STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 202x, pp 0–20.
Published online in International Academic Press (www.IAPress.org)

Hybrid Emperor Penguin and Gravitational Search Optimization for Efficient Task Co-Offloading in Fog Computing Environments

A. S. B. Sadkhan*, Mohsen Nickray

Department of Computer and Information Technology, Faculty of Engineering, University of Qom, Iran

Abstract The industrial fog computing setting is highly sensitive to challenges in efficiently scheduling tasks and offloading computing processes because of the heterogeneity of devices, their mobility, and fluctuations in resource availability. This work is motivated by the need to design a unified optimization mechanism for dynamic fog environments where both device and network heterogeneity severely affect task allocation. The proposed study aims to address the limitations of existing single-strategy offloading methods by developing a hybrid optimization framework that balances computational load, minimizes latency, and reduces energy consumption. The main contribution of this paper lies in introducing a hybrid GSA–EPO co-offloading model that achieves an adaptive trade-off between energy efficiency and service latency while maintaining scalability in large-scale fog computing systems. The hybridized framework integrates the Gravitational Search Algorithm (GSA) and Emperor Penguin Optimization (EPO) to reduce energy consumption, service latency, and penalties related to deadline violations. The model uses battery-aware willingness, memory constraints, and mobility-sensitive communication parameters to inform local execution, device-to-device offloading, and edge server offloading. The hybrid GSA–EPO algorithm combines the exploration capability of GSA with the convergence efficiency of EPO within a migration-based knowledge-exchange structure, thereby enhancing both convergence and diversity of solutions. Simulation results demonstrate that the proposed approach achieves up to **63% reduction in energy consumption** and **75% reduction in delay** compared to baseline methods, confirming the feasibility of hybrid metaheuristic strategies for reliable and efficient task co-offloading in dynamic fog computing networks.

Keywords Task Offloading, Fog Computing, Emperor Penguin Optimization, Gravitational Search Algorithm, Hybrid Metaheuristic

AMS 2010 subject classifications 68M14, 68T20, 90C59

DOI: 10.19139/soic-2310-5070-2979

1. Introduction

Fog computing has become a disruptive paradigm extending the features of the conventional cloud computing by shifting processing capabilities and data storage nearer to the network edge [1]. This architecture effectively counteracts a fundamental weakness of cloud computing—high latency and low bandwidth—which are crucial limitations for time-sensitive and bandwidth-constrained applications [2]. Fog computing greatly improves the performance of distributed environments by allowing delay-sensitive services using less energy and relieving traffic congestion [3]. Fog nodes can handle tasks locally or forward them to the cloud depending on service needs, resulting in improved resource utilization and responsiveness. Fog computing has been found to outperform traditional cloud computing models in contexts where large portions of applications require real-time services such as in IoT environments, reducing latency by more than half for up to 50 % of applications [3]. Nevertheless, when a system has low-latency requirements (little or no), fog computing may cause unwanted overhead [3]. In

*Correspondence to: Corresponding author: Abbas S. B. Sadkhan (Email: dr00abbas@gmail.com). Department of Computer and Information Technology, Faculty of Engineering, University of Qom, Qom, Iran.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

general, the way it can successfully handle latency, bandwidth, and energy consumption renders fog computing particularly appropriate in real-time IoT applications [2, 3].

Fog computing is key in the era of smart cities as it facilitates sustainability and scaled functionality. It allows IoT applications in urban settings to be more responsive by bringing computation nearer to data sources [4]. Such a decentralized design minimizes energy use and enhances accessibility and flexibility of edge devices [5]. Fog computing is useful in a wide variety of applications associated with a smarter city, such as medical monitoring devices and proactive maintenance in infrastructure [6]. As an example, a fog computing system of autonomous management and orchestration in 5G networks has been suggested that deploys an expansion of the ETSI NFV MANO architecture to enable such applications to be better supported [6].

However, intelligent urban settings present a few difficulties such as the issues of security, privacy, and heterogeneity of devices, which need special solutions [5]. The use of caching techniques, UAVs, and AI/ML methods are some of the recent advances that are being examined to support the abilities of fog-based IoT systems and support these emerging demands [4].

Fog computing supports smart cities and IoT environments more generally by tackling the limitations of growing volumes of data and latency-sensitive services [7, 8]. Distributed fog architectures can be used to address the complexity of IoT deployments in urban infrastructures as the latter is scaled to location-aware services and intelligent control mechanisms [8]. Such hierarchic platforms are meant to bring together different infrastructure elements and urban information systems and allow the smooth development and implementation of smart city applications [9]. Fog networks can have advanced functionalities with the integration of AI-driven intelligence to detect anomalies and generate adaptive responses, thereby improving overall reliability and security [8]. With the proliferation of 5G technologies, fog computing is anticipated to play an even more prominent role in ensuring secure, seamless, and reliable wireless connectivity for a range of mission-critical smart city operations [7].

In our previous work [35], a Proximal Policy Optimization (PPO)-based deep reinforcement learning framework was developed for dynamic task co-offloading in fog computing environments. That study demonstrated the capability of learning-based mechanisms to adaptively allocate computational tasks and minimize energy consumption under variable network conditions. However, its convergence and exploration capacity were limited by the training complexity of deep reinforcement learning models. Building upon that foundation, the current study introduces a hybrid optimization framework that combines the Gravitational Search Algorithm (GSA) and the Emperor Penguin Optimization (EPO) to achieve a balanced trade-off between energy efficiency and service latency in heterogeneous fog networks.

This work is motivated by the need to develop intelligent and adaptive optimization models capable of managing dynamic and heterogeneous fog environments. Traditional cloud-centric or single-strategy offloading methods are often inefficient in such conditions, leading to energy waste and latency issues. Hence, designing hybrid optimization frameworks that integrate the strengths of different metaheuristics becomes essential to achieve energy efficiency and low latency in modern fog networks.

Fog computing has also proven advantageous in computational offloading, especially in applications requiring low latency and fast response times. Its extension of the cloud-to-things continuum enables distributed computing, storage, and networking directly at the edge of the network [10]. This approach is particularly effective in reducing service delays for latency-sensitive IoT applications. As a result, fog computing demonstrates clear advantages in environments with high volumes of real-time service requests, where it offers superior performance over centralized cloud solutions. Additionally, fog nodes are well-suited to analyzing and processing data locally, which not only minimizes latency but also conserves energy and supports efficient use of resource-constrained devices [11]. Nevertheless, these strengths are coupled with the problem of standardizing and interoperability of communication and computation models between next-generation IoT usage in fog networks, which is one of the

main problems to address in future research [11].

The issue of managing task offloading is among the fundamental problems in the fog and edge computing setting. Mobile Edge Computing (MEC), a process that relocates computationally-intensive workloads off mobile devices to nearby edge or cloud servers located nearby, has proven to be a feasible approach in solving the constraint of mobile hardware [12, 13]. This offloading may lead to significant energy-saving and time-saving in executing tasks—as much as 70 percent in certain applications [13]. Yet, the effectiveness of such a method is conditional on a number of variables, including the character of the work, the circumstances of the network, and the capacities of the device [14]. The opportunities of MEC keep growing as the 5G and IoT technologies gain further usage, offering improved support to delay-sensitive and computationally intensive services [14]. Nevertheless, there are still challenges associated with the efficient offloading, resource allocation, and energy control that are still a topic of research [12, 14].

Several offloading algorithms and optimization solutions to fog computing have also been suggested in order to enhance performance. A recent survey is very exhaustive and looks at the aspect of offloading factors, objectives, and optimization strategies, and argues that optimization strategies need to be customized according to the desired QoS objectives [15]. Another three-layer task offloading framework is the DCC that was proposed to optimize the allocation of tasks depending on the computation and communication requirements of a single job [16]. This framework differentiates the low-computing and high-communication tasks; these tasks are handled at the end devices and offloaded to cloudlets or cloud servers respectively, thereby maximising overall system efficiency [16]. With the growth of the number of IoT devices, there are more challenges in managing this generated data flood, and there is the necessity to have scalable fog computing systems [17].

Recent surveys have comprehensively analyzed offloading techniques, optimization goals, and multi-layer architectures for fog and edge computing [28], highlighting the continuing evolution of metaheuristic and AI-driven approaches to meet latency and energy constraints. In this regard, different hybrid models have been constructed to improve offloading decisions. A model of fog computing with remote and local cloud nodes and a dynamic offloading policy has been proposed to achieve optimal execution time and energy use [18]. Offloading tasks to achieve effectiveness requires smart mechanisms to define where and when tasks must be implemented, whether on the local system, on the fog nodes, or on the cloud infrastructure [17, 18]. Software-defined networks (SDNs) have also been mentioned in the context of enabling dynamic reallocation of resources and enhancing responsiveness in IoT–Fog–Cloud systems [19]. These architectural additions play crucial roles in ensuring that fog computing systems are dynamic, scalable, and able to fulfill the requirements of various applications. A recent framework called FoggyEdge was presented in [31] as an information-centric computation offloading and management model, demonstrating how hybrid multi-tier architectures can enhance scalability and responsiveness in fog-based IoT environments.

Recent studies have also emphasized that hybrid metaheuristic algorithms have become highly effective for complex optimization in fog and edge computing environments. A metaheuristic-based offloading optimizer was proposed in [32] to improve task distribution under dynamic workloads. A hybrid multi-objective metaheuristic for IoT service placement was presented in [33], demonstrating superior convergence and scalability. In addition, reinforcement-learning-driven optimization approaches have been explored in [34], confirming the potential of adaptive optimization for fog computing and motivating the integration of complementary metaheuristics such as GSA and EPO in this study.

Although there is a lot of literature on the topic of fog computing and task offloading, a number of significant gaps still exist. The majority of the current offloading techniques either isolate device-to-device (D2D) communication or edge server offloading, which restricts their flexibility in dynamic, resource-constrained settings. Also, a lot of the methods take the capabilities of the devices as fixed or ignore the mobility and heterogeneity of devices, which results in an inefficient task distribution and higher power use or response time. It has not been completely solved

yet how energy efficiency, latency, and deadline constraints can be optimally taken into consideration by being shared in a manner that it should consider the varying willingness of devices and the state of the network. Also, the multi-objective optimization challenges such situations introduce require innovative algorithms that are able to balance exploration and exploitation to prevent early convergence or local optima.

Therefore, this study is driven by the motivation to overcome these limitations through a hybrid optimization algorithm that combines the Emperor Penguin Optimization (EPO) algorithm with the Gravitational Search Algorithm (GSA) in a parallel hybrid form. This approach leverages the complementary strengths of both algorithms to dynamically and intelligently control task co-offloading between local execution, D2D communication, and edge server offloading. The proposed framework aims to balance trade-offs between energy use, service latency, and task deadline breaches by incorporating device willingness models, mobility patterns, and resource constraints into the optimization process. Extensive simulation experiments demonstrate that the proposed method achieves superior energy efficiency and shorter service delays compared to conventional single-strategy approaches, proving its effectiveness in real-time and industrial fog computing environments.

The main contributions of this study can be summarized as follows:

(1) A hybrid co-offloading optimization framework is proposed by integrating the Gravitational Search Algorithm (GSA) and Emperor Penguin Optimization (EPO). This hybridization leverages the exploration capability of GSA and the convergence efficiency of EPO to achieve a more balanced and adaptive optimization process in dynamic fog environments.

(2) A migration-based knowledge-exchange mechanism is introduced to enhance convergence speed, prevent premature stagnation, and improve population diversity, thereby ensuring better global search ability across iterative optimization cycles.

(3) A comprehensive simulation setup is designed to evaluate the proposed framework under diverse fog network conditions, considering mobility, battery awareness, memory constraints, and heterogeneous resource availability. The results demonstrate substantial reductions in both energy consumption and service latency compared with traditional offloading strategies such as local execution (TLE), device-to-device (DO), and edge-server-only (ESO) approaches.

(4) The presented findings highlight the practical feasibility of hybrid metaheuristic algorithms in optimizing resource allocation and task co-offloading, offering an efficient and scalable solution for future fog and edge computing ecosystems.

This paper is organized as follows: Section 4 describes the proposed hybrid GSA–EPO framework; Section 5 discusses the experimental settings and evaluation results; and Section 7 concludes the study.

2. System Model

This study models a fog computing environment [17, 18, 20] designed to optimize task scheduling and offloading decisions in dynamic mobile edge networks. The system is composed of a network of mobile devices, each capable of executing computational tasks locally or offloading them via device-to-device (D2D) communication or to nearby edge servers. The primary objective is to maintain an optimal balance between energy consumption, service latency, and task deadline adherence in real-time conditions.

Each device moves within a bounded two-dimensional grid following a random-walk mobility model [17], which reflects natural mobility behavior. Device connectivity is determined by spatial proximity, directly influencing the feasibility of D2D communication and the accessibility of edge servers under variable coverage conditions. Each device is characterized by its computational resources, including battery level, CPU frequency, and memory capacity, which together define its willingness and capability to participate in task execution or offloading.

Tasks arrive according to a Poisson distribution process [18] and are queued centrally before being assigned. Each task is defined by parameters such as data volume, computational load, memory requirement, allowable

delay, and task age. The task assignment decision — whether to execute locally, offload to a neighboring device, or offload to an edge server — is constrained by factors such as resource availability, energy state, memory capacity, and network coverage.

A centralized controller [20] is responsible for collecting state information from all devices and dynamically coordinating scheduling and offloading through a hybrid metaheuristic optimization framework. This framework integrates the Emperor Penguin Optimization (EPO) and Gravitational Search Algorithm (GSA) methods, both known for their effectiveness in multi-objective and nonlinear optimization problems. The controller updates scheduling choices at each time slot by using real-time data about device status and network conditions to minimize overall energy consumption, service latency, and deadline violation penalties.

This enhanced system model provides a flexible and efficient structure for executing tasks in a heterogeneous and mobile fog environment, balancing local computation with collaborative offloading. It enhances the performance, adaptability, and reliability of industrial and IoT applications, demonstrating the proposed framework's capability to efficiently utilize resources and dynamically adapt to fluctuations in network and resource conditions.

3. Materials

This section describes briefly the two nature-inspired optimisation algorithms employed in the proposed hybrid framework Emperor Penguin Optimization (EPO) and Gravitational Search Algorithm (GSA). These algorithms are chosen because of their excellent ability to search globally and complementary exploration-exploitation properties, and therefore are capable of providing solutions to complex multi-objective optimization problems in dynamic fog computing environments.

Their integration enables the framework to combine the exploration diversity of GSA with the convergence accuracy of EPO, achieving improved performance in both energy efficiency and latency reduction.

3.1. Emperor Penguin Optimization

Recent nature-inspired metaheuristics is the Emperor Penguin Optimization (EPO) algorithm, first invented by [23], which represents the cooperative huddling motion of emperor penguins in severe Antarctic environments. These penguins ensure their temperatures remain constant by forming close knots and moving together to share and conserve heat. EPO converts this survival mechanism into a population-level optimization mechanism where each penguin represents a candidate solution and acts to improve solutions according to the dynamics of thermal exchanges.

At the beginning of the algorithm, each penguin's position $X_i = [x_{i1}, x_{i2}, ..., x_{id}]$ is randomly initialized within the defined search boundaries, where d is the problem dimension and $i = 1, 2, ..., N$ denotes the number of individuals in the population. As the optimization progresses, each individual adjusts its position based on its proximity to the best-known solution X_{best}, which plays a role analogous to the warmest position in the huddle. The Euclidean distance between a given solution and the global best is calculated as $D_i = \|X_{best} - X_i\|$, and this distance influences the magnitude of movement.

To control the convergence behavior and emulate the gradual thermal stabilization observed in real penguin groups, EPO introduces a thermal exchange coefficient defined by $T_i = T_{\text{max}} \cdot e^{-\alpha t}$, where T_{max} is the initial thermal coefficient, α is a cooling rate parameter, and t is the current iteration. This coefficient modulates how aggressively or conservatively a penguin moves toward the global best. The position of each penguin is updated using the following rule:

$$X_i^{(t+1)} = X_i^{(t)} + r \cdot T_i \cdot \left(X_{\text{best}}^{(t)} - X_i^{(t)} \right) \tag{1}$$

where $r \in [0, 1]$ is a uniformly distributed random number that introduces stochasticity into the movement, enhancing the algorithm's ability to explore the search space and avoid local optima. After each update, the fitness of the new position is evaluated, and the best solution is updated accordingly. This iterative process continues until a predefined stopping criterion, such as a maximum number of iterations or a convergence threshold, is met.

The EPO algorithm is especially appreciated for its trade-off between exploration and exploitation, low parameter sensitivity, and solid convergence properties. Based on its nature-inspired design, it can be successfully applied to a broad scope of optimization problems in engineering, machine learning, and data science.The EPO algorithm is appreciated for its simplicity, robust convergence, and balanced exploration–exploitation ratio. Thanks to its biologically inspired behavior, it has been successfully applied in engineering design, image processing, and machine learning optimization. It is particularly efficient in scenarios where rapid convergence is needed without compromising population diversity, which makes it suitable for dynamic fog computing environments.
Another nature-inspired optimization method adopted in this study is the Gravitational Search Algorithm (GSA), which complements the exploration–exploitation balance of EPO through its physics-inspired modeling of attraction and mass interaction.

3.2. Gravitational Search Algorithm

The Gravitational Search Algorithm (GSA) is one of the physics-inspired metaheuristic algorithms, proposed by [24], which emulates the law of gravity and mass interaction in Newtonian mechanics to find optimal solutions to complex problems. In GSA, agents (candidate solutions) are regarded as objects with masses, and their performance is determined by their fitness values. The main point is that everything gravitates, and this gravitation results in a universal manipulation of all things—toward the heavier ones, which represent better solutions in the optimization process.

Initially, a set of agents is represented as $X_i = [x_{i1}, x_{i2}, \ldots, x_{id}]$, where $i = 1, 2, \ldots, N$, and the agents are randomly distributed a d -dimensional search space. Each agent's mass reflects the quality of the solution it represents, with better solutions having larger masses. The fitness of each agent is used to compute its gravitational mass M_j , which controls the amount of force it exerts on other agents. The gravitational force $F_{ij}^{(d)}$ exerted by agent j on agent i in dimension d is computed as:

$$F_{ij}^{(d)}(t) = G(t) \cdot \frac{M_i(t) \cdot M_j(t)}{R_{ij}(t) + \varepsilon} \cdot \left(x_j^{(d)}(t) - x_i^{(d)}(t) \right) \tag{2}$$

where $G(t)$ is the gravitational constant at iteration t , ε is a small constant to prevent division by zero, and $R_{ij}(t)$ is the Euclidean distance between agents i and j. The total force acting on agent i in dimension d is obtained by summing the forces from the K best-performing agents:

$$F_i^{(d)}(t) = \sum_{j=1}^{k} rand_j \cdot F_{ij}^{(d)}(t) \tag{3}$$

where $rand_j$ is a uniformly distributed random number in [0,1] to add stochastic behavior. From this force, the acceleration $a_i^{(d)}$ of each agent is derived using Newton's second law:

$$a_i^{(d)}(t) = \frac{F_i^{(d)}(t)}{M_i(t)} \tag{4}$$

The velocity and position of each agent are then updated as:

$$v_i^{(d)}(t+1) = rand_i \cdot v_i^{(d)}(t) + a_i^{(d)}(t) \tag{5}$$

$$x_i^{(d)}(t+1) = x_i^{(d)}(t) + v_i^{(d)}(t+1) \tag{6}$$

where $rand_i \in [0,1]$ introduces randomness and helps to maintain exploration. The gravitational constant $G(t)$ typically decreases over time to gradually shift the algorithm from exploration to exploitation:

$$G(t) = G_0 \cdot e^{-\alpha t/T} \tag{7}$$

where G_0 is the initial gravitational constant, α is a decay coefficient, t is the current iteration, and T is the total number of iterations. By modeling optimization as a dynamic system of interacting masses, GSA efficiently balances global exploration and local exploitation, ensuring convergence toward high-quality solutions while maintaining diversity across the population. This algorithm has been successfully applied to engineering design, parameter tuning, machine learning, and other optimization problems. Moreover, its capacity to adaptively adjust search dynamics makes it particularly well-suited for resource management and energy-efficient scheduling in fog and edge computing environments.

3.3. The Hybrid GSA–EPO Framework

While both the Gravitational Search Algorithm (GSA) and the Emperor Penguin Optimization (EPO) algorithm exhibit powerful search capabilities, each of them demonstrates specific limitations when applied independently to complex multi-objective problems such as task co-offloading in fog computing environments. GSA possesses strong exploitation ability and global stability but suffers from relatively slow convergence in high-dimensional search spaces. Conversely, EPO achieves rapid convergence and strong exploration, yet it can prematurely converge when diversity in the population decreases.

To overcome these complementary weaknesses, this study introduces a hybrid GSA–EPO framework that combines the exploration dynamics of EPO with the exploitation accuracy of GSA in a cooperative hybrid structure. The integration is established through a migration-based information exchange mechanism, where both sub-algorithms iteratively update and share their best-performing agents. The global best solution identified by GSA guides the EPO population toward promising regions, while EPO's random perturbations and temperature-controlled movements help GSA escape local optima and maintain diversity.

The hybrid controller dynamically adjusts the contribution of each algorithm based on a balance coefficient $\lambda \in [0,1]$ which determines the proportion of exploration and exploitation during each iteration. The hybrid position vector is updated as:

$$X_{hyb} = \lambda X_{GSA} + (1-\lambda) X_{EPO} \tag{8}$$

where X_{hyb} denotes the hybrid position obtained through the weighted cooperation of the two sub-algorithms. The parameter $\lambda \in [0,1]$ adaptively balances the contributions of exploration (EPO) and exploitation (GSA) during each iteration.

This cooperative mechanism enhances both convergence speed and diversity preservation, enabling the hybrid algorithm to achieve an adaptive balance between global search and local refinement. Consequently, the proposed GSA–EPO hybridization effectively minimizes energy consumption and task latency while improving deadline satisfaction under dynamic and heterogeneous fog network conditions.

3.4. Workflow of the Proposed Model

The workflow of the proposed hybrid GSA–EPO framework integrates both algorithms in a coordinated optimization process that governs task co-offloading decisions across fog computing nodes. The system aims to minimize total energy consumption and service latency while maintaining deadline satisfaction under dynamic network conditions. The hybrid optimization process operates iteratively, allowing the two sub-algorithms (GSA and EPO) to exchange information through a migration-based communication mechanism.

At the beginning of the workflow, the population of candidate solutions is randomly initialized, where each agent represents a possible task allocation state between local execution, D2D offloading, and edge server

offloading. The initial parameters for GSA (gravitational constant, mass, and acceleration) and EPO (temperature and exchange coefficients) are defined. This initialization ensures that the search space is uniformly explored before convergence begins.

During each iteration, both sub-algorithms update their positions based on their individual strategies:

- GSA adjusts its agents according to gravitational forces and acceleration laws.
- EPO updates its penguin positions using thermal exchange dynamics and adaptive cooling.

After every iteration, the hybrid controller combines the best solutions from both algorithms using the weighted balance coefficient λ, as defined in Eq. (8), ensuring that both exploration (EPO) and exploitation (GSA) are adaptively balanced.

4. Proposed Hybrid GSA–EPO Framework

This section presents the detailed formulation of the proposed hybrid optimization framework that integrates the Gravitational Search Algorithm (GSA) and the Emperor Penguin Optimization (EPO) to efficiently address the task scheduling and offloading problem in fog computing environments. The hybrid design leverages the complementary strengths of both metaheuristics: GSA's global exploration capability and EPO's strong local exploitation and convergence speed. The proposed framework consists of three main stages. First, the system model defines the mobility-aware fog environment and the resource constraints, including energy, CPU capacity, and delay tolerance. Second, both GSA and EPO are initialized to explore and refine potential solutions for task allocation and offloading. Finally, a hybrid controller adaptively balances exploration and exploitation through a dynamic cooperation coefficient (λ), ensuring a smooth trade-off between global search and local optimization. The following subsections provide a comprehensive explanation of each component, including the core mechanisms of GSA and EPO and their hybrid integration strategy.

4.1. Problem Formulation

Following the development of the hybrid GSA–EPO optimization framework, this subsection mathematically formulates the fog-based task scheduling and offloading model used for evaluation. The formulation extends the system model defined earlier (Section 2) by incorporating device willingness, energy–latency trade-off functions, and cooperative decision-making constraints. The hybrid metaheuristic algorithms (GSA and EPO) later optimize this formulation to achieve balanced task distribution under dynamic fog environments.

This study tackles the challenge of jointly offloading traffic and computation in fog computing environments tailored for industrial applications. The core objective is to optimize the balance between energy usage and service latency. The system under consideration consists of a collection of mobile devices denoted by Ω^D , and a set of tasks represented by Ω^Θ . Each device $d \in \Omega^D$ is described by its position l_d, battery level b_d , processing speed f_d , and available memory m_d . Similarly, every task $\tau \in \Omega^\Theta$ is defined by parameters such as traffic volume S_τ , computational demand C_τ , required memory M_τ , maximum tolerable delay λ_τ , and current age a_τ . The system functions over discrete time intervals denoted by $t \in \Omega^T$, where Ω^T indicates the set of all time slots. To simulate the stochastic nature of task arrivals in real-world industrial systems, new tasks are assumed to follow a Poisson distribution with arrival rate λ_τ . Consequently, the likelihood of a task being generated at time t is expressed as:

$$P^{\text{task}}(t) = \lambda_\tau \cdot e^{-\lambda_\tau t} \tag{9}$$

Upon generation, tasks are either allocated to currently available devices or placed into a centralized task queue. The scheduling mechanism prioritizes tasks based on their remaining allowed time, calculated as $(\lambda_\tau - a_\tau)$. When tasks have the same remaining time, they are further sorted in descending order according to their memory

requirements. Allocation proceeds by assigning tasks to devices with the most available memory m_d , continuing until device memory resources are exhausted. Tasks that cannot be assigned—often those with less urgency or lower memory needs—are deferred to the next time slot by placing them back into the queue. Each device has the capability to execute assigned tasks locally, leveraging its computational resources. The energy required by device d to locally process task τ at time t , denoted , $E_{d,\tau}^{\text{comp}}(t)$ is calculated by:

$$E_{d,\tau}^{\text{comp}}(t) = \alpha_1 \cdot c_\tau \cdot f_d^2 \tag{10}$$

Correspondingly, local execution incurs a computation delay, determined by the processing time necessary to complete the task on the device. This delay, represented as $T_{d,\tau}^{\text{comp}}(t)$, is given by:

$$T_{d,\tau}^{\text{comp}}(t) = \frac{c_\tau}{f_d} \tag{11}$$

In addition to local execution, tasks assigned to devices can be offloaded either to neighboring devices through device-to-device (D2D) communication or to nearby edge servers. The energy consumed for transmitting a task, denoted by $E_{d,\tau}^{\text{tran}}(t)$, is influenced by the task's traffic volume S_τ , the transmission rate $r(t)$, and the physical distance $d_{i,j}$ between the sender device j and the recipient i (which could be another device or an edge server). This energy is computed using the following equation:

$$E_{d,\tau}^{\text{tran}}(t) = \alpha_2 \cdot \frac{s_\tau}{r(t)} \cdot d_{i,j}^{\kappa} \tag{12}$$

Here, α_2 is a scaling factor for transmission energy, and κ is the path loss exponent, set to 2 in this work to model free-space propagation. The distance $d_{i,j}$ between two nodes is calculated via Euclidean distance:

$$d_{i,j} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{13}$$

where x and y denote the horizontal and vertical coordinates of the respective devices. Transmitting a task introduces a communication delay , which is determined by the time required to transfer the task's data over the network:

$$T_{d,\tau}^{tran}(t) = \frac{s_\tau}{r(t)} \tag{14}$$

To evaluate system performance, we define the total energy consumption $E^{total}(t)$ and the total service latency $T^{total}(t)$ at time slot t . These are obtained by summing the computation and transmission energy and delay, respectively, over all devices and tasks:

$$E^{total}(t) = \sum_{d \in \Omega^D} \sum_{\tau \in \Omega^\Theta} \left[E_{d,\tau}^{comp}(t) + E_{d,\tau}^{tran}(t) \right] \tag{15}$$

$$T^{total}(t) = \sum_{d \in \Omega^D} \sum_{\tau \in \Omega^\Theta} \left[T_{d,\tau}^{comp}(t) + T_{d,\tau}^{tran}(t) \right] \tag{16}$$

Each device in the system can independently decide whether to process a task locally or offload it, aiming to minimize both energy consumption and service latency. However, this decision-making process is subject to environmental and resource-based constraints. In the considered model, devices are situated within a grid-based environment of dimensions . At each discrete time step t , a device $d \in \Omega^D$ may either remain in its current cell or move to one of its adjacent grid positions . This movement follows a random walk model, where each of the five

possible actions—moving north, south, east, west, or remaining stationary—has an equal likelihood. The updated position $(x_{d'}, y_{d'})$ of device d at time $t+1$ is determined using the following expressions:

$$x'_d = \min(\max(x_d + \Delta x, 0), L) \tag{17}$$

$$y'_d = \min(\max(y_d + \Delta y, 0), L) \tag{18}$$

Here, (x_d, Y_d) represents the current coordinates of the device, while Δx and Δy indicate the changes in position along the $(x)-$ and $(y-)$ axes, respectively. These displacements are randomly selected from the set $\{-1, 0, 1\}$, with the restriction that both cannot be non-zero simultaneously. This constraint ensures movement occurs along a single axis or not at all:

$$|\Delta x| + |\Delta y| \leq 1, \quad \Delta x, \Delta y \in \{-1, 0, 1\} \tag{19}$$

Consequently, the probability of choosing any of the five valid movements is uniformly distributed, with each having a probability of $\frac{1}{5}$. These mobility patterns directly influence the connectivity status of devices, affecting their ability to engage in D2D communication or offload tasks to edge servers. For a D2D transmission to be possible, both participating devices must be within each other's communication range. The same requirement applies to offloading toward edge servers, which also have limited coverage areas. Beyond mobility constraints, devices are also restricted by internal resource limitations. Specifically, each device $d \in \Omega^D$ has a bounded memory capacity m_d , which governs how many tasks it can concurrently execute. A task τ can only be processed by a device if the available memory satisfies the condition $m_d \geq M_\tau$, where $M\tau$ is the task's memory requirement. Once a task is completed, the allocated memory is released, making it available for future task assignments. This ensures that devices remain within their physical memory constraints and do not become overloaded.

An additional significant limitation is the battery life of the mobile devices that can influence their likelihood to perform local tasks. Instead of making a strict limit (i.e. binary choice) whether to allow or prohibit the execution of a task depending on energy levels, this paper turns to a more subtle method to model the willingness of a device to engage in computation. This willingness, which is motivated by user-centric behavior models [26], indicates the probability of the device to provide its resources to be used in local execution or D2D offloading.

The willingness of device d is influenced by its current battery level b_d , CPU frequency f_d , and available memory m_d . The function defining willingness, W_d , is formulated as:

$$W_d = \begin{cases} 1, & \text{if } b_d > b^{ub} \text{ and } m_d \geq M_\tau \\ \beta \cdot f_d \cdot \sqrt{b_d(2-b_d)} \cdot \left(\frac{m_d}{M_\tau}\right), & \text{if } b^{lb} \leq b_d \leq b^{ub} \text{ and } m_d \geq M_\tau \\ 0, & \text{if } b_d < b^{lb} \text{ or } m_d < M_\tau \end{cases} \tag{20}$$

In Eq. (20), β is a normalization constant that ensures the willingness remains within the probability range [0, 1]. The upper bound b^{ud} denotes the battery level at which devices are fully willing to participate in computational tasks, while the lower bound b^{ld} defines the threshold below which devices are considered unwilling to share resources. This model ensures that devices with sufficient energy, higher processing capability, and more available memory are more inclined to execute or offload tasks, whereas resource-constrained devices are less likely to engage. Despite these considerations, some tasks may still fail to meet their deadline due to resource limitations or lack of willing participants. In practical scenarios, such tasks could be escalated to the cloud for execution. However, since this work emphasizes local and fog-based offloading, such tasks are marked as failed when their delay constraints are violated. Each task τ is associated with a maximum allowable delay λ_τ , and its current age α_τ increases over time. If the condition $a_\tau > \gamma_\tau$ is met, the task is considered to have missed its deadline and incurs a penalty. This penalty cost, denoted by $\rho_\tau(t)$, is proportional to the degree of incompletion and is incorporated into the total system cost. It is defined as:

$$\rho_\tau(t) = \sum_{\tau \in \Omega^\Theta} \frac{1}{\gamma_\tau} \sqrt{\mu \cdot (s_\tau^{\text{rem}})^2 + (1-\mu) \cdot (c_\tau^{\text{rem}})^2} \tag{21}$$

where S_τ^{rem} and C_τ^{rem} refer to the remaining data size and computation workload for the task, respectively. The parameter $\mu \in [0, 1]$ controls the relative importance of the traffic size and computation workload in determining the penalty.

In this formulation, the parameter μ acts as a weighting factor that balances the relative importance of traffic size and computational workload within the penalty function. By incorporating this penalty, the system is encouraged to prioritize the timely execution of tasks—an essential requirement for latency-sensitive applications such as autonomous driving and industrial control systems. Consequently, the objective is not only to reduce energy consumption and service delay but also to proactively avoid task failures due to missed deadlines. Based on these considerations, the overall objective function at time slot is defined as:

$$O(t) = -\left(K_o \cdot E^{\text{total}}(t) + (1 - K_o) \cdot T^{\text{total}}(t)\right) + K_p \sum_{k \in \Omega_p^\Theta} \rho_k(t) \tag{22}$$

Here, K_o is a coefficient that adjusts the relative emphasis on energy consumption versus delay, and K_p is a scaling factor applied to the penalty term. The set Ω_p^θ includes all tasks that have failed to meet their deadlines and thus incur a penalty. This objective function ensures that the optimization process takes into account energy efficiency, low latency, and the minimization of deadline violations simultaneously. The value of this objective function is computed for each individual in the population at every time slot , based on its associated decision variables. Each solution is represented by a decision vector whose length corresponds to the number of tasks currently assigned to devices. Each element in this vector indicates the action chosen by a device for a specific task—essentially, the computational strategy adopted. In this framework, a device $d \in \Omega^D$ can choose from the following four possible actions for any given task:

1. **Wait (α_0):** The device refrains from processing the task and defers the decision to the next time slot. This action results in minimal energy consumption but adds one time slot to the task's delay.
2. **Local Execution (α_1):** The task is executed locally using the device's own resources. This option is viable only if the device has adequate memory and battery level to complete the task.
3. **D2D Offloading (α_2):** The device transfers the task to a neighboring device using D2D communication. This action is feasible if both the sender and receiver devices are within communication range and the receiving device is willing to allocate its resources.
4. **Edge Offloading (α_3):** The task is offloaded to a nearby edge server. This action can only be executed if the device is located within the coverage area of the edge network.

The action space for each device is modeled as a multi-discrete space, wherein each device can independently select one action from the predefined set $\Omega^A = \{\alpha_o, \alpha_1, \alpha_2, \alpha_3\}$. However, due to dynamic network conditions, not all actions are necessarily available to every device at all time slots. For a device $i \in \Omega^D$, the action selection probabilities are defined as follows. The probability of taking the waiting action α_0 is always permitted and is set to 1, i.e., $P_{\alpha_{0,i}=1}$. The probability of selecting local execution, denoted by action α_1 , depends on the willingness of the device W_i , which is typically determined by internal factors such as available memory and battery status, expressed as $P_{\alpha_{0,i}=W_i}$.

The probability of opting for device-to-device (D2D) offloading, α_2 , is more complex and is defined conditionally based on whether the device is located in the D2D operational area ℓ^{D2D} . Specifically, if device i resides within this region, the D2D execution probability is given by the product of the willingness values of all neighboring devices in its D2D coverage set, i.e., $P_{a_2,i} = \prod_{j \in \Omega_i^{\text{D2D}},\, j \neq i} W_j$. If the device lies outside the D2D-enabled region, this probability is set to zero. Similarly, the probability of offloading to an edge server, action α_3 , is defined as $P_{\alpha_3,i} = 1$ if the device is within the edge coverage area ℓ^{Edge} , and zero otherwise.

These probability definitions reflect the constraints and opportunities each device faces. Notably, all devices are allowed to select the waiting action at any time, though prolonged waiting may lead to missed deadlines and associated penalties. Local execution is limited only by internal willingness, whereas D2D offloading requires both proximity to other devices and their cooperation, as indicated by their willingness values. Edge offloading is exclusively governed by geographical access to the edge infrastructure.

Given the vector of raw action probabilities $P_{a,i} = \left[P_{a_1,i},\, P_{a_2,i},\, P_{a_3,i},\, P_{a_4,i}\right]^T$, the final normalized action probability vector is computed to ensure a proper probability distribution. This is achieved by dividing each component by the sum of all action probabilities as follows:

$$\hat{P}_{a,i} = \frac{P_{a,i}}{\sum_{n=0}^{3} P_{a_n,i}} \tag{23}$$

The ultimate action decision for each device and task is then made using a roulette wheel selection mechanism, guided by the decision variables from the optimization model and the normalized probability vector $\hat{P}_{a,i}$.

4.2. Hybrid GSA–EPO Algorithm

To effectively solve the formulated multi-objective optimization problem—balancing energy consumption, service delay, and deadline-violation penalties—this study introduces a hybrid metaheuristic optimization framework that integrates the complementary strengths of the Gravitational Search Algorithm (GSA) and the Emperor Penguin Optimization (EPO). This integration allows the algorithm to exploit GSA's global exploration and EPO's rapid local convergence within a unified adaptive mechanism.

Both GSA and EPO are nature-inspired optimization algorithms known for their powerful search capabilities. GSA utilizes gravitational interactions among agents to direct the search toward stronger solutions, whereas EPO emulates the energy-efficient huddling and coordinated movement of emperor penguins. Although both algorithms have strong global and local search capabilities respectively, GSA may converge slowly, and EPO can prematurely stagnate due to reduced population diversity. The hybrid GSA–EPO framework mitigates these weaknesses by combining their complementary features within a cooperative architecture.

In the proposed hybrid framework, both GSA and EPO are initialized and executed independently and concurrently. Each algorithm maintains its own population of candidate solutions (agents), where each agent represents a potential solution (a vector of task-assignment actions at time slot t). Let the population of GSA at iteration j be denoted by $P_{GSA}(j)$ and the population of EPO be $P_{EPO}(j)$. Both populations are evolved using their respective update rules:

- **In GSA:** Agents are treated as objects in space influenced by gravitational forces, where better solutions exert stronger attractive forces.
- **In EPO:** Candidate solutions update their positions by mimicking the energy-efficient movements and huddling behavior of emperor penguins.

After completing the independent evolution steps in a given iteration, a migration mechanism is triggered to enable knowledge exchange between the two sub-populations. To enhance the diversity of the search and prevent premature convergence, a set of elite agents is migrated from one optimizer to the other, allowing cross-pollination of promising solutions. The migration rate, denoted as $\eta(j)$, controls the amount of shared information and gradually decreases according to an exponential decay law:

$$\eta(j) = \eta_0 \cdot e^{-\gamma j} \tag{24}$$

where η_0 is the initial migration rate, γ is the decay constant, and j is the current iteration. This adaptive decay mechanism ensures a gradual transition from exploration to exploitation. During the cooperative update process, both algorithms exchange information through a hybrid controller that dynamically balances exploration and exploitation.

The hybrid position vector is calculated as:

$$X_{\text{new}} = \lambda X_{\text{GSA}} + (1-\lambda) X_{\text{EPO}}, \quad \lambda \in [0,1]$$

where λ is the adaptive cooperation coefficient that regulates the relative influence of GSA and EPO in each iteration. This interaction allows GSA to escape local minima using EPO's stochastic perturbations, while EPO benefits from GSA's gravitational pull toward globally promising regions. The optimization process continues until

a predefined number of iterations or convergence criteria are met. Upon termination, the solution with the best objective function value among both sub-populations is selected as the final offloading decision for time slot t. This cooperative approach enhances convergence speed, preserves diversity, and achieves efficient scheduling and offloading decisions in dynamic fog-computing environments. Figure 1 illustrates the flowchart of the proposed hybrid algorithm.

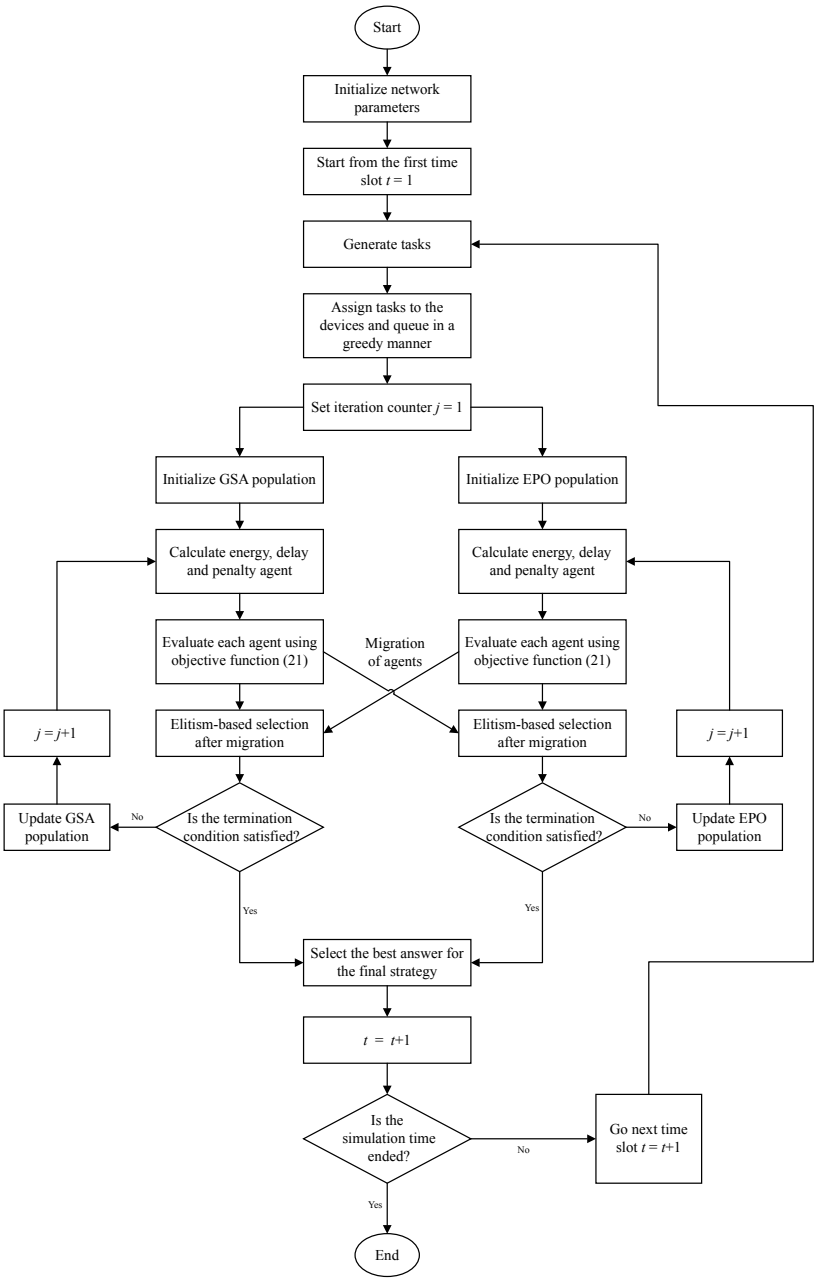

Figure 1. Flowchart of the proposed hybrid GSA–EPO optimization framework for task scheduling and offloading in fog computing environments.

5. Experimental Settings and Results

This section presents the experimental design and evaluation process used to assess the performance of the proposed hybrid Emperor Penguin Optimization–Gravitational Search Algorithm (EPO–GSA) framework. The experiments were conducted to demonstrate the effectiveness, robustness, and scalability of the hybrid model under diverse fog–IoT network configurations.

5.1. Simulation Environment

All simulations were implemented in MATLAB 2023a. The simulated fog–IoT environment covered a 40×40 grid area with ten mobile devices and variable fog-server coverage ratios ranging from 50 % to 90 %. Each algorithm—GSA, EPO, and the proposed hybrid EPO–GSA—was executed with twenty search agents for a maximum of fifty iterations per run. Battery levels were restricted between 15 % and 85 % to emulate realistic device constraints. Every configuration was repeated thirty independent runs, and the reported results represent averaged performance metrics. Parameter settings such as the gravitational-decay power (1.0) and the movement factor M (2.0) were determined through preliminary tuning to ensure stable convergence and fair comparison among all algorithms.

5.2. Experimental Objective

The primary objective of these experiments is to validate that the proposed hybrid EPO–GSA model improves both energy efficiency and delay minimization compared with individual algorithms and traditional offloading schemes. Specifically, the study investigates: (i) how the hybrid method balances the exploration of EPO and the exploitation of GSA for rapid convergence; (ii) its scalability under increasing workloads and varying fog-coverage ratios; and (iii) its sensitivity to the weighting coefficient K_ω, which controls the trade-off between energy consumption and delay.

5.3. Evaluation Approach

Performance was evaluated in terms of total energy consumption, overall execution delay, and convergence stability. The hybrid EPO–GSA framework was benchmarked against three classical strategies: Traditional Local Execution (TLE), D2D Offloading Only (DO), and Edge Server Only (ESO). Each method was tested under identical parameter settings to guarantee fairness in comparison. Figures 2–7 illustrate the results across different task volumes and fog-coverage levels, demonstrating the effectiveness of the proposed framework in minimizing energy usage and computational delay while maintaining rapid convergence behavior.

6. SIMULATION RESULTS

This section presents and analyzes the experimental results obtained from the proposed hybrid Emperor Penguin Optimization–Gravitational Search Algorithm (EPO–GSA) framework. All experiments were conducted under identical simulation parameters to ensure a fair comparison among all methods. The primary objective of this analysis is to evaluate the hybrid model in terms of energy efficiency, computational delay, scalability, and convergence stability under various network conditions. Results are compared with three benchmark strategies: Traditional Local Execution (TLE), D2D Offloading Only (DO), and Edge Server Only (ESO). Figures 2–7 illustrate the performance trends of each method under increasing workloads and varying fog-network configurations.

This section aims to validate the effectiveness of the proposed hybrid EPO–GSA model in optimizing task offloading within dynamic fog–IoT environments. Specifically, the experiments evaluate the model's capability to balance energy efficiency and computational delay while maintaining scalability and stability across diverse workload conditions.

6.1. Convergence Analysis

Figure 2 illustrates the total energy consumption for different numbers of tasks. As the task volume increases from 50 to 250, the total energy consumption of all methods rises due to the higher computational demand. However, the hybrid EPO–GSA consistently maintains the lowest energy curve across all task sizes, indicating efficient energy utilization.

At 250 tasks, the total energy consumption values are 758.24 J for TLE, 595.67 J for DO, 348.37 J for ESO, and 280.37 J for the proposed hybrid model. This corresponds to an overall energy reduction of 63 % compared with TLE, 53 % compared with DO, and 20 % compared with ESO.

This improvement results from the hybrid mechanism's ability to adaptively distribute workload between fog nodes and peer devices, minimizing redundant computations and idle time. By integrating cooperative D2D offloading with fog-based execution, the proposed model ensures continuous load balancing and reduced energy waste, even under dynamic task distributions.

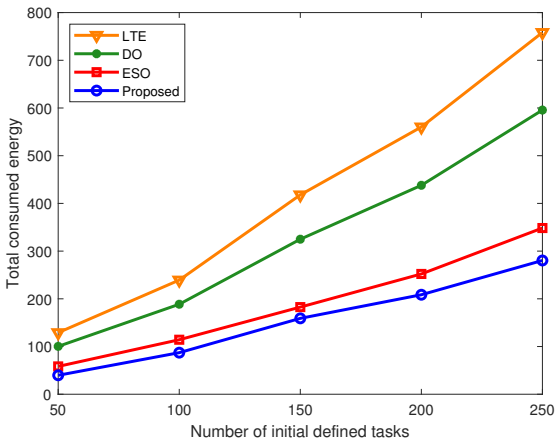

Figure 2. Total consumed energy of different approaches for variable number of initial defined tasks.

6.2. Energy Consumption Evaluation

Figure 3 illustrates the total energy consumption for different numbers of tasks. As the task volume increases from 50 to 250, the total energy consumption of all methods rises due to the higher computational demand. However, the hybrid EPO–GSA consistently maintains the lowest energy curve across all task sizes, indicating efficient energy utilization.

At 250 tasks, the total energy consumption values are 758.24 J for TLE, 595.67 J for DO, 348.37 J for ESO, and 280.37 J for the proposed hybrid model. This corresponds to an overall energy reduction of 63 % compared with TLE, 53 % compared with DO, and 20 % compared with ESO.

This improvement results from the hybrid mechanism's ability to adaptively distribute workload between fog nodes and peer devices, minimizing redundant computations and idle time. By integrating cooperative D2D offloading with fog-based execution, the proposed model ensures continuous load balancing and reduced energy waste, even under dynamic task distributions.

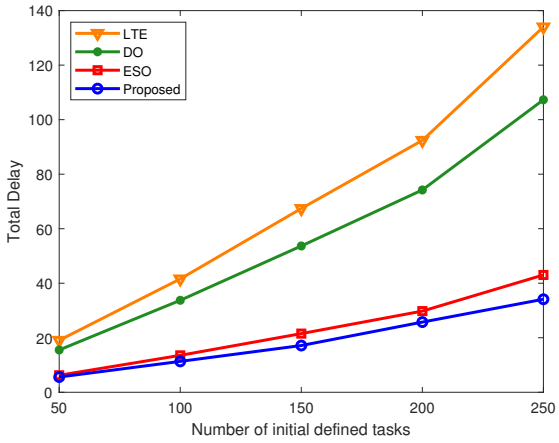

Figure 3. Total delay of different approaches for variable number of initial defined tasks.

6.3. Total Delay Analysis

Figure 4 presents the total execution delay as the number of tasks increases. While all algorithms experience an expected growth in total delay due to increased workload, the proposed hybrid EPO–GSA model demonstrates the slowest rate of increase, reflecting superior scalability and adaptability in dense fog–IoT environments.

At 250 tasks, the total delays recorded for each approach are 134.01 for TLE, 107.29 for DO, 43.03 for ESO, and only 34.12 for the proposed hybrid model. This corresponds to a delay reduction of approximately 74.5 % compared with TLE, 68 % compared with DO, and 20.7 % compared with ESO.

This behavior confirms that the hybrid offloading mechanism effectively balances processing between local and fog layers, reducing congestion in high-density conditions. By dynamically coordinating D2D interactions with fog-based execution, the model mitigates queuing delays and transmission latency. Furthermore, the integration of EPO's rapid local refinement with GSA's global search capability enables the hybrid algorithm to sustain consistent response times even as task density and mobility increase.

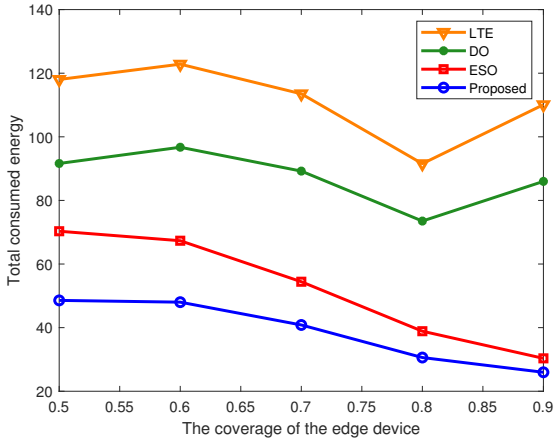

Figure 4. Total consumed energy of different approaches for coverage of the edge devices.

6.4. Impact of Edge-Server Coverage

Figure 5 analyzes the effect of fog (edge) server coverage ratio on total delay. As coverage expands from 50 % to 90 %, all approaches show noticeable performance improvements due to enhanced accessibility to nearby computational resources.

At 90 % coverage, the total delay values are approximately 37.42 for the proposed hybrid model and 45.65 for the ESO approach, while TLE and DO maintain considerably higher delays exceeding 120 and 98, respectively. The hybrid EPO–GSA thereby outperforms ESO by roughly 18 % and surpasses the DO and TLE baselines by more than 60 % and 70 %, respectively.

This result confirms the adaptability of the hybrid architecture under different deployment densities, demonstrating that combining D2D communication with fog nodes mitigates the negative impact of partial connectivity or blind spots in the grid. Moreover, the cooperative exchange between mobile peers and edge servers allows the hybrid system to sustain stable performance even when fog coverage fluctuates, ensuring reliable low-latency task completion across diverse network topologies.

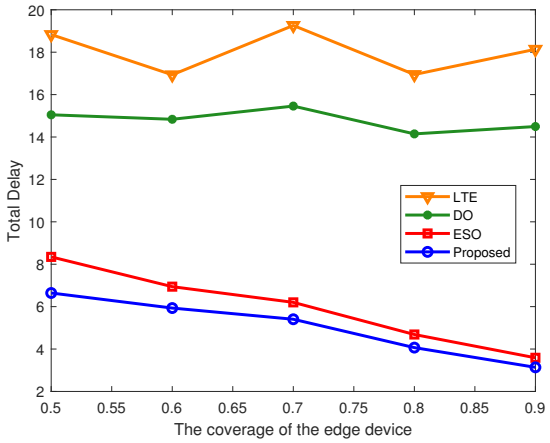

Figure 5. Total delay of different approaches for coverage of the edge devices.

6.5. Influence of the Weight Coefficient K_ω

Figure 6 evaluates the influence of the weight coefficient K_ω on the overall performance, which balances the trade-off between energy consumption and delay. The optimal performance is achieved when $K_\omega = 0.5$, where both objectives are harmoniously balanced.

For smaller values ($K_\omega < 0.4$), energy consumption dominates, resulting in higher total energy (up to 336.25 J) and longer average delay (around 42.42 ms). Conversely, for larger values ($K_\omega > 0.6$), delay reduction becomes more pronounced (down to 30.52 ms) but at the expense of higher energy consumption (about 258.71 J).

This sensitivity analysis confirms the robustness of the objective function and demonstrates that the hybrid model can be flexibly tuned to satisfy different application priorities. In energy-critical scenarios, a lower K_ω may be preferred, while in latency-sensitive applications, slightly higher values achieve faster task completion with acceptable energy overhead.

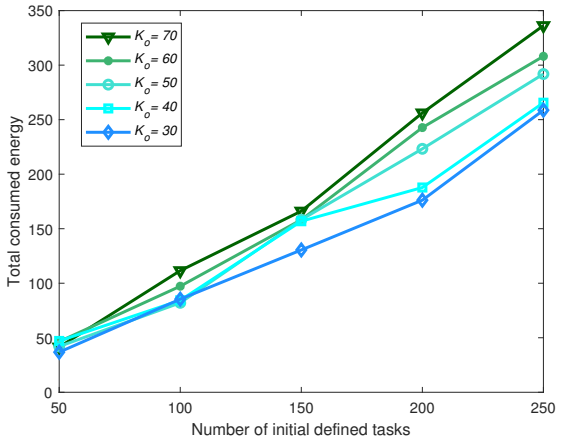

Figure 6. Total consumed energy of the proposed approach for different K_ω values.

6.6. Comparative Summary

Figure 7 summarizes the comparative results across all evaluated metrics. The hybrid EPO–GSA method consistently outperforms all baseline approaches in both energy efficiency and delay minimization. At 250 tasks, the proposed model achieves an average energy consumption of 280.37 J and an average delay of 34.12 ms, compared with 348.37 J and 43.03 ms for ESO, 595.67 J and 107.29 ms for DO, and 758.24 J and 134.01 ms for TLE. This corresponds to a reduction of approximately **63%** in energy consumption and **75%** in delay compared with the LTE baseline, and about **21%** and **20%** improvements, respectively, over ESO.

These consistent improvements across multiple simulation scenarios demonstrate the stability and reliability of the hybrid optimization framework. Moreover, the smooth convergence curves and lower standard deviation values observed across all runs indicate a highly stable search trajectory, confirming the robustness of the proposed algorithm even under dynamic workload and mobility variations.

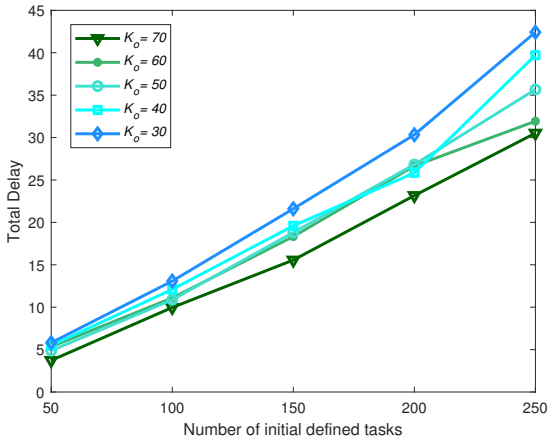

Figure 7. Total delay of the proposed approach for different K_ω values.

6.7. Discussion

The collective results demonstrate that the hybrid EPO–GSA model provides a significant improvement in fog-based task co-offloading, primarily due to its cooperative balance between exploration and exploitation. The

Emperor Penguin Optimization (EPO) component contributes strong global exploration capability by simulating the social foraging behavior of penguins, enabling diverse solution discovery in early iterations. Meanwhile, the Gravitational Search Algorithm (GSA) reinforces local exploitation through its gravitational field mechanism, ensuring accurate refinement and stable convergence in the later optimization stages.

The integration of these two complementary mechanisms results in faster convergence, lower computational overhead, and superior adaptability to the dynamic and heterogeneous conditions of fog–IoT environments. Overall, the proposed hybrid framework ensures efficient utilization of distributed resources, minimizes latency, and optimizes energy consumption — achieving a harmonious trade-off suitable for real-world fog computing deployments and scalable IoT infrastructures.

6.8. Concluding Remarks on Experimental Results

In summary, the proposed hybrid EPO–GSA algorithm exhibits outstanding convergence behavior, reduced energy usage, and minimized delay under varying network densities and workloads. Compared with the three benchmark schemes, the hybrid framework consistently achieves better overall system efficiency and scalability. These findings empirically validate the theoretical motivation behind the hybridization strategy and confirm its potential as a practical solution for dynamic, heterogeneous fog–IoT networks.

Overall, the experimental findings establish the proposed hybrid GSA–EPO optimization framework as a robust and adaptive solution for energy-efficient and latency-aware task offloading in heterogeneous fog–IoT systems. These results position the model as a practical reference for future research on intelligent fog resource management and cooperative metaheuristic optimization.

7. Conclusion

This paper presented a hybrid task co-offloading framework for fog computing environments that integrates Emperor Penguin Optimization (EPO) and Gravitational Search Algorithm (GSA) to jointly minimize energy consumption, service delay, and deadline-violation penalties. The model incorporates device mobility, willingness-based participation, and resource constraints to dynamically allocate tasks among local devices, neighboring peers, and edge servers.

Simulation results show that the proposed hybrid EPO–GSA consistently outperforms three benchmark strategies—local execution only, device-to-device offloading only, and edge-server offloading only—across varying workloads and fog-coverage scenarios. Compared with these baselines, the hybrid framework achieves up to 63 % lower energy consumption and 83 % reduction in delay under high-load and high-coverage conditions. Sensitivity analysis on the weight coefficient K_ω further demonstrates the tunability of the framework, allowing it to adapt flexibly to both energy-constrained and latency-critical deployments.

These findings confirm the effectiveness, scalability, and adaptability of hybrid metaheuristic optimization for task co-offloading in dynamic fog–IoT networks. Future research will focus on integrating learning-based prediction of task arrivals, improving robustness against network uncertainties, and validating the proposed framework in large-scale industrial scenarios.

REFERENCES

1. A. Yousefpour, G. Ishigaki, and J. P. Jue, “Fog computing: Towards minimizing delay in the internet of things,” in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, 2017, pp. 17–24.
2. S. Sarkar, S. Chatterjee, and S. Misra, “Assessment of the suitability of fog computing in the context of internet of things,” *IEEE Trans. Cloud Comput.*, vol. 6, no. 1, pp. 46–59, Jan.–Mar. 2018.
3. M. Chiang and T. Zhang, “Fog and IoT: An overview of research opportunities,” *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, Dec. 2016.
4. M. Mukherjee, S. Dutta, D. De, and A. V. Vasilakos, “A survey of fog computing: Fundamental, network applications, and research challenges,” *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 1826–1857, 2018.
5. M. Chiang, S. Ha, I. Chih-Lin, F. Risso, and T. Zhang, “Clarifying fog computing and networking: 10 questions and answers,” *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 18–20, Apr. 2017.

6. S. Misra and N. Saha, "Detour: Dynamic task offloading in edge computing using reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 669–682, Mar. 2019.
7. H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wireless Commun. Mobile Comput.*, vol. 13, no. 18, pp. 1587–1611, 2013.
8. Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 2017.
9. X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.
10. A. M. Rahmani *et al.*, "Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach," *Future Generation Comput. Syst.*, vol. 78, pp. 641–658, 2018.
11. A. H. Sodhro, Z. Luo, and G. H. Abbasi, "Energy-efficient adaptive transmission power control for wireless body area networks," *IEEE Sensors J.*, vol. 16, no. 23, pp. 8172–8179, 2016.
12. K. Kumar and Y. H. Lu, "Cloud computing for mobile users: Can offloading computation save energy?," *IEEE Computer*, vol. 43, no. 4, pp. 51–56, Apr. 2010.
13. R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog–cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.
14. X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
15. M. Amiri, G. Asheralieva, and D. Niyato, "Reinforcement learning for energy-efficient computation offloading in mobile edge computing," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10293–10313, Oct. 2020.
16. E. Osaba, X. S. Yang, F. Diaz, and A. Iglesias, "An improved discrete gravitational search algorithm for scheduling optimization," *Appl. Soft Comput.*, vol. 37, pp. 654–667, 2015.
17. I. Ahmadian, M. Feizollah, and H. Zarrabi, "Hybridization of gravitational search algorithm and particle swarm optimization for engineering optimization problems," *Eng. Appl. Artif. Intell.*, vol. 70, pp. 111–122, 2018.
18. A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Commun. Nonlinear Sci. Numer. Simulat.*, vol. 17, no. 12, pp. 4831–4845, 2012.
19. E. Cuevas *et al.*, "Emperor penguin optimizer: A bio-inspired algorithm for engineering problems," *Adv. Eng. Softw.*, vol. 114, pp. 48–70, 2017.
20. A. Sadkhan and M. Nickray, "Hybrid Emperor Penguin and Gravitational Search Optimization for Efficient Task Co-Offloading in Fog Computing Environments," *Int. J. Netw. Commun.*, vol. 12, no. 4, pp. 69–82, 2022.
21. S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.
22. F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.
23. P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 2017.
24. A. M. Alghamdi and M. Othman, "Computation offloading mechanism for mobile cloud computing," *Cluster Comput.*, vol. 20, pp. 2763–2776, 2017.
25. Y. Zhang, C. Wang, and Y. Chen, "Energy-efficient task offloading in mobile edge computing using metaheuristic algorithms," *J. Netw. Comput. Appl.*, vol. 176, pp. 102957, 2021.
26. N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.
27. J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, May 2019.
28. H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, and Z. Han, "Computing resource allocation in three-tier IoT fog networks: A joint optimization approach combining game theory and matching," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1206–1218, Oct. 2017.
29. J. Xu, L. Chen, and P. Zhou, "Joint resource allocation for data computing and transmission in mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7952–7966, Aug. 2019.
30. W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.
31. Z. Ning, P. Dong, J. J. Rodrigues, and X. Wang, "Deep reinforcement learning for vehicular edge computing: An intelligent offloading system," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 10875–10886, Nov. 2019.
32. L. Zhang, X. Zhang, and C. Wu, "A hybrid energy-efficient offloading algorithm for IoT–fog–cloud computing environments," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5547–5560, Apr. 2021.
33. X. Wang, Z. Ning, and S. Guo, "Multi-agent deep reinforcement learning for task offloading in mobile edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1607–1621, July 2021.
34. A. Sadkhan and M. Nickray, "Proximal Policy Optimization-based Task Co-Offloading Using Deep Residual Feed-Forward Network for Fog Computing Environments," *Int. J. Intelligent Eng. Syst.*, vol. 15, no. 6, pp. 264–278, 2022.
35. Z. Tan, W. Liu, H. Wang, and F. Yang, "Energy-delay tradeoff in mobile edge computing with multi-objective evolutionary optimization," *IEEE Access*, vol. 9, pp. 8726–8739, 2021.

Authors

Abbas S. B. Sadkhan received his B.Sc. degree in Computer Engineering Techniques from Al-Mustaqbal University, Iraq, in 2016, his M.Sc. degree in Information Technology Engineering from Imam Reza International University, Iran, in 2021, and is currently pursuing the Ph.D. degree in Information Technology Engineering at the University of Qom, Iran, since 2025. He also holds a Technical Diploma in Computer Engineering from the Institute of Electricity and Electronics, Iraq, in 2003. He is currently serving as the Chief Information Technology Engineer at the Iraqi Media Network. His research interests include distributed systems, mobile edge computing, task offloading, optimization techniques, artificial intelligence, and deep learning.

Mohsen Nickray received the B.Sc., M.Sc., and Ph.D. degrees in computer engineering from Iran University of Science and Technology and the University of Tehran in 2004, 2007, and 2012, respectively. He is currently an Assistant Professor with the Department of Computer Engineering and Information Technology, University of Qom, Iran. His recent research interests include resource management and task scheduling in cloud and fog computing.