IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 January 2023, accepted 29 January 2023, date of publication 1 February 2023, date of current version 7 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3241279

RESEARCH ARTICLE

A Fast Converging and Globally Optimized Approach for Load Balancing in Cloud Computing

MANA SALEH AL RESHAN[1], DARAKHSHAN SYED[2], NOMAN ISLAM[3], ASADULLAH SHAIKH[1], (Senior Member, IEEE), MOHAMMED HAMDI[1], MOHAMED A. ELMAGZOUB[1], GHULAM MUHAMMAD[2], AND KASHIF HUSSAIN TALPUR[2]

[1]College of Computer Science and Information Systems, Najran University, Najran 61441, Saudi Arabia
[2]Computer Science Department, Bahria University, Karachi 75500, Pakistan
[3]Department of Computer Science, Karachi Institute of Economics and Technology, Karachi 76400, Pakistan

Corresponding author: Mohamed A. Elmagzoub (meabdullah@nu.edu.sa)

This work was supported by the Deanship of Scientific Research at Najran University through the National Research Priorities under Grant NU/NRP/SERC/11/18.

ABSTRACT Cloud Computing is the dynamic provisioning of resources to provide services to end-users over the internet. The realization of cloud computing requires addressing several challenges, such as resource discovery, security, scheduling, and load balancing. Among these research issues, load balancing is the most challenging one. Therefore, in the past few years, research into various static and dynamic algorithms to achieve optimal results is gaining importance. This research proposes Swarm Intelligence (SI) as a load-balancing solution for cloud computing. Several alternatives in the literature (like genetic algorithm, ACO, PSO, BAT, GWO, and many others) are investigated, but none consider the load balancing convergence time with global optimization. Among these algorithms, this research emphasizes Particle Swarm Optimization (PSO) and Grey Wolf Optimization (GWO). A combined approach of GWO-PSO that capitalizes on the benefits of fast convergence and global optimization is proposed in this paper. These two techniques enhance system efficiency and resource allocation, working together to solve the load-balancing challenge. Compared to other traditional approaches, the findings of this research are promising while achieving globally optimized fast convergence and reducing overall response time. On average, the overall response time of the proposed technique is reduced to 12% as compared to other algorithms. Furthermore, the best optimal value obtained from the objective function of the proposed GWO-PSO algorithm improves PSO to 97.253% in terms of convergence.

INDEX TERMS Cloud computing, load balancing, swarm intelligence, particle swarm optimization, grey wolf optimization.

I. INTRODUCTION

Cloud computing is the delivery of resources over the internet. These resources include computing, storage, databases, and networking [1]. The actual realization of a cloud environment has been fraught with various difficulties. Amongst them includes resource discovery, scheduling, security, and privacy. Load balancing is one of the most pressing concerns among these issues. It refers to how the load is distributed among multiple machines [2]. Load balancing refers to the delivery and distribution of the required workload over numerous computer platforms [3]. Load balancing proposes methods for maximizing system output production, resource utilization, and performance parameters of virtual machines (VMs). To achieve efficient use of resources, the cloud system employs a variety of load-balancing algorithms. Some of these algorithms are presented in [4] and [5]. Gutierrez-Garcia and Ramirez-Nafarrate [6], in their research, discussed the objective of load balancing: to reduce reaction time and boost resource usage, resulting in higher productivity at

The associate editor coordinating the review of this manuscript and approving it for publication was Nitin Gupta.

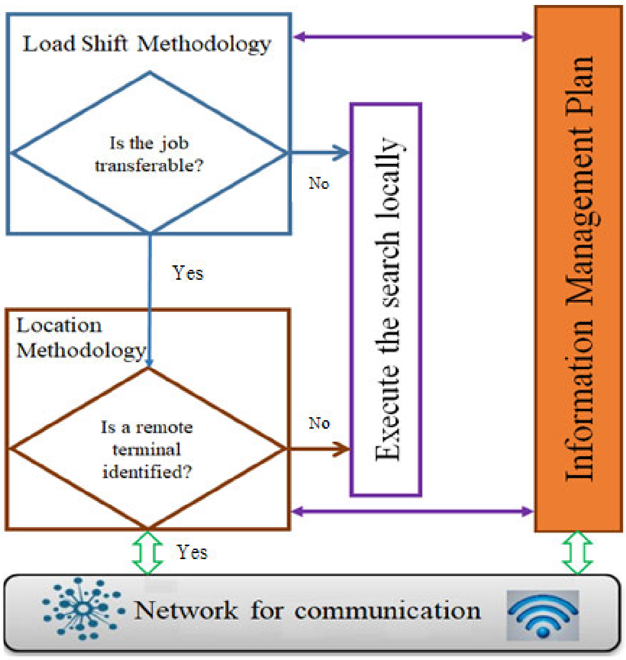


FIGURE 1. The architectural components of dynamic load balancing [5].

reduced costs. In addition, load balancing aims to provide sustainability and adaptability for applications that grow in size and demand more resources. It also prioritizes complex tasks' equitable distribution.

Before moving further, this research presents a detailed discussion about how the approaches to load balancing have been classified in the literature. Load balancing in a cloud-based system can be divided into two types [7]: (1) static algorithms and (2) dynamic algorithms.

- In a static method, the available base stations ignore location. All of the links are understood ahead of time and their characteristics. The execution of this kind of approach is predefined. It's simple to use and doesn't rely on real-time data from the current system.
- On the contrary, the machine's current status is considered for the dynamic balancing process. Changes in node architecture drive its operation. Dynamic algorithms are difficult to implement, but they efficiently distribute resources and, in that way, they balance load more effectively (Figure 1).

Due to the critical nature of run-time dynamics, the demand for dynamic algorithms is high. In this research, the targeted algorithms are also inspired by dynamic algorithms presented in literature coupled with Swarm Intelligence (SI) techniques. Several swarm intelligence techniques have been presented in [8], [9], [10], [11], and [12]. SI studies decentralized, evolvable systems, whether developing or developed, and their mutual behaviour and performance [13]. SI systems focus on the integrated behaviours that emerge from individuals' local interactions with one another, including their surroundings [14]. Ant and termite communities, groups of fish, swarms, and herds of cattle are examples of such systems.

For scholars, utilizing the concept of SI in cloud technology has gained prominence [15]. The following few lines highlight the importance of swarm intelligence algorithms. It produces cost-effective, optimized solutions to cloud-based applications for effective implementation, infrastructure sustainability, and privacy concerns. Swarm intelligence's immense understanding, ranging from an animal's glimpse to cutting-edge methodologies, may be maximized for handling diverse cloud computing difficulties. For example, VM allocation [16] is a common problem in the cloud context that can be recast using SI advanced approaches. SI can be applied in different areas that deal with routing and specialized task scheduling methods. These two SI applications eventually led to the most discussed topic in cloud computing: "load balancing" SI makes load balancing simply by relying on the animals for motivation [17], [18]. Consequently, this collaboration can be used to manage load in the cloud more effectively. Similarly, swarm tactics present a very clever and decentralized solution, and these methods function in the way cloud computing requires to manage demand efficiently. As a result, these collective, informed, and decentralized insect behaviours have evolved into a paradigm for addressing the challenging issues of load balancing in a cloud-based framework. Various state-of-the-art hybridized techniques are also recently presented by researchers. For example, Ahmed et al. [19] suggest a composite version of the Generalized Normal Distribution Optimizer (GNDO) with Simulated Annealing (SA) entitled Binary Simulated Normal Distribution Optimizer (BSNDO). The proposed technique uses SA as a localized search to improve classification performance. The new method is analyzed to its predecessors and various popular FS techniques on 18 well-known UCI samples. Furthermore, this approach is evaluated on high-dimensionality microarray data samples to demonstrate its utility in real-world datasets.

Yaun et al. [20] proposed Elite opposition-based learning and a chaotic k-best gravitational search (EOCS) technique. The key concept is to improve global exploration capabilities and convergence rates. Regarding exploration accuracy and reliability, the EOCS-based grey wolf optimizer (EOCSGWO) algorithm exceeds its competition. Zhao et al. [21] introduced the artificial hummingbird algorithm (AHA), a new bio-inspired optimization algorithm for solving optimization issues. The AHA algorithm mimics hummingbirds' unique flight abilities and foraging approaches in the ecosystem. The findings of AHA's validation are compared to those of several methods using two sets of quantitative test functions. The results demonstrate that AHA outperforms other meta-heuristic methods in determining high-quality alternatives while requiring fewer control parameters.

To discuss the importance of the swarm intelligence technique. This research presents a novel method to distribute workloads in a cloud-based environment. A load-balancing mechanism should take the convergence time into account. Even though fast convergence can quickly relieve overburdened VMs, Cloud Services' can promptly recover their performance, and unexpected outcomes can be avoided (e.g., overloaded central systems may hang or eventually crash). Hence, the paper presented a combination of grey wolf and particle swarm optimization to provide global optimization and fast convergence. To emphasize further, the following are the main contributions of this manuscript:

- A hybrid technique based on grey wolf and particle swarm optimization is presented in this paper.
- The optimization approach offered in this paper is a combination of two factors. The technique aims for global optimization and rapid convergence, supported by results.
- GWO-PSO algorithm is developed for efficient load balancing – the achieved factors are tested on the MATLAB software package, showing good global optimization and fast convergence.
- Presents the analytical results of the simulation to show convergence evaluation based on the value(s) of the objective function. The targeted algorithms are PSO, Social Spider(SSO), ABC, BAT, GWO, and the proposed GWO-PSO, with 30 agents (VMs).
- Comparative analysis based on overall response time by using cloud analyst is performed. The comparison is made with other traditional algorithms used for load balancing

The rest of the paper is structured as follows. Section II presents the related work. Section III focuses on SI algorithms for load balancing, while Section IV explores the proposed algorithm used in the study. Then, section V presents the simulation results to measure the optimization, and convergence of Particle Swarm Optimization (PSO), Grey Wolf Optimization (GWO), and amalgamative GWO-PSO. Finally, Section VI offers conclusions and future directions.

II. RELATED WORK

The load balancing issue is a typical challenge in cloud infrastructure, where the best solution is sought within a complex space to manage resources better. The investigation of Swarm Intelligence is motivated by the practical idea of using "collective intelligence." In an environment, this intelligence is dispersed, coordinated, and diffused [22]. Swarm intelligence is a well-known subfield of load balancing in Cloud Computing. Many algorithms are used in these techniques, such as genetic algorithms (GA), ant colony optimization (ACO), honey bee optimization (HBO), water wave algorithm (WWA), particle swarm optimization (PSO), grey wolf optimization (GWO), etc.

GAs are artificial life methods that incorporate biological processes as a template to create programming code. After then, this computer software learns in the same way that live systems do [23]. Makasarwala and Hazari [24] utilize GA for load balancing in Cloud Computing. It balances load with an issue of reduced resource utilization. To overcome the disadvantage of GA, another variation is presented by Kaur and Sengupta [25]. The improved GA (IGA) keeps continuous track of all the available virtual machines and the current arrival of tasks. In that way, the algorithm improves the utilization of resources and saves energy. But still, the response time is high. Therefore, a hybrid methodology has been developed to intelligently encounter task scheduling in cloud technology. Ali Saadat and Ellips Masehian [26] proposed two components to provide load balancing efficiently. The first module employs a Genetic Algorithm to find the best work arrangements, while the second focuses on fuzzy logic. It successfully implements the objective function of defining occupied server states based on task buffers where service availability is a fuzzy performance, GA optimization accessibility, stability, and universality. This study also conducts computer tests with the best solution. It ultimately leads to a higher user retention rate. Saadat and Masehian [27] proposed employing multi-agent GA to facilitate effective load balancing. The proposed method considers the users' priorities and, at the same time, also targets the completion time of the first task. Jayswal and Saxena [28] proposed utilizing multi-agent GA to achieve effective load balancing. The suggested method considers the priority of the users and the completion time of the first job. The load is balanced across all VMs based on CPU memory capacity at the request of a given user. The fitness function is the difference between each host's and the system's average load. For modelling the tests, the data are evaluated using a cloud analyst. It runs load-balancing tests in the background and displays the results in a graphical style with a significant level of customization.

The unbalanced load challenge is the focus of Li and Wu [29]. The ant colony algorithm uses the core aspect to find the best dynamic scheduling sequence. The scheduler resembles an ant, and the scheduling phase is compared to the foraging behaviour of ants. Finally, a node with good performance and low load witty minimum ending time are chosen to handle the assigned assignment. Ragmani et al. [30] improve the AC technique by incorporating a fuzzy logic module. This fuzzy-based approach reduced the response time by more than 80%.

Honey Bee or Artificial Bee Colony (ABC) is another method of optimization that is excellent for inspection but not so good for customized modification [31]. The b algorithm is recommended for applying optimization approaches using honey bees' sophisticated foraging activity. The aggregation of honey bees is known as a swarm, capable of completing the whole task of cooperation. Various academics utilized the ABC algorithm to improve load balancing [32]. Ullah [33] proposes workload scheduling using artificial bee foraging (TSABF) optimization to define an optimal task plan for virtual machines. The activities are planned ahead of time using the ABC method. The goal of task prevention in the proposed framework is to reduce the time required to complete jobs with various priorities. Li and Han [34] examine and resolve the problem of flexible work scheduling.

Honeybees utilize the hybrid discrete optimization technique to tackle the issue of load balancing in cloud computing. Shen et al. [35] examine the ABC optimization technique using a load balance algorithm and present several ways to improve overall load balance efficiency and responsiveness. The ABC technique is optimized, and the intelligent grid cloud resource characteristics are employed to cluster virtual machines (VMs). A simulation study confirms the proposed method's efficiency.

Venkatachalam and Bhalaji [36] present the experimental results of WWA for load balancing in cloud computing. The quality of service parameters reveals that the WWA method is also used for load balancing in cloud computing. With file-sharing services, a basic cloud with three VMs is emulated. The findings show that WWA outperforms the competition regarding maximum throughput, time, resource consumption, and scalability. Gulbaz et al. [37] describe the Balancer Genetic Algorithm (BGA), a unique balancing scheduler designed to increase makespan and load balancing. Inadequate task scheduling can result in a resource utilization overhead since specific resources continue to operate. BGA implements a load balancing method that considers the actual load in terms of million operations assigned to VMs. It is also emphasized that multi-objective optimization should enhance load balancing and makespan. Several batch sizes and skewed, normal, and homogeneous workload distributions were used in the experiments. BGA has shown considerable improvement compared to state-of-the-art systems for makespan, throughput, and load balancing. Miao et al. [38] proposed an APDPSO algorithm to overcome the associated issue with the PSO algorithm. The measure drawback experienced by PSO is randomness in particles' movement, which ultimately affects the discretization strategies. The proposed algorithm utilizes the stored reasonable solutions to update the personal best positions of particles with a certain probability. A discretization method is also applied to PSO for continuous management of change in the velocity and position vectors of the particles. Improved BAT algorithm [38] is proposed to obtain more optimum and most satisfactory results. To make it possible, the algorithm needs to be executed iteratively. Whenever there is a task for processing, the BAT algorithm finds the optimal server among the available. At the same time, the load scheduler also identifies the job type and resource required and selects the optimal VM for task execution. If the available server meets the requirements efficiently, the load is assigned; otherwise, if the load is higher, it is distributed to more than one server. This algorithm maintains load balancing by keeping all the servers busy, neither underloaded nor overloaded. Future research could look into new meta-heuristic evolutionary techniques for balancing load in terms of energy consumption, and quality of service [39]. Social Spider Algorithm (SSA) is also used for task scheduling which is an efficient approach to adapting the global best match, but when the number of VMs increases, it shows slow convergence [40].

Table 1 shows the summary of the discussed research work.

The various approaches to load balancing are discussed. However, there is a research gap where different hybrid algorithms are used to overcome the shortcomings of individual load balancing. In this discussion, several hybrid approaches to load balancing are presented. In the context of electric power infrastructures, Shaheen et al. [41] provide an implementation of the hybrid Grey Wolf and Particle Swarm Optimization (GWO-PSO) technique to resolve the effective reactive power distribution challenge.

A unique hybrid maximal power spot tracking approach built on the principles of particle swarm and grey wolf optimization is presented by Chtita et al. [42]. The proposed method not only overcomes the typical drawbacks of existing methods but also offers a straightforward and reliable tracking of maximum power points to manage partial shadowing in solar systems successfully. Leveraging GWO-based PSO, Gohil and Patel [43] introduce a novel dynamic load balancing technique in cloud technology and contrast it with several existing optimization algorithms. It aids in enhancing resource allocation equity and system performance. The research experiments' showed better convergence but did not focus on other quality of service parameters. Alabdalbari and Abed [44] moreover provide novel hybrid modelling data to show that the suggested Hybridized GWO-PSO strategy is significant from the perspective of path optimization problems. Task scheduling is also discussed by Senthil et al. [45] as a type of non-deterministic, difficult polynomial challenge that can be resolved using optimization techniques. They describe a model hybrid technique that improves response time by integrating PSO and grey wolf techniques. For science-based procedures, Kaur and Aron [46] introduced the integrated load balancing technique, which combines the tabu search, Grey Wolf and Ant Colony Optimization (ACO). The suggested model implements load balancing at the fog layer to improve resource utilization. Kaur and Dhindsa [47] discussed several strategies and techniques for managing parallel jobs and services to improve CPU utilization in the cloud infrastructure. They proposed the usage of GWO and PSO for efficient distribution of workload.

Table 2 shows the summary of the hybrid approaches discussed to load balancing, specifically targeting PSO and GWO. The targeted QoS parameters comprise Power(P), Cost(C), Response Time(RT), Throughput(T), Energy Consumption(E), and Execution time(ET). It has been observed that very few algorithms are available that targets convergence and global optimization at the same time. The novelty of this paper lies in a proposal for such an efficient algorithm.

After analyzing several SI-based algorithms with certain variations for load balancing in cloud computing, this research focuses on Particle Swarm Optimization(PSO) and Grey Wolf Optimization(GWO) techniques to achieve load balancing. With the help of Table 1 and 2, it is observed that most algorithms targeting several QoS parameters still lack convergence. If the proposed technique achieves fast convergence, the method will outperform the other traditional algorithms. Additionally, rapid convergence reduces response time, cost, and, eventually high throughput.

TABLE 1. Summary of the discussed algorithm with the pros and cons.

Algorithms	Targeted QoS	Targeted Application(s)	Advantages	Disadvantages
Genetic	Makespan	Load balancing in cloud computing, optimized task scheduling	It enables more effective utilisation of resources and provides a more effective load balancing strategy	The performance diminishes when the search space is extended. It does not provide the equivalent level of priority period as the other options
Ant colony optimisation	Resources utilisation	Load balancing in cloud computing, resource scheduling in cloud computing	Optimised resource utilisation via load distribution efficiency among virtual machines	Slow convergence
Artificial bee colony	Throughput	Load balancing in cloud computing	The optimum throughput limit is increased	The lack of accompanying information leads the procedure to slow down, and the techniques raise the computation cost when used in a sequence of events (i.e., slow convergence)
BAT algorithm	Accuracy	Avoid overloading of resources in cloud infrastructure	It has a high degree of accuracy. It beats in terms of processing cost.	There was no mathematical evaluation to connect the variables (i.e., better convergence)
Social spider algorithm	Resource utilisation	Load balancing in cloud computing	Achieve global best solution	High response time (slow convergence)
Particle Swarm Optimization	Resource utilisation	Optimised resource utilisation in a cloud computing environment	It achieves better resource utilisation	Slow convergence
Grey Wolf optimisation	Optimal resource utilization, reduce cost and time	Load balancing in cloud computing	Fast convergence rate	Inconsistent in terms of global optimisation

TABLE 2. Summary of the discussed hybridized algorithms targeting PSO and GWO on the basis of QoS.

Hybridized Approaches (Year)	Authors	Targeted Application(s)	P	C	RT	T	E	ET
[41](2021)	Mohamed A.M.Shaheen, Hany M.Hasanien, and Abdulaziz Alkuhayli	Optimal reactive power dispatch	✓	✗	✗	✗	✗	✗
[42](2022)	Smail Chtita, Saad Motahhir, Aboubakr El Hammoumi, Aissa Chouder, Abou Soufiane Benyoucef, Abdelaziz El Ghzizal, Aziz Derouich, Mohamed Abouhawwash, and S. S. Askar	Maximum power point tracking	✓	✗	✗	✗	✗	✗
[43](2018)	Bhavesh N. Gohil, and Dhiren R. Patel	Load balancing in cloud Computing	✗	✓	✗	✗	✗	✗
[44](2022)	Yad Abdulrahem Alabdalbari and Issa Ahmed Abed	Path planning optimization	✗	✗	✗	✓	✗	✗
[45](2021)	Avinashi Malleswaran Senthil Kumar, Parthiban Krishnamoorthy, Sivakumar Soubraylu, Jeya Krishnan Venugopal and Kalimuthu Marimuthu	Task Scheduling in cloud computing	✗	✗	✓	✗	✗	✗
[46](2021)	Mandeep Kaur and Rajni Aron	Resource utilization in cloud computing	✗	✗	✓	✗	✓	✓
[47](2018)	Rupinder Kaur and Kanwalvir Singh Dhindsa	Effective scheduling cloud computing	✗	✗	✓	✗	✗	✓
	Our proposed algorithm	Load balancing in cloud computing	✗	✓	✓	✓	✗	✓

III. TARGETED SI ALGORITHMS FOR LOAD BALANCING IN CLOUD INFRASTRUCTURE

A. PATICLE SWARM OPTIMIZATION (PSO)

The particle swarm optimization (PSO) technique is among the literature's most popular and widely utilized optimization techniques. Because of its flexibility, good global optimization, and a minimal number of characteristics, the PSO algorithm has been effectively used in many research discussions to tackle various optimization issues (Figure 2). Pbest is the last best value, while Gbest is the global best value. For example, Agarwal et al. [48] discuss how a swarm of flying birds chooses a landing spot but how deciding where to land is difficult.

Vidya et al. [49] highlight multiple things that influence the landing. Food availability (incoming requests) and the threat of carnivores (overloaded VMs') are examples of these circumstances. Under the currently available circumstances, the birds fly in lockstep (i.e., continuous iterations) until they find the optimal landing spot targeted VM), at which point they all land at the same time (assign tasks to the targeted VM). Pan and Chen [50] perform performance evaluations on various other algorithms with Particle swarm optimization. The research combines load with a shorter reaction time for every given activity. PSO has cheaper computation costs and is simpler to understand and implement. The PSO algorithm has been chosen as one of the methods in this amalgamative study since it is successful in various challenges from diverse disciplines.

Jordehi and Jasni [51] present an optimization method for task scheduling to balance load more efficiently. Results from simulation prove it is a better performance technique. Guliyev et al. [52] proposed a framework to optimize load in cloud environments when the network is complicated. The PSO algorithm was used to create the method. The study claimed improved PSO algorithm performs exceptionally well. Various changes define the particle's position and velocity in this upgraded version. The rules for ongoing updates are eventually reevaluated. Agarwal et al. [53] implement PSO for load balancing to form the relevance for scheduling procedures with proportional weight. These weights assist in determining the best option. In this study, a new objective function is developed, which differs from current ones and work schedules for various virtual machines. Li and Wu [29] developed a method to improve the average load in cloud systems. The downside of PSO is that it is susceptible to beginning conditions, which is considered in this study. A favourable result will not be feasible if there is a problem with the beginning population. A hybrid technique has been presented as a solution to this problem. This idea enables us to do jobs at a high rate in this manner. The usual load time of IPSO is 0.457 milliseconds, and Firefly is 0.47 milliseconds, whereas the overall load time of the combined IPSO-Firefly technique is 0.259 milliseconds.

Pan and Chen [54] provides another technique to balance the load in cloud environments in case the infrastructure of the cloud is complicated. The PSO algorithm was used to create the model. In this paper, the author proposed an improved PSO algorithm that performs exceptionally well. Various changes define the particle's position and velocity in this upgraded version.

1) IMPORTANCE OF PSO

The most common application of the population-based method PSO (Particle Swarm Optimization) is the practical approach of optimal control approaches. PSO is one of several swarm intelligence approaches used to address optimization issues. PSO is a non-deterministic, stochastic optimal control technique that uses a community-based search process to achieve global optimization. Its main benefit is that it is relatively easy to implement and requires few characteristics to alter. The adoption of parameters is the fundamental issue with present approaches for resource scheduling with the goal of load balancing. A workload of metrics is minimized throughout the routing process, which equals the current techniques. Metrics are assigned appropriate load coefficients to illustrate their degree of relevance. These load variables demonstrate the importance of metrics in the scheduling phase. The more loads are assigned to the criterion, the better the answer. Changing the value of the load coefficients makes it simple to regulate the adequacy of the target function.

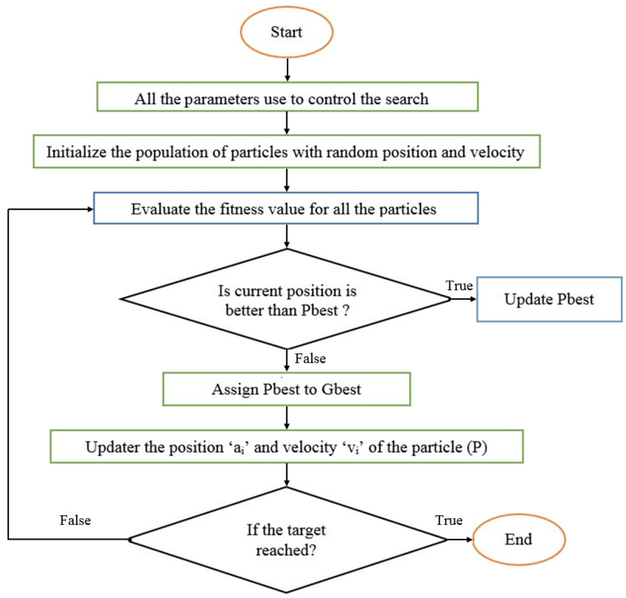


FIGURE 2. Flowchart of PSO.

2) THE BASIC MATHEMATICAL MODEL OF PSO FOR LOAD BALANCING IN CLOUD ENVIRONMENT

PSO algorithm is a multi-agent concurrent search method in which every particle poses a feasible solution in the swarm [55]. All particles go through a multidimensional exploration region, where each particle adjusts its position based on its own and neighbouring experiences. There are "M" virtual machine networks of interconnected nodes. The VMs are denoted by the list VM = ($VM_1, VM_2, \ldots, VM_M$), where V_j is the maximum resource ability that VM "j" can give, and j belongs to [1,M]. Here, there is a collection of tasks denoted by T.

$$T = \{T_1, T_2, \ldots, T_N\}$$

where "T_k" means the task number "k", where k belongs to [1,N], and N is the total length of the task series. The model of tasks is defined as 'T_k" (RequiredTime, RequiredResource). The "RequiredTime" denotes the task's execution time, and the Required Resource denotes the need for resources for the execution of the task. For the objective model, PSO utilizes one zero matrices named "Z" to show the mapping of tasks to VMs. This zero matrix can be shown as:

$$Z = \begin{bmatrix} Z_{1,1}, Z_{1,2}, \ldots, Z_{1,N} \\ Z_{2,1}, Z_{2,2}, \ldots, Z_{2,N} \\ . \\ . \\ . \\ Z_{M,1}, Z_{M,2}, \ldots, Z_{M,N} \end{bmatrix}$$

Considering the discussed models, the following equations (Eq. 1 and Eq. 2) are used:

$$\text{Time}_{VM} = {}^{\text{M}}\text{max}_{j=1}\left[\sum_{k=1}(Z_{j,k} * T_k \cdot \text{Required Time}\right] \quad (1)$$

$$RU_{VM} = {}^{\text{N}}\sum_{k=1}\left(\frac{{}^{\text{N}}\sum_{k=1}(z_{j,k} * T_k \cdot \text{Required Resource})}{VM_j}\right) \quad (2)$$

where, Time_{VM} = the time required for virtual machines to complete all jobs.

RU_{VM} = the number of resources virtual machines (VMs) use throughout the task execution.

This leads to the objective function of the task scheduling model, i.e., to minimize the Time_{VM} and maximize the RU_{VM}. The fitness function is developed to determine how effectively the particle is positioned:

$$f = \text{Min}\left(\frac{\text{Time}_{VM}}{\text{RU}_{VM}}\right) \quad (3)$$

If there are M dimension particle $\text{P} = (\text{p}_1, \text{p}_2, \ldots, \text{p}_M)$, where p_j is the identification index of VM on which the j^{th} task is processed. An M dimension velocity "V_i" is defined in the same instance.

Apart from this, the inertia weight with updating positions and velocities is given by the following equations:

$$\text{q} = \left|\frac{f_a - f_b}{\max(f_a - f_b)}\right| \quad (4)$$

Vj^{q+1}=w. vx $V_j + 1 = qVj + i_1n_1 * (psobest - pj) + i_2n_2 * (pobest - pj), j$ is for iteration, where,

q = interia

i_1, i_2 = acceleration coefficients

n_1, n_2 = Random numbers distributed arbitrary

pobest = In a population, the best place of entire particles

B. GREY WOLF OPTIMIZATION

Patel et al. [56] proposed the GWO technique to balance the load dynamically. The hunting style of wolves inspires it. The algorithm adapts the main idea of how wolves live in four-level packs. Similarly, the load balancer is separated into four tiers, i.e. omega, delta, beta, and alpha (Figure 3). The modification of GWO is presented as Fuzzy GWO by Xingjun et al. [57]. To further improve the exploitation and exploration criteria. As wolves initially detect the overloaded, the system records all of the information while /hlconsidering

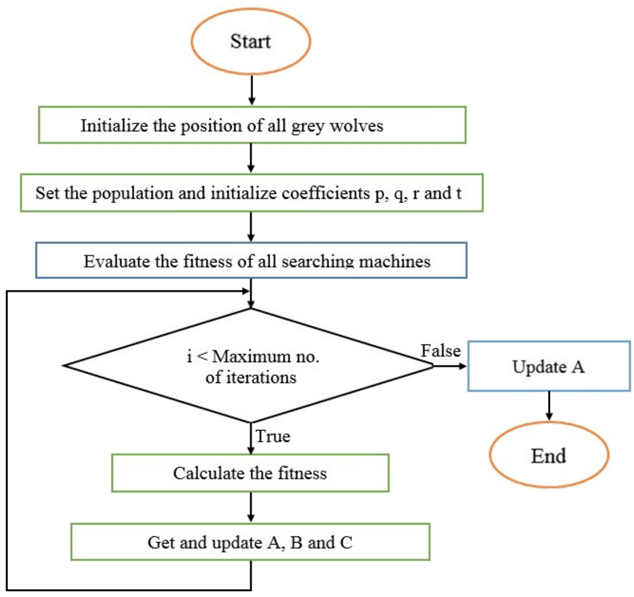


FIGURE 3. Flowchart of GWO.

the allocated demands to the VMs. The least loaded machine for assigning the allocation is identified whenever an assignment request is made, much as the wolves' second action is to determine the status. The approach is subsequently enhanced by incorporating fuzzy logic with GWO to stabilise the system.

Natesan and Chokkalingam [58] gather all workload and resource details and determine effective allocation controls. The scheduler is allotted a resource after examining the best option. The cloud load balancer refreshes the resource components based on processing time, space utilization, and connectivity parameter values. Ouhame et al. [59] present another Hybrid approach for resource allocation among VMs. The research presents a combined GWO-ACO algorithm. This technique improves the quality parameters and increases the efficiency up to 1.25%

The idea proposed in this research is to build a customized load-balancing method in cloud services and the Internet of Things. With a balanced load, the goal is to achieve a faster convergence with better response time in the proposed amalgamative PSO-GWO Algorithm.

1) THE BASIC MATHEMATICAL MODEL OF GWO FOR LOAD BALANCING IN CLOUD ENVIRONMENT

An acceptable solution is obtained in the form of Alpha (A). As a result, beta (B) comes in second, and delta (C) comes in third. Omega (O) is considered the best option for the remaining candidates and mandates optimization in the Optimization technique. Wolves are trailing these three wolves. The action surrounding the target can be demonstrated using the following equations:

$$D = |S \cdot P_x(t) - P(t)| \quad (5)$$

$$P(t+1) = P_x(t) - R * D \quad (6)$$

where,

t is the recent iteration

R and S are the coefficient vectors. P_x is the target position vector

P are the grey wolf position vector

The equations used to find vector R and vector S are given below:

$$R = 2 * a * r_1 - a \tag{7}$$

$$C = 2 * r_2 \tag{8}$$

As the number of iterations increases, a portion of "a" is decreased linearly from 2 to 0, while r1 and r_2 are random vectors that fall in the range [0, 1]. The ultimate scenario P (t+1) is determined by the search field locations of alpha, beta, and delta. The following expressions are used to describe these occurrences:

$$D_A = |S_1 P_A - P| \tag{9}$$

$$D_B = |S_2 P_B - P| \tag{10}$$

$$D_C = |S_3 P_C - P| \tag{11}$$

$$P_1 = |P_A R_1 - (D_A)| \tag{12}$$

$$P_2 = |P_B R_2 - (D_B)| \tag{13}$$

$$P_3 = |P_C R_3 - (D_C)| \tag{14}$$

$$P(t+1) = (P_1 P_2 P_3)/3 \tag{15}$$

If R>1, optimal solutions go farther from the objective, but if R=1, they converge.

Finally, the GWO algorithm comes to a close using the previously specified objective.

IV. PROPOSED ALGORITHM

PSO is suitable for global optimization but suffers from minimal local confinement. Exploration, exploitation, exclusion of local optima, and convergence are all powerful features of GWO. However, some of the balance between exploration and exploitation dynamics are still reliant on or constrained by it. As a result, the idea is to integrate both approaches to attain globally optimal minima without becoming trapped in local minima. It is recommended that the first GWO is utilized to generate the best location as alpha. Afterward, rather than determining the perfect location, PSO is carried out with the help of alpha.

As a result, the fundamental value added by this technique is the development of an objective function that effectively regulates load in a cloud infrastructure.

A. ALGORITHM OF PSO

The algorithm is designed by considering PSO as an SI-based optimization technique inspired by the collective behaviour of birds and shoals of fish. It mainly utilizes a community search technique to find the optimum solution to a problem by suspended particles in the search area. Particles in PSO float in a multidimensional search space, and each particle modifies its position throughout flying based on its knowledge and that of its neighbours.

Algorithm 1 PSO's Algorithm [60]

1) Initialize the dimension(d)
2) Set all the parameters to use to control the search
3) Initialize the population of particles with random position and velocity
4) Evaluate the fitness value for all the particles
    a) Continuously compare the fitness value (F) of each particle(P) with the recent particle's best (Rbest)
    b) If the current value is better than the last best(Pbest) value than switch the better value to the best and update the location
    c) Compare the fitness value (F) with the global best (Gbest)
    d) If the current value is better than the Gbest value then switch the better value to the Gbest and update the location
    e) Updater the position 'ai' and velocity 'vi' of the particle(P) as using:

       $v_i = w \cdot v_i + C_1 \cdot rand_1 \cdot (Pbest_d - Pbest_{p,d}) + C_2 \cdot rand_2 \cdot (Gbest_d - Rbest_{p,d})$

       $Rbest_{p,d} = v_i + Rbest_{p,d}$

       where C_1, C_2 = acceleration coefficients
       $rand_1$, $rand_2$ = random numbers distribute arbitrary
5) Repeat all the sub-steps of 3 until the stopping criteria met

B. ALGORITHM OF GWO

The algorithm is designed for the hunting style of the wolves. The strongest option for representing the social behaviour of the grey wolf is alpha, supported by beta and delta. Beta and delta follow alpha. In contrast, the remaining possibilities are categorized as omega. The alpha, beta, and delta wolves oversee the hunting process to optimise GWO, while the omega generally follows these three wolves.

GWO is the most refined model for discovering the best answer without getting stuck in early convergence, efficient local and global search, balanced investigation, and utilization of resources.

Algorithm 2 GWO's Algorithm

1) Set the population := 'Wi' for i = (1,2, ...,n) [Here, n is the population size]
2) Define A, B, and c
3) Evaluate the fitness values f(xi), for i = (1,2, ...,n)
4) Find X(alpha), X(beta) and X(gamma)
5) while (T<maxValue) [Here, maxValue is the maximum value of generations]
    a) for i=1 to n
        i) Update the position
        ii) Update A, B, and c
        iii) Calculate the f(xi)
    b) Update X(alpha), X(beta) and X(gamma)
6) X(alpha)

C. AMALGAMATIVE ALGORITHM OF GWO-PSO

Undoubtedly, PSO achieves promising global optimization, and at the same time, sticking to the local minima is unavoidable. The proposed algorithm is first using the grey Wolf technique, and this technique entirely runs the GWO Algorithm to generate 'Alpha' (A) as the best possible solution at the end. Then, the fitness value of the machines is re-evaluated as per the PSO policy. Then, all the comparisons of 'Pbest'

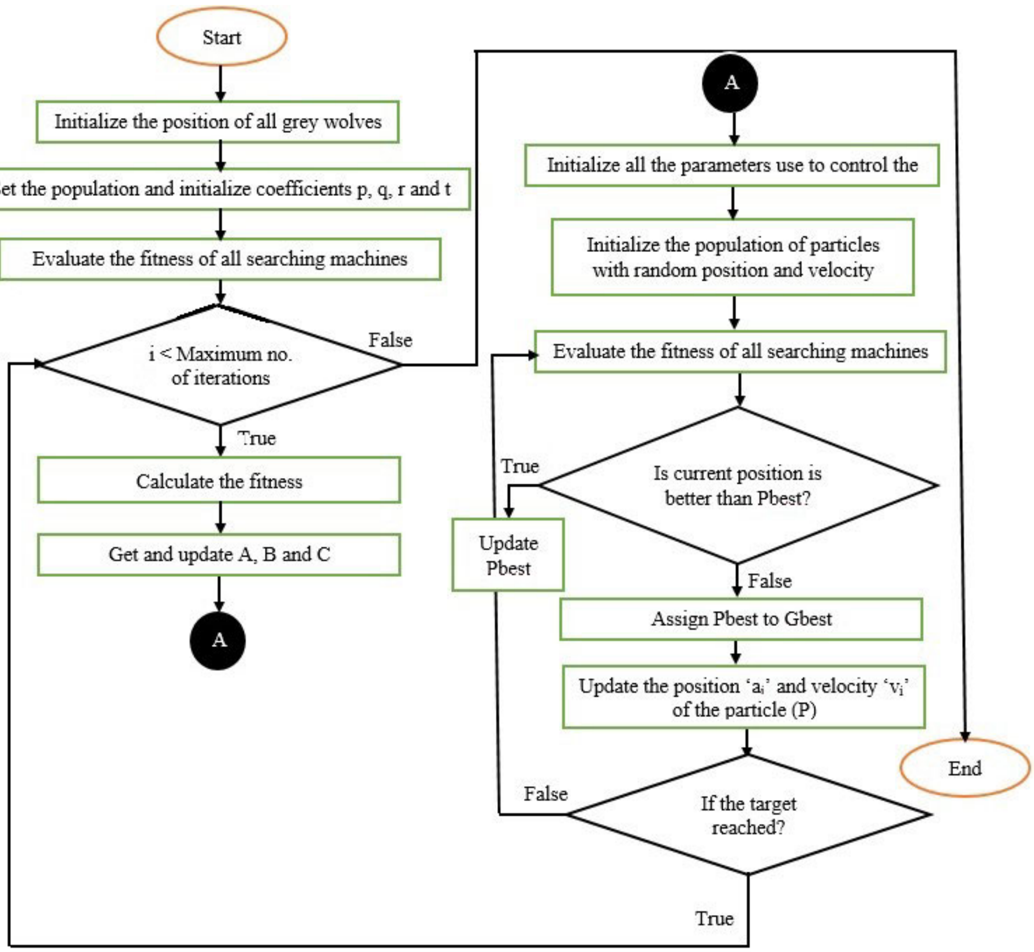


FIGURE 4. Flowchart of GWO-PSO algorithm.

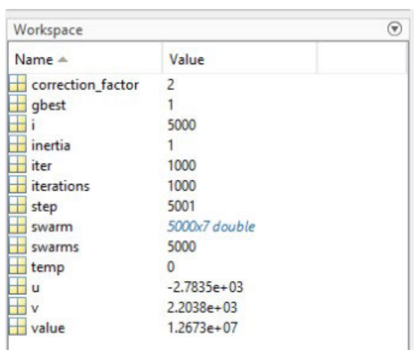
and 'Gbest' are performed. Finally, the position and velocity are evaluated, but it is not based on 'Pbest' this time.

GWO achieves the best solution as 'A' which is passed an input to update the velocity of the search agent by switching the 'Pbest' of PSO with 'A'. After that, all iterations complete the velocity that is updated again to achieve fast convergence and optimized load balancing (Figure 4).

V. SIMULATION RESULTS

Experiments are performed, and results are collected for these algorithms to measure the optimization and convergence of PSO, GWO, and amalgamative GWO-PSO. The experimentation is done using the MATLAB platform. Matlab is a Mathworks-developed scientific computer language that runs in interpretation mode on various operating systems. It's incredibly powerful, easy to use, and found in almost every research and engineering setting. With the inclusion of 'toolkits,' additional capabilities were added to the Matlab software by application developers for certain jobs or fields; therefore, the tool becomes increasingly dominant and customized.

Figure 5, shows the workspace parameters set to run PSO and globally optimized but bad convergence rate, respectively. This indicates that this method is not salable at all.


Workspace

Name	Value
correction_factor	2
gbest	1
i	5000
inertia	1
iter	1000
iterations	1000
step	5001
swarm	5000x7 double
swarms	5000
temp	0
u	-2.7835e+03
v	2.2038e+03
value	1.2673e+07

FIGURE 5. The values of control parameters of PSO.

Table 3 presents the analytical results to show convergence based on the value(s) of the objective function. The targeted algorithms are PSO, Social Spider(SSO), ABC, BAT, GWO, and the proposed GWO-PSO, with 30 agents (VMs). These analytical results are obtained based on the objective function. As can be seen that the proposed GWO-PSO is converging toward the value of 3.0 after 500 iterations. Figure 6 shows the convergence issue.

According to the results presented in Table 3. it can be concluded that the best optimal value obtained from the

Algorithm 3 GWO-PSO's Algorithm

1) Initialize the position of all grey wolves (total search agents: 'T')
2) Set the population as 'Wi' (i = 1,2, ...,n)
3) Initialization of coefficients p, q, r and t
4) Evaluate the fitness of all search machines
5) Let A= the best, B = second and C = third searching machine.
6) Start searching till the max no. of iteration
   a) Repeat for a search agent
      i) Update the position
      ii) Decrement the value of p from 2 to 0
      iii) Update the remaining coefficients
      iv) Calculate the fitness of each search agent
   b) Update A, B, and C
   c) Go for the next iteration and repeat until the maximum iteration is reached
7) Evaluate the best possible result 'A'
8) Evaluate the fitness value for all the particles
   a) Continuously compare the fitness value (F) of each particle(P) with the recent particle's best (Rbest)
   b) If the current value is better than the last best(Pbest) value then switch the better value to the best and update the location
   c) Compare the fitness value (F) with the global best (Gbest)
   d) If the current value is better than the Gbest value then switch the better value to the Gbest and update the location
   e) Updater the position 'ai' and velocity 'vi' of the particle(P) as using:

      $v_i = w \cdot v_i + C_1 \cdot rand_1 \cdot (A - Pbest_{p,d}) + C_2 \cdot rand_2 \cdot (Gbest_d - Rbest_{p,d})$

      $Rbest_{p,d} = v_i + Rbest_{p,d}$
9) Repeat all the sub-steps of 15 until the stopping criteria met
10) $v_i = w \cdot v_i + C_1 \cdot rand_1 \cdot (Pbest - Pbest_{p,d}) + C_2 \cdot rand_2 \cdot (Gbest_d - Rbest_{p,d})$

    $Rbest_{p,d} = v_i + Rbest_{p,d}$

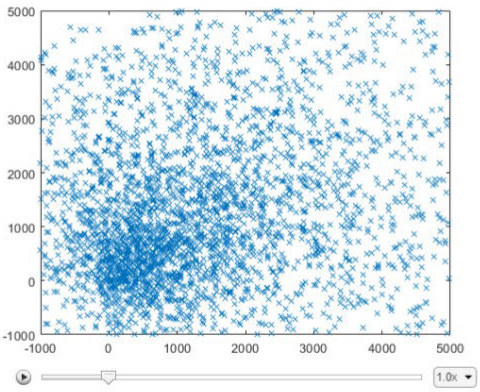


FIGURE 6. Good global optimization, but convergence issue is found.

objective function of the proposed GWO-PSO algorithm, improves PSO to 97.253% in terms of convergence by using the following equation:16

$$\%increase = \frac{|Newvalue - Originalvalue|}{OriginalValue} * 100 \quad (16)$$

Figure 7, 8, and 9 show the convergence evaluation of the Artificial Bee colony, Social Spider, and Bat Algorithms, respectively, with 30 agents (VMs).

GWO is proposed to make convergence possible, and the algorithm is also executed using MATLAB.

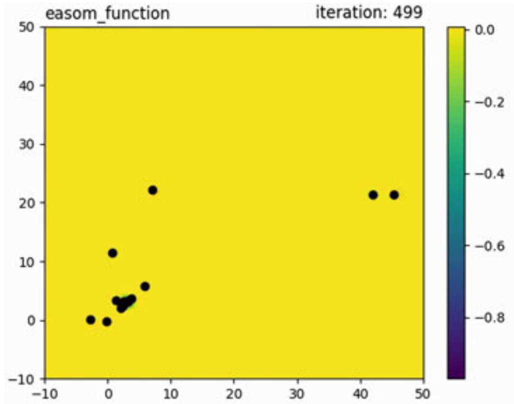


FIGURE 7. Slow convergence of Artificial Bee colony(ABC).

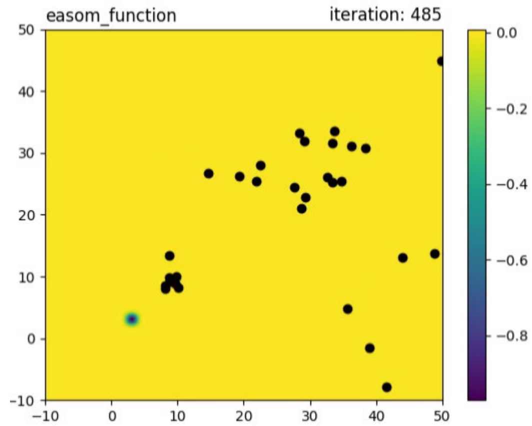


FIGURE 8. Slow convergence found in Social Spider(SSO).

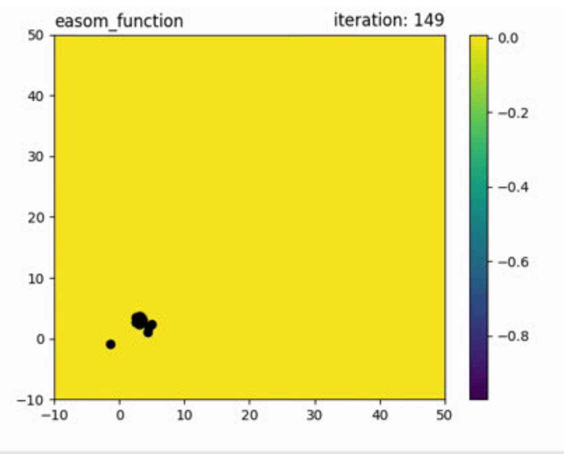


FIGURE 9. Convergence of Bat Algorithms is better than ABC and SSO techniques.

Figures 10 and 11 show the control parameters values and the resolved convergence issue, respectively.

The simulation results obtained from GWO support the theoretical explanation of the algorithm. The algorithm is converging, but with tremendous dynamic load increase in cloud infrastructure, it is necessary to make the algorithm converge faster. Therefore the amalgamative GWO-PSO is implemented, and simulations are performed. Figure 12

TABLE 3. Analytical results of simulation on the basis of value(s) of the objective function.

Targeted technique	Best Optimal Value	Average Value	Worst Value
GWO-PSO	3.001	3.0154	4.136
GWO	18.126	21.216	26.137
BAT	27.244	30.345	35.016
ABC	71.164	93.165	113.757
SSO	91.135	123.676	128.183
PSO	109.245	120.805	132.112

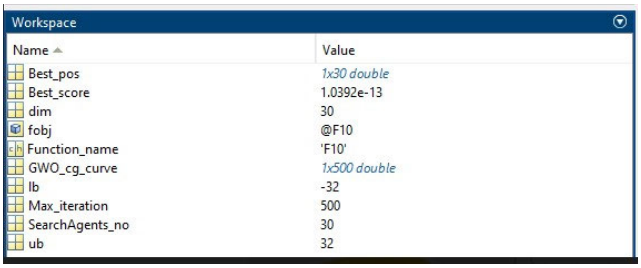


FIGURE 10. The values of control parameters of GWO.

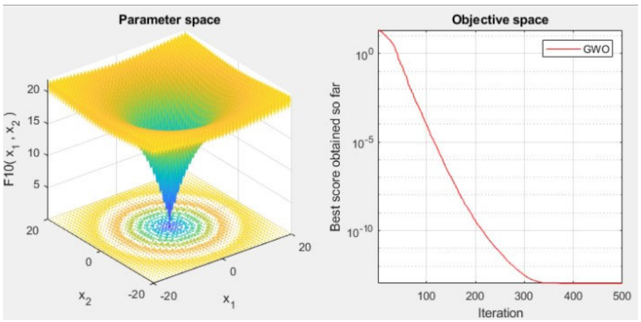


FIGURE 11. Better convergence found in GWO.

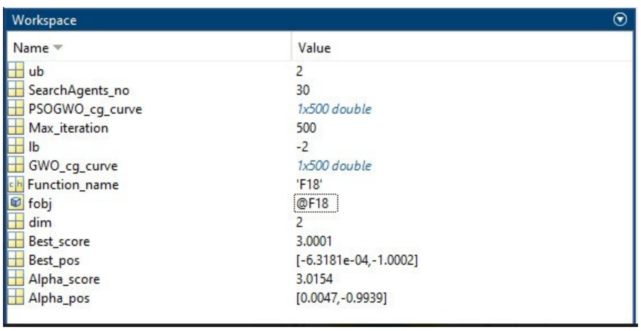


FIGURE 12. The values of control parameters of GWO-PSO.

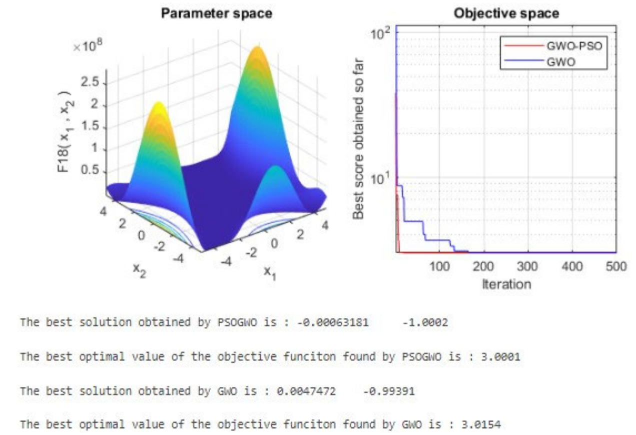


FIGURE 13. Fast convergence is achieved in GWO-PSO as compared to GWO.

shows the control parameters and fast convergence of the GWO-PSO approach compared to GWO.

Referring to Figure 6,7,8,9, 11 and 13, respectively, it is concluded that PSO (Figure 6) is promising in terms of global optimization, but convergence issue is found. ABC (Figure 7) and SSO (Figure 8) both have slow convergence that leads to delayed response time. Therefore, they are inefficient for load balancing when high user retention is high. Furthermore, it shows that these methods are neither scalable nor do they entertain concurrency. For more evidence, let's analyze the results of the BAT (Figure 9) algorithm. Its convergence is better than the other two, leading to a fast response time, but it is only efficient in the case of fewer agents (i.e., virtual machines). Hence, BAT can provide scalability but cannot facilitate concurrency when the number of users increases. Comparatively, GWO (Figure 11) outperforms to achieve the least response time with the best convergence. However, it can not perform global optimization to balance the load efficiently with the least response time. This is because the algorithm cannot utilize the available resources that need to be optimized. It can be concluded from Figure 13 that the hybrid approach achieved fast convergence with a more optimized solution. GWO-PSO is the technique through which the load is balanced by considering the globally optimized solution with the least response time. GWO-PSO also provides scalability and supports concurrency even when there is high user retention (i.e., fast convergence). Eventually, it supports the presented mathematical model of GWO and PSO.

Finally, a set of other experiments is performed to validate the research. Simulation and experimentation are the best way to test an algorithm in cloud computing for each VM's load balancing and scheduling algorithm. The CloudAnalyst tool is used, and the parameters of request per user per hour, data size per request (bytes), peak hours start(GMT), peak hours end (GMT), average Peak users, and average off-peak users are configured as 60, 100, equation, 9, 1000 and 100, respectively. To obtain the results under the created scenario, the values of user bases have been initialized as 10, 20, 30, and 50; data centres as 5; virtual machines as 10, respectively.

TABLE 4. Hypothesis t-test results for all the discussed PSO and GWO-based methods.

Method No.	P-value	Best Method	Method No.	Method
1 and 2	1	2	1	Hybrid GWO-PSO [41]
2 and 3	0.981027	3	2	Dynamic GWO-PSO [42]
3 and 4	0.998132	4	3	GWO based PSO [43]
4 and 5	0.999567	5	4	Hybridized $PSO_GWO1001[44]$
5 and 6	0	5	5	Our Proposed Method
5 and 7	0	5	6	Integrated PSO and GWO [45]
5 and 8	0	5	7	Tabu-GWO-ACO [46]
			8	Tabu-GWO-ACO [47]

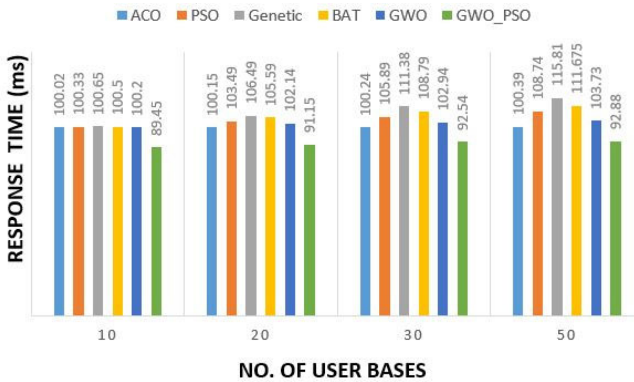


FIGURE 14. Comparative analysis of overall response time. (When User Bases=10, 20, 30 and 50 and Data Center=5.)

After analyzing the results, the findings conclude that based on overall response time, the performance of GWO-PSO outperforms all the other algorithms, as evident from Figure 14. Therefore, it has been concluded from analytical and simulation results that the best optimal value obtained from the objective function of the proposed GWO-PSO algorithm improves PSO to 97.253% in terms of convergence.

A. HYPOTHESIS TESTING AND T-TesT RESULTS

In this paper, we perform a t-test in the following manner. The t-statistic test examines two techniques at a time. The best is put up against the subsequent method, and so on. The P-value is generated for each assessment by the t-test. 0.5 is chosen as the major significance for comparing the two populations. The P value provides information about how effective a technique is. The first approach is superior to the second when the P-value is minimal (0-0.5), whereas the second method is superior to the first when the P-value is greater (0.5-1). Table 4 shows the P-values obtained from the t-distribution statistical tests for each discussed technique. A unique number is assigned for each method to test the methods. As can be seen, the proposed method outperforms all the other methods.

VI. CONCLUSIONS AND FUTURE WORK

This research introduced a hybrid fast-converging load balancing technique that accomplished globally optimal rapid convergence for balancing loads among VMs to keep the servers up and running. The implementation of GWO is straightforward because GWO hardly depends on the control parameters. Comparative analysis has shown that both ABC, PSO, and SSO algorithms have slow convergence, i.e., high response compared to BAT and GWO, but BAT degrades when user retention increases. Results have shown that PSO is appropriate for global optimization, and GWO is better for convergence. So to enable the possibility of making the convergence faster so that it's possible to converge at the globally optimized solution, PSO and GWO are combined. The scheduling algorithm is flexible enough to accommodate concurrent requests that are contending for VM due to the fast convergence. The simulation also supports this critical thinking, and the suggested algorithm works according to expectations.

The computational complexity of PSO depends on the number of iterations k, the size of the vector, and the number of particles so that it can be approximated as O(k*n). Similarly, the complexity of GWO depends on k, the size of the vector, and the number of search agents a. So the complexity of GWO can be approximated as O(k*a). Overall, the complexity of the proposed algorithm is O(k*n + k*a).

This section concludes the work by highlighting machine and deep learning potential for load balancing. The findings of this research indicate that there continue to be a lot of load-balancing mechanism concerns that are resolved in the long term by using an effective and intelligent task scheduling algorithm, especially in terms of extra QoS metrics and technique complexity evaluations. Given these factors, machine learning (and deep learning) are promising for enhancing the already existing and our suggested solutions. Machine learning algorithms have been effectively used in various industries, including manufacturing, pattern identification, and language modelling [61]. However, for deep learning to be successfully utilised, its variables must be properly configured to produce sufficient results. A deep machine learning network's effectiveness is primarily affected by two important parameters: the number of hidden layers and the number of neurons in each layer. The users' jobs selecting these crucial parameters are considerably easier by manual parameter configuration and grid search procedures. The suggested algorithm tries to improve the probability that load balancing will succeed by achieving fast convergence, a short response time, increased throughput, and a low execution time, but not accuracy. Machine learning, in contrast, only focuses on accuracy and does not care about achievement or success. In that, The same outcomes can be produced with guaranteed accuracy by combining the suggested technique with machine learning. This work can be extended to evaluate the quality parameters in the future. Here, another point of consideration is that the basic GWO algorithm balances the load so that half of the iterations are set aside for exploration. In contrast, the other half is set aside for exploitation. In GWO, the ideal balance of exploration and exploitation is neglected, and a closer solution can guarantee the impact of the perfect balance between these two. A modified version of GWO could be proposed to address these issues, focusing on the essential realistic balance between exploration and exploitation. The ultimate goal in the future is to integrate the upgraded Grey wolf algorithm with PSO to produce even better outcomes. The suggested method could also benefit from other technologies like the Internet of Things (IoT), fog computing, prediction systems, etc. Therefore, the suggested plan is to use this technique to optimize the smart city concept by including cognitive IoT.

DATA AVAILABILITY

A standalone online version of the algorithm is made available at GitHub at the following URL: https://github.com/DrNomanIslam/Amalgamated-load-balancing

ACKNOWLEDGMENT

The authors are thankful to the Deanship of Scientific Research at Najran University for funding this work under the National Research Priorities funding program grant code (NU/NRP/SERC/11/18).

CONFLICTS OF INTEREST

The authors declared that they have no conflicts of interest to report regarding the present study.

REFERENCES

[1] D. Syed, U. B. Masood, U. E. Farwa, and M. Khurram, "Cloud based smart irrigation for agricultural area of Pakistan," *Comput. Eng. Appl. J.*, vol. 4, no. 3, pp. 153–164, Oct. 2015.

[2] A. Garg, K. Patidar, G. K. Saxena, and M. Jain, "A literature review of various load balancing techniques in cloud computing environment," *Int. J. Enhanced Res. Manag. Comput. Appl.*, vol. 5, no. 2, pp. 11–14, 2006.

[3] R. Nema, "A review: Load balancing algorithm using cloud analyst environment," *Int. J. Sci. Manage. Eng. Res. (IJSMER)*, vol. 1, no. 5, pp. 192–197, 2016.

[4] S. Joshi and U. Kumari, "Load balancing in cloud computing: Challenges & issues," in *Proc. 2nd Int. Conf. Contemp. Comput. Informat. (IC3I)*, Dec. 2016, pp. 120–125.

[5] P. Kumar and R. Kumar, "Issues and challenges of load balancing techniques in cloud computing: A survey," *ACM Comput. Surv.*, vol. 51, no. 6, pp. 1–35, Nov. 2019.

[6] J. O. Gutierrez-Garcia and A. Ramirez-Nafarrate, "Collaborative agents for distributed load management in cloud data centers using live migration of virtual machines," *IEEE Trans. Services Comput.*, vol. 8, no. 6, pp. 916–929, Nov. 2015.

[7] S. Jain and A. K. Saxena, "A survey of load balancing challenges in cloud environment," in *Proc. Int. Conf. Syst. Model. Advancement Res. Trends (SMART)*, 2016, pp. 291–293.

[8] R. Z. Khan and M. O. Ahmad, "Load balancing challenges in cloud computing: A survey," in *Proc. Int. Conf. Signal, Netw., Comput., Syst.* Cham, Switzerland: Springer, 2016, pp. 25–32.

[9] P. Arya and P. Tripathi, "Various issues & challenges of load balancing over cloud: A survey," *Int. J. Eng. Comput. Sci.*, vol. 5, no. 8, pp. 17517–17524, 2016.

[10] V. R. Kanakala, V. K. Reddy, and K. Karthik, "Performance analysis of load balancing techniques in cloud computing environment," in *Proc. IEEE Int. Conf. Electr., Comput. Commun. Technol. (ICECCT)*, Mar. 2015, pp. 1–6.

[11] R. Kumar and T. Prashar, "Performance analysis of load balancing algorithms in cloud computing," *Int. J. Comput. Appl.*, vol. 120, no. 7, pp. 19–27, Jun. 2015.

[12] M. Awan and M. A. Shah, "A survey on task scheduling algorithms in cloud computing environment," *Int. J. Comput. Inf. Technol.*, vol. 4, no. 2, pp. 441–448, 2015.

[13] S. Ouhame and Y. Hadi, "A hybrid grey wolf optimizer and artificial bee colony algorithm used for improvement in resource allocation system for cloud technology," *Int. J. Online Biomed. Eng. (iJOE)*, vol. 16, no. 14, pp. 4–17, 2020.

[14] M. A. Mosa, A. S. Anwar, and A. Hamouda, "A survey of multiple types of text summarization with their satellite contents based on swarm intelligence optimization algorithms," *Knowl.-Based Syst.*, vol. 163, pp. 518–532, Jan. 2019.

[15] A. Ullah, N. M. Nawi, J. Uddin, S. Baseer, and A. H. Rashed, "Artificial bee colony algorithm used for load balancing in cloud computing: Review," *IAES Int. J. Artif. Intell. (IJ-AI)*, vol. 8, no. 2, pp. 156–167, Jun. 2019.

[16] G. Muthsamy and S. R. Chandran, "Task scheduling using artificial bee foraging optimization for load balancing in cloud data centers," *Comput. Appl. Eng. Educ.*, vol. 28, no. 4, pp. 769–778, Jul. 2020.

[17] J.-Q. Li and Y.-Q. Han, "A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system," *Cluster Comput.*, vol. 23, no. 4, pp. 2483–2499, Dec. 2020.

[18] A. Saadat and E. Masehian, "Load balancing in cloud computing using genetic algorithm and fuzzy logic," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2019, pp. 1435–1440.

[19] S. Ahmed, K. H. Sheikh, S. Mirjalili, and R. Sarkar, "Binary simulated normal distribution optimizer for feature selection: Theory and application in COVID-19 datasets," *Expert Syst. Appl.*, vol. 200, Aug. 2022, Art. no. 116834.

[20] Y. Yuan, X. Mu, X. Shao, J. Ren, Y. Zhao, and Z. Wang, "Optimization of an auto drum fashioned brake using the elite opposition-based learning and chaotic k-best gravitational search strategy based grey wolf optimizer algorithm," *Appl. Soft Comput.*, vol. 123, Jul. 2022, Art. no. 108947.

[21] W. Zhao, L. Wang, and S. Mirjalili, "Artificial hummingbird algorithm: A new bio-inspired optimizer with its engineering applications," *Comput. Methods Appl. Mech. Eng.*, vol. 388, Jan. 2022, Art. no. 114194.

[22] V. K. Ojha, A. Abraham, and V. Snášel, "Metaheuristic design of feedforward neural networks: A review of two decades of research," *Eng. Appl. Artif. Intell.*, vol. 60, pp. 97–116, Apr. 2017.

[23] P. K. Sa, M. N. Sahoo, M. Murugappan, Y. Wu, and B. Majhi, *Progress in Intelligent Computing Techniques: Theory, Practice, and Applications: Proceedings of ICACNI 2016*, vol. 2. Cham, Switzerland: Springer, 2017.

[24] H. A. Makasarwala and P. Hazari, "Using genetic algorithm for load balancing in cloud computing," in *Proc. 8th Int. Conf. Electron., Comput. Artif. Intell. (ECAI)*, Jun. 2016, pp. 1–6.

[25] S. Kaur and J. Sengupta, "Load balancing using improved genetic algorithm (IGA) in cloud computing," *Int. J. Adv. Res. Comput. Eng. Technol. (IJARCET)*, vol. 6, no. 8, pp. 1323–2278, 2017.

[26] J. O. Gutierrez-Garcia and A. Ramirez-Nafarrate, "Collaborative agents for distributed load management in cloud data centers using live migration of virtual machines," *IEEE Trans. Serv. Comput.*, vol. 8, no. 6, pp. 916–929, 2015.

[27] A. Saadat and E. Masehian, "Load balancing in cloud computing using genetic algorithm and fuzzy logic," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2019, pp. 1435–1440.

[28] A. K. Jayswal and P. C. Saxena, "Multi-agent genetic algorithm for efficient load balancing in cloud computing," *Int. J. Innov. Technol. Exploring Eng.*, vol. 9, no. 4, pp. 45–51, Feb. 2020.

[29] G. Li and Z. Wu, "Ant colony optimization task scheduling algorithm for SWIM based on load balancing," *Future Internet*, vol. 11, no. 4, p. 90, Apr. 2019.

[30] A. Ragmani, A. Elomri, N. Abghour, K. Moussaid, and M. Rida, "An improved hybrid fuzzy-ant colony algorithm applied to load balancing in cloud computing environment," *Proc. Comput. Sci.*, vol. 151, pp. 519–526, Jan. 2019.

[31] M. A. Mosa, A. S. Anwar, and A. Hamouda, "A survey of multiple types of text summarization with their satellite contents based on swarm intelligence optimization algorithms," *Knowl.-Based Syst.*, vol. 163, pp. 518–532, Jan. 2019.

[32] A. Ullah, N. M. Nawi, J. Uddin, S. Baseer, and A. H. Rashed, "Artificial bee colony algorithm used for load balancing in cloud computing: Review," *Int. J. Artif. Intell. (IJ-AI)*, vol. 8, no. 2, pp. 156–167, Jun. 2019.

[33] G. Muthsamy and S. R. Chandran, "Task scheduling using artificial bee foraging optimization for load balancing in cloud data centers," *Comput. Appl. Eng. Educ.*, vol. 28, no. 4, pp. 769–778, Jul. 2020.

[34] J.-Q. Li and Y.-Q. Han, "A hybrid multi-objective artificial bee colony algorithm for flexible task scheduling problems in cloud computing system," *Cluster Comput.*, vol. 23, no. 4, pp. 2483–2499, Dec. 2020.

[35] L. Shen, J. Li, Y. Wu, Z. Tang, and Y. Wang, "Optimization of artificial bee colony algorithm based load balancing in smart grid cloud," in *Proc. IEEE Innov. Smart Grid Technol. Asia (ISGT Asia)*, May 2019, pp. 1131–1134.

[36] A. V and N. Bhalaji, "Load balancing in cloud computing using water wave algorithm," *Concurrency Comput., Pract. Exper.*, vol. 34, no. 8, Apr. 2022, Art. no. e5492.

[37] R. Gulbaz, A. B. Siddiqui, N. Anjum, A. A. Alotaibi, T. Althobaiti, and N. Ramzan, "Balancer genetic algorithm—A novel task scheduling optimization approach in cloud computing," *Appl. Sci.*, vol. 11, no. 14, p. 6244, Jul. 2021.

[38] Z. Miao, P. Yong, Y. Mei, Y. Quanjun, and X. Xu, "A discrete PSO-based static load balancing algorithm for distributed simulations in a cloud environment," *Future Gener. Comput. Syst.*, vol. 115, pp. 497–516, Feb. 2021.

[39] H. S. Alatawi and S. A. Sharaf, "Hybrid load balancing approach based on the integration of QoS and power consumption in cloud computing," *Int. J.*, vol. 10, no. 2, pp. 1–12, 2021.

[40] M. A. Elmagzoub, D. Syed, A. Shaikh, N. Islam, A. Alghamdi, and S. Rizwan, "A survey of swarm intelligence based load balancing techniques in cloud computing environment," *Electronics*, vol. 10, no. 21, p. 2718, Nov. 2021.

[41] M. A. M. Shaheen, H. M. Hasanien, and A. Alkuhayli, "A novel hybrid GWO-PSO optimization technique for optimal reactive power dispatch problem solution," *Ain Shams Eng. J.*, vol. 12, no. 1, pp. 621–630, Mar. 2021.

[42] S. Chtita, S. Motahhir, A. El Hammoumi, A. Chouder, A. S. Benyoucef, A. El Ghzizal, A. Derouich, M. Abouhawwash, and S. S. Askar, "A novel hybrid GWO–PSO-based maximum power point tracking for photovoltaic systems operating under partial shading conditions," *Sci. Rep.*, vol. 12, no. 1, pp. 1–15, Jun. 2022.

[43] B. N. Gohil and D. R. Patel, "A hybrid GWO-PSO algorithm for load balancing in cloud computing environment," in *Proc. 2nd Int. Conf. Green Comput. Internet Things (ICGCIoT)*, Aug. 2018, pp. 185–191.

[44] A. A. Alabdalbari and I. A. Abed, "New robot path planning optimization using hybrid GWO-PSO algorithm," *Bull. Electr. Eng. Informat.*, vol. 11, no. 3, pp. 1289–1296, Jun. 2022.

[45] A. M. Senthil Kumar, P. Krishnamoorthy, S. Soubraylu, J. K. Venugopal, and K. Marimuthu, "An efficient task scheduling using GWO-PSO algorithm in a cloud computing environment," in *Proc. Int. Conf. Intell. Comput., Inf. Control Syst.* Cham, Switzerland: Springer, 2021, pp. 751–761.

[46] M. Kaur and R. Aron, "FOCALB: Fog computing architecture of load balancing for scientific workflow applications," *J. Grid Comput.*, vol. 19, no. 4, pp. 1–22, Dec. 2021.

[47] R. Kaur and D. K. S. Dhindsa, "Efficient task scheduling using load balancing in cloud computing," *Int. J. Adv. Netw. Appl.*, vol. 10, no. 3, pp. 3888–3892, 2018.

[48] R. Agarwal, N. Baghel, and M. A. Khan, "Load balancing in cloud computing using mutation based particle swarm optimization," in *Proc. Int. Conf. Contemp. Comput. Appl. (IC3A)*, Feb. 2020, pp. 191–195.

[49] M. M. Golchi, S. Saraeian, and M. Heydari, "A hybrid of firefly and improved particle swarm optimization algorithms for load balancing in cloud environments: Performance evaluation," *Comput. Netw.*, vol. 162, Oct. 2019, Art. no. 106860.

[50] K. Pan and J. Chen, "Load balancing in cloud computing environment based on an improved particle swarm optimization," in *Proc. 6th IEEE Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Sep. 2015, pp. 595–598.

[51] A. R. Jordehi and J. Jasni, "Particle swarm optimisation for discrete optimisation problems: A review," *Artif. Intell. Rev.*, vol. 43, no. 2, pp. 243–258, 2015.

[52] R. M. A. Guliyev, Y. N. Imamverdiyev, and F. J. Abdullayeva, "PSO-based load balancing method in cloud computing," *Autom. Control Comput. Sci.*, vol. 53, no. 1, pp. 44–55, 2019.

[53] R. Agarwal, N. Baghel, and M. A. Khan, "Load balancing in cloud computing using mutation based particle swarm optimization," in *Proc. Int. Conf. Contemp. Comput. Appl. (IC3A)*, Feb. 2020, pp. 191–195.

[54] K. Pan and J. Chen, "Load balancing in cloud computing environment based on an improved particle swarm optimization," in *Proc. 6th IEEE Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Sep. 2015, pp. 595–598.

[55] Z. Liu and X. Wang, "A PSO-based algorithm for load balancing in virtual machines of cloud computing environment," *Adv. Swarm Intell.*, vol. 7331, pp. 142–147, Apr. 2021.

[56] D. Patel, M. K. Patra, and B. Sahoo, "GWO based task allocation for load balancing in containerized cloud," in *Proc. Int. Conf. Inventive Comput. Technol. (ICICT)*, Feb. 2020, pp. 655–659.

[57] L. Xingjun, S. Zhiwei, C. Hongping, and B. O. Mohammed, "A new fuzzy-based method for load balancing in the cloud-based Internet of Things using a grey wolf optimization algorithm," *Int. J. Commun. Syst.*, vol. 33, no. 8, p. e4370, May 2020.

[58] G. Natesan and A. Chokkalingam, "An improved grey wolf optimization algorithm based task scheduling in cloud computing environment," *Int. Arab J. Inf. Technol.*, vol. 17, pp. 73–81, Jan. 2020.

[59] S. Ouhame, Y. Hadi, and A. Arifullah, "A hybrid grey wolf optimizer and artificial bee colony algorithm used for improvement in resource allocation system for cloud technology," *Int. J. Online Biomed. Eng. (iJOE)*, vol. 16, pp. 4–17, Nov. 2020.

[60] M. Kalra and S. Singh, "A review of metaheuristic scheduling techniques in cloud computing," *Egyptian Informat. J.*, vol. 16, no. 3, pp. 275–295, 2015.

[61] B. Qolomany, M. Maabreh, A. Al-Fuqaha, A. Gupta, and D. Benhaddou, "Parameters optimization of deep learning models using particle swarm optimization," in *Proc. 13th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2017, pp. 1285–1290.

MANA SALEH AL RESHAN received the B.S. degree in information systems from King Khalid University, Abha, Saudi Arabia, in 2007, the M.S. degree (Hons.) in computer, information, and network security from DePaul University, Chicago, USA, in 2011, and the Ph.D. degree in computer science from The Catholic University of America (CUA), Washington, DC, USA, in 2019. He was a Teaching Assistant at the College of Computer Science and Information Systems, Najran University, Saudi Arabia, from 2007 to 2009, a Lecturer, in 2012, and where he is currently an Assistant Professor and the Head of the Network Engineering Department. His current research interests include computer network and security, system security, wireless and mobile security, body area networks, and cloud security.

DARAKHSHAN SYED received the Master of Engineering (M.Eng.) degree in computer networking and performance evaluation from the NED University of Engineering and Technology. She is currently pursuing the Ph.D. degree with the Computer Science Department, Bahria University, Karachi. She is currently working as a Lecturer with the Computer Science Department, Bahria University. She is an experienced Tutor of computer courses such as Programming Languages, Object-Oriented Programming, Web Programming Languages, Information Security, Cryptography and Data Security, Introduction to Computer Science and Information Technology, Professional Practices, Artificial Intelligence, Software Engineering, Microprocessors, and Mobile Application and Development. She was a research assistant with a demonstrated history of working in the education management industry. She is skilled in Microsoft Word, Proteus, C++, Technical Documentation, and Raspberry Pi.

NOMAN ISLAM received the B.S. degree in computer science from the University of Karachi, and the M.S. and Ph.D. degrees in computer science from FAST-NUCES, Karachi. He was a Postdoctoral Research Fellow at the University of Kuala Lumpur, Malaysia. He is currently working as a Professor with the Karachi Institute of Economics and Technology (KIET). He has published more than 50 research publications in reputable conferences and journals. His research interests include wireless/ad hoc networks, deep learning, and cloud computing.

ASADULLAH SHAIKH (Senior Member, IEEE) received the B.Sc. degree in software development from the University of Huddersfield, U.K., the M.Sc. degree in software engineering and management from Goteborg University, Sweden, and the Ph.D. degree in software engineering from the University of Southern Denmark. He was a Researcher at UOC Barcelona, Spain. He is currently a Professor, the Head of research and graduate studies, and the Coordinator of seminars and training with the College of Computer Science and Information Systems, Najran University, Najran, Saudi Arabia. He has more than 140 publications in the area of software engineering in international journals and conferences. He has vast experience in teaching and research. His current research interests include UML model verification, UML class diagrams verification with OCL constraints for complex models, formal verification, and feedback technique for unsatisfiable UML/OCL class diagrams. He is an Editor of the *International Journal of Advanced Computer Systems and Software Engineering (IJACSSE)* and on the International Advisory Board of several conferences and journals. Further details can be obtained using www.asadshaikh.com.

MOHAMMED HAMDI received the bachelor's degree in computer science from Jazan University, Jazan, Saudi Arabia, in 2009, and the Master of Science and the Doctor of Philosophy (Ph.D.) degrees in computer science from Sothern Illinois University-Carbondale, IL, USA, in 2013 and 2018, respectively. He is currently an Associate Professor and the Dean with the College of Computer Science and Information System, Najran University. His research interests include databases, query optimization, data mining, big data, and security.

MOHAMED A. ELMAGZOUB was born in Riyadh, Saudi Arabia, in 1985. He received the B.S. degree (Hons.) in electrical and electronic engineering from Omdurman Islamic University, Sudan, in 2007, and the master's degree in electrical communication engineering and the Ph.D. degree in electrical engineering (communications) from Universiti Teknologi Malaysia (UTM), Malaysia, in 2012 and 2016, respectively. He was worked as a Telecommunication Engineer at the Communication Department, Dams Implementation Unit, Sudan, from 2008 to 2009. He is currently an Assistant Professor with the Department of Network and Communication Engineering, Najran University. He has authored ten ISI papers, two of them are highly cited review articles in the area of passive optical networks and hybrid optical, and wireless communication systems. His research interests include passive optical networks, hybrid optical and wireless communication systems, optical OFDM communication, MIMO, and next-generation access networks.

GHULAM MUHAMMAD received the B.E. degree in computer systems and the M.E. degree in communication systems and networks from the Mehran University of Technology, Jamshoro, and the Ph.D. degree in parallel processing from the Beijing Institute of Technology, Beijing, China. He joined the Computer Science Department, Bahria University Karachi Campus, as an Associate Professor. After the Ph.D., he has published more than 25 research articles in different international and nationally recognized journals. He is an Expert in multi-discipline domains such as cloud computing, networking, artificial intelligence, machine learning, SDN, the Internet of Things (IoT), and robotics. He conducts workshops on cloud computing.

KASHIF HUSSAIN TALPUR received the B.S. degree in information technology from the Mehran University of Engineering and Technology, Jamshoro, Pakistan, in 2006, and the M.S. and Ph.D. degrees in information technology from the Universiti Tun Hussein Onn Malaysia, Johor, Malaysia, in 2015 and 2018, respectively. He has also served as a Postdoctoral Researcher at the Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, China. He is currently with Bahria University, Karachi, Pakistan, as an Associate Professor. His major research interests include machine learning, optimization, fuzzy neural networks, and metaheuristic algorithms.