCOA have been compared with other algorithm like CS, PSO and ROA. The proposed RCOA technique had produced an improvement of 4.96%, 2.86% and 2% for makespan and had reduced the energy to the tune of 12.73%, 5.64%, and 1.48% when compared with Particle Swarm Optimization (PSO), Cuckoo Search (CS), Rider Optimization Algorithm (ROA) algorithms respectively when used with 75 VMs. As a future-work, additional Quality-of-Service parameters might be considered which can increase the effectiveness and efficiency of the introduced algorithm and at the same time it might also work under real-time situations.

Conflicts of Interest

The authors declare no conflict of interest

Author Contributions

"Conceptualization, Pradeep Krishnadoss; methodology, Pradeep Krishnadoss; software, Pradeep Krishnadoss; validation, Pradeep Krishnadoss; formal analysis, Pradeep Krishnadoss; investigation, Pradeep Krishnadoss; resources, Pradeep Krishnadoss; data curation, Chirag Chandrashekar; writing—Chirag Chandrashekar; writing—review and editing, Chirag Chandrashekar; visualization, Vijayakumar Kedalu Poornachary; supervision, Vijayakumar Kedalu Poornachary".

References

[1] K. Pradeep and T. P. Jacob, "A hybrid approach for task scheduling using the cuckoo and harmony search in cloud computing environment", *Wireless Personal Communications*, Vol. 101, No. 4, pp. 2287-2311, 2018.

[2] M. A. Khan, "A cost-effective power-aware approach for scheduling cloudlets in cloud computing environments", *The Journal of Supercomputing*, Vol. 78, No. 1, pp. 471-496, 2022.

[3] P. Albert and M. Nanjappan, "WHOA: Hybrid Based Task Scheduling in Cloud Computing Environment", *Wireless Personal Communications*, Vol. 121, No. 3, pp. 2327-2345, 2021.

[4] M. Nanjappan, G. Natesan, and P. Krishnadoss, "An adaptive neuro-fuzzy inference system and black widow optimization approach for optimal resource utilization and task scheduling in a cloud environment", *Wireless Personal Communications*, Vol. 121, No. 3, pp. 1891-1916, 2021.

[5] J. K. V. Thekkepuryil, D. P. Suseelan, and P. M. Keerikkattil, "An effective meta-heuristic based multi-objective hybrid optimization method for workflow scheduling in cloud computing environment", *Cluster Computing*, Vol. 24, No. 3, pp. 2367-2384, 2021.

[6] R. Medara and S. R. Singh, "Energy efficient and reliability aware workflow task scheduling in cloud environment", *Wireless Personal Communications*, Vol. 119, No. 2, pp. 1301-1320, 2021.

[7] A. Mohammadzadeh, M. Masdari, F. S. Gharehchopogh, and A. Jafarian, "A hybrid multi-objective metaheuristic optimization algorithm for scientific workflow scheduling", *Cluster Computing*, Vol. 24, No. 2, pp. 1479-1503, 2021.

[8] S. Mangalampalli, S. K. Swain, and V. K. Mangalampalli, "Prioritized Energy Efficient Task Scheduling Algorithm in Cloud Computing Using Whale Optimization Algorithm", *Wireless Personal Communications*, pp. 1-17, 2021, https://doi.org/10.1007/s11277-021-09018-6.

[9] R. Masadeh, N. Alsharman, A. Sharieh, B. A. Mahafzah, and A. Abdulrahman, "Task scheduling on cloud computing based on sea lion optimization algorithm", *International Journal of Web Information Systems*, Vol. 17 No. 2, pp. 99-116, 2021.

[10] P. Krishnadoss and P. Jacob, "OLOA: based task scheduling in heterogeneous clouds", *International Journal of Intelligent Engineering and Systems*, Vol. 12, No. 1, pp. 114-122, 2019, doi: 10.22266/ijies2019.0228.12.

[11] P. Krishnadoss, J. Ali, M. Nanjappan, P. Krishnamoorthy, and V. K. Poornachary, "CCSA: Hybrid cuckoo crow search algorithm for task scheduling in cloud computing", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp. 241-250, 2021, doi: 10.22266/ijies2021.0831.22.

[12] P. Krishnadoss and P. Jacob, "OCSA: Task Scheduling Algorithm in Cloud Computing Environment", *International Journal of Intelligent Engineering and Systems*, Vol. 11, No. 3, pp. 271-279, 2018, doi: 10.22266/ijies2018.0630.29.

[13] K. Pradeep and T. P. Jacob, "CGSA scheduler: A multi-objective-based hybrid approach for task scheduling in cloud environment", *Information Security Journal: A Global Perspective*, Vol. 27, No. 2, pp. 77-91, 2018.

[14] D. Binu and B. S. Kariyappa, "RideNN: A New Rider Optimization Algorithm-Based Neural Network for Fault Diagnosis in Analog Circuits", *IEEE Transactions on Instrumentation and Measurement*, Vol. 68, No. 1, pp. 2-26, 2019.

[15] T. S. Gnanasekar and D. Samiappan, "Impact of hybridized rider optimization with cuckoo search algorithm on optimal VANET routing", *International Journal of Communication Systems*, Vol. 34, No. 16, p. e4954, 2021.