Research Article

International Journal of Advanced Technology and Engineering Exploration, Vol 10(105)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2022.10100466

An efficient load balancing in cloud computing using hybrid Harris hawks optimization and cuckoo search algorithm

Alok Kumar Pani[1*], M. Manohar[1], Merin Thomas[2] and Pankaj Kumar[3]
Department of Computer Science and Engineering, CHRIST (Deemed to be University), Bengaluru, India[1]
School of Computer Science, RV University, Bengaluru, India[2]
Department of Computer Science and Engineering, Motihari College of Engineering, Bihar, India[3]

Received: 26-November-2022; Revised: 20-August-2023; Accepted: 22-August-2023

Abstract
Cloud computing has rapidly emerged as a burgeoning research field in recent times. However, despite this growth, a comprehensive examination of this domain reveals persistent issues in the application of cloud-based systems concerning workload distribution. The abundance of resources and virtual machines (VMs) within cloud computing underscores the importance of efficient task allocation as a critical process. Within the infrastructure as a service (IaaS) architecture, load balancing (LB) remains a pivotal but challenging task. The occurrence of overloaded or underloaded hosts/servers during cloud access is undesirable, as it leads to operational delays and system performance degradation. To address LB issues effectively, it is imperative to deploy a proficient access scheduling algorithm capable of distributing tasks across the available resources. A novel approach was introduced by combining the Harris hawk's optimization and cuckoo search algorithm (HHO-CSA), with a specific focus on critical service level agreement (SLA) parameters, particularly deadlines, to uphold LB in a cloud environment. The primary objective of the hybrid HHO-CSA methodology is to provide task attributes, resource allocation, VMs prioritization, and quality of service (QoS) to clients within cloud computing applications. The outcome analysis reveals that the proposed hybrid HHO-CSA algorithm results in a resource utilization reduction of 52%, with an execution time of 529.84 ms and a makespan of 638.88 ms. These values outperform those of existing SLA-based LB algorithms. Effective task scheduling plays a pivotal role in ensuring the seamless execution of tasks within a cloud system, while LB significantly aligns with the SLAs available to users. Drawing insights from the existing literature, the suggested hybrid HHO-CSA method addresses the research gap by effectively mitigating the challenges.

Keywords
Cloud computing, Hybrid Harris hawks optimization-cuckoo search algorithm, Load balancing, Quality of service, Service level agreement, Task scheduling.

1.Introduction
One of the most common services used by people worldwide is cloud computing. Users get immediate access to a variety of computer components or services, including servers, apps, storage, and network. Task planning is crucial in the cloud environment. Applications for users, or tasks at a particular time, are premeditated for specific properties.

The concentration is mostly on minimizing the spectrum of construction and resource utilization.

There are currently a lot of heuristic algorithms available for work schedules. However, to boost performance and increase task planning effectiveness, more adjustments and modifications are required [1]. Cloud computing is an internet-based mechanism for providing end users with on-demand computing resources via virtualization. Elasticity and scalability are two services provided by cloud computing. It benefits from the fact that scheduling an application is a dynamic process that responds to user demand and the condition of virtual machines (VMs) in data centers. The execution performance of scientific applications must be obtained while keeping costs to a minimum, despite some considerable challenges [2]. Users of the cloud can benefit from virtualization and dynamic task-

*Author for correspondence

scheduling tools. Therefore, effective scheduling is significantly increased, to improve the ratio and execution time of resource usage in applications based on cloud [3]. Task scheduling involves giving task deadlines and completion times to cloudlets, and load balancing (LB) involves performing workload movement in the event of VMs violations, to keep the workload balanced in the cloud environment [4]. Task scheduling in the cloud is done by adhering to service level agreement (SLA) norms for customers and cloud developers, and it substantially aids in LB while performing tasks [5]. It doesn't take much time to cache these files because a dataset is regularly uploaded to the cloud environment. Several cloud storages are used to organize data collection amongst cloud-based businesses such as in colleges, banks, and hospitals that have their localized cloud. Several firms reserve the public cloud resources to reduce the cloud and operational costs. Scheduling the work to meet the bare minimal deadline is difficult [6]. A scheduler must employ beneficial ways to deal with the many work types and the changing environment. Task scheduling effectiveness is one of the key difficulties involved in the transfer of duties [7].

The Harris hawk optimizer (HHO) is a metaheuristic method that mimics the social interactions in the field. It is employed in the solution of numerous actual engineering issues, including those involving image segmentation/selection, renewable energies [8], and many others [9]. The suggested technique makes use of the capabilities of direct implications in the continuous-valued subspace and uses random-key encoding to construct a tour while maintaining the fundamental characteristics of the HHO [10]. Solutions are converted from continuous to discrete space using random-key encoding [11]. HHO has many advantages that makes it appealing to researchers to utilize this method [12]. The five levels of cloud computing environment which are hardware / data center, infrastructure, platform, end users and application, are used to illustrate tasks in cloud computing. The network's mathematical model is built to identify the optimization issue and later in order to analyze the usual infrastructure as a service (IaaS) provider's behavior [13]. Here, the method for computing the optimal solution was provided by two objectively optimized solution set cases, which were later contrasted with a linearization method [14, 15]. The research's primary objective was to improve the LB procedure to give effective results. This study's contribution can be summed up as follows

- The cloud's VMs violation problem is addressed by a new LB algorithm name called hybrid Harris hawk's optimization - cuckoo search algorithm (HHO-CSA), which also offers high-quality workload scheduling and balancing services.
- A thorough planning and execution of the hybrid HHO-CSA is demonstrated and contrastively analysed with several already-in-use metaheuristics like ant colony optimization (ACO) and particle swarm optimization (PSO).
- The experimental findings show that the hybrid HHO-CSA performs more effectively in terms of system resource consumption for both smaller and larger workloads on cloud computing.

This paper is structured as follows; section 2 deliberates the existing works; the efficient LB process performed by the proposed hybrid HHO-CSA algorithm is illustrated in section 3. Experimental analysis and its evaluations are presented in section 4, discussion about the overall analysis is explained in section 5 and finally, conclusion of the paper is stated in section 6.

2.Literature review

A LB algorithm for the data centres with SLA has been shown by Shafiq et al. [16]. Task scheduling significantly complies with the standards of SLA, a document made available to consumers by cloud developers, and contributes significantly to LB. The LB algorithm considers crucial SLA criteria like deadlines. Considering the quality of service (QoS) task parameters, the proposed approach aimed to optimize resources and enhance LB. Based on the results of the literature, the suggested SLA-LB algorithm resolved the difficulties raised as well as bridged the existing research gap. However, the makespan is decreased if the proposed technique considers a smaller size.

Zhu et al. [17] designed pile-Hadoop distributed file system (PHDFS) to improve the speed of cloud computing for HDFS input in deep learning. PHDFS offered a transitional file known as the merged file to combine the array of heaps. The PHDFS management module first checked to see if a combined file already existed or not before making a written request to it. In test scheduling, PHDFS performed better than HDFS once it contained smaller file sizes. PHDFS considerably decreased reading latency for tiny files and increased the use of traditional methods. However, because of the sheer volume of files and the small file sizes associated with deep learning datasets, execution of HDFS severely raised the performance costs.

Abualigah and Diabat [18] designed mirjalili antlion optimizer (MALO) to tackle task scheduling issues and have a balanced task distribution in systems of cloud computing. The MALO approach functioned as per the modified basic antlion optimizer that employed differential evaluation algorithm and it was critical to evaluate its output using a global search methodology. MALO performed better than other known algorithms while handling the task scheduling issues. MALO was appropriate for significant scheduling issues because it converged more quickly than the other methods for bigger search spaces. Moreover, when the MALO was compared to other competitive optimization algorithms in a range of tasks, it produced an efficiency span measure and provided better results.

Abualigah and Alkhrabsheh [19] discovered multi-verse optimizer - genetic algorithm (MVO-GA) to improve the efficiency of task transfers via cloud networks based on the workload of cloud resources. MVO-GA was used to plan the transfer task for the load in cloud resources. By using mutation processes and crossover to optimize the initiated activities schedule, the GA enhanced the standard MVO. The MVO-GA approach effectively reduced the transfer time for large cloud workloads and successfully scheduled a larger number of activities, which justified its efficacy. However, to increase the hybrid multiverse optimizer's search capabilities and use evolutionary algorithm, more improvements would be needed in MVO-GA.

Zhang et al. [20] generated an efficient priority and relative distance (EPRD) to reduce the task scheduling time for workflow applications with precedence constraints, while also maintaining the necessity. To meet a limit, EPRD seeked to reduce the scheduling time of directed acyclic grip (DAG) applications using the right VMs instances. To compare time savings with end-to-end deadline constraints, an effective EPRD method was used to evaluate the makespan. In contrast to the scenario where TD = 3.0 TC, EPRD extended the deadline for submissions. When it came to reducing makespan, EPRD worked well. Furthermore, EPRD formalized resource management in cloud computing centers as a combinatorial optimization problem.

Sanaj and Prathap [21] designed a chaotic squirrel search algorithm (CSSA) to provide better multitask scheduling in IaaS environment. The usage of cloud computing in visualization software allows the monitoring of the CSSA process. As the application process is implemented, resources are monitored and handled for the users. In comparison to alternative algorithms for work schedules in a cloud context, the proposed CSSA algorithm lowered the cost by 30% while also enabling great efficiency. However, CSSA still required adjustments to boost productivity and boost task planning efficiency.

Praveenchandar and Tamilarasi [22] presented the dynamic resource allocation (DRA) technique with increased power management and improved task scheduling to increase the effectiveness of the resource allocation procedure. DRA was utilized to distribute resources in response to client requests. The resources had different numbers of VMs available, which were all prepared for distribution by user requests. The DRA provided correct updated values in the resource tables, and effective resource allocation was made possible through better task scheduling mechanism and less power usage strategy. Yet, even when the system was overloaded, there was some inefficiency in task scheduling and energy usage. However, to maximize energy efficiency in the allocation process for effective tasks, the task scheduling algorithm DRA needed to be improved.

Sefati et al. [23] proposed grey wolf optimization algorithm to reliably maintain proper LB. But this method had scalability and security issues. Talaat et al. [24] introduced an effective dynamic LB technique (EDLB) using convolutional neural network and modified PSO. If the server hosting that task had an unexpectedly high demand, it would result in real-time task failure by EDLB which was the major drawback of this work. Singh et al. [25] proposed a fog-cluster-based LB approach along with a refresh period to optimize the use of all the resources in the fog sub-system. Nabi et al. [26] proposed an adaptive PSO-based task scheduling approach for cloud computing to reduce the task execution time, and increase throughput as well as average RU ratio (ARUR). Rana et al. [27] proposed a hybrid whale optimization algorithm (HWOA) with differential evolution (DE) for multi-objective VM scheduling in cloud computing. Gupta et al. [28] proposed an artificial neural network – whale optimization (ANN-WHO) algorithm to improve the fault tolerance of the cloud environment and to facilitate improvement of the system performance at the same time using machine learning techniques. Latchoumi and Parthiban [29] proposed Quasi Oppositional Dragonfly algorithm for LB to achieve optimal resource scheduling in cloud. But this work had certain limitations like weak security measures and low optimization of resources.

Annie and Radhamani [30] proposed an efficient LB scheme with HHO and pigeon inspired optimization (PIO) algorithms. But this algorithm was more complex and had uncontrollable number of tenants. Also, the cost and time increased with the increase in the VMs. Kruekaew and Kimpan [31] proposed a multi-objective artificial bee colony (ABC) q-learning largest job first (MOABCQ_LJF) method for efficient LB tasks. There was uncertainty if the algorithm was optimal and the system's performance could not be optimized in every test dataset. Honey bee foraging behaviour and LB min-min scheduling in cloud computing have been proposed by Thapliyal and Dimri [32]. While the competition for load management solutions has increased due to the increasing expansion of cloud users, cloud storage LB has not been taken into account. The LB techniques described here were adaptable and fault tolerant, but there was still a lot of need for further study in LB. Its algorithm is used in data centers to optimize cloud computing applications. The SLA parameters were not considered in this work, which impacted in optimizing cloud resources. Nazir et al. [33] proposed a framework of LB for cross-region tasks. High-cost time, lack of energy efficiency, were the limitations of this work. Shekhar and Sharvani [34] proposed a multi-tenant LB for cost effective resource allocation. But this method had poor data security management. Hung et al. [35] proposed migration-based LB of VMs using two stage genetic mechanism. But this approach lacked hardware resources. Saif et al. [36] proposed an autonomic chicken swarm optimized inter cloud load balancer (CSO-ILB) to ensure the elasticity of the cloud system and balance the user workload among the available containers in a multi-cloud environment. The experimental analysis observed that the task migration from the containers disrupted the communication flow between the containers of similar hosts, which was a limitation of this work. Abedi et al. [37] developed an improved firefly algorithm (IFA) based on LB optimization to solve DRA problem, hence this development was called IFA-DRA. Adil et al. [38] proposed a novel hybrid approach called content-aware machine learning based LB schedular (CA-MLBS). Task Scheduling was done by advanced phasmatodea population evolution (APPE) algorithm which was presented by Zhang et al. [39]. By enhancing the convergence of the closest optimal solutions, the method reduces the amount of time needed to locate solutions. Additionally, the assessment function aims to identify the ideal solutions by taking the makespan, resource cost, and LB level into account. Al-yarimi et al. [40] had explored the contemporary approach of using the Bollinger Band model for statistical analysis of the load factor or the chosen metric for each of the VM's integral system.

Iqbal et al. [41] proposed enhanced time-constraint aware (TCA) tasks scheduling mechanism based on predictive optimization for efficient LB. The proposed enhanced TCA tasks scheduling mechanism was an improved variant of fair emergency first (FEF) scheduling that considers accurate prediction measures and tasks' optimal time to schedule tasks efficiently. Murad et al. [42] proposed a noble mechanism called optimized min-min (OMin-Min) algorithm, inspired by the Min-Min algorithm. In cloud computing, Bal et al. [43] proposed a combination of resource allocation task scheduling-hybrid machine learning (RATS-HM) technique. A modified workflow scheduling algorithm for cloud computing was proposed by Ahmed and Omara [44]. Tasks are allocated to resources in accordance with task-VMs phase, LB was carried out while considering task length and demand on available VMs. Nan et al. [45] proposed new task scheduling scheme based on genetic algorithm (GA) for edge computing to resolve the task scheduling problem. The simulation result shows that the proposed algorithm had a beneficial effect on energy consumption and LB, and also reduced time delay. However, these methods have limitations such as high-power consumption, makespan task scheduling, time delay, network bandwidth.

From the research discussed above it is clearly evident that virtualization is crucial to cloud computing, and that problems like improper task scheduling in VMs maintenance, quickly deteriorate the cloud's efficiency. Furthermore, this results in an uneven distribution of workload across servers. As a result, there is still an opportunity for advancement in cloud computing technologies in terms of resource mapping and task scheduling. To effectively use resources without compromising the SLA, QoS metrics should be taken into account, as well as parameters like deadlines and priorities. One of the difficulties with cloud technology is resource allocation, which affects LB. This challenge also occurs in case if the priority between users and resources needs to be distributed equally. Hence the overall limitations observed from the existing works are scalability issues, poor security measures, high power consumption, time delay in task scheduling, low optimization of resources and lack of energy efficiency. To overcome these issues, an efficient LB is undertaken for the cloud computing, using a hybrid HHO-CSA, proposed in this work.

3.Proposed methodology

In the context of cloud computing, the proposed model and LB are explained in this section. The provision of higher quality services in applications of cloud computing clients is the main objective of this method. Multiple procedures are part of it: Task scheduling processes to give cloudlets (tasks) due dates and completion times, and LB processes to perform workload relocation in the event of VMs violations in a cloud environment, so that the LB is maintained as shown in *Figure 1*. The suggested algorithm's flow diagram shows the steps involved in LB and task scheduling. In the first step, the code is started, in the second step each V_m randomly assigns tasks to V_{ms}, in the third step million instructions per second(MIPS) is calculated and the base total workload for each V_m is shared, then in the fourth step, the expected completion time is calculated for all the tasks, the fifth step is for calculating the deadline and the violation cost of each vector machine and the next step is to find the vector machine that violates SLA. If it violates, then the vector machine migrates the workload with the aid of hybrid HHO-CSA, otherwise, the ready queue and expected completion time of the corresponding vector machine is updated.

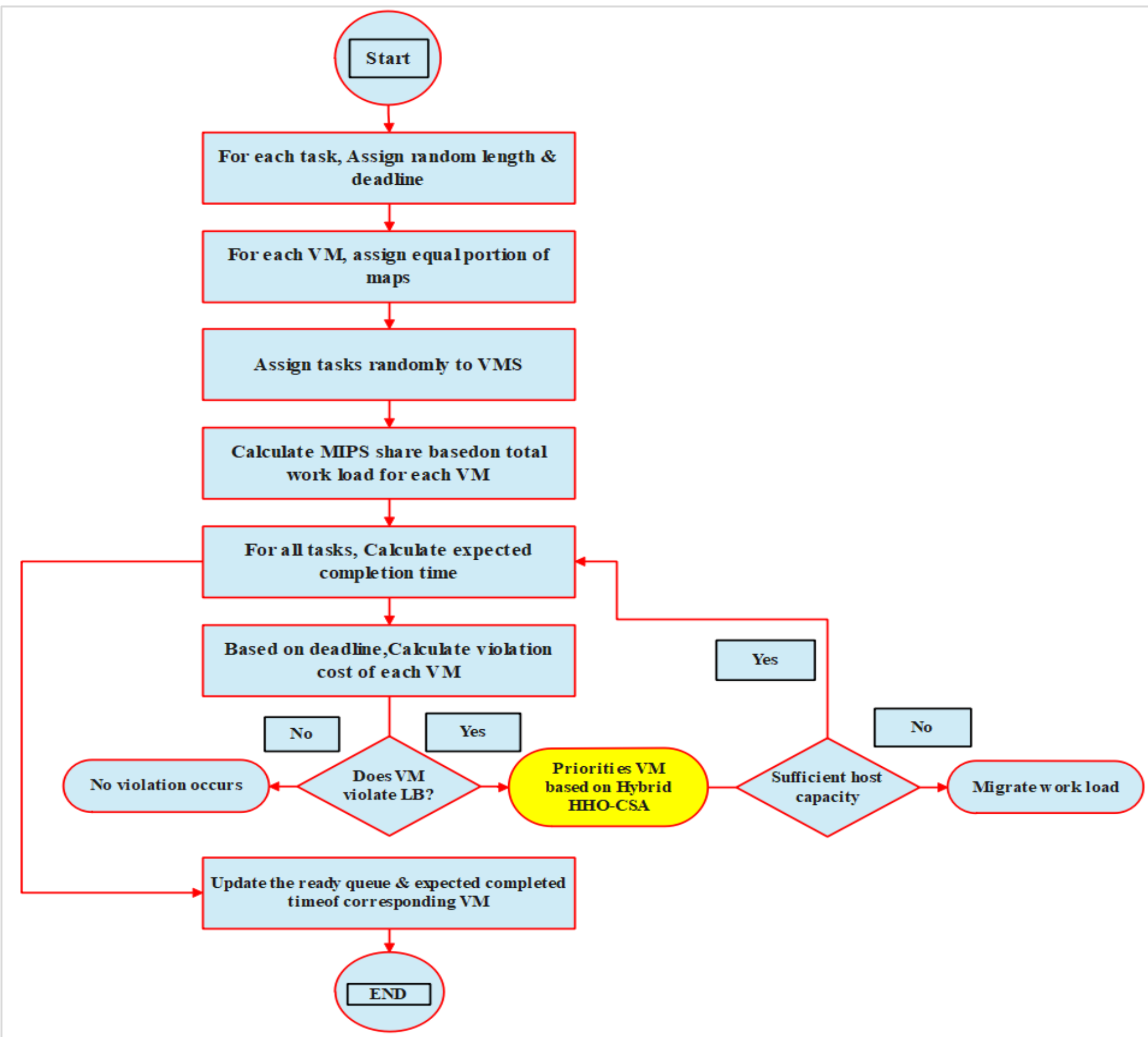

Figure 1 Block diagram for the proposed algorithm

3.1Harris hawks optimization and cuckoo search algorithm

This study introduces the hybrid HHO algorithm which is a new type of data stimulation that combines the DE and the hybrid HHO algorithms. The hybrid HHO method makes use of a few HHO algorithm parameters as well as the benefits of DE's local search approach. Five unique client identifier (UCI) benchmarks are exploited to assess the result of hybrid HHO. The HHO, when compared to several other algorithms performed better in error rate. This was backed by HHO prey investigation, startling jump, and various attack strategies. The use of conventional detachment is classified in the following Equation 1.

$$X_{n+1} = X_n - \frac{F(X_n)}{F'(X_n)} \tag{1}$$

In situations where $F'(Xn)$ is the Jacobian of F, with algebraic problems (x). Mathematical and MATLAB use Newton's method-based built-in functions to find the roots of nonlinear equations because of these Equation 2.

$$X(t+1) = \begin{cases} X_{rand(t)-r_1}|X_{rand(t)-2r2X(t)|9\geq 0.5} \\ X_{rabbit(t)-Xm(t)-rs(LB+r4(UB-LB))9\leq 0.5} \end{cases} \tag{2}$$

$X_{rabbit(t)}$ is stated as rabbit, $X(t)$ is where the hawks currently are, and $X(t)$ is where the HHO will be in the following iteration. Xm is a representation of the average location of HHO at this time (t) Equation 3.

$$Xm(t) = \frac{1}{N}\sum_{I=1}^{N} Xi(t) \tag{3}$$

$Xi(t)$ denotes the position of the Harris hawks in iteration t; N stands for the maximum number of HHO.

Ani and Guira cuckoos are two species that exhibit brood parasitism, which serves as the basis for the CSA. By laying eggs in other nests, these cuckoos display an aggressive method of reproduction. The host birds discard these eggs to start new nests elsewhere after learning they are not their own. A dynamic adaptive CSA takes advantage of adaptive step control to boost subgroup cooperation, accelerate convergence, and boost optimization precision. Despite being used and researched in many different domains, the computer Science algorithm still has some flaws. The typical computer Science method lacks a useful procedure to increase the search depth due to the high randomness of Levy's flight, and as a result, the convergence accuracy is moderately high. Equation 4 presents the computations of the proposed algorithm.

$$\alpha(t_i+1) = \begin{cases} \alpha_{\max}\exp\left(\frac{-t_1}{t_{\max}}\right) T < 0.35 \\ \alpha(t_i)\, 0.35 \leq T \leq 0.65 \\ \alpha_{\min}\exp\left(\frac{-t_1}{t_{\max}}\right) T > 0.65 \end{cases} \tag{4}$$

While the other parameters are left unchanged, the parameter max was fixed to successive values. The objective function was chosen after combining optimization of various functions as shown in Equation 5.

$$X_i(t+1) = x_i^t + \alpha \times L(\lambda) \tag{5}$$

where$X(t)$ indicates the generation's nest position, and L () Levy-flight random search pathways in Equation 6.

$$f(t) = min(t_{i1} + t_{i2} + t_{i3}), max(V_I) < V_i max \tag{6}$$

3.2Hybrid for Harris hawks optimization and cuckoo search algorithm

The fitness value is then determined using the Euclidean norm, often known as the norm-2. Here, a solution through a lower norm is supplementarily better than a higher norm. Therefore, a norm=0 is an exact solution. The average distance between the origin and vector is represented by this norm $f(x) = f_1, f_2..$ as stated Equation 7.

$$Fitness = \|f(x)\|2 \tag{7}$$

$$\sqrt{f_1^2 + f_2^2 + \ldots f_n^2}$$

The random integers L_i and D_i are inputs for the proposed work, and appropriate mapping of V_m is derived as output. To begin with, several V_m are assigned in the same part of MIPS to end the process. Then, $i = 1\ to\ m$ and $j = 1ton$ are the values assigned for processing the formula $C_{ij} = \frac{L_i}{MIPS}$which is used to calculate the value of I. For j, it is done until all the tasks are allocated to appropriate V_m. If $V_{ms}V_{ms+1}$, where $s \leq 6$ then the process of the proposed work is reconfigured to another V_m. During MIPS, if a host is running V_{mi}, then sufficient workload migration is processed. When the process is not classified then values are assigned as 0, else; C_{ij} is recomputed for each V_{mi} or else; if no violation occurs, an upgrade of a relay C_{ij} is expected and queued together with V_{mi}, to calculate average, after which all the tasks are exited separately for V_{mi} and R_v.

4.Results

This experiment aims to demonstrate how resource use increases in a dynamic cloud environment while makespan and execution time decrease. In the algorithm testing phase, task scheduling is taken into account ahead of time. As a result, if the workload violates the LB, the task may be suspended during execution or transferred to another resource to finish, as revealed in *Table 1*. Here, the iterations are varied from 50 to 100 at a fitness range of 0 to 1.

Table 1 Comparison table for same arrival time and random arrival time

Output						
Same arrival time						
Cloud ID	**Status**	**DC ID**	**VM ID**	**Time**	**Start time**	**Finish time**
8	Success	2	3	89.111	0.1	89.21
24	Success	2	6	124.01	0.1	124.11
16	Success	2	5	161.41	0.1	161.51
14	Success	2	4	181.82	0.1	181.92
15	Success	2	4	193.95	0.1	193.19
21	Success	2	6	224.52	0.1	224.62
3	Success	2	1	252.18	0.1	252.28
11	Success	2	3	311.72	0.1	311.82
10	Success	2	3	365.60	0.1	365.70
Random arrival time						
Cloud ID	**Status**	**DC ID**	**VM ID**	**Time**	**Start time**	**Finish time**
13	Success	2	2	95.78	0.1	95.68
2	Success	2	3	122.25	0.1	122.15
5	Success	2	2	171.13	8.1	171.03
6	Success	2	1	224.42	0.1	224.32
7	Success	2	4	252.18	10.1	252.08
8	Success	2	1	412.46	4.1	412.36
9	Success	2	2	452.81	1.1	452.18

Several QoS performance indicators of cloudlets are taken into account during the scheduling process, including:

Arrival time: Either the algorithm receives a request from the user, or the time is indicated when the cloudlets arrive. When using CloudSim, this is referred to as the cloudlet start time. A default setting in cloud computing is that all cloudlets come simultaneously. Based on the code used in this technique, the cloud computing is assigned in a gradual sequence to VMs. We can construct an algorithm using this parameter to work the environment in a dynamic where each request's time of arrival differs.

Task length: Tasks are measured in terms of their size in bytes; smaller tasks result in more resource use. Each Cloudlet in CloudSim needs to have a length value that specifies whether it is a light, heavy, or normal request. Each Cloudlet in this research has a randomly chosen length that has been identified. To distinguish between distinct client requests, every cloudlet ought to have a random value. The total workload of the cloud environment was represented by setting a length to a lethargically value. In this experiment calculate the load for each VMs and the length parameter is a crucial input for this parameter. The parameter allows the identification to complete in-time requests for VMs, which allows for the assessment.

Makespan: One of the most crucial factors that cloud service providers (CSPs) take into account while developing a LB algorithm is this period allotted in the task's completion. Each Cloudlet in this deadline has differed from the experiment value, therefore the client receives a contract based on their requirements and the condition was different from the cloud provider's service expectations. Therefore, using random deadline values is advised rather than static ones. Makespan is a crucial property since it symbolizes LB; if the requests take longer than expected to complete, express if the LB has been violated.

4.1Evaluation of 2 VM

Additionally, the suggested technique of hybrid HHO-CSA enhances cloud environment resource consumption. According to the graph, the technique yields an average RU of 70% for 2 VMs and 40 workloads. The varied makespan and Execution times in each situation can cause the RU value to change. *Table 2* aims to demonstrate how resource use increases in a dynamic cloud environment while makespan and resource utilization decrease.

Table 2 Performance analysis of 2 VMs

No. of clouds	Makespan (ms)	Resource utilization (%)
10	260.4400938	78
15	380.7141928	76
20	510.0912354	74
25	614.1059191	68
30	750.6493054	69
35	865.4092714	73
40	890.851151	70

Prior task scheduling was considered during algorithm testing, while the workload violates the condition, the task may be suspended during execution or transferred to another resource to finish as shown in *Table 2*.

4.2Evaluation of 4 VM

This experiment aims to demonstrate how resource use increases in a dynamic cloud environment while makespan 4 VMs with 40 tasks may be suspended during transferred to another resource for complete the processing. *Table 3* represents the computational time for 40 cloudlets in 4 VMs.

Table 3 Computational time for 4 VMs

No. of clouds	Computational Time (ms)
10	200.9588411
15	280.1049563
20	362.7542894
25	422.1371608
30	530.5813692
35	625.637351
40	610.8145012

The makespan time is the primary parameter used for comparison in this study. The primary goals of the proposed hybrid HHO-CSA are used improve the allocation and usage of cloud resources to reduce the amount of time needed. To increase the features of the cloud the work compares the proposed method in the study to a relevant task that has been done in general terms for 4 VMs as shown in *Table 4*.

Table 4 Performance analysis of makespan and resource utilization for 4 VMs

No. of clouds	Average makespan (ms)	Resource utilization (%)
10	301.5310114	82
15	425.7085131	81
20	510.8855832	78
25	570.5558001	72
30	750.1528740	70
35	865.2081364	74
40	890.3224651	70

4.3Evaluation of different load and capacity

For analysis purposes, *Table 5* aims to validate how resource utilization decreases in a dynamic cloud environment while evaluating a load of 2 VMs, 4 VMs, 6 VMs, and 8 VMs with a capacity of 100 tasks.

In *Table 5*, the performances of resource utilization are calculated for 2, 4, 6 and 8 VM with 100 tasks. *Table 5*, clearly shows that the resource utilization has achieved 54% for 2 VMs, 48% for 4 VMs, 41% for 6 VMs and 39% for 8 VMs at 100 tasks. *Figure 2* shows the fitness function graph for different algorithms.

Table 5 Performance analysis of resource utilization for 100 tasks

Resource utilization (%)				
Capacity	Load			
No. of tasks	2 VMs	4 VMs	6 VMs	8 VMs
10	78	82	69	67
20	74	78	63	64
30	69	70	57	57
40	70	70	52	50
50	67	64	50	48
60	65	60	48	45
70	61	57	47	44
80	59	53	45	41
90	57	51	43	40
100	54	48	41	39

From the *Figure 2*, it evidently shows that the proposed approach of hybrid HHO-CSA outperformed the existing SLA-LB [16], MALO technique [18], and MOABCQ_LJF method [31] for the performance metric (Fitness function). For instance, the suggested HHO-CSA method achieved the least values in all task situations, as shown in *Figure 2*, based on the average values of the fitness function. Additionally, the suggested algorithm's stability is adequately demonstrated while resolving the tasks scheduling problem at various scales.

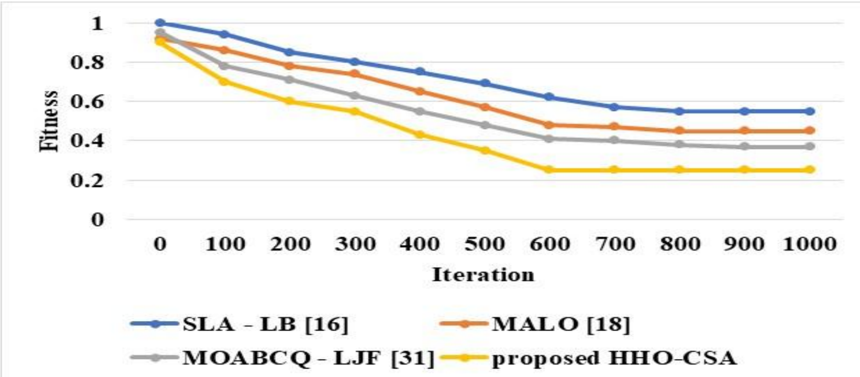

Figure 2 Fitness Vs Iteration graph for proposed HHO-CSA

4.4Comparative analysis

To evaluate the outcomes of the suggested algorithm, hybrid HHO-CSA, a comparative analysis is carried out with the most recent research algorithms. The makespan time is the primary comparison parameter used in this study. The suggested hybrid HHO-CSA is used to improve the usage and distribution of cloud resources. Results for 6 VMs with 10 to 40 clouds were achieved and tabulated in *Table 6*. The previous research (SLA-LB) [16] in the field and the algorithm that is being offered in this study are being compared here.

The experiment considered a wide range of task lengths and findings of 40 tasks that makespan in the re-approach increases 25 to 40 clouds. In the case of comparison, the existing SLA-LB [16] needs a normal resource utilization of roughly 70% for 40 tasks in 6 VMs, while the suggested technique (HHO-CSA) yields a resource utilization of 52% in 6 VMs, which is marginally better.

The existing SLA-LB [16] has been limited to projects with a length of 400,000 MI, the suggested hybrid HHO and CSA algorithm can handle requests for longer tasks with a duration of 1000000 MI. A longer task will result in a longer makespan because makespan is dependent on the load on the VMs. However, for the case of 25–40 tasks, the suggested approach by name of hybrid HHO-CSA lowers the makespan time if a smaller size is taken into account. In this paper, the statistical analysis is conducted according to the values of the makespan measure as shown in *Table 7*.

Table 6 Comparison analysis of execution time for 6 VMs

	Existing SLA-LB [16]			Proposed HHO-CSA		
No. of clouds	Execution time (ms)	Resource utilization (%)	Makespan (ms)	Execution time (ms)	Resource utilization (%)	Makespan (ms)
10	250.551	82	300.18	215.20	69	224.34
15	360.69	81	420.61	291.58	66	289.37
20	405.932	78	510.73	330.28	63	346.39
25	426.537	72	623.37	390.46	59	402.29
30	550.481	70	752.18	439.94	57	471.18
35	645.838	74	835.82	488.43	55	533.61
40	635.697	70	901.46	529.84	52	638.88

Table 7 Comparison of makespan by MALO and HHO-CSA generated using different sizes of tasks

Task Size	MALO [18]			HHO-CSA		
	Best	Worst	Average	Best	Worst	Average
100	64	75	70	56	71	64
200	109	126	120	106	119	119
300	229	252	240	201	232	218
400	318	335	323	298	313	302
500	436	458	446	435	460	452
600	527	557	543	509	555	533
700	609	632	620	603	625	610
800	703	731	720	689	719	700

Task Size	MALO [18]			HHO-CSA		
	Best	Worst	Average	Best	Worst	Average
900	796	836	810	768	790	800
1000	894	925	900	884	921	881
2000	1680	1845	1757	1510	1610	1668

This analysis is conducted to check whether the makespan measure values achieved by the proposed HHO-CSA is significantly less than that of MALO [18] for all tasks cases using the same termination criteria. This measure is one of the main tests used to measure the effectiveness of schedules. The results indicate that nine out of ten cases (100, 200, 300, 400, 600, 700, 800, 900 and 1000 tasks) showed significant improvement in makespan value for HHO-CSA, which means that there is a significant difference between the performance of the proposed HHO-CSA and the original MALO for these instances. But, the other cases (500 tasks) have no significant improvement. Thus, the main goal here is to find a small makespan value and maximum resource utilization. *Table 8* represents a comparison of the proposed method performance in terms of the degree of imbalance (DI) to assess the LB of the system.

Table 8 Comparative analysis of DI in hybrid HHO-CSA

Task	Task scheduling approach	
	Existing MOABCQ_LJF [31]	Proposed HHO-CSA
200	0.200	0.144
400	0.166	0.155
600	0.116	0.098
800	0.115	0.093
1000	0.093	0.089

The experiments were tested on 100 VMs with 200, 400, 800 and 1000 tasks. The proposed method (HHO-CSA) was compared with the existing MOABCQ_LJF method [31]. The proposed HHO-CSA can distribute tasks better than the existing MOABCQ_LJF at 4.5%.

5.Discussion

This section provides the discussion about the proposed hybrid HHO-CSA algorithm’s findings with respect to makespan, execution time, resource utilization as well as DI, and these results are compared with the existing SLA-LB [16], MALO method [18] and MOABCQ_LJF methods [31]. The major goal of this study is to maintain LB using the HHO and CSA method. From the result analysis, it clearly shows that proposed HHO-CSA processed the LB process with better performances in terms of Execution time (529.84 ms), Resource Utilization (52%), and makespan (638.88 ms) than the existing SLA-LB [16] method. The proposed HHO-CSA achieved the better improvement in makespan measures while comparing the MALO method [18]. While considering the DI performance, the proposed HHO-CSA achieved the better performance and have the lowest tested values when compared to MOABCQ_LJF method [31]. The proposed hybrid approach of HHO-CSA and existing MOABCQ_LJF method were contrasted with 100 VMs and 1000 workloads to test the trials. Overall, task distribution is improved by the proposed HHO-CSA by 4.5% over the existing MOABCQ_LJF. While compared with three existing methods, proposed HHO-CSA improved the LB.

5.1Limitation

From the result analysis, it has been demonstrated that QoS criteria of proposed model greatly increases the resource usage while lowering the makespan and offering for VMs allocation. Additionally, the proposed HHO-CSA is useful for a variety of applications, such as location-aware services, cloud-based recording services, etc. However, the issue of workload migration is still not totally resolved. Even if VMs is in an SLA violation status, which means it doesn't follow the deadline and requirements specified. As a result, CSP create unique SLA contracts for each client based on their requirements, and also the scheduling requires that the deadline parameter as random values to demonstrate the algorithm's violation problem. Furthermore, CSA still requires additional work and adjustments to increase productivity and task planning efficiency.

A complete list of abbreviations is shown in *Appendix I.*

6.Conclusion and future work

In recent trends, efficient task allocation has become a critical process in cloud computing due to the presence of limited resources and VMs. According to this research, task scheduling plays a significant role in LB within a cloud context. Enhanced task scheduling improves the LB procedure, which, in turn, contributes to the effective utilization of cloud resources. The objective of this work is to enhance task scheduling using LB techniques. The result analysis clearly demonstrates that the proposed HHO-CSA approach outperforms the existing SLA-LB technique and the MALO approach in terms of Execution time (529.84 ms), Resource Utilization (52%), and makespan (638.88 ms) when managing the LB process. Particularly when compared to the MALO approach, the proposed HHO-CSA demonstrates more effective improvement in the makespan measures. By optimally allocating combined resources for task completion, this technique effectively addresses SLA violations of VMs. In the future, this research will be further extended by analyzing the LB process using other metaheuristics or nature-inspired algorithms under various scenarios.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

Author's contribution statement

For this research work all authors' have equally contributed in Conceptualization, methodology, validation, resources, writing—original draft preparation, writing—review and editing.

References

[1] Wu Z, Sun J, Zhang Y, Zhu Y, Li J, Plaza A, et al. Scheduling-guided automatic processing of massive hyperspectral image classification on cloud computing architectures. IEEE Transactions on Cybernetics. 2020; 51(7):3588-601.

[2] Chen X, Cheng L, Liu C, Liu Q, Liu J, Mao Y et al. A WOA-based optimization approach for task scheduling in cloud computing systems. IEEE Systems Journal. 2020; 14(3):3117-28.

[3] Manikandan N, Gobalakrishnan N, Pradeep K. Bee optimization based random double adaptive whale optimization model for task scheduling in cloud computing environment. Computer Communications. 2022; 187:35-44.

[4] Seyfollahi A, Ghaffari A. Reliable data dissemination for the internet of things using Harris hawks optimization. Peer-to-Peer Networking and Applications. 2020; 13:1886-902.

[5] Parida BR, Rath AK, Mohapatra H. Binary self-adaptive salp swarm optimization-based dynamic load balancing in cloud computing. International Journal of Information Technology and Web Engineering. 2022; 17(1):1-25.

[6] Wei X. Task scheduling optimization strategy using improved ant colony optimization algorithm in cloud computing. Journal of Ambient Intelligence and Humanized Computing. 2020:1-2.

[7] Ismayilov G, Topcuoglu HR. Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing. Future Generation Computer Systems. 2020; 102:307-22.

[8] Iranmanesh A, Naji HR. DCHG-TS: a deadline-constrained and cost-effective hybrid genetic algorithm for scientific workflow scheduling in cloud computing. Cluster Computing. 2021; 24:667-81.

[9] Velliangiri S, Karthikeyan P, Xavier VA, Baswaraj D. Hybrid electro search with genetic algorithm for task scheduling in cloud computing. Ain Shams Engineering Journal. 2021; 12(1):631-9.

[10] Gohil BN, Patel DR. Load balancing in cloud using improved gray wolf optimizer. Concurrency and Computation: Practice and Experience. 2022; 34(11):e6888.

[11] Gharehchopogh FS, Abdollahzadeh B. An efficient harris hawk optimization algorithm for solving the travelling salesman problem. Cluster Computing. 2022; 25(3):1981-2005.

[12] Kheradmand B, Ghaffari A, Gharehchopogh FS, Masdari M. Cluster-based routing schema using Harris hawks optimization in the vehicular Ad Hoc networks. Wireless Communications and Mobile Computing. 2022; 2022:1-15.

[13] Seyfollahi A, Abeshloo H, Ghaffari A. Enhancing mobile crowdsensing in fog-based internet of things utilizing Harris hawks optimization. Journal of Ambient Intelligence and Humanized Computing. 2021:1-6.

[14] Xu X, Chen Y, Yuan Y, Huang T, Zhang X, Qi L. Blockchain-based cloudlet management for multimedia workflow in mobile cloud computing. Multimedia Tools and Applications. 2020; 79:9819-44.

[15] Imene L, Sihem S, Okba K, Mohamed B. A third generation genetic algorithm NSGAIII for task scheduling in cloud computing. Journal of King Saud University-Computer and Information Sciences. 2022; 34(9):7515-29.

[16] Shafiq DA, Jhanjhi NZ, Abdullah A, Alzain MA. A load balancing algorithm for the data centres to optimize cloud computing applications. IEEE Access. 2021; 9:41731-44.

[17] Zhu Z, Tan L, Li Y, Ji C. PHDFS: optimizing I/O performance of HDFS in deep learning cloud computing platform. Journal of Systems Architecture. 2020; 109:101810.

[18] Abualigah L, Diabat A. A novel hybrid antlion optimization algorithm for multi-objective task scheduling problems in cloud computing environments. Cluster Computing. 2021; 24:205-23.

[19] Abualigah L, Alkhrabsheh M. Amended hybrid multi-verse optimizer with genetic algorithm for solving task scheduling problem in cloud computing. The Journal of Supercomputing. 2022; 78(1):740-65.

[20] Zhang L, Zhou L, Salah A. Efficient scientific workflow scheduling for deadline-constrained parallel tasks in cloud computing environments. Information Sciences. 2020; 531:31-46.

[21] Sanaj MS, Prathap PJ. Nature inspired chaotic squirrel search algorithm (CSSA) for multi objective task scheduling in an IAAS cloud computing atmosphere. Engineering Science and Technology, an International Journal. 2020; 23(4):891-902.

[22] Praveenchandar J, Tamilarasi A. Dynamic resource allocation with optimized task scheduling and improved power management in cloud computing. Journal of Ambient Intelligence and Humanized Computing. 2021; 12:4147-59.

[23] Sefati S, Mousavinasab M, Zareh FR. Load balancing in cloud computing environment using the grey wolf optimization algorithm based on the reliability: performance evaluation. The Journal of Supercomputing. 2022; 78(1):18-42.

[24] Talaat FM, Ali HA, Saraya MS, Saleh AI. Effective scheduling algorithm for load balancing in fog environment using CNN and MPSO. Knowledge and Information Systems. 2022; 64(3):773-97.

[25] Singh P, Kaur R, Rashid J, Juneja S, Dhiman G, Kim J, et al. A fog-cluster based load-balancing technique. Sustainability. 2022; 14(13):1-14.

[26] Nabi S, Ahmad M, Ibrahim M, Hamam H. AdPSO: adaptive PSO-based task scheduling approach for cloud computing. Sensors. 2022; 22(3):1-22.

[27] Rana N, Abd LMS, Abdulhamid SI, Misra S. A hybrid whale optimization algorithm with differential evolution optimization for multi-objective virtual machine scheduling in cloud computing. Engineering Optimization. 2022; 54(12):1999-2016.

[28] Gupta P, Bhagat S, Saini DK, Kumar A, Alahmadi M, Sharma PC. Hybrid whale optimization algorithm for resource optimization in cloud E-healthcare applications. Computers, Materials & Continua. 2022; 71(3):5659-76.

[29] Latchoumi TP, Parthiban L. Quasi oppositional dragonfly algorithm for load balancing in cloud computing environment. Wireless Personal Communications. 2022; 122(3):2639-56.

[30] Annie PPG, Radhamani AS. A hybrid meta-heuristic for optimal load balancing in cloud computing. Journal of Grid Computing. 2021; 19(2):21.

[31] Kruekaew B, Kimpan W. Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning. IEEE Access. 2022; 10:17803-18.

[32] Thapliyal N, Dimri P. Load balancing in cloud computing based on honey bee foraging behavior and load balance min-min scheduling algorithm. International Journal of Electrical and Electronics Research. 2022; 10(1):1-6.

[33] Nazir J, Iqbal MW, Alyas T, Hamid M, Saleem M, Malik S, et al. Load balancing framework for cross-region tasks in cloud computing. Computers, Materials & Continua. 2022; 70(1):1479-90.

[34] Shekhar CA, Sharvani GS. MTLBP: a novel framework to assess multi-tenant load balance in cloud computing for cost-effective resource allocation. Wireless Personal Communications. 2021; 120:1873-93.

[35] Hung LH, Wu CH, Tsai CH, Huang HC. Migration-based load balance of virtual machine servers in cloud computing by load prediction using genetic-based methods. IEEE Access. 2021; 9:49760-73.

[36] Saif MA, Niranjan SK, Murshed BA, Ghanem FA, Ahmed AA. CSO-ILB: chicken swarm optimized inter-cloud load balancer for elastic containerized multi-cloud environment. The Journal of Supercomputing. 2023; 79(1):1111-55.

[37] Abedi S, Ghobaei-arani M, Khorami E, Mojarad M. Dynamic resource allocation using improved firefly optimization algorithm in cloud environment. Applied Artificial Intelligence. 2022; 36(1):2055394.

[38] Adil M, Nabi S, Aleem M, Diaz VG, Lin JC. CA-MLBS: content-aware machine learning based load balancing scheduler in the cloud environment. Expert Systems. 2023; 40(4):e13150.

[39] Zhang AN, Chu SC, Song PC, Wang H, Pan JS. Task scheduling in cloud computing environment using advanced phasmatodea population evolution algorithms. Electronics. 2022; 11(9):1-16.

[40] Al-yarimi FA, Althahabi S, Eltayeb MM. Optimal load balancing in cloud environment of virtual machines. Computer Systems Science & Engineering. 2022; 41(3):919-32.

[41] Iqbal N, Khan AN, Rizwan A, Qayyum F, Malik S, Ahmad R, et al. Enhanced time-constraint aware tasks scheduling mechanism based on predictive optimization for efficient load balancing in smart manufacturing. Journal of Manufacturing Systems. 2022; 64:19-39.

[42] Murad SS, Badeel RO, Salih N, Alsandi A, Faraj R, Ahmed AR, et al. Optimized Min-Min task scheduling algorithm for scientific workflows in a cloud environment. Journal of Theoretical and Applied Information Technology. 2022; 100(2):480-506.

[43] Bal PK, Mohapatra SK, Das TK, Srinivasan K, Hu YC. A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques. Sensors. 2022; 22(3):1-16.

[44] Ahmed S, and Omara FA. A modified workflow scheduling algorithm for cloud computing environment. International Journal of Intelligent Engineering and Systems. 2022; 15(5):336-52.

[45] Nan Z, Wenjing L, Zhu L, Zhi L, Yumin L, Nahar N. A new task scheduling scheme based on genetic algorithm for edge computing. Computers, Materials & Continua. 2022; 71(1):843-54.

Alok Kumar Pani pursued his Bachelor of Engineering from Biju Patnaik University of Technology, Odisha in 2006 and Master of Technology from Indian Institute of Technology (ISM), Dhanbad in 2014. He is currently pursuing Ph.D. and working as an Assistant Professor in Department of Computer Science and Engineering, CHRIST (Deemed to be University), Bengaluru. He has 14 years of teaching experience and 1 year of Industry Experience. His primary research interests are in the fields of Databases, Algorithms and Computational Complexity.
Email: alok.kumar@christuniversity.in

M. Manohar was brought up in Udupi and currently he lives in Bangalore, Karnataka. He was qualified Bachelor in CSE in the year 1999 at Dr AIT, Bangalore. He has his Master in CSE in the year 2004 at Dr AIT, Bangalore. He completed his Ph.D in CSE in the area of data mining and big data in the year 2016 at Jain University, Bangalore. He is an Associate Professor in the Computer Science and Engineering Department at Faculty of engineering of CHRIST (Deemed to be University), Bengaluru. He is an educator by choice and vocation, with an experience of 13 years in Teaching. He has many publications in both conference and Journals. He is a member of Professional body IACSIT, IET.
Email: manohar.m@christuniversity.in

Merin Thomas is an Associate Professor in the School of Computer science, RV University, Bengaluru, India. Her research specialization is in Natural Language Processing in the domain of Artificial Intelligence and Machine Learning.
Email: thomasmerin287@gmail.com

Pankaj Kumar received his B Sc. Engineering in Information Technology from MIT Muzaffarpur and his M.Tech in Computer Science and Engineering from IIT(ISM) Dhanbad. He is currently working as an Assistant Professor with Department of Computer Science and Engineering at Motihari College of Engineering (Government of Bihar). His current research interests include Cryptography, Information Security, and Database Management System.
Email: paku9780@gmail.com

Appendix I

S. No.	Abbreviation	Description
1	ABC	Artificial Bee Colony
2	ACO	Ant Colony Optimization
3	ANN-WHO	Artificial Neural Network-Whale Optimization
4	ARUR	Average Resource Utilization Ratio
5	CA-MLBS	Content-Aware Machine Learning Based Load Balancing Schedular
6	CSSA	Chaotic Squirrel Search Algorithm
7	CSO-ILB	Chicken Swarm Optimized Inter-Cloud Load Balancer
8	CSP	Cloud Service Provider
9	DAG	Directed Acyclic Grip
10	DE	Differential Evolution
11	DI	Degree of Imbalance
12	DRA	Dynamic Resource Allocation
13	EDLB	Efficient Dynamic Load Balancing
14	EPRD	Efficient Priority and Relative Distance
15	FEF	Fair Emergency First
16	GA	Genetic Algorithm
17	GWO	Grey Wolf Optimization
18	HDFS	Hadoop Distributed File System
19	H-HHO	Hybrid- Harris Hawks Optimization
20	HHO	Harris Hawk Optimizer
21	HHO-CSA	Harris Hawk's Optimization-Cuckoo Search Algorithm
22	HWOA	Hybrid Whale Optimization Algorithm
23	IaaS	Infrastructure as a Service
24	IFA	Improved Firefly Algorithm
25	IFA-DRA	Improved Firefly Algorithm-Dynamic Resource Allocation
26	LB	Load Balancing
27	MALO	Mirjalili Antlion Optimizer
28	MIPS	Million Instructions Per Second
29	MOABCQ_LJF	Multi-Objective Artificial Bee Colony q-learning Largest Job First
30		
31	MVO	Multi-Verse Optimizer
32	MVO-GA	Multi-Verse Optimizer-Genetic Algorithm
33	OMin-Min	Optimized Min-Min
34	PHDFS	Pile-Hadoop Distributed File System
35	PIO	Pigeon Inspired Optimization
36	PSO	Particle Swarm Optimization
37	QoS	Quality of Service
38	RATS-HM	Resource Allocation Task Scheduling-Hybrid Machine Learning
39	SLA	Service Level Agreement
40	SLA-LB	Service Level Agreement-Load Balancer
41	TCA	Time-Constrained Aware
42	UCI	Unique Client Identifier
43	VMs	Virtual Machines