Hybrid Enhanced Optimization-Based Intelligent Task Scheduling for Sustainable Edge Computing

Mohamed Abd Elaziz, Ibrahim Attiya, Laith Abualigah, Muddesar Iqbal, Amjad Ali, Ala Al-Fuqaha, *Senior Member, IEEE*, and Shaker El-Sappagh

***Abstract*—The demand for task scheduling in Internet of Things (IoT)-based edge and cloud computing environments is experiencing exponential growth due to the need to address real-world issues, such as load instability, slow convergence rates, and under-utilization of virtual machine devices. In this paper, a hybrid enhanced optimization method called RFOAOA is designed to solve challenging task scheduling scenarios in edge-cloud computing-based IoT environments. The proposed method leverages the strengths of two powerful search operators, such as Red Fox Optimization (RFO) and Arithmetic Optimization Algorithm (AOA). To evaluate the effectiveness of the proposed method, we conducted experiments on real and synthetic workload traces of NASA Ames iPSC/860 and HPC2N. The comparative analysis demonstrates that the proposed algorithm achieves better performance in terms of Makespan time and energy consumption and outperforms the other state-of-the-art scheduling methods.**

***Index Terms*—Swarm intelligence, red fox optimization, task scheduling, edge intelligence, sustainable edge computing.**

Manuscript received 4 March 2023; revised 15 June 2023; accepted 11 September 2023. Date of publication 2 October 2023; date of current version 26 April 2024. This work was supported by the Academy of Scientific Research and Technology (ASRT), Egypt, under Grant REPECT10069. *(Corresponding authors: Muddesar Iqbal; Ala Al-Fuqaha.)*

Mohamed Abd Elaziz is with the Faculty of Computer Science and Engineering, Galala university, Suez 435611, Egypt, also with the Department of Mathematics, Faculty of Science, Zagazig University, Zagazig 44519, Egypt, and also with the Academy of Scientific Research and Technology (ASRT), Cairo, Egypt (e-mail: abd_el_aziz_m@yahoo.com).

Ibrahim Attiya is with the Department of Mathematics, Faculty of Science, Zagazig University, Zagazig 44519, Egypt, also with the Academy of Scientific Research and Technology (ASRT), Cairo, Egypt, and also with the Faculty of Computer Science and Engineering, New Mansoura University, New Mansoura, Egypt (e-mail: ibrahimateya@zu.edu.eg).

Laith Abualigah is with the Computer Science Department, Al Al-Bayt University, Mafraq 25113, Jordan, also with the Department of Electrical and Computer Engineering, Lebanese American University, Byblos 13-5053, Lebanon, also with the Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan, also with the MEU Research Unit, Middle East University, Amman 11831, Jordan, and also with the Applied Science Research Center, Applied Science Private University, Amman 11931, Jordan (e-mail: Aligah.2020@gmail.com).

Muddesar Iqbal is with the Renewable Energy Laboratory, Communications and Networks Engineering Department, College of Engineering, Prince Sultan University, Riyadh 11586, Saudi Arabia (e-mail: miqbal@psu.edu.sa).

Amjad Ali and Ala Al-Fuqaha are with the Division of Information and Computing Technology, College of Science and Engineering, Hamad Bin Khalifa University, Qatar Foundation, Doha, Qatar (e-mail: amjad.khu@gmail.com; aalfuqaha@hbku.edu.qa).

Shaker El-Sappagh is with the Faculty of Computer Science and Engineering, Galala University, Suez 435611, Egypt, and also with the Information Systems Department, Faculty of Computers and Artificial Intelligence, Benha University, Banha 13511, Egypt (e-mail: shaker.elsappagh@gu.edu.eg).

Digital Object Identifier 10.1109/TCE.2023.3321783

I. Introduction

CLOUD Computing (CC) has gained widespread adoption across various disciplines [1]. However, the rise of the Internet of Things (IoT) has introduced several challenges related to CC. Specifically, CC struggles to efficiently handle the growing demand for fine-tuned and location-aware services in IoT systems, primarily due to its limited scalability and high development costs. As the number of IoT devices increases, a massive amount of data needs to be processed at data stations [2]. Applying the traditional CC model to handle this huge amount of data would result in significant latency and system congestion. To address these issues, fog computing (FC) is introduced by Cisco [3] and further discussed in [4]. FC serves as an intermediary by providing just-in-time computing resources between cloud hubs and IoT devices [5]. It extends the capabilities of CC to overcome the limitations associated with by providing decentralized data storage and computing facilities at the edge of the network, enabling real-time processing, and reducing the burden on data stations [6]. The core advantages of FC include localized processing, reduced data transmission over the Internet, and enhanced security measures that instill user confidence [7]. However, it require efficient tasks assignment to devices within the IoT fog-cloud computing ecosystem, considering the various ranges of capabilities and capacities [8], [9].

The IoT paradigm has become pervasive across various aspects of our daily lives [10], and the efficient operation of IoT applications often relies on timely and accurate responses, such as gaming, virtual reality, and autonomous vehicle movement [11]. To minimize response times, task offloading techniques have been proposed, which involve leveraging edge computing to perform IoT tasks [12], [13]. In common, task scheduling has two main steps; in the first one, the tasks of an application are located in a logical distribution [14]. The second step aims to map tasks to possible computing devices on the given system produced by obtaining computational time or Makespan of the employment [15]. Task scheduling is studied as a optimization problem in disseminated computing models [16] and identified as an NP-hard problem [17]. In [18], a fog-based computing scheduler method is proposed. The results indicate a 32% improvement in network performance compared to the First-Come-First-Served approach. To further enhance the quality of service for users, an energy-aware task scheduling scheme for FC is introduced that exploits the Marine

Predators' optimizer concept [19]. Further, to reduce the Makespan and total time in a FC ecosystem, a multi-objective task scheduling method is introduced in [20]. Additionally, a scheduling method is presented in [21] to balance application performance and the cost of utilizing cloud devices. However, in IoT applications, the trade-off between the Makespan and time cost becomes crucial when scheduling multiple tasks.

The rapid growth of IoT, edge, and cloud computing environments has led to an increased demand for efficient task scheduling algorithms. These environments face various challenges, including load instability, slow convergence rates, and under-utilization of virtual machine devices, which hinder the optimal utilization of resources and the timely execution of tasks. Addressing these challenges is crucial to ensure the effective functioning of IoT-based systems in real-world scenarios. Therefore, in this paper, we propose a hybrid enhanced optimization method called RFOAOA to tackle the complex task scheduling scenarios in edge-cloud computing-based IoT environments. The RFOAOA method leverages the strengths of two powerful search operators, namely, Red Fox Optimization (RFO) and Arithmetic Optimization Algorithm (AOA), to efficiently allocate tasks and resources. To evaluate the effectiveness of the proposed method, extensive experiments are conducted under various workload scenarios of varying sizes. These experiments aimed to measure the performance of RFOAOA in terms of Makespan time and energy consumption. Furthermore, a comparative analysis is presented to assess the superiority of the proposed method against other state-of-the-art scheduling methods. The main contribution of this paper can be summarized as follows:

- We propose a hybrid enhanced optimization method called RFOAOA to tackle the complex task scheduling in edge-cloud based IoT environments. The RFOAOA leverages the strengths of two powerful search operators, namely, RFO and AOA, to efficiently allocate tasks and resources.
- A detailed comparative analysis is presented to assess the superiority of the proposed method against state-of-the-art including Marine Predators Algorithm (MPA), RFO, Artificial Ecosystem-based Optimization (AEO), Salp Swarm Algorithm (SSA), Whale Optimization Algorithm (WOA), and Particle Swarm Optimization (PSO) in terms of Makespan time and energy consumption.
- The effectiveness of proposed algorithm is validated via real and synthetic workload traces of NASA Ames iPSC/860 and HPC2N of "Parallel Workload Archive" repository [22].

The arrangement of the rest of sections is as follows. Section II presents the related work. Section III provides system model for our task scheduling problem. Section IV provides a background study related to RFO and AOA and discussed our proposed intelligent task scheduling scheme in detail. Section V presents the simulation results and discussions. Finally, Section VI summarises the paper and presents some future works.

II. Related Work

In this section we discuss the state-of-the-art related to task scheduling in IoT edge computing. The task scheduling problem in fog-cloud computing environments has emerged as a significant challenge due to the increasing influx of big data streams from IoT devices. With the high number of user applications and devices in this era, an efficient scheduling technique is crucial to effectively allocate tasks among available resources. Such efficient methods can optimize resource utilization, reduce execution time, and enhance overall system performance. Thus, various methods have been introduced in the literature to solve the task scheduling problem and to find the efficient solutions for different edge-based IoT applications.

To address the cognitively demanding scheduling challenges in IoT systems, a linked task scheduling problem is formulated as a constrained optimization problem within the cloud architecture. The objective is to meet mixed target deadlines while considering energy usage and improving energy efficiency [23]. Experimental findings demonstrate the effectiveness of the proposed method in minimizing power under task computation. In another study, a transformed Henry gas solubility method utilizing the leading operators of the whale optimizer is introduced in [24]. The method incorporates extensive opposition-based learning to determine task scheduling. Furthermore, an enhanced version of the wild horse optimization algorithm based on Levy flight has been applied to handle task scheduling in a CC environment [25]. This approach aims to improve the efficiency and effectiveness of task scheduling in CC. Iftikhar et al. proposed an AI-based energy-efficient task scheduling method for CC environments [26]. The proposed scheme exploits the metaheuristic technique, such as the RFO, that is inspired by the hunting behavior of red foxes and their evolutionary population dynamics when avoiding hunters [27]. The RFO algorithm combines local and global search techniques with a reproduction approach. Abualigah et al. in [28] introduced the AOA algorithm. AOA is a robust optimizer that utilizes changes in arithmetic operator values, including multiplication, division, subtraction, and addition, during the calculation process.

In [29], the authors proposed a novel approach that utilizes multi-agent reinforcement learning techniques for task allocation in cooperative edge cloud computing environments. In the proposed approach, multiple autonomous agents collaborate to learn and make decisions regarding task allocation. Each agent represents a computational resource, such as an edge device or a cloud server. Similarly, in [30], the authors introduced a multi-agent reinforcement learning scheme for cooperative task offloading in distributed edge cloud computing. The proposed method utilizes multiple agents that collaborate and learn to make optimal decisions regarding task offloading, aiming to improve performance and resource utilization in edge cloud environments. In [31], the authors proposed a approach that combines graph convolutional networks and reinforcement learning techniques for dependent task allocation in edge computing. By leveraging the power of graph convolutional networks, the authors aim to improve the allocation of tasks in edge computing environments. Similarly, in [32] the authors proposed a cost-efficient and quality-of-experience-aware player request scheduling and rendering server allocation in edge-computing-assisted multiplayer

cloud gaming. The objective of the proposed scheme is to optimize resource allocation to minimize costs while ensuring a high-quality gaming experience for players. In [33], the focus of the authors is to improve scalability, sustainability, and availability in edge-cloud gaming through workload distribution. The authors proposed a method to effectively distribute gaming workloads across edge and cloud resources to enhance system scalability, reduce energy consumption, and ensure high availability for gaming services. In [34], the authors proposed a task allocation optimization scheme called EASE: an energy-aware job scheduling algorithm for vehicular edge networks that utilize renewable energy resources. The proposed scheme optimizes task allocation by considering the availability and efficiency of renewable energy sources. The proposed algorithm minimizes energy consumption by intelligently scheduling jobs based on factors, such as task energy demand, vehicle energy generation capacity, and vehicle mobility patterns. In [35], the authors addressed the challenge of scheduling in air-ground collaborative mobile edge computing systems with a focus on minimizing the Age of Information (AoI). The Age of Information represents the time elapsed since the last received update, which is critical for real-time applications. The proposed scheme aims to optimize the allocation of computing resources to minimize the AoI and improve the overall system performance by considering the specific characteristics of air-ground collaborative scenarios.

III. System Model

In this Section, we present the mathematical formulation of task scheduling problem for fog-cloud based IoT environment. We assume there are a large number of physical servers available in the data center with variable specifications (i.e., storage capacity, RAM size, CPU cores, and network bandwidth). These servers can be scaled up or down to achieve the Service Level Agreements (SLAs) and Quality of Service (QoS) requirements [9]. Let's Assume $VM = \{VM_1, \ldots, VM_m\}$ represents a collection of virtual machines in a data center. The processing capabilities of each VM_j are measured in MIPS (millions of instructions per second). Let $T = \{T_1, \ldots, T_n\}$ denotes a set of application tasks provided by cloud users to be completed on the data center's collection of VMs. The task length TL_i of each task/job T_i is measured in millions of instructions (MI). The Expected Time to Compute (ETC) is used to keep track of the time it takes to conduct a specific job (service) on different virtual machines [36]. ETC_{ij} refers to the ETC of the task i on VM_j computed as follows:

$$ETC_{ij} = \frac{TL_i}{VMP_j}, \quad 1 \le i \le n, 1 \le j \le m \tag{1}$$

where TL_i is the length of task i and VMP_j refers to the processing power of VM_j. The objective of our study is to enhance the QoS in terms of Makespan and energy efficiency. Makespan refers to the total time required to complete all computational tasks. Therefore, proper job mapping is required to necessitate shorter Makespan. Makespan (MKS) is calculated as follows:

TABLE I
Key Notations

Abbr.	Definition
TL_i	Length of task i
VM	Set of virtual machines in data centre
VM_j	Virtual machines j
λ	Balancing parameter
m	Total number of virtual machines
TE_j	Total execution time of virtual machine VM_j
α_j	Consumed energy by VM_j in the idle state
β_j	Consumed energy by VM_j in active state
F	Fitness function
ETC_{ij}	ETC of the task i on virtual machine VM_j
$a \in [0, d_{ib}]$	Scaling hyper-parameter
n	Number of tasks
N	Number of solutions
t_{max}	Number of iterations
TE_{ng}	Total energy consumed by a cloud system

$$MKS = \max_{j \in 1,2,\ldots,m} \sum_{i=1}^{n} ETC_{ij}. \tag{2}$$

The amount of total energy utilized by computing resources to complete the total tasks is called energy consumption and the objective is to kept it minimum to improve system performance and to deliver the required QoS. However, the total energy consumed by VM_j equals the amount of energy used in its active and idle states [37]. However, it is predicted that a VM roughly consumes 60% of total energy in idle state only [38]. Thus, energy consumption $Eng(VM_j)$ for a particular VM_j can be calculated as follows:

$$Eng(VM_j) = \left(TE_j \times \beta_j + (MKS - TE_j)\alpha_j\right) VMP_j, \tag{3}$$

$$\beta_j = 10^{-8} \times VMP_j^2, \tag{4}$$

$$\alpha_j = 0.6 \times \beta_j. \tag{5}$$

where TE_j refers the total execution time of VM_j. α_j and β_j denote the consumed energy by VM_j in the idle state and in active state, respectively. Thus, the total energy consumption ($TEng$) of a cloud system can be calculated as:

$$TEng = \sum_{j=1}^{m} Eng(VM_j), \tag{6}$$

where m represents the total number of virtual machines. As energy consumption and Makespan have more impact on a cloud system overall performance. Therefore, our primary goal is to improve the Makespan while reducing total energy consumption. Thus, our task scheduling problem become a bi-objective optimization problem whose fitness function is defined as follows:

$$F = (1 - \lambda) \times MKS + \lambda \times TEng. \tag{7}$$

where λ represents the balancing parameter between the factors in the fitness function. Thus, our task scheduling objective is finding a schedule with the lowest F. The key notations and symbols are presented in Table I.

IV. Proposed Hybrid Enhanced Optimization Algorithm for Intelligent Task Scheduling

In this section, firstly we discuss the background knowledge related to red fox and arithmetic optimization algorithms and then we present our proposed hybrid enhanced optimization scheme for intelligent task scheduling in edge-cloud based IoT environment.

A. RFO Optimization Scheme

The RFO optimization technique is inspired by the hunting behavior of red foxes and their evolutionary population dynamics when avoiding hunters [27]. In this technique, first parameters values are allocated and then the initial population of N foxes initialized using the following formula:

$$X_{ij} = l_j + r \times \left(u_j - l_j\right).\ i = 1, \ldots, N,\ j = 1, \ldots, n,\ r \in [0, 1] \quad (8)$$

where l_j and u_j are the limits of search space, n denotes the dimension of each agent. Then, the optimal X_b is determined by calculating the objective function for each X_i. The following Equation is used to reallocate the solution (X_{ir}).

$$X_{ir} = X_i + a \times sign(X_b - X_i) \quad (9)$$

where $a \in [0, d_{ib}]$ represents scaling hyperparameter chosen randomly and d_{ib} is defined as:

$$d_{ib} = \sqrt{||X_i - X_b||} \quad (10)$$

If reallocation has a smaller fitness value than the prior position, X_i should be replaced with X_{ir}. Otherwise, X_i should be kept. Update the parameter μ that represents the noticing of hunting fox. The value of observation radius r is then adjusted using the following calculation if the fox is not noticed:

$$r = \begin{cases} a\frac{\sin(\phi_0)}{\phi_0} & if\ \phi_0 \neq 0 \\ \theta & otherwise \end{cases} \quad (11)$$

where $\theta \in [0, 1]$ is a random variable that represents the impact of poor weather circumstances like fog, rain etc. Following that, the agents will be updated using the following spatial coordinate equations [27]:

$$\begin{gathered} x_0^n = a \times r \times \cos(\phi_1) + x_0^{act} \\ x_1^n = a \times r \times \sin(\phi_1)\ + a \times r \times \cos(\phi_2) + x_1^{act} \\ x_2^n = ar \times \sin(\phi_1) + a \times r \times \sin(\phi_2) \\ +a \times r \times \cos(\phi_3) + x_2^{act} \\ \vdots \\ x_{n-1}^n = a \times r \times \sin(\phi_1) + \cdots + a \times r \times \sin(\phi_{n-1})\ + x_{n-1}^{act} \end{gathered} \quad (12)$$

In eq. (12), $\phi_1,\ \phi_2, \ldots, \phi_{n-1} \in \langle 0,\ 2\pi \rangle$ are randomly generated angular values against each point. This set of equations shows a fox's behavior after it notices the victim and attempts to attack it. The next step is to calculate each agent's fitness value and then sort the agents based on their fitness values. Hunters may kill the worst agents. In the meantime, the new agents are substituted in X using the following equation:

$$X = \begin{cases} \text{Nomadic agent} & if\ > 0.5 \\ \text{Reproduction of the alpha couple} & Otherwise \end{cases} \quad (13)$$

In the first scenario of Eq. (13), the new agents depart the environment as nomadic agents and look for space to rebuild their herd outside of the area. Within the search domain, but outside the habitat, the agent is chosen at random. The habitat's center (C_H) is calculated as follows:

$$C_H = \frac{X_b + X_\beta}{2}, \quad (14)$$

where X_b and X_β denote the first and second best solutions, respectively. Similarly, in the second situation of Eq. (13), new agents can be produced from the alpha couple using the following equation:

$$X = \frac{X_b + X_\beta}{2}. \quad (15)$$

B. AOA Optimization Scheme

Abualigah et al. developed the AOA optimization algorithm [28]. In this Section, we discuss the traditional AOA's exploration and exploitation phases that depend on mathematics functions named Addition (A '+'), Multiplication (M '*times*'), Division (D '*div*'), and Subtraction (S '−'). Depending on the formulation, AOA may tackle small or big optimization problems because it is a gradient-free and population-based approach. As shown in Eq. (16), the improvement procedures in AOA work with a randomized set of competing solutions (X).

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,n} \\ x_{2,1} & \cdots & x_{2,j} & \cdots & x_{2,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,n} \end{bmatrix} \quad (16)$$

In Eq. (16), X stands for the solutions obtained using Eq. (17) given below:

$$X_{ij} = LB_j + rand \times \left(UB_j - LB_j\right), \quad (17)$$

where X_i represents a solution to task scheduling problem at hand. The upper and lower borders (UB_j and LB_j) confine the solutions. Before the AOA starts, the search stage should be chosen (i.e., exploration or exploitation) given in Eq. (18), which is used in the search criteria to measure the math optimizer accelerated (M_{OA}).

$$M_{OA}(t) = Min + t \times \left(\frac{Max_{OA} - Min_{OA}}{M_{it}}\right) \quad (18)$$

AOA's exploration operators randomly explore the search field in numerous regions in an attempt to discover the optimal solution utilizing the main search technique (division and multiplication methods) given in Eq. (19). In general, M_{OA} controls this process of searching (i.e., search by running M or D). The operator D is controlled by $r2 = 0.5$ in this step (the first factor in Eq. (19), whereas M is disregarded before the operator completes its present task. Otherwise, the

second function (M) would be used to conduct the current mission instead of the D ($r2 \in [0, 1]$ is a stochastic number to choose the search technique in each phase randomly). For the exploratory modules, the following position-updating formulae are used:

$$x_{ij}(t+1) = \begin{cases} X_{bj} \div (M_{OP} + \epsilon) \times D_{UL} & r2 < 0.5 \\ X_{bj} \times M_{OP} \times D_{UL} & otherwise \end{cases} \quad (19)$$

In Eq. (19), $D_{UL} = ((UB_j - LB_j) \times \mu + LB_j)$. X_{bj} is the best solution found so far. ϵ stands for a small integer value, UB_j and LB_j stands for the upper and lower bound values at dimension j_{th}, respectively. $\mu = 0.5$ denotes the control function applied to switch between search strategies.

$$M_{OP}(t) = 1 - \frac{t^{1/\alpha}}{M_{it}^{1/\alpha}}, \quad (20)$$

where (M_{OP}) is the probability of the math optimizer and (M_{it}) is the cumulative number of generations. $\alpha = 5$ stands for the variable employed to find the precision of exploitation over the iterations. In case if $M_{OA} < r1$ the exploitation strategy is performed. In this stage, the S and A functions in AOA extensively study the exploration areas in numerous concentrated locations and run out to identify a superior solution using these search approaches (i.e., Subtraction (S) and Addition (A) represented in Eq. (21)).

$$x_{i,j}(t+1) = \begin{cases} X_{bj} - M_{OP} \times D_{UL}, & r3 < 0.5 \\ X_{bj} + M_{OP} \times D_{UL}, & otherwise \end{cases} \quad (21)$$

where D_{UL} is defined in Eq. (19).

C. Proposed RFOAOA Intelligent Task Scheduling Algorithm

In this Section, we discuss the proposed task scheduling algorithm for edge-cloud based IoT environments. The proposed algorithm aims to enhance the behavior of RFO using the operators of AOA. RFOAOA begins by initializing population X with N agents. The Makespan value (i.e., fitness) of X_i is determined by the following Eq. (3). The last step is to determine the agent has the lower Makespan value and utilize that as the best solution X_b. Following that, the improved RFOAOA updates the solutions using RFO within the exploration capability. However, throughout the exploitation phase, the X will be changed depending on the quality of the competition between the RFO and the AOA. The updating process is ended when it reaches the termination condition, and X_b presents the output value. Figure 1 depicts the RFOAOA framework for SIoTC, and the entire explanation is presented with more information in the following subsections.

Initial Stage: The primary goal of this step is to generate the initial population, which reflects the solution to the discrete optimization issue that is the task scheduling problem. However, the traditional RFO is designed to solve the real-valued optimization problems, but the developed task scheduling is a discrete optimization problem. Therefore, the proposed RFOAOA tackle this issue by first generating $X_i, i = 1, 2, \ldots, N$ as given in following Eq. (22):

$$X_{ij} = floor\left(Lb_{ij} + \alpha \times \left(Ub_{ij} - Lb_{ij}\right)\right),\ \alpha \in [0, 1],\ j = 1, 2, \ldots, n \quad (22)$$

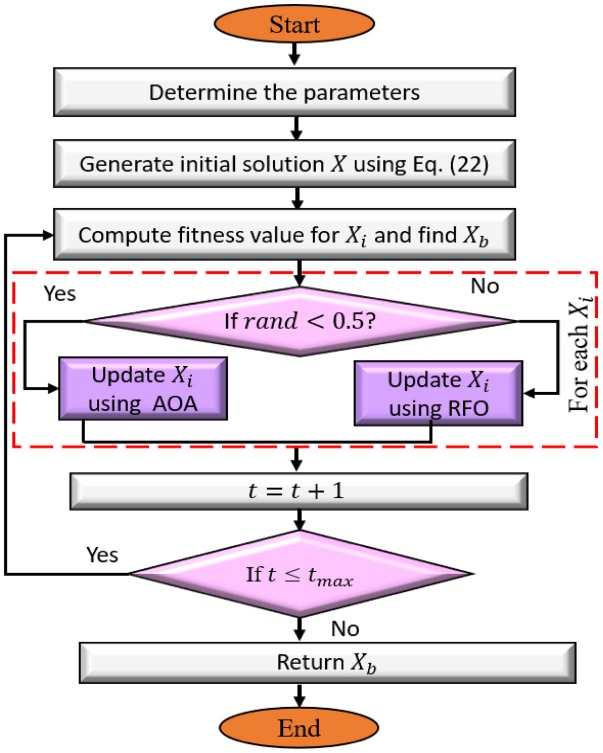

Fig. 1. Procedural flow of the proposed RFOAOA algorithm.

Algorithm 1 Proposed RFOAOA Algorithm

```
Initialize: list of VMs (m), tasks (n), solutions (N), total number of iterations (t_max).
Build solutions (X) using Eq. (22).
t = 0.
while t <= t_max do
    Assess the quality of X_i (i.e., F_i) using Eq. (7).
    Determine X_b with smallest F_b.
    for i = 1 : N do
        t = t + 1.
        if rand > 0.5 then
            Enhance X_i using Eqs. (9)–(15).
        else
            Enhance X_i using Eqs. (18)–(21).
Return X_b.
```

According to the task scheduling problem formulation, the lower boundary of the search domain is $Lb = 1$, while the upper boundary is $Ub = m$. The function $floor(.)$ is applied to convert real values to integers.

Updating Stage: In this phase, population X is enhanced using the proposed RFOAOA, To begin, compute the fitness value for X_i. Then find X_b with the best *Fit* (i.e., smallest Makespan). Using the RFO operators to update X in the exploration stage is the next step. When X_i enters the exploitation phase, the operators of AOA are used. Finally, the stopping criteria are evaluated, and if they are met then updating of X is stopped, and the output X_b is returned; otherwise, the updating operation is iterated again. The proposed RFOAOA algorithm pseudo-code is illustrated in Algorithm 1.

TABLE II
SIMULATION PARAMETERS

Cloud Entity	Parameter	Value
Client	No. of clients	100 - 200
Datacenter	No. of datacenters	1
Host	No. of hosts	2
Cloud nodes:		
	No. of nodes	10
	CPU power	3000 - 5000 MIPS
	RAM	8 GB
	Storage	1 TB
	Bandwidth capacity	0.5 Gb/s
Fog nodes:		
	No. of nodes	10
	CPU power	1000 - 2800 MIPS
	RAM	2 GB
	Storage	0.5 TB
	Bandwidth capacity	1 Gb/s

Time Complexity: The time complexity of the proposed RFOAOA depends on the number of tasks n, number of solutions N, and number of iterations t_{max}. So, the time complexity of RFOAOA is calculated as follows:

$$O(RFOAOA) = N \times n + t\Big(K_1 \times N^2 \times n^2\Big) + K_2 \times (N \times (n+1)) \quad (23)$$

where K_1 and K_2 refers to the number of solutions updated using Eqs. (9)-(15) (i.e., RFO) and Eqs. (18)-(21) (i.e., AOA). Therefore, complexity of RFOAOA is formulated as $O(N^2 \times n^2 \times t_{max})$.

V. PERFORMANCE EVALUATION

In this Section, we provides a comprehensive assessment of the proposed scheme with other state-of-the-art task scheduling schemes. Firstly, we present the simulation environment and settings of the parameters. Then, we discuss the performance measuring metrics employed to examine the effectiveness of the proposed scheme. Finally, simulation results and related discussions are provided.

A. Experimental Setup

The experimentation is performed using MATLAB R2018a which is a well-known simulator [39] on a desktop machine configured with a 2.40 GHz Intel Core i5 CPU, 4.00 GB RAM, and Windows 10 OS. In our experiments, the cloud-fog framework involves fog nodes with limited processing capacities, but they are closer to the intelligent IoT appliances and have a minimum delay. Conversely, cloud nodes can process the IoT tasks rapidly, but they require a long time to receive them. Thus, in order to optimize the overall performance, the proposed scheduling algorithm maintains the balance between both the fog and the cloud nodes. The cloud-fog environment comprises on a single data center, two host machines, and ten cloud nodes of varying configurations. Besides, it involves ten fog nodes of varying configurations too.

To test the effectiveness of RFOAOA algorithm, we used the real and synthetic workload traces from NASA Ames iPSC/860 and HPC2N "Parallel Workload Archive" repository [22]. These workload traces archives are provided in standard workload format (SWF) for the scientific community. The NASA Ames iPSC/860 log comprises of 14,794 tasks and HPC2N log contains 527,371 tasks. The synthetic workload, on the other hand, involves 1500 tasks generated with lengths varying from 2,000 to 56,000 MI. Table IV provides the specifics of the synthetic workload.

TABLE III
PARAMETER SETTINGS FOR PERFORMANCE ANALYSIS

Algorithm	**Parameter**	**Value**
RFO	$c1$	0.18
	$c1$	0.82
AEO	$ran_1, ran_2, ran_3, ran_4$	[0,1]
PSO	w	$0.9 \rightarrow 0.4$
	c_1, c_2	1.49
MPA	$FADs$	0.2
	P	0.5
WOA	a	2
	b	1
MRFO	S	2
	r_1, r_2, r_3	[0,1]
RFOAOA	$c1$	0.18
	$c1$	0.82
	Max_{OA}	0.5
	Min_{OA}	0.2

TABLE IV
PROPERTIES OF THE SYNTHETIC WORKLOAD

Parameter	Value
Tasks length	[2000, 56000] MI
Data size	[400, 600] MB
Tasks number	[300, 1500]

B. Evaluation Metrics

The objective of our study is to ensure a minimum total energy consumption along with a lower Makespan. In order to validate the performance of the proposed RFOAOA algorithm against other state-of-the-art methods, we evaluate performance in terms of Makespan time and total energy consumption.

1) *Makespan:* Represents the time to accomplish the last completed task. A minimum Makespan ensures appropriate mapping of user tasks to CNs and it is measured according to Eq. (2).
2) *Total energy consumption:* denotes the energy required by the infrastructure including all fog and cloud nodes. For an efficient system, the VM's energy consumption should be minimized. Thus, the total energy consumption can be calculated using Eq. (6).

C. Results and Discussions

For comparative analysis, six optimization algorithms including MPA, RFO, AEO, SSA, WOA, and PSO are selected. Parameters settings of RFOAOA and other algorithms are summarized in Table III. These parameters are selected based on the original implementation of each method. For simulation results an average of 30 independently executions is taken against each benchmark problem to provide more reliable results. Moreover, for fair comparative analysis, we select the population size to 50 and $\omega = 0.7$.

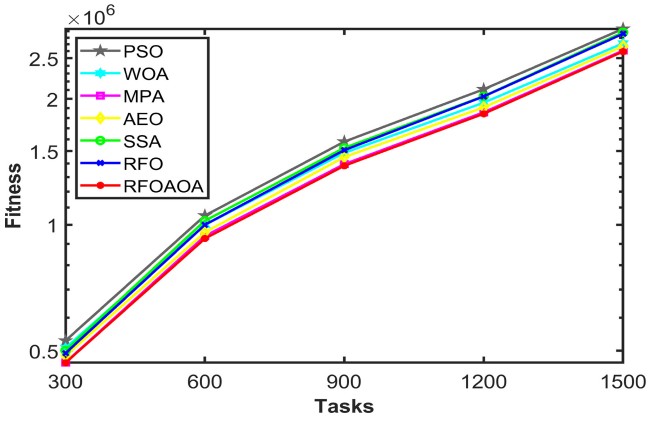

Fig. 2. Comparison against fitness values using the synthetic workload.

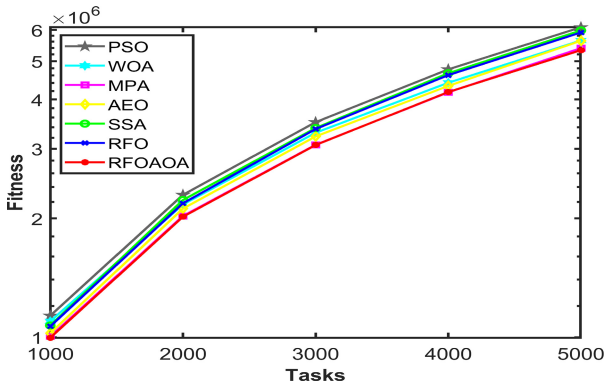

Fig. 3. Comparison against fitness values using the NASA iPSC workload.

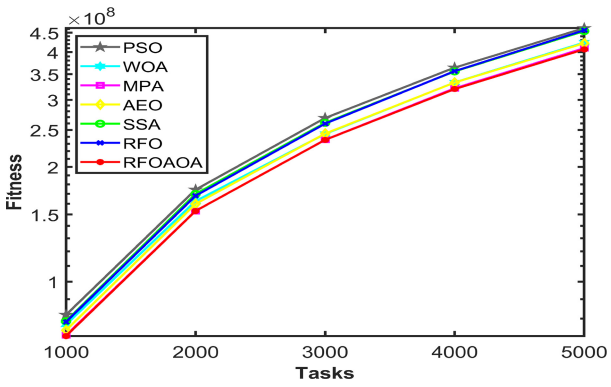

Fig. 4. Comparison against fitness values using HPC2N workload.

Figures 2 - 4 demonstrate the comparison of the proposed algorithm with other state-of-the-art optimization algorithms in terms of fitness function against no. of tasks. Figure 2 shows that the proposed RFOAOA algorithm achieves the best fitness value for the synthetic workload when the number of tasks ranges from 300-1500. Figure 3 indicated that the RFOAOA consistently outperforms other optimization algorithms when evaluated on the NASA Ames iPSC/860 workload traces. Similarly, Figure 4 illustrates the impressive performance of the proposed algorithm against the HPC2N workload traces when number of tasks varies from 1000 - 5000. Thus, based on these findings, it is clear that the proposed RFOAOA consistently exhibits superior performance compared to the other

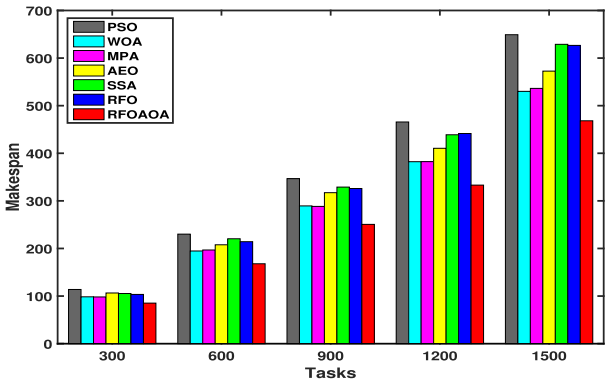

Fig. 5. Average Makespan of all algorithms using the synthetic workload.

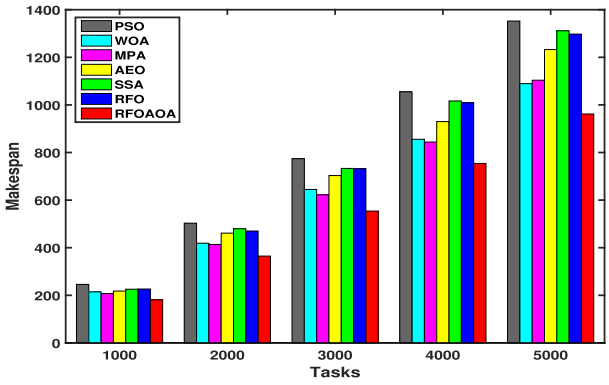

Fig. 6. Average Makespan of all algorithms using NASA iPSC workload.

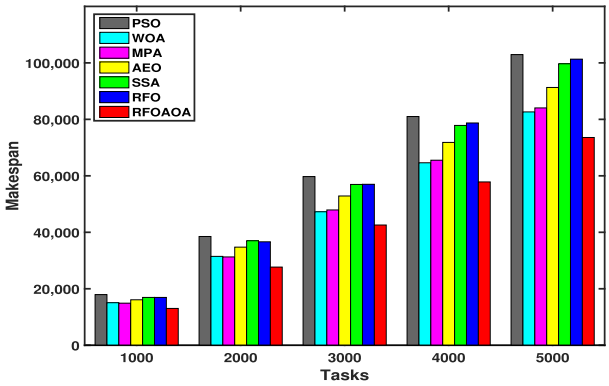

Fig. 7. Average Makespan of all algorithms using HPC2N workload.

optimization approaches across all workload instances in terms of the fitness function.

Figures 5 - 7 present a comparison of the experimental results for synthetic and real workload traces in terms of average Makespan. As shown in Figure 5, the proposed RFOAOA algorithm consistently achieves best average Makespan value against the synthetic wordload traces compared to the standard RFO algorithm and other competing algorithms. Similarly, Figure 6 demonstrates that the proposed RFOAOA algorithm outperforms the other optimization techniques in terms of average Makespan against the real NASA iPSC workload traces. Furthermore, when considering the HPC2N workload traces, Figure 7 reveals that the proposed algorithm achieves the minimum average Makespan compared to the other state-of-the-art optimization schemes.

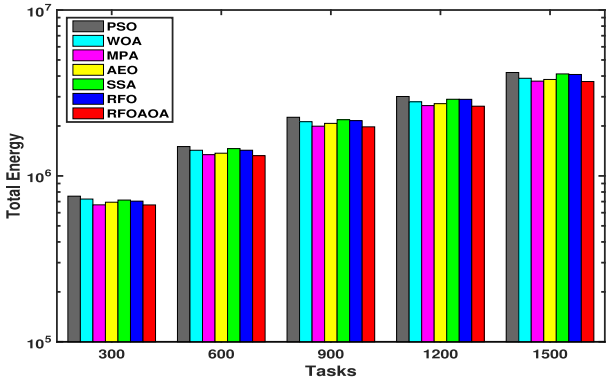

Fig. 8. Total energy consumption against synthetic workload traces.

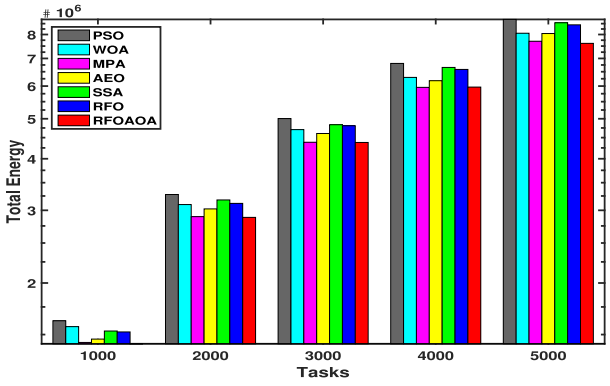

Fig. 9. Total energy consumption against NASA iPSC workload traces.

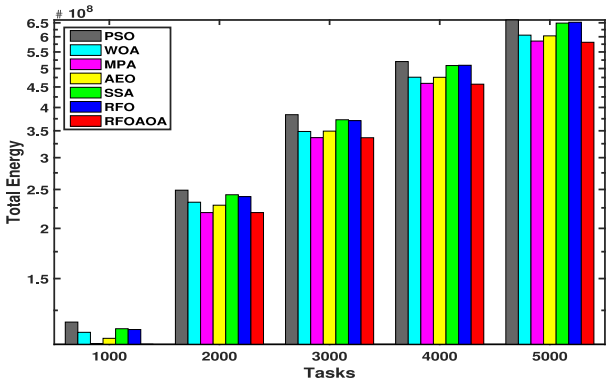

Fig. 10. Total energy consumption against HPC2N workload traces.

Figures 8 - 10 demonstrate the overall energy consumption of the proposed RFOAOA algorithm and other state-of-the-art optimization algorithms under synthetic and real workload traces. Figure 8 shows that the proposed RFOAOA algorithm achieves the lowest energy consumption for the synthetic dataset when compared to the conventional RFO algorithm and other compared algorithms. Similarly, Figure 9 shows that the RFOAOA algorithm outperforms the existing approaches in terms of total energy consumption against the NASA Ames iPSC workload traces. Figure 10 reveals that the proposed RFOAOA achieves minimum energy consumption compared to other approaches for the real HPC2N workload traces. Thus, overall RFOAOA achieves better performance compared to the other optimization algorithms in terms of energy consumption against all workload traces. Thus, the simulation results show that the proposed RFOAOA algorithm achieves better results in terms of Makespan and energy consumption against all workload traces. The integration of RFO and AOA significantly enhances search efficiency and leads to improved solutions for large-scale workload traces. However, the proposed optimization algorithm has some limitations. For example, increased time complexity and selection of optimal value of initial population.

To sum up, the comparison results shown in the figures indicate that RFOAOA attains very promising results according to the performance metrics, such as Makespan and energy consumption on all dataset instances under consideration. Hence, integrating AOA with the RFO can improve search efficiency and provide better large-scale datasets solutions. However, the developed model still has some limitations such as the time complexity still need to be improved. In addition, selecting the optimal initial population is considered a limitation of the developed method, similar to other MH techniques.

VI. Conclusion

In this paper, we proposed a hybrid enhanced optimization scheme called RFOAOA for efficient task scheduling in sustainable edge computing environments. The RFOAOA algorithm combines the strengths of the red fox optimization and arithmetic optimization algorithms. We evaluate the efficiency of our proposed algorithm using different evaluation criteria and compare it with state-of-the-art methods, including RFO, AEO, SSA, WOA, MPA, and PSO, under various workload scenarios. Our proposed algorithm outperforms the state-of-the-art methods in terms of Makespan time and energy efficiency. These findings highlight the superior performance of RFOAOA in scheduling outputs and reinforce its potential for practical implementation. However, scalability is one of the major limitations of the proposed scheme, as the performance of the proposed algorithm may be constrained when applied to large-scale edge computing systems with a significant number of tasks and resources. For future work, we suggest extending the proposed algorithm to address other optimization-related problems in cloud-edge environments, such as workflow scheduling and virtual machine placement. Moreover, it would be interesting to investigate the performance of the proposed algorithm in other applications, such as vehicle routing, production scheduling, assembly line balancing, and healthcare facility planning, etc.

References

[1] A. Arunarani, D. Manjula, and V. Sugumaran, "Task scheduling techniques in cloud computing: A literature survey," *Future Gener. Comput. Syst.*, vol. 91, pp. 407–415, Feb. 2019.

[2] E. Ahmed et al., "The role of big data analytics in Internet of Things," *Comput. Netw.*, vol. 129, pp. 459–471, Dec. 2017.

[3] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.

[4] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, 2014.
[5] A. Yousefpour et al., "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *J. Syst. Architect.*, vol. 98, pp. 289–330, Sep. 2019.
[6] J. Luo et al., "Container-based fog computing architecture and energy-balancing scheduling algorithm for energy IoT," *Future Gener. Comput. Syst.*, vol. 97, pp. 50–60, Aug. 2019.
[7] P. Zhang, M. Zhou, and G. Fortino, "Security and trust issues in fog computing: A survey," *Future Gener. Comput. Syst.*, vol. 88, pp. 16–27, Nov. 2018.
[8] C.-G. Wu, W. Li, L. Wang, and A. Y. Zomaya, "An evolutionary fuzzy scheduler for multi-objective resource allocation in fog computing," *Future Gener. Comput. Syst.*, vol. 117, pp. 498–509, Apr. 2021.
[9] I. Attiya, L. Abualigah, D. Elsadek, S. A. Chelloug, and M. A. Elaziz, "An intelligent chimp Optimizer for scheduling of IoT application tasks in fog computing," *Mathematics*, vol. 10, no. 7, p. 1100, 2022.
[10] A. Ali et al., "Quality of service provisioning for heterogeneous services in cognitive radio-enabled Internet of Things," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 328–342, Jan.–Mar. 2018.
[11] M. Gorlatova, H. Inaltekin, and M. Chiang, "Characterizing task completion latencies in fog computing," 2018, *arXiv:1811.02638*.
[12] M. Aazam, S. Zeadally, and K. A. Harras, "Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities," *Future Gener. Comput. Syst.*, vol. 87, pp. 278–289, Oct. 2018.
[13] A. Ali, M. E. Ahmed, F. Ali, N. H. Tran, D. Niyato, and S. Pack, "Non-parametric Bayesian channels cLustering (NOBEL) scheme for wireless multimedia cognitive radio networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2293–2305, Oct. 2019.
[14] M. Ghobaei-Arani, A. Souri, and A. A. Rahmanian, "Resource management approaches in fog computing: A comprehensive review," *J. Grid Comput.*, vol. 18, no. 1, pp. 1–42, 2020.
[15] P. Hosseinioun, M. Kheirabadi, S. R. K. Tabbakh, and R. Ghaemi, "A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm," *J. Parallel Distrib. Comput.*, vol. 143, pp. 88–96, Sep. 2020.
[16] M. Al-Khafajiy, T. Baker, H. Al-Libawy, A. Waraich, C. Chalmers, and O. Alfandi, "Fog computing framework for Internet of Things applications," in *Proc. IEEE 11th Int. Conf. Develop. eSyst. Eng. (DeSE)*, 2018, pp. 71–77.
[17] M. Ghobaei-Arani, A. Souri, F. Safara, and M. Norouzi, "An efficient task scheduling approach using moth-flame optimization algorithm for cyber-physical system applications in fog computing," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 2, 2020, Art. no. e3770.
[18] B. Jamil, M. Shojafar, I. Ahmed, A. Ullah, K. Munir, and H. Ijaz, "A job scheduling algorithm for delay and performance optimization in fog computing," *Concurrency Comput. Pract. Exp.*, vol. 32, no. 7, 2020, Art. no. e5581.
[19] M. Abdel-Basset, R. Mohamed, M. Elhoseny, A. K. Bashir, A. Jolfaei, and N. Kumar, "Energy-aware marine predators algorithm for task scheduling in IoT-based fog computing applications," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5068–5076, Jul. 2021.
[20] I. M. Ali, K. M. Sallam, N. Moustafa, R. Chakraborty, M. J. Ryan, and K.-K. R. Choo, "An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems," *IEEE Trans. Cloud Comput.*, vol. 10, no. 4, pp. 2294–2308, Oct.–Dec. 2022.
[21] X.-Q. Pham, N. D. Man, N. D. T. Tri, N. Q. Thai, and E.-N. Huh, "A cost-and performance-effective approach for task scheduling based on collaboration between cloud and fog computing," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 11, p. 15, 2017.
[22] "Parallel workloads archive." Accessed: Jul. 20, 2022. [Online]. Available: http://www.cse.huji.ac.il/labs/parallel/workload/logs.html
[23] J. Xu, Z. Hao, R. Zhang, and X. Sun, "A method based on the combination of laxity and ant colony system for cloud-fog task scheduling," *IEEE Access*, vol. 7, pp. 116218–116226, 2019.
[24] M. A. Elaziz and I. Attiya, "An improved Henry gas solubility optimization algorithm for task scheduling in cloud computing," *Artif. Intell. Rev.*, vol. 54, no. 5, pp. 3599–3637, 2021.
[25] G. Saravanan, S. Neelakandan, P. Ezhumalai, and S. Maurya, "Improved wild horse optimization with Levy flight algorithm for effective task scheduling in cloud computing," *J. Cloud Comput.*, vol. 12, no. 1, p. 24, 2023.
[26] S. Iftikhar et al., "HunterPlus: AI based energy-efficient task scheduling for cloud–fog computing environments," *Internet Things*, vol. 21, Apr. 2023, Art. no. 100667.
[27] D. Połap and M. Woźniak, "Red fox optimization algorithm," *Exp. Syst. Appl.*, vol. 166, Mar. 2021, Art. no. 114107.
[28] L. Abualigah, A. Diabat, S. Mirjalili, M. A. Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Comput. Methods Appl. Mech. Eng.*, vol. 376, Apr. 2021, Art. no. 113609.
[29] S. Ding, "Multi-agent reinforcement learning for task allocation in cooperative edge cloud computing," in *Proc. Int. Conf. Service Orient. Comput.*, 2021, pp. 283–297.
[30] S. Ding and D. Lin, "Multi-agent reinforcement learning for cooperative task offloading in distributed edge cloud computing," *IEICE Trans. Inf. Syst.*, vol. 105, no. 5, pp. 936–945, 2022.
[31] S. Ding, D. Lin, and X. Zhou, "Graph convolutional reinforcement learning for dependent task allocation in edge computing," in *Proc. IEEE Int. Conf. Agents (ICA)*, 2021, pp. 25–30.
[32] Y. Gao, C. Zhang, Z. Xie, Z. Qi, and J. Zhou, "Cost-efficient and quality-of-experience-aware player request scheduling and rendering server allocation for edge-computing-assisted multiplayer cloud gaming," *IEEE Internet Things J.*, vol. 9, no. 14, pp. 12029–12040, Jul. 2022.
[33] I. Jaya, Y. Li, and W. Cai, "Improving scalability, sustainability and availability via workload distribution in edge-cloud gaming," in *Proc. 30th ACM Int. Conf. Multimedia*, 2022, pp. 2987–2995.
[34] G. Perin, F. Meneghello, R. Carli, L. Schenato, and M. Rossi, "EASE: Energy-aware job scheduling for vehicular edge networks with renewable energy resources," *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 1, pp. 339–353, Mar. 2023.
[35] Z. Qin et al., "AoI-aware scheduling for air-ground collaborative mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 22, no. 5, pp. 2989–3005, May 2023.
[36] I. Attiya, M. A. Elaziz, L. Abualigah, T. N. Nguyen, and A. A. A. El-Latif, "An improved hybrid swarm intelligence for scheduling IoT application tasks in the cloud," *IEEE Trans. Ind. Informat.*, vol. 18, no. 9, pp. 6264–6272, Sep. 2022.
[37] I. Attiya, L. Abualigah, S. Alshathri, D. Elsadek, and M. A. Elaziz, "Dynamic jellyfish search algorithm based on simulated annealing and disruption operators for global optimization with applications to cloud task scheduling," *Mathematics*, vol. 10, no. 11, p. 1894, 2022.
[38] S. K. Mishra, D. Puthal, J. J. P. C. Rodrigues, B. Sahoo, and E. Dutkiewicz, "Sustainable service allocation using a metaheuristic technique in a fog server for industrial applications," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4497–4506, Oct. 2018.
[39] A. Karimiafshar, M. R. Hashemi, M. R. Heidarpour, and A. N. Toosi, "Effective utilization of renewable energy sources in fog computing environment via frequency and modulation level scaling," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 10912–10921, Nov. 2020.

Mohamed Abd Elaziz received the B.S. and M.S. degrees in computer science and the Ph.D. degree in mathematics and computer science from Zagazig University, Egypt, in 2008, 2011, and 2014, respectively, where he was an Assistant Lecturer with the Department of Computer Science from 2008 to 2011. He is an Associate Professor with Zagazig University. He is the author of more than 430 articles. His research interests include metaheuristic technique, medical application, digital twins, renewable energy, security IoT, cloud computing, machine learning, signal processing, image processing, and evolutionary algorithms. He is one of the 2% influential scholars, which depicts the 100 000 top-scientists in the world. He is one of the highest cited researchers according to WOS 2022.

Ibrahim Attiya received the B.Sc. and M.Sc. degrees in computer science from the Department of Mathematics, Faculty of Science, Zagazig University, Egypt, in 2005 and 2012, respectively, and the Ph.D. degree in computer science and technology from the University of Science and Technology Beijing, Beijing, China, in 2017. He was a Postdoctoral Researcher with the School of Computer Science and Technology, Wuhan University of Technology from 2019 to 2021. He is currently an Associate Professor with the Faculty of Science, Zagazig University. His current research interests include cloud computing, fog computing, Internet of Things, scheduling and load balancing algorithms, resources management, energy-aware, and optimization techniques.

Laith Abualigah received the degree in computer information system and the master's degree in computer science from Al-Albayt University, Jordan, in 2011 and 2014, respectively, and the Ph.D. degree from the School of Computer Science, Universiti Sains Malaysia, Malaysia, in 2018. He is an Associate Professor with the Prince Hussein Bin Abdullah College for Information Technology, Al Al-Bayt University. He is also a Distinguished Researcher with the University of Science, Malaysia. He has published more than 400 journal papers and books, which collectively have been cited more than 15 200 times (H-index = 50). His main research interests focus on arithmetic optimization algorithm, bio-inspired computing, nature-inspired computing, swarm intelligence, artificial intelligence, meta-heuristic modeling, and optimization algorithms, evolutionary computations, information retrieval, text clustering, feature selection, combinatorial problems, optimization, advanced machine learning, big data, and natural language processing. According to the report published by Clarivate, he is one of the highly cited researchers in 2021 and 2022 and the 1% influential Researchers, which depicts the 6938 top scientists in the world. He is the First Researcher in the domain of Computer Science in Jordan for 2021. According to the report published by Stanford University in 2020, he is one of the 2% influential scholars, which depicts the 100 000 top scientists in the world. He currently serves as an Associate Editor for the *Journal of Cluster Computing* (Springer), the *Journal of Soft Computing* (Springer), and the *Journal of Engineering Applications of Artificial Intelligence* (Elsevier).

Muddesar Iqbal is an academic entrepreneur whose research, teaching, and enterprise activities go hand in hand to create technical solutions to social problems. He has co-invented four patented inventions in SDN, IoT, cloud, and autonomous vehicles and co-founded several TLR 9 and above Digital Products. He has won over 17 R&D and capacity building grant Grants from different national and international funding agencies and has published more than 100 peer-reviewed research articles in reputable journals and conferences. His research interests included B5G and 6G network and communication technologies, Internet of sense for 6G, enabled industry 5.0 social and collaborative applications, Intelligent Internet of Things, and collaborative cognitive communication systems. He has won several awards and honors, including Awards of Appreciation for Tutoring and the Prize Winner twice from the Association of Business Executives, U.K.

Amjad Ali received the B.S. and M.S. degrees in computer science from the COMSATS Institute of Information Technology, Pakistan, in 2006 and 2008, respectively, and the Ph.D. degree from the Electronics and Radio Engineering Department, Kyung Hee University, South Korea, in 2015. From 2015 to 2022, he served as an Assistant Professor with the Department of Computer Science, COMSATS University Islamabad, Lahore Campus, Pakistan. From 2018 to 2019, he was a Postdoctoral Fellow with the Department of Information and Communication Engineering, Inha University and a Research Professor with the School of Electrical Engineering, Korea University, South Korea. He is currently working as a Postdoctoral Researcher with the College of Science and Engineering, Hamad Bin Khalifa University, Doha, Qatar. His main research interests include multimedia transmission, cognitive radio networks, stochastic network optimization, machine learning, Internet of Things, cloud and edge computing, 6G networks, and metaverse.

Ala Al-Fuqaha (Senior Member, IEEE) received the Ph.D. degree in computer engineering and networking from the University of Missouri—Kansas City, Kansas City, MO, USA. He is currently a Professor with the Information and Computing Technology Division, College of Science and Engineering, Hamad Bin Khalifa University. His research interests include the use of machine learning in general and deep learning in particular in support of the data-driven and self-driven management of large-scale deployments of IoT and smart city infrastructure and services, wireless vehicular networks (VANETs), cooperation and spectrum access etiquette in cognitive radio networks, and management and planning of software-defined networks. He serves on the editorial boards of multiple journals, including IEEE COMMUNICATIONS LETTER, *IEEE Network Magazine*, and *Arabian Journal for Science and Engineering* (Springer). He also served as the Chair, the Co-Chair, and a Technical Program Committee Member of multiple international conferences, including IEEE VTC, IEEE Globecom, IEEE ICC, and IWCMC. He is an ABET Program Evaluator.

Shaker El-Sappagh received the bachelor's degree in computer science from the Information Systems Department, Faculty of Computers and Information, Cairo University, Egypt, in 1997, the master's degree from Cairo University in 2007, and the Ph.D. degree in computer science from the Information Systems Department, Faculty of Computers and Information, Mansura University, Mansura, Egypt, in 2015. In 2003, he joined the Department of Information Systems, Faculty of Computers and Information, Minia University, Egypt, as a Teaching Assistant. Since June 2016, he has been with the Department of Information Systems, Faculty of Computers and Information, Benha University as an Assistant Professor. He worked as a Research Professor with the UWB Wireless Communications Research Center, Department of Information and Communication Engineering, Inha University, South Korea, from 2018 to 2020. He worked as a Research Professor with the Centro Singular de Investigación en Tecnoloxías Intelixentes, Universidade de Santiago de Compostela, Santiago de Compostela, Spain, in 2021. He has been an Associate Professor with Galala University, Egypt, since 2021. He has also been a Senior Researcher with the College of Computing and Informatics, Sungkyunkwan University, South Korea, since 2021. He has publications in clinical decision support systems and semantic intelligence. His current research interests include machine learning, medical informatics, (fuzzy) ontology engineering, distributed and hybrid clinical decision support systems, semantic data modeling, fuzzy expert systems, and cloud computing. He is a reviewer in many journals, and very interested in the diseases' diagnosis and treatment research.