Received: 4 Septemb er 2019 Revised: 11 April 2020 Accepted: 4 May 2020
DOI: 10.1002/dac.446 7

RESE ARCH ARTICL E

WILEY

Crow – peng uin optimizer for multiobjective task scheduli ng strategy in cloud computing

Harvinder Singh [1] | Sanjay Tyagi [2] | Pardeep Kumar [2]

[1]Departm ent of Virtualizati on, School of Computer Science, University of Petroleum and Energy Studies (UPES), Dehradun , Uttarakha nd, India
[2]Departm ent of Computer Science & Applicatio ns, Kurukshetr a University , Kurukshetr a, Haryana, India

Correspon dence
Harvinde r Singh, Assistant Profes sor, University of Petroleum and Energy Studies, Dehradun , India.
Email: harvinderh2 7@gmail.com

Summary

Task scheduling in the cloud is the multiobjecti ve opt imization pro blem, and most of the task sched uling pro blems fail to offer an effective trade-off between the load, resource utilization, makespan, and Quality of Service (QoS). To bring a balance in the trade-of f, this paper proposes a m etho d, termed as crow – pengui n optimi zer for multiobjecti ve task scheduling strategy in cloud computing (CPO-MTS). The proposed algorithm decides the optimal execution of the available tasks in the availabl e cloud resources in minim al time. The proposed algorithm is the fusion of the Cro w Search optimization Algori thm (CSA) and the Penguin Searc h Optim ization Algorithm (PeSOA), and the optima l alloc ation of the tasks depend s o n the newly design ed optimi zation algor ithm. The proposed algor ithm exhibi ts a better convergence rate and converges to the global optimal solut ion rather than the local optima. The formulation of the mult iobjectives aims at a maximum value through attain ing the maximum QoS and re source utilization and minimum load and makespa n, respectively. The experimentati on is performed using three setups, and the analysi s proves that the method attained a better QoS , m akespa n, Resource Utilization Cost (RUC), and load at a rate of 0.4729, 0.0432, 0.0 394, and 0.0298, respectively.

KEYWOR DS

cloud computing , crow search optimiza tion algorithm, multiobj ective functions, penguin search optimiza tion algorithm, task schedulin g

1 | INTRODUCTION

Cloud computing refers to the on-d emand business computing that offers convenience to users by enabling access to the shared configurable resources that are availabl e o n the internet.[1] The performance of the cloud computing networks is en hanced by optimizing the resource allocation of the re sources to various tasks, and the re sources available in the cloud are he terogeneous and geog raphically distributed[2].Thus, the hectic challen ge of cloud computing[3] is regarding the scheduling poli cy. In other words, the challenge regarding the allocation of the tasks is about how the tasks are scheduled to attai n a n optimal system perform ance.[4] Thus , task scheduling is the major process that contributes a lot in the cloud computing te chnology.[5] The role of the effective task schedu ling mec hanism is that it concentrates not only on attaining the requirements of the user but also in enhancing the effici ency of the cloud computing system. The task scheduling strategy is recogn ized as a typi cal NP-h ard problem.[6] Since the task scheduling issue is listed as the NP-hard problem, the time and cost of determining the optimal solution from a huge number of the possible solution are large.[7,8] Hence, the need for the optimal solution is to schedule the current jobs/tasks within the given constraints, and the main constraint is the QoS.[9] Also, there should be a balance between the QoS and the fairness among the tasks.[10,11]

The process of scheduling the task based on QoS is the optimization problem that enables the optimal mapping, which displays the map between the individual task and the available resources and said to have a considerable effect on the QoS of the entire cloud computing systems.[12,13] In addition to this, the workload scheduling in the cloud is an interesting and nourishing research area that aims at the allocation of the essential resources to the tasks to attain the formulated objective function.[14,15] The objective function should be tuned towards the maximum or the minimum value to obtain a particular result, and the parameters employed for accomplishing the objectives include the makespan, execution time, resource utilization, and throughput. The workload management[16] is performed based on the characteristics of the load to balance the workload among the existing servers.[17] Additionally, the load-aware QoS task scheduling assigns the devices to run the tasks based on the certain constraints that assure the efficiency.[18,19]

In the recent decades, the quality of the better solution is attained, and almost all the researchers concentrated in establishing the nature-inspired meta-heuristic algorithms like Ant Colony Optimization (ACO), Simulated Annealing (SA), Particle Swarm Optimization (PSO),[20] Grey Wolf Optimizer,[21] and Genetic Algorithm (GA), for solving the multiobjective workflow scheduling in the cloud.[22] The main aim of the multiobjective workflow scheduling problem is to minimize the makespan and execution cost towards obtaining the objective function.[23] In general, the expense of faster resources is larger than the expense of slower resources. Thus, the meta-heuristics aims at balancing between the makespan and execution cost to optimize the problem of generating the optimal solution.[24] Moreover, to improve the effectiveness of the task scheduling, researchers apply multiple criteria in the optimization problem termed as the multiobjective task scheduling[25] that is a significant topic in the cloud computing that includes multiple parameters, such as makespan, cost, QoS, load, and network parameters.[26]

This paper proposes a CPO algorithm for scheduling the task among the available virtual machines. The cloud environment suffers from a lot of problems, and the main problem is regarding the execution time and the cost of execution. To ensure effectiveness, the multiobjective constraints like makespan, load, resource utilization, and the QoS are formulated. The optimal tuning of scheduling the task is obtained using the proposed CPO algorithm that uses the objective function based on the four constraints. The CPO algorithm is the integration of the CSA and the PeSOA. The proposed CPO schedules the tasks to the virtual machines by minimizing the objective function. The proposed algorithm exhibits the benefits of CSA and PeSOA, and thus, effective task scheduling is enabled.

1.1 | The contributions of the paper

1.1.1 | CPO algorithm

The CPO algorithm is developed by modifying the CSA in which the PeSOA is integrated into the position update step of the CSA. The proposed CPO converges fast to the global optimal solution.

1.1.2 | Objective function based on the multiobjective constraints

The objective function is designed based on the four constraints, such as makespan, resource utilization, load, and the QoS. The CPO algorithm ensures the minimum value of the objective function through minimizing the makespan and load while maximizing the resource utilization and QoS.

The organization of the paper: Section 1 provides the Introduction about the task scheduling and the contribution of the proposed method, and Section 2 discusses the existing methods of task scheduling along with their merits and demerits. Section 3 presents the system model of the cloud computing environment. In Section 4, the proposed CPO algorithm is explained, and Section 5 presents the results and discussion of the proposed method, and finally, Section 6 concludes the paper.

2 | RELATED W ORKS

Hua e t al.[2] proposed a m eth od of task scheduling, named the PSO-based Adapt ive Multiobjectiv e Task Scheduling that aimed at reducing the processing time and the tran smission time. The method offered quasi-optimal solutions in task completi on time, average cost, and energy consumption, but the dynamic depl oyment and service availability are not considered. Dhinesh Babu et al.[1] introdu ced a task scheduling strategy , called the honey bee behavior inspire d load balancing that redu ced the wa iting time o f tasks on queue but failed in scheduling the dependent tasks. Zhan et al.[5] proposed a load aware GA for scheduling the tasks, and the method was found to optimize the makespan and the time required for the load bal ancing simultaneously. However, multiobjecti ve of tasks are not consi dered for scheduling . Zuo et al.[12] presented a task schedu ling mechanism based on the self-adaptive learning PSO, which is capable of handling large-sized constraints but takes huge run time. Shakkeera and Tamil Selvan[14] present ed an app roach, termed as QoS and Load Balancin g Aware (QALBA) approach that sched ules the task using the Enriched -Look ahea d H EFT algorithm (E-LHEFT), which reduced the latency and makespan. However, the challenge is regarding res ource uti lization that depend s o n the size of the task group. Sheng and Li[15] proposed a m eth od, template-base d GA, to schedule the task based on the QoS constraints. The mai n advantage is that the meth od offered less makespan, but the major shortcoming of the method is that the cost factor is not considered. Singh and Chana[23] proposed an efficient cloud workload manageme n t framework that determined the workloads in the cloud for further analysis and clu stering for which the K-means algor ithm is employed. The wei ghts are assigned to the K-means algorithm to meet their QoS requirements. The method is mo re suitable for the heterogeneous cloud workloads, but the miss rate of the task's deadline remai ned high. Zuo et al.[6] proposed a m ethod using ACO for scheduling the task in the cloud. The method determined the optimal solution obtained through the feedback network, but the deadline violation re mained high.

Arunarani et al.[27] provided a compr ehensive survey of task scheduling techniqu es and the associated metrics suitable for cloud computing en vironments. They discussed the issues re lated to schedu ling meth odologies and the limitations to overcome. Keshanchi et al.[28] introduced a powerfu l and improved geneti c algorithm for task scheduling . This algorithm used the advanta ges of the evoluti onary geneti c algor ithm along with heurist ic techniqu es. Kumar and Venkatesan[29] introduced an efficien t task scheduling algorithm. I n this method, user tasks were stored in the queue mana ger. The priority was calculat ed, and suitable resources were allocated for the task if it is a repeated task. New tasks were analyzed and stored in the on-demand queue. The output of the on-demand queue was given to the Hybrid Genetic-P article Swarm Optimization (HGPS O) algorithm, which was deve loped by integrating the geneti c algorithm and PSO algorithm. H GPSO algorithm evaluates suitable resources for the user tasks, which are in the on-demand queue .

Elaziz et al.[30] developed an alternative technique for the cloud task scheduling problem, which aims t o m inim ize makespa n that required sched uling several tasks on differen t V Ms. This metho d was based on the improvement of the Moth Search A lgorithm (MSA) usin g the Diff erential Evolution (DE). Panda and Jana[31] developed an ener gy-efficient task sched uling algorithm (ETSA) to overcome the demerits assoc iated with task consol idation and scheduling . This algorithm takes into acco unt the completio n time and total utilization of a task on the re sources and follows a normali zation procedur e t o m ake a scheduling decision. Also, it pro vided an elegant trade-off between energy efficiency and mak espan.

Previous studies[1,2,5,6,12, 14,5,23,27 – 31] utilize the various optimi zation algorithms for task sched uling. In this work, a new optimization algorithm, named CPO, is developed for task s cheduling with faster conv ergence.

2.1 | Research gaps

The limitations of the existing task scheduling algor ithms are presented below.

- The main aim of a task scheduling strategy is to determine a trade-of f between user re quirements and resource utilization. The main problem is that the tasks that are perform ed by various users are with varying requirements on the computing time, memory space, data traffic, res ponse time, and so on.[2]
- One of the hectic chall enges in the current cloud solutions is regarding the ser vices provided to the users such that the services should yield the expected QoS level offered by the user. Cloud service providers should manage and verify whether a sufficient amount of resources are allot ted to meet the required QoS re quirements of cloud service consumers includi ng the dead line, execution time, energy consump tion, and budget restrictions.[23]

- Additionally, more recent and effective optimization algorithms are needed to solve the multiple objective-based task scheduling since the traditional algorithms are only used in most of the works.

The proposed CPO algorithm tries to solve the challenges of the existing task scheduling algorithms and performs the task scheduling with better QOS, makespan, RUC, and load.

3 | SYSTEM MODEL OF THE CLOUD COMPUTING ENVIRONMENT

Cloud computing is the process of providing multiple services to users with the required QoS based on the pay-for-use provision concept. In addition to the QoS, the cloud service providers should provide services such that the energy is conserved and enables an effective trade-off in QoS, resource efficiency, and energy efficiency. The cloud environment comprises many physical machines with each physical machine connected to some virtual machines. The physical machines allow the resources to the users, which should enable configuring the resources to the users faster and through the energy-cost constraint. The problem is regarding the efficient allocation of the task to the virtual machines such that the load in the network and makespan are reduced and to achieve the customer satisfaction. Let us consider the cloud consists of q number of physical machines that are represented as,

$$p = \{p_1, p_2, \ldots, p_k, \ldots, p_q\} \quad (1)$$

where q is the total number of physical machines present in the cloud, $p_1, p_2, \ldots, p_k, \ldots, p_q$ are the individual physical machines, and p_k is the k^{th} physical machine in the cloud. The total virtual machines corresponding to the k^{th} physical machine is,

$$|V^k| = \{V_1^k, V_2^k, \ldots, V_j^k, \ldots, V_g^k\} \quad (2)$$

where $|V^k|$ represents the total virtual machines corresponding to the k^{th} physical machine, g is the total number of virtual machines, and V_g^k represents the g number of virtual machines corresponding to the k^{th} physical machine. V_j^k is the j^{th} virtual machine of the k^{th} physical machine. Let the total task be notated as T, and the total number of tasks is given as

$$T = \{T_1, T_2, \ldots, T_i, \ldots, T_n\} \quad (3)$$

where T represents the total tasks, T_i denotes the i^{th} task, and n denotes the total number of tasks in the cloud.

4 | PROPOSED METHOD OF MULTIOBJECTIVE TASK SCHEDULING

The proposed method aims at offering satisfaction to the customers through efficiently scheduling the tasks. The proposed method schedules the task based on the proposed CPO algorithm, which allocates the efficient task to be executed in the virtual machine free from load and time lag. The multiple constraints considered for framing the objective function is maximum QoS, maximum resource utilization, minimum makespan, and minimum load. The proposed CPO algorithm allocates the tasks based on the constraints mentioned above such that the effectiveness is enabled in the user – provider environment of the cloud. Figure 1 illustrates the proposed method of task scheduling using the CPO.

In a cloud computing network, the resource parameters used by the physical machines and the virtual machines are listed below. The resource parameters of the physical machines are the bandwidth required for external communication and internal communication. The bandwidth required for enabling the communication among the virtual machines of the same physical machine is termed as the bandwidth of the internal communication, and the bandwidth required for the communication of the virtual machines operating under different physical machines is termed as the bandwidth of the external communication. The resource parameters of the virtual machines are defined by three main factors, such as Million Instruction Per Second (MIPS) denoted as I, processor (P), and the memory (m), which are responsible for the effective task scheduling in the cloud.

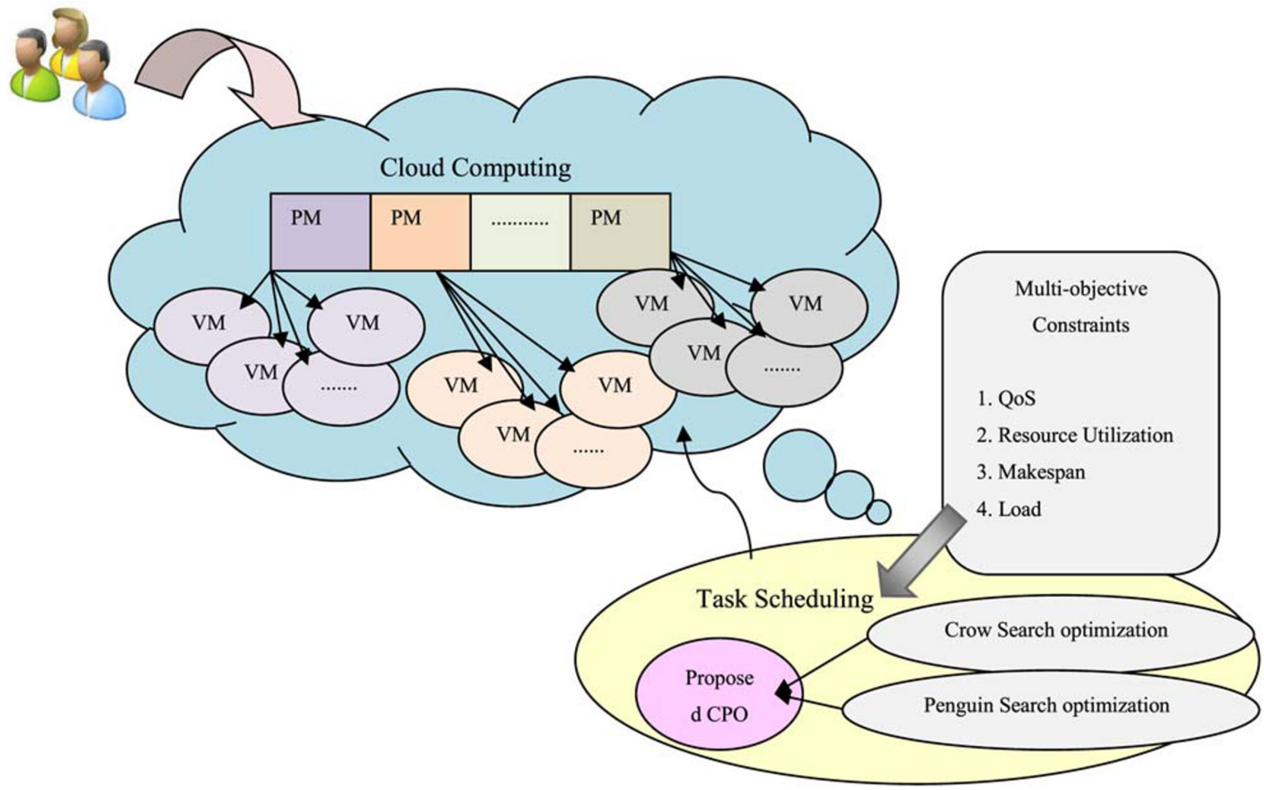


FIGURE 1 Block diagram of the proposed method of task scheduling strategy

4.1 | Multiobjective constraints

This section presents the multiobjective constraints used for designing the objective function. The values of the constraints vary between 0 and 1.

4.1.1 | Makespan

It is the sum of the execution time required to run all the tasks, and the makespan should be minimum.

$$K = \sum_{i=1}^{n} \max E_i \quad (4)$$

where n refers to the total number of the tasks and E_i refers to the execution time of the i^{th} task. The calculation of the execution time is based on the following formula.

$$E_i = \frac{L}{\frac{1}{3} \times \left(\frac{M^a \times I_j}{c_1} + \frac{M^a \times P_j}{c_2} + \frac{M^a \times m_j}{c_3} \right)}; 1 \leq j \leq g \quad (5)$$

where c_1, c_2, and c_3 are the constants. M^a refers to the task assignment matrix that is determined based on the tasks allocated to the virtual machine. The task assignment matrix displays the virtual machine that runs the task. In other words, one can say that the element in the matrix gains the value 1 when the task is executed in the corresponding virtual machine. The data length is denoted as L that denotes the number of tasks waiting in the queue for execution.

$$c_1 = \max_j I_j \quad (6)$$

$$c_2 = \max_j P_j$$

$$c_3 = \max m_j \quad (8)$$

where c_1, c_2, and c_3 are the constants that are calculated as the maximum values of the MIPS, processor, and memory, respectively. I, P, and m correspond to the MIPS, processor, and the memory of the virtual machines.

4.1.2 | Resource utilization

It is determined based on the task time matrix (M^T) and the resource parameters of the virtual machine. The task time matrix (M^T) is computed based on the task assignment matrix (M^a) and the execution time required for executing the task, which is given by,

$$M^T = M^a \times E_i; \quad (9)$$

The resource utilization of the system is given by,

$$R_U = \frac{1}{n \times U_N} \sum_{j=1}^{g} \sum_{i=1}^{n} M^T_{ji} \times \left(\frac{I_j}{c_1} + \frac{P_j}{c_2} + \frac{m_j}{c_3} \right), \quad (10)$$

where U_N refers to the normalized value. The resource utilization should be in maximum while providing service to the customer so that the performance of the system is enhanced. M^T_{ji} indicates the task time matrix of the i^{th} task executed in the j^{th} virtual machine. I_j, P_j, and m_j are the MIPS, processor, and memory of the j^{th} virtual machine. g corresponds to the total number of the virtual machines present in the cloud for allocating resources.

4.1.3 | Load

The load of the cloud depends on the capacity of the cloud and the resource utilized by the system such that the minimum load indicates the improved performance of the cloud.

$$\text{Load} = \frac{R_U}{C}, \quad (11)$$

where C denotes the capacity of the cloud.

$$C = \frac{1}{n} \sum_{j=1}^{g} K \times \left(\frac{I_j}{c_1} + \frac{P_j}{c_2} + \frac{m_j}{c_3} \right); \quad (12)$$

The above formula indicates the capacity of the cloud that depends on the MIPS, processor, and the memory of the virtual machine operating under a physical machine.

4.1.4 | Quality of service

The QoS provided to the customer should be in maximum. The QoS of the system depends on the average of the QoS of the network and the QoS of the system.

$$QoS = \frac{1}{2} \times \left(QoS^{system} + QoS^{network} \right); \quad (13)$$

where QoS^{system} indicates the QoS of the system and $QoS^{network}$ denotes the QoS of the network.

QoS of the network

The QoS of the network depends on the QoS of internal communication and external communication. Internal and external communications are identified based on the physical machines, which execute the task. If the communication occurs in the virtual machines of the same physical machine, it is referred to as internal communication, and if the communication takes place in the virtual machines of different physical machines, it is referred to as external communication. The QoS of the network should be in maximum.

$$QoS^{network} = QoS^{Int} + QoS^{ext}, \quad (14)$$

where QoS^{Int} denotes the QoS of the internal communication and QoS^{ext} represents the QoS of the external communication, formulated as,

$$QoS^{Int} = \sum_{z=1}^{N^{int}} \frac{BW^{Int}}{L^{k}}, \quad (15)$$

$$QoS^{Ext} = \sum_{l=1}^{N^{ext}} \frac{BW^{ext}}{L^{l}}, \quad (16)$$

where BW^{ext} and BW^{int} indicate the bandwidth of external and internal communication and L^k corresponds to the data length of the k number of tasks associated with internal communication. The total number of tasks exhibiting internal communication in the network is represented as N^{int}. L^l implies the data length of the l number of tasks associated with external communication. The QoS of the network depends on the bandwidth of the communication and the length of the task.

QoS of the system

The QoS of the system depends on the QoS of the CPU, memory, and the MIPS.

$$QoS^{system} = \frac{1}{3} \times \left[QoS^{CPU} + QoS^{m} + QoS^{I} \right], \quad (17)$$

where QoS^{CPU}, QoS^{m}, and QoS^{I} refer to the QoS of the CPU, memory, and MIPS. QoS^{CPU} is calculated as the ratio of the CPU utilization to the capacity of the CPU, and it should be a maximum value. The CPU utilization is based on the task time matrix, the processor value, and the capacity of the CPU, which depends on the makespan.

$$QoS^{CPU} = 1 - \left[\frac{U_{CPU}}{C_{CPU}} \right] = \left[1 - \frac{\frac{1}{n} \sum_{j=1}^{c} \sum_{i=1}^{n} M_{ji}^{T} \times \left[\frac{P_j}{c_2} \right]}{\frac{1}{n} \sum_{j=1}^{c} K \times \left[\frac{P_j}{c_2} \right]} \right], \quad (18)$$

where U_{CPU} refers to the CPU utilization factor, K is the makespan, and C_{CPU} is the capacity of the CPU.

$$QoS^{m} = 1 - \left[\frac{U_{mem}}{C_{CPU}} \right] = \left[1 - \frac{\frac{1}{n} \sum_{j=1}^{c} \sum_{i=1}^{n} M_{ji}^{T} \times \left[\frac{m_j}{c_3} \right]}{\frac{1}{n} \sum_{j=1}^{n} K \times \left[\frac{m_j}{c_3} \right]} \right], \quad (19)$$

where U_{mem} is the memory utilization factor. QoS^{m} refers to the QoS of the memory of the system, and it is computed based on the task time matrix and the MIPS value as given in Equation 19. Similarly, Equation 20 depicts the QoS of the MIPS of the system that depends on the MIPS utilized by the task to be executed in the virtual machine.

$$QoS^{I} = 1 - \left[\frac{U_{MIPS}}{C_{MIPS}} \right] = \left[1 - \frac{\frac{1}{n} \sum_{j=1}^{c} \sum_{i=1}^{n} M_{ji}^{T} \times \left[\frac{I_j}{c_1} \right]}{\frac{1}{n} \sum_{j=1}^{c} K \times \left[\frac{I_j}{c_1} \right]} \right], \quad (20)$$

where the capacity and utilization of the MIPS are denoted as U_{MIPS} and C_{MIPS} respectively.

4.2 | Proposed CPO algorithm for scheduling the tasks in the cloud

The proposed CPO algorithm is used for scheduling the tasks in the clo ud. The proposed CPO algorithm is the hybrid i-zation of the CSA and the PeSOA that offers the advantages of both the algorithms. The issues faced by the individual optimization algorithms are regard ing the premature and slow convergence, but these are tackled by using the hybrid i-zation of the meta-heuristics.

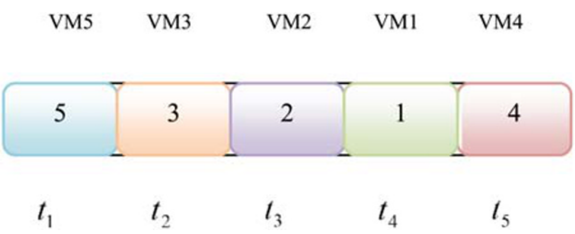
The PeSOA[32] is design ed based on the hunting behav ior of penguins that genera tes the optimal global solut ion through the collabor ative synchro nization in their dives during collectiv e hunting and nutrition processes. The vocal i-zation of the penguins is similar causing the uniqu e identification and re cognition of the pengui ns, and this is the hectic challenge to identify the penguin from the large number of colon ies offering great similarity. The pengui ns are orga-nized in the groups, and the numb er vari es based on the availabi lity of the food in a particular location, and they have the te ndency to hunt in groups and traverse randomly until they possess the oxygen reserves in locating the food. The importa nce of the algorithm is that the global optima l s olution is obtained with bett er convergence. The PeSOA is robust, and it offers a simple balance betwee n the global minima and the other local minima, and the global solution is obtained even when the numb er of penguins is large. It is noted from Gher aibia and Moussaoui[32] that the hybridization of the PeSO A with the other optimization algorithms yields further improv ement in the perf ormance.

The CSA[33] is the populat ion-based optimization algorithm that highl ights the search ing mechanism of the crow s, which live in a flock. The CSA is simple as there are only two adjustable parameters, namely, flight length and aware-ness pro bability, and its importa nce is inher ited in most of the app lications that possess the variable nature of the objec-tive functions, constraint s, and the decision variables. The main character istics of the crow are discussed here. The crow fol lows other crows brilliantly to determine the hidden place of the food, and they sup port their caches from being pilfered that depend on the awareness pro bability. The importance of the awareness probability of the CSA is that the intensifica tion and diversificati on of the search pro cess are controll ed by the awareness probabili ty. When the aw are-ness pro bability takes the low value, the search process of the CSA is perform ed at the local level enabling the intensifi-cation, but when the awareness probability is greater then, the diver sification is enabled. Thus, the meta-heur istic CSA optimization intends to attai n a proper bal ance betwee n diversificati on and intensifica tion. It is clear from Alireza[33] that the convergence rate is good, and the s olution is obtained in around 1 s.

As known from the meta-heuristics, the proposed CPO is good at solving the NP-hard problem and takes very less time to converge to the global optimal solution. The object ive of the proposed algorithm is to minimize the makespan, minimize the networ k load, maximize resource utilizat ion, and maximize the QoS in cloud computing. The proposed optimization algorithm offers a platform for solving real-world optimization problems. The compu tational cost of the method is very low, and this meth od requires less computation. Moreover, this method provides a good trade-off between the convergence and accuracy making the algor ithm to operate with good capacity.

4.2.1 | Solution encoding

The solution encoding aims to represent the solu tion obtained using the proposed CPO. The tasks are optima lly s ched-uled among the available resources. Let us consider that there are five virtual machines and the five tasks $\{t_1, t_2, t_3, t_4, t_5\}$. Figure 2 illustrates the solut ion vector implying that the tasks t_5, t_3, t_2, t_1, and t_4 are execu ted in the virtual machines 1, 2, 3, 4, and 5, respectiv ely. The tasks are alloc ated based on the multiconst raints that aim towards attai ning the opti-mum scheduling. Task 1 is execut ed in the virtual machine 5. Simil arly, the tasks t_2, t_3, t_4, t_5 are executed in virtual machines



FIGU RE 2 Solution encoding of the proposed CPO

4.2.2 | Objective function

The objective function of the proposed CPO is formulated based on Equation 21 depending on the makespan, resource utilization, load, and QoS. The objective function aims at offering the minimum value.

$$F = \frac{\mathrm{Makespan}}{N_{\mathrm{makespan}}} + [1 - R_U] + \mathrm{Load} + [1 - \mathrm{QoS}], \quad (21)$$

where R_U refers to resource utilization.

4.2.3 | Algorithmic steps

The detailed explanation of the proposed CPO algorithm is depicted in . In the first step of the proposed optimization process, the population of the crow is initialized. The population is initialized randomly, and the crow memorizes the location of the hidden layer, and the memory of the crow is updated for the total crow population. The first iteration aims at the selection of the best memory randomly. Whenever a crow tries to change the position for locating the new position of the hidden food, it follows the other crow that insists the action of thievery. During the position update, the feasibility of the new position of the crow is determined, and the new position of the crow is memorized, or the crow returns to its initial position in case of the nonfeasible solution. The objective function is employed to evaluate the fitness of the updated position such that the position that offers the minimum value of the fitness is selected as the best solution. During the update in the new position of the crow based on the proposed CPO, there are two major conditions.

Condition 1: Whenever the position of the randomly chosen v^{th} crow exceeds the awareness probability of the chosen v^{th} crow, then the position update follows Equation 34 to update their positions.

Condition 2: When the position of the randomly chosen v^{th} crow is less than the awareness probability of the v^{th} crow, then the position of the crow is determined randomly. This step happens in the case of the nonfeasible solution. The following are the detailed algorithmic steps of the proposed CPO algorithm.

a Initialization: The initial step is the initialization that initializes the total crow population to determine the best position of the crow. The total population of the crow is given by,

$$W = \{W_1, W_2, \ldots, W_u, \ldots, W_a\}, \quad (22)$$

where, a indicates the total number of crows present in the population and W_u represents the u^{th} crow present in the population. The crows are found distributed in the D dimensional search area. The position of the crows is denoted as,

$$\begin{bmatrix} Y_1^1 & Y_2^1 & \cdots & Y_D^1 \\ Y_1^2 & Y_2^2 & \cdots & Y_D^2 \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ Y_1^a & Y_2^a & \cdots & Y_D^a \end{bmatrix}. \quad (23)$$

The memory of the crow is given by

$$\begin{bmatrix} l_1^1 & l_2^1 & \cdots & l_D^1 \\ l_1^2 & l_2^2 & \cdots & l_D^2 \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ l_1^a & l_2^a & \cdots & l_D^a \end{bmatrix}. \quad (24)$$

During the initial step of the optimization process, the crow has no knowledge regarding the position of the hidden food, and therefore, the position of the hidden food is assumed to be in their initial positions. All the crows memorize the position of the hidden food and the memory of the hidden food for all the crow present in the flock is formulated based on Equation 24.

b Evaluation of the objective function: The optimal selection of the best crow is based on the fitness measure that depends on the makespan, resource utilization, load, and the QoS. The fitness is said to be in minimum, and for attaining the minimum value of the fitness, the makespan should be in minimum, resource utilization should be in maximum, and load should be in minimum with maximized QoS. The designed objective function of the paper is shown in Equation 21.

c Obtain the new position of the crow using the proposed CPO algorithm: The position of the crow is updated based on the proposed CPO algorithm that comprises two conditions. Let us denote the predecessor crow as v, and the u^{th} crow follows the v^{th} crow to update the position. It is necessary to validate the feasibility of the newly generated positions such that if the position is feasible, the position is updated based on Equation 39. In the case of a nonfeasible solution, the crow sustains in its initial position and undergoes a random search. The range of the random number varies between 0 and 1. The position update equation of the proposed CPO algorithm is designed as shown in Equations 25 to 39 such that the newly developed position update equation of the proposed CPO converges quickly to the global optimal solution and memorizes the optimal position for searching the hidden food in the forthcoming search mechanisms. The position update in the CPO is as follows:

The position update equation of the CSA is given as,

$$Y^{u,t+1} = Y^{u,t} + r_u \times f^{u,t} \times \left(x^{v,t} - Y^{u,t}\right) \quad (25)$$

In Equation 25, r_u denotes the random number of the u^{th} crow, and its value varies in the value range of $0 \le r_u \le 1$, and $f^{u,t}$ implies the flight length of the u^{th} crow, $Y^{u,t}$ indicates the position of the crow in the current iteration or the initial position of the crow in case of the first iteration. $x^{v,t}$ indicates the memory of the v^{th} crow.

The position update of the PeSOA is given in Equation 26:

$$D_{new} = D_{LastLast} + rand() \left| X_{LocalBest} - D_{LocalLast} \right| \quad (26)$$

where rand () refers to a random number, $D_{LastLast}$ is the last solution, $X_{LocalBest}$ is the best solution, and $D_{LocalLast}$ is the local last solution. In the proposed algorithm, $D_{LastLast}$ refers to the best position of the crow and is represented as x^*, $X_{LocalBest}$ is inferred as the memory of the v^{th} crow, $x^{v,t}$; and $D_{LocalLast}$ is replaced by the current position of the crow, $Y^{u,t}$. Thus, Equation 27 symbolizes that the position of the u^{th} crow in the next iteration is updated based on the current best position of the crow, current position of the crow, and the memory of the v^{th} crow.

$$Y^{u,t+1} = x^* + r_u \left| x^{v,t} - Y^{u,t} \right| \quad (27)$$

where r_u refers to the random number.

In the hybridization of the PeSOA and the CSA, there is an assumption that when the memory of v^{th} crow is better when compared with the position of the u^{th} crow, then the absolute value is not essential. This assumption is described in Equation 28.

$$\text{If } x^{v,t} > Y^{u,t} \text{ then, } \left| x^{v,t} - Y^{u,t} \right| = x^{v,t} - Y^{u,t} \quad (28)$$

Thus, the absolute value $\left| x^{v,t} - Y^{u,t} \right|$ in Equation 26 is substituted as $\left(x^{v,t} - Y^{u,t}\right)$. Substituting (28) in (27) gives,

$$Y^{u,t+1} = x^* + r_u x^{v,t} - r_u Y^{u,t} \quad (29)$$

$$r_u Y^{u,t} = x^* + r_u x^{v,t} - Y^{u,t+1} \quad (30)$$

$$Y^{u,t} = \frac{1}{r_u} \times \left[x^{?} + r_u x^{v,t} - Y^{u,t+1} \right] \quad (31)$$

Substituting Equa tion 31 in equation 25, we get,

$$Y^{u,t+1} = \frac{1}{r_u} \times \left[Y^{?} + r_u x^{v,t} - Y^{u,t+1} \right] + r_u \times f^{u,t} \times \left[x^{v,t} - \frac{1}{r_u} \times \left[Y^{?} + r_u x^{v,t} - Y^{u,t+1} \right] \right], \quad (32)$$

$$Y^{u,t+1} = \frac{1}{r_u} \times (Y)^{?} + r_u x^{v,t} \frac{1}{r_u} \times Y^{u,t+1} + r_u \times f^{u,t} \times x^{v,t} - r_u \times f^{u,t} \frac{1}{r_u} \left[Y^{?} + r_u x^{v,t} - Y^{u,t+1} \right], \quad (33)$$

$$Y^{u,t+1} = \frac{1}{r_u} \times (Y)^{?} + r_u x^{v,t} \frac{1}{r_u} \times Y^{u,t+1} + r_u \times f^{u,t} \times x^{v,t} - \frac{r_u \times f^{u,t}}{r_u} (Y)^{?} + r_u x^{v,t} \frac{r_u \times f^{u,t}}{r_u} Y^{u,t+1}, \quad (34)$$

$$Y^{u,t+1} = Y^{u,t+1} \left[\frac{r_u \times f^{u,t}}{r_u} - \frac{1}{r_u} \right] + \frac{1 - r_u \times f^{u,t}}{r_u} (Y)^{?} + r_u x^{v,t} + r_u \times f^{u,t} \times x^{v,t}; \quad (35)$$

$$Y^{u,t+1} - Y^{u,t+1} \left[\frac{r_u \times f^{u,t} - 1}{r_u} \right] = \frac{1 - r_u \times f^{u,t}}{r_u} (Y)^{?} + r_u x^{v,t} + r_u \times f^{u,t} \times x^{v,t}; \quad (36)$$

$$Y^{u,t+1} \left[1 - \frac{r_u \times f^{u,t} - 1}{r_u} \right] = \frac{1 - r_u \times f^{u,t}}{r_u} (Y)^{?} + r_u x^{v,t} + r_u \times f^{u,t} \times x^{v,t}; \quad (37)$$

$$Y^{u,t+1} \left[\frac{r_u - r_u \times f^{u,t} - 1}{r_u} \right] = \frac{1 - r_u \times f^{u,t}}{r_u} (Y)^{?} + r_u x^{v,t} + r_u \times f^{u,t} \times x^{v,t}; \quad (38)$$

$$Y^{u,t+1} = \left[\frac{r_u}{r_u - r_u \times f^{u,t} - 1} \right] \left[\frac{1 - r_u \times f^{u,t}}{r_u} (Y)^{?} + r_u x^{v,t} + r_u \times f^{u,t} \times x^{v,t} \right]; \quad (39)$$

where $Y^{u,t+1}$ is the position of the u^{th} crow in the next iteration.

Equation 39 is the position update equation introdu ced using the pro posed CPO algorithm that determines the position of the u^{th} crow when the random numb e r o f the v^{th} crow (R_v) exceeds the awareness probabili ty of the v^{th} crow (AP_v^t). In case, if the random number of the v^{th} crow remains less than the awareness probability of the v^{th} crow, then the position update of the crow is perform ed through random search, and in this cond ition, the initial position of the crow is retained as the position of the crow instead of going for the new position update.

d Compute the objective funct ion for the newly generated position of the crows: The object ive funct ion is verifie d for the newly updated position of the u^{th} crow for the selection of the best solution.

e U pdate the memory of the crow: The memory of the newly updated position is updated using the formula shown below. The main function of memorizi ng is to keep the record of the availabili ty of the hidde n food for under going future search mechanism s.

$$l^{u,t+1} = \begin{cases} Y^{u,t+1}; & \text{fit}(Y^{u,t+1}) \text{ is better than fit } (Y^{u,t}) \\ Y^{u,t}; & \text{Other wise} \end{cases} \quad (40)$$

The position of the u^{th} crow is updated only when the fitness of the new position is better than the initial position of the u^{th} crow. This est imation is based on fitne ss, and the crow updates its position only when the fitness attain s a minimum value. Only the position corresponding to the minim um valu e o f the fitness is inf erred as the new position and proceeds with the position update, which is memorized, or the crow drops the position update and remains in its initial position when the fitness of the newly determined position is greater than the initial position of the crow and memorizes the initial position of the crow.

f Stopping criterion: The steps are iterated continuously from the position update until the memory update until the maximum iterations are reached. Finally, the best position of the crow is determined depending on the objective function.

Algorithm 1

Pseudocode of the proposed CPO algorithm

Proposed CPO algorithm

Input: Crow population

Output: $Y^{u,T+1}$ best position of the crow (optimal task schedule)

Begin
1 Initialize the population
2 Calculate the objective function using equation 21
3 $(R_v > AP_{v,t})$
4 {
5 Update the position based on the equation 39 using the proposed CPO algorithm
6 Else
7 Update the position based on the random search
8 }
End if
9 Compute the objective function of the new position of the u^{th} crow
10 If fit $(Y^{u,T+1}) >$ fit $(Y^{u,T})$
11 {
12 Update the memory of the crow
13 Else
14 Remain in the initial position, $Y^{u,T}$
15 }
16 Repeat steps until $(T+1) \leq T$
17 End

5 | RESULTS AND DISCUSSION

This section presents the results and the discussion of the proposed method and provides an elaborate comparison with the existing method to prove the superiority of the proposed method.

5.1 | Experimental setup

The experimentation of the proposed technique of task scheduling in the cloud is done in the system with 2-GB RAM, Intel core processor, and Windows 10 Operating System. The proposed task scheduling mechanism is implemented using cloudsim tool with JAVA. The experimentation is carried out using three setups. The setups use two physical machines and three virtual machines, and the analysis is carried out based on the task size.

5.1.1 | Input parameters

Table 1 shows the parameters used for experimentati on and the correspond ing values.

5.2 | Competing methods

The existi ng algorithms e mployed for comparis on include CSA,[33] PSO,[12] Artificial-Be e Colony (ABC),[1] GA,[5] ACO,[6] and PeSOA.[32] The existing methods are compared with other existing methods for proving their effectiveness.

5.3 | Performance metrics

The metrics used for the experi mentation are QoS, RUC, Makespan, and Load, which are explai ned in Sectio n 4.1.

$$RUC = 1 - R_U \quad (41)$$

Based on these four measures, the analysis is carried out with competing methods.

5.4 | Comparative analysis

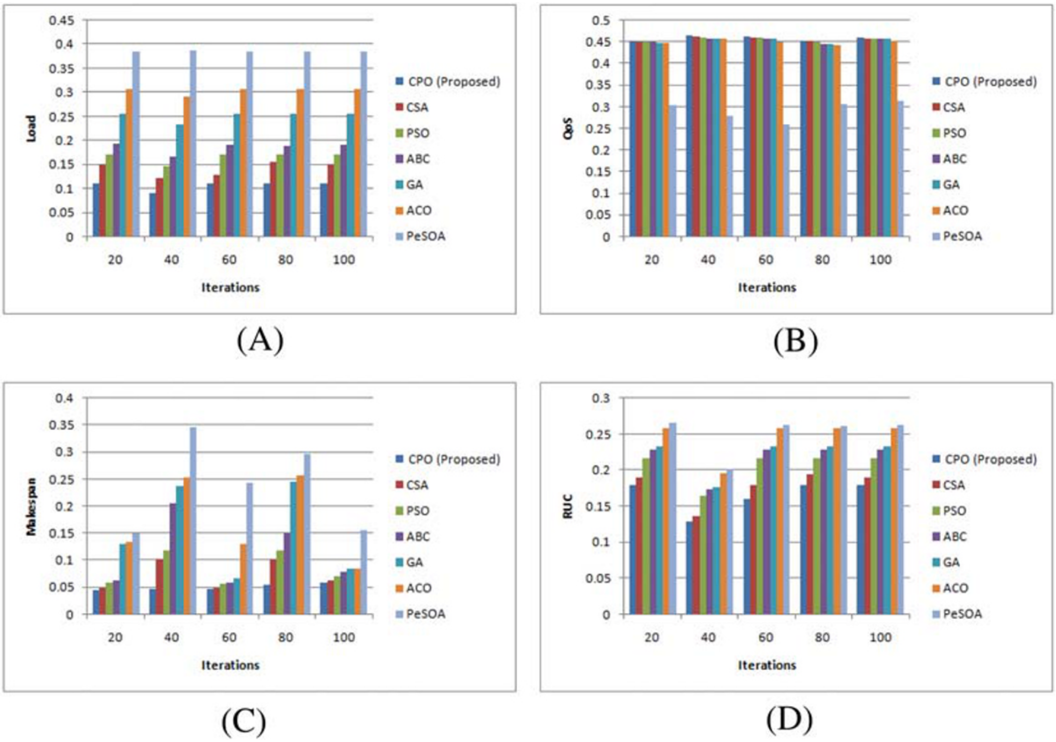
In this section, the compara tive analysis of the proposed method is provided, and the analysis is performed using three setups.

5.4.1 | Setup 1 w ith the task size as 100

For an effective method, the load should be in minimum . Figure 3 shows the comparative analysis of the proposed method and other existing optimization methods in terms o f the performance metrics. Figure 3A shows the analysis in terms o f the load . The load of the proposed CPO meth od is minim ized when compared with the existing methods, like CSA, PSO, ABC, GA, ACO, and PeSO A. With the increasin g number of iterati ons, the load valu e decreases. At the maximum iterati on of 100, the algorithms like CPO, CSA, PSO, ABC, GA, ACO, and PeSO A are 0.1 097, 0.1508, 0.1706, 0.19, 0.256, 0.3 072, and 0.384, respectively. Note that the proposed method offered a m inim um value for the load. Figure 3B shows the analysis in term s o f the QoS. The QoS of the proposed CPO method is in maximu m when compared with the existing methods CSA , PSO, A BC, GA, ACO, and PeSOA. At the maximu m iteration of 100, CPO, CSA, PSO, ABC, GA ACO, and PeSOA offered maximum QoS at a rate of 0.4604, 0.4582, 0.4577, 0.4575, 0.4574, 0.4 531, 0.3131, respectively. Note that the proposed metho d offered a m aximum value for the QoS. Figure 3C shows the analysis in terms o f the makespa n. The makespan of the proposed CPO method is in minim um when compared with the existing methods. At the maximu m iteration of 100, the algorithms CPO, CSA, PSO, ABC, GA, ACO, and PeSOA offered minim um makespa n o f 0.0 584, 0.0625, 0.0703, 0.0768, 0.0 826, 0.0828, and 0.1558. Figure 3D shows the analysis in te rms of the

TABLE 1 Parameter description

Type	Parameters	Value
Datacenter	Number of physical machines	70
	Number of virtual machines	100
Virtual machine	MIPS	5000 – 15 000
	Processor	1 – 100
	Memory	1 – 100
Task	Total number of tasks	300
	Length of task	0 – 1



FIGU RE 3 Comparative analysis using the setup 1: (A) load, (B) QoS, (C) makespan, (D) RUC

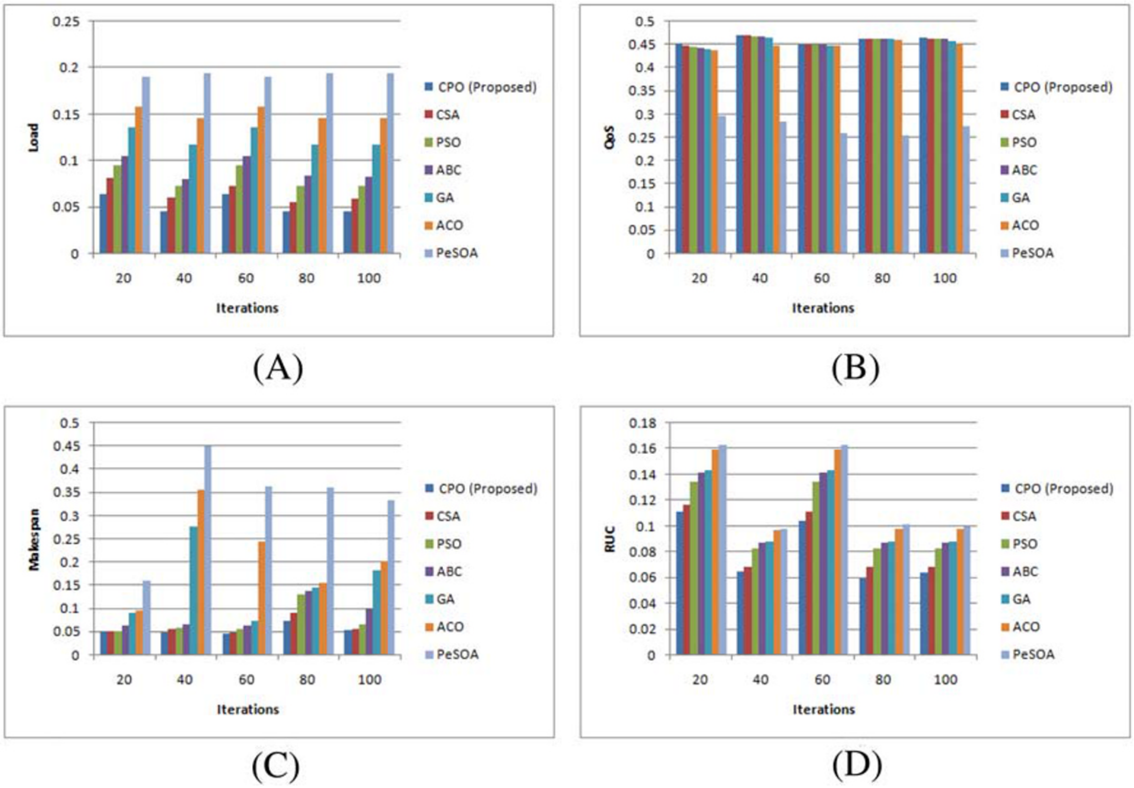
RUC. The RUC of the proposed CPO metho d is m inim um than that of the existing metho ds. At the maximum iterati on of 100, the algorithms CPO, CSA, PSO, ABC , GA, ACO, and PeSOA offered the RUC at a rate of 0.1793, 0.1889, 0.217, 0.2291, 0.2323, 0.2577, and 0.2632, re spectively.

5.4.2 | Setup 2 w ith the task size as 200

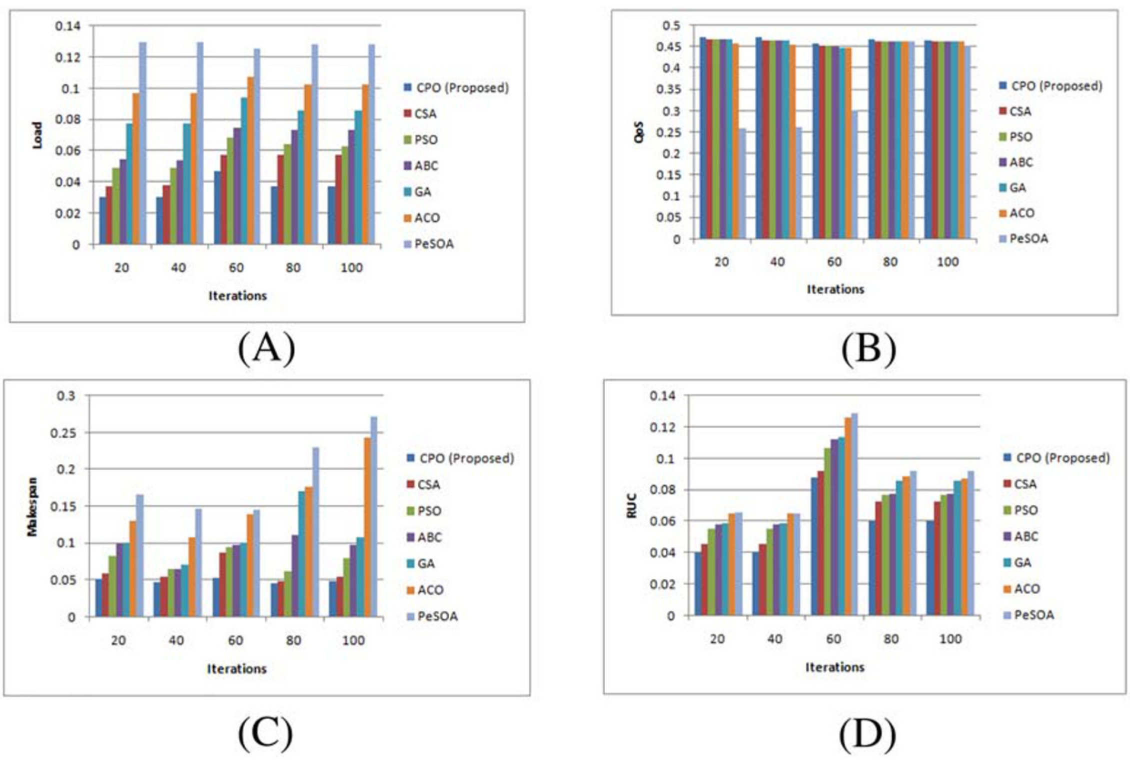
Figure 4A s hows the analysis in term s of the load. The load of the proposed CPO met hod has to be in minimum when compared with the existing methods CSA, PSO, ABC, GA, ACO, and PeSOA. With the increasing number of iterations, the load value decreases. At the maximu m iteration of 100, CPO, CSA, PSO, ABC , GA, ACO, and PeSOA are 0.045, 0.059, 0.0727, 0.0 823, 0.1 164, 0.1 455, and 0.194, respectively. Figure 4B shows the analysis in terms of the QoS. The QoS of the pr oposed CPO method is in maximum. At the maximum iteration of 100, the algor ithms like proposed CPO, CSA, PSO, ABC, GA, ACO, and PeSOA offered maximum QoS of 0.4645, 0.4 628, 0.4624, 0.4620, 0.4585, 0.4535, and 0.2745, respect ively. Figure 4C shows the analysi s in terms of the mak espan. The mak espan of the proposed CPO method is in minimum when compared with the existing meth ods. At the maximum iteration of 100, the algor ithms CPO, CSA, PSO, ABC , GA, ACO, and PeSOA offered minim um mak espan at a rate of 0.0523, 0.0543, 0.0649, 0.0998, 0.1820, 0.2022, and 0.3 334, respectiv ely. Figure 4D shows the analysis in terms of the RUC. The RUC of the proposed CPO method is in minimum when compared with the existing metho ds. At the maximum iteration of 100, the algorithms CPO, CSA, PSO, ABC , GA, ACO, and PeSO A offered RUC of 0.0 634, 0.0 679, 0.0 822, 0.0 868, 0.0 88, 0.0976, and 0.0997, respectively.

5.4.3 | Setup 3 w ith the task size as 300

Figure 5A shows the analysi s in terms of the load. The load of the proposed CPO method is in minimum when compared with the existing methods, such as CSA, PSO, A BC, GA, ACO, and PeSOA algorithms. With the increasing number of iterations, the load value decr eases. At the maximum iteration of 100, the algorithms CPO, CSA, PSO, ABC, GA, ACO,



FIGU



FIGU RE 5 Comparative analysis using the setup 3: (A) load, (B) QoS, (C) makespan, (D) RUC

TABLE 2 Comparative discussion of the proposed method

Methods	Load	QoS	Makespan	RUC	Computational time (s)
CPO (proposed)	0.0298	0.4729	0.0432	0.0394	2
CSA	0.0367	0.4696	0.0465	0.0453	3
PSO	0.0485	0.4683	0.0496	0.0548	3.5
ABC	0.0537	0.4681	0.0577	0.0578	3
GA	0.0776	0.4665	0.0667	0.0587	4
ACO	0.097	0.4618	0.0828	0.0651	4.5
PeSOA	0.128	0.4617	0.1452	0.0651	4

in terms o f the QoS. The QoS of the proposed CPO method is in maximu m At the maximum iteration of 100, the algorithms CPO, CSA , PSO, A BC, GA, ACO, and PeSOA offered maximu m QoS of 0.4654, 0.4 629, 0.4624, 0.4623, 0.4621, 0.4618, and 0.453, respectively. Figure 5C shows the analysis in term s o f the mak espan. The mak espan of the proposed CPO metho d i s i n m inim um when compared with the existi ng methods CSA, PSO, ABC, GA, ACO, and PeSOA. At the maximum iteration of 100, the algor ithms CPO, CSA, PSO, ABC, GA, ACO, and PeSOA offered minim um makespan of 0.0471, 0.0544, 0.0784, 0.0964, 0.1 073, 0.2 433, and 0.2723. Figure 5D shows the analysis in terms o f the RUC. At the maximum iteration of 100, the algorithms CPO, CSA, PSO, ABC , GA, ACO, and PeSOA offered minim um RUC of 0.0597, 0.0723, 0.0764, 0.0774, 0.0858, 0.0868, and 0.0 916.

5.5 | Comparative discussion

From Table 2, it is clear that the proposed meth od acquired a m aximum QoS , wherea s the load, makespa n, and RUC are minimum. The QoS obtained using the proposed method is 0.4729, whereas other existin g m ethods like the CSA, PSO, PSO, ABC , GA, ACO, and PeSOA are found to be 0.4 696, 0.4683, 0.4681, 0.4 665, 0.4618, and 0.4 617, respectively. The RUC o f the proposed CPO is found to be 0.0394, which is a m inimum value when compared with the RUC o f the existing meth ods. The existing methods such as CSA, PSO, PSO, ABC, GA, ACO, and PeSOA attained the RUC of 0.0453, 0.0548, 0.0578, 0.0587, 0.0651, and 0.0651, respectively . The load is found to be 0.0298, 0.0 367, 0.0485, 0.0537, 0.0776, 0.097, and 0.128, respectively, for CPO, CSA, PSO, PSO, ABC, GA, ACO, and PeSOA. The proposed method attained a m inim um makespa n o f 0.0432, while the existing methods such as CSA, PSO, PSO, ABC, GA, ACO, and PeSOA attai ned the makespan of 0.0465, 0.0496, 0.0577, 0.0667, 0.0 828, and 0.1452, res pectively. The compu tational time o f the proposed meth od is 2 s while the existin g m ethods such as CSA, PSO, PSO, ABC, GA, ACO, and PeSOA have the compu tational time o f 3 , 3.5, 3, 4, 4.5, and 4 s , respectiv ely.

6 | CO NCLU SIO N

In this paper , a new algor ithm, t e r med a s the CPO algorithm, has been proposed for scheduling the task among the available re sources. The main intention of the algorithm is to manage the cloud from the existing problems like the execution time and the execution cost. These problems have been tackled by employing multiobjecti ve constrai nts, such as makespa n, load, resource utilizat ion, and QoS. The optima l tuning of the task schedu ling using the proposed CPO algorithm employs the multiobjecti ves. The CPO algorithm, which is the fusion of the CSA and the PeSOA, offers bett er convergence rate. The proposed method exhibits a bett er scheduling mechanism and is found to be bet ter when compared to the other existing algorithms. The proposed algorithm is experimented using three setups consisting of two physical machines and three virtual machines in the cloud area. The analysi s using the setups proves that the proposed method attained a m aximum QoS of 0.4729, minimum RUC a t a rate of 0.0394, minimum makespa n a t a rate of 0.0432, and minim um load at a rate of 0.0298, respectively.

ORCID
Harvinder Singh https://orcid.o rg/0000-0001-982 3-8559

RE FER E NCES

1. Dhinesh Babu LD, Venkat P, Krishna Q. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Appl Soft Comput . May 2013;13(5):2292-2303.
2. Hua H, Guangquan X, Shanchen P, Zenghua Z. AMTS: adaptive multi-objective task scheduling strategy in cloud computing. Chin Commun . April 2016;13(4):162-171.
3. Polepally V, Chatrapati KS. Exponential gravitational search algorithm-based VM migration strategy for load balancing in cloud comp ing. Int J Model Simul Sci Comput . 2018;9(1).
4. Liu Y, Xu X, Lin Z, Wang L, Zhong RY. Workload-based multi-task scheduling in cloud manufacturing. Robot Comput Integr Manuf . June 2017;45:3-20.
5. Zhan ZH, Zhang GY, Lin Y, Gong YJ, Zhang J, " Load balance aware genetic algorithm for task scheduling in cloud computing, " Asia Pacific Conference on Simulated Evolution and Learning, pp. 644-655, 2014.
6. Zuo L, Shu L, Dong S, Zhu C, Hara T. A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing. IEEE Access . December 2015;3:2687-2699.
7. Mao X, Li C, Yan W, " Shumeng Du Optimal Scheduling Algorithm of MapReduce Tasks Based on QoS in the Hybrid Cloud Parallel a Distributed Computing, " 17th International Conference on Applications and Technologies (PDCAT), December 2016.
8. Banerjee S, Mukherjee I, Mahanti P. Cloud computing initiative using modified ant Colony framework. World Acad Sci Eng Technol . 2009;56:221-224.
9. Dutta D, Joshi RC. " A genetic: algorithm approach to cost-based multi-QoS job scheduling in cloud computing environment " , I n Pr ceedings of the International Conference & Workshop on Emerging Trends in Technology, pp. 422-427, ACM, February, 2011.
10. Ali HGEDH, Saroit IA, Kot AM. Grouped tasks scheduling algorithm based on QoS in cloud computing network. Egypt Inf J . March 2017;18(1):11-19.
11. Yu L, Zhang C, Li B, Niu J. DeMS: a hybrid scheme of task scheduling and load balancing in computing clusters. J Netw Comput Ap April 2017;83:213-220.
12. Zuo X, Zhang G, Tan W. Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud. IEEE Trans Autom Sci Eng . April 2014;11(2):564-573.
13. Varalakshmi P, Ramaswamy A, Balasubramanian A, Vijaykumar P. An optimal workflow based scheduling and resource allocation in cloud. In: Advances in Computing and Communications . Berlin, Heidelberg: Springer; 2011:411-420.
14. Shakkeera L, Selvan LT. QoS and load balancing aware task scheduling framework for mobile cloud computing environment. Int J Wir less Mobile Comput . 2016;10(4):309-316.
15. Sheng X, Li Q. " Template-based genetic algorithm for QoS-aware task scheduling in cloud computing, " International Conference on Advanced Cloud and Big Data (CBD), August 2016.
16. Ni L-N, Zhang J-Q, Yan C-G, Jiang C-J. A heuristic algorithm for task scheduling based on mean load on grid. J Comput Sci Technol July 2006;21(4):559-564.
17. Kaur K, Kaur N, Kaur K. A novel context and load-aware family genetic algorithm based task scheduling in cloud computing. Data E Intel Comput . June 2017;521-531.
18. Yu X, Xiea Z-Q, Yang J. A load balance oriented cost efficient scheduling method for parallel tasks. J Netw Comput Appl . March 201 37-46.
19. Li Z, Ge J, Yang H, et al. A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds. Futu Gener Comput Syst . December 2016;65:140-152.
20. Kulkarni YR, Senthil Murugan T. Hybrid weed-particle swarm optimization algorithm and CMixture for data publishing. Multimed R 2019;2(3):33-42.
21. Waykar SB, Bharathi CR. Intent aware optimization for content based lecture video retrieval using grey wolf optimizer. J Eng Res . 20 7(3):123 – 141.
22. Li K, Xu G, Zhao G, Dong Y, Wang D. " Cloud task scheduling based on load balancing ant colony optimization " , I n Proceedings of IEEE Sixth Annual Chinagrid Conference (ChinaGrid), pp. 3 9, 2011.
23. Singh S, Inderveer Chana J. QRSF: QoS-aware resource scheduling framework in cloud computing. Super Comput . 2015;71:241-292.
24. Verma A, Kaushal S. A hybrid multi-objective particle swarm optimization for scientific workflow scheduling. Parallel Comput . February 2017;62:1-19.
25. Lakra AV, Yadav DK. Multi-objective tasks scheduling algorithm for cloud computing throughput optimization. Proc Comput Sci . 20 48:107-113.
26. Zhu L, Li Q, He L. Study on cloud computing resource scheduling strategy based on the ant colony optimization algorithm. IJCSI Int Comput Sci Issues . 2012;9(5):54 – 58.
27. Arunarani AR, Manjula D, Sugumaran V. Task scheduling techniques in cloud computing: a literature survey. Future Gener Comput Syst . February 2019;91:407-415.
28. Keshanchi B, Souri A, Navimipour NJ. An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: formal verification, simulation, and statistical testing. J Syst Softw . February 2017;124:1-21.
29. Senthil Kumar AM, Venkatesan M. Task scheduling in a cloud computing environment using HGPSO algorithm. Clust Comput . Janu 2019;22(S1):2179-2185.

30. Elaziz MA, Xiong S, Jayasena KPN, Li L. Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution. Knowledge-Based Syst . April 2019;169:39-52.
31. Panda SK, Jana PK. An energy-efficient task scheduling algorithm for heterogeneous cloud computing systems. Clust Comput . June 2019;22(2):509-527.
32. Gheraibia Y, Moussaoui A. Penguins Search Optimization Algorithm (PeSOA). In: Ali M, Bosse T, Hindriks KV, Hoogendoorn M, Jonker CM, Treur J, eds. Recent T rends i n A pplied A rtificial I ntelligence. IEA/AIE 2013. Lecture N ote s in Computer Sc ience . Heidel ber g : S pr inger; 20 13 .
33. Askarzadeh A. A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct . 2016;169:1-12.

How t o cite this article: Singh H, Tyagi S, Kuma r P. Crow – penguin opt imizer for mult iobjective task schedu ling strategy in cloud computing. Int J Commun Syst . 2020;e4467. https ://doi.org/10 .1002/dac.4467