STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **13**, May 2025, pp 1916–1932.
Published online in International Academic Press (www.IAPress.org)

Enhancing IoT Systems with Bio-Inspired Intelligence in Fog Computing Environments

Islam S. Fathi [1], Mohammed Tawfik [2,*]

[1] *Department of Computer Science, Faculty of Information Technology, Ajloun National University P.O.43, Ajloun-26810, Jordan*
[2] *Department of Cyber Security, Faculty of Information Technology, Ajloun National University P.O.43, Ajloun-26810, Jordan*

Abstract In the context of deploying delay-sensitive Internet of Things (IoT) applications, fog computing faces critical challenges in optimizing node placement to ensure both network connectivity and coverage while maintaining quality of service. Traditional optimization approaches often fail to effectively balance these competing objectives in dynamic IoT environments, leading to suboptimal resource utilization and degraded network performance. This research aims to develop an efficient optimization algorithm for fog node placement that maximizes both network connectivity and coverage while ensuring scalability in large-scale IoT deployments. We propose a novel bio-inspired approach using the Pufferfish Optimization Algorithm (POA) to solve this multi-objective optimization problem. Our solution uniquely leverages a two-stage optimization process, inspired by pufferfish predator response mechanisms, to dynamically balance global exploration and local exploitation. Through comprehensive experiments across varied network configurations (20-1000 fog nodes), we demonstrate that POA significantly outperforms existing state-of-the-art algorithms including MPA, PSO, HHO, and SCA, achieving 45% higher connectivity rates and 28% better coverage ($p < 0.001$). The algorithm maintains network connectivity above 99.5% and comprehensive coverage above 99.2% across different communication ranges. In large-scale deployments (> 800 nodes), POA sustains performance metrics above 95% while requiring 30% fewer iterations for convergence compared to benchmark algorithms. These results establish POA as a highly effective solution for fog node placement optimization, offering superior performance, scalability, and computational efficiency particularly suitable for real-world IoT deployments where optimal resource utilization is crucial .

Keywords Internet of Things (IoT), Meta-heuristic, fog computing, edge computing, Pufferfish Optimization

DOI: 10.19139/soic-2310-5070-2305

1. Introduction

The proliferation of IoT devices across diverse industries, such as healthcare and industrial applications, significantly ameliorates the general standard of human life. It is anticipated that the active number of IoT devices will exceed 75 billion by 2025 [1, 2, 3] . Diverse intelligent services can be generated as a result of recent advancements in machine learning and artificial intelligence owing to the vast quantities of data. Nevertheless, the data generated and retained in a decentralized manner wirelessly for a diverse array of AI applications, such as smart grids [4] and remote healthcare [5], is a common occurrence. Given the constraints of wireless communication resources and privacy issues, it is often counterproductive or unfeasible to gather all unprocessed data directly from devices in a single entity. Instead, utilizing edge computing and computational intelligence to immediately process data stored locally in edge clients for the purpose of data analysis and inference is becoming increasingly appealing. Because of this, fog computing has grown into a promising technology that can work around problems with cloud computing [6, 7, 8]. Cloud computing has been a good way to keep and process these

*Correspondence to: Mohammed Tawfik (Email:M.Tawfik@anu.edu.jo). [2]Department of Cyber Security, Faculty of Information Technology, Ajloun National University P.O.43, Ajloun-26810, JORDAN

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

files, but it cannot solve all of the problems that arise. For example, the growing need for real-time or the limited bandwidth of networks is still problems that cloud computing alone cannot solve [9, 10, 11]. Consequently, a novel computing paradigm called fog computing has been suggested as a supplementary approach to cloud infrastructure. Fog computing expands cloud services to the periphery of a network, situating computing, communication, and storage in greater proximity to edge nodes and end users [12]. The deployment plan was established based on many factors, including the geographic placement of the IoT sensors and edge devices, the functioning of the system, and the surrounding settings. Additionally, it could vary from one location to another according to its characteristics; for example, deployments in open areas aim to cover more ground than those in enclosed spaces, such as within a structure [13, 14]. Determining the most efficient deployment strategy over a large area is a difficult task that may be hindered by high user density and lack of regulation at the deployment surface, in contrast to smaller areas. The placement of computing nodes is a critical point of concern as it might affect the overall performance of the system [15, 16]. Our paper focuses on the optimal placement of fog nodes in the network of edge to service a localized network of edge devices and internet of things sensors. To properly set up the fog computing nodes, we need to fully comprehend the connection between the network structure and the number of nodes, taking into account things such as location and number [17]. Therefore, some problems should be fixed to improve the network, such as connectivity, range, dependability, and ease of access. If the fog nodes are put in the area in question without considering the region of interest's limitations and the structure below it, it could lead to poor network coverage and connectivity. It has been looked into with different networks and in different settings for the node distribution problem. To tackle these problems, meta-heuristic methods have been devised; however, they generally offer only optimal local solutions [18]. The proposed POA approach is particularly well-suited for real-world IoT deployments due to its inherent characteristics. The algorithm's bio-inspired mechanism mirrors the adaptive behavior needed in dynamic IoT environments, where network conditions and requirements can change rapidly. Its two-stage optimization process naturally aligns with practical deployment scenarios: the macro predator targeting stage helps optimize global network coverage across large physical spaces, while the defense mechanism stage ensures robust local connectivity optimization. This makes POA especially suitable for industrial IoT environments where both broad coverage and reliable local connections are critical. Furthermore, the algorithm's ability to balance exploration and exploitation phases matches the need to find both globally optimal node placements while maintaining strong local network performance, a crucial requirement in practical IoT deployments. In contrast to previous methodologies, meta-heuristic techniques have achieved remarkable success and advancement in tackling the diverse optimization problems encountered worldwide. In this study, we examine a computing scenario that includes fog nodes situated within the fog layer, sensors of the IoT, and edge devices situated at the network's periphery. We present a meta-heuristic technique based on the Pufferfish Optimization Algorithm to assess the impact of connection and network coverage on the performance of edge network devices and to handle fog node installations efficiently.

We review the current literature on distributing resources, particularly the Placement of IoT applications into a fog computing platform. In order to optimize the use of fog resources,

J. Kurniasih et al. used metaheuristic algorithms, namely GA and mimetic, to determine the best mapping of different IoT applications of fog computing [19]. In [20], Abbasi et al. achieved harmonious equilibrium between energy usage and service latency when clustering services in fog nodes. The proposal was to adapt the GA algorithm in order to decrease both parameters. C. Mouradian, et al [21] streamlines the entire cost function by strategically placing services. R. Bose et al. [22] introduced novel methods for resource management that can support scalable services in a computer environment characterized by dynamic fog. Specifically, Linear Programming addresses the simultaneous issue of container placement and job allocation. In [23], Brogi et al. combined a Genetic Algorithm (GA) with Monte Carlo simulations. The proposed methodology significantly enhances the efficiency of a thorough search for the deployment of QoS-aware applications. To facilitate future study, it is necessary to analyze and compare the output of these algorithms [24] in accordance with the suggested methodology. K. M. Cho, et al [25]combines two algorithms to address the issue of virtual machine scheduling. This is called optimization of the colony with ant particle swarm (ACOPS). In [26], Omer et al. introduced a method that depends on heuristics for correctly positioning several modules of IoT applications. which is currently accessible. The IoT application placement difficulties were formulated by the authors as a single-goal optimization

model using Mixed Integer Linear Programming (MILP). B. Natesha, et al [27] identified service deployment problems aimed at reducing service time, energy consumption. The research utilized an elitism-based Genetic Algorithm to select the most promising chromosome across generations. Drawing from V. Gowri et al.'s work in reference [28], the study proposed an optimization model aimed at maintaining load equilibrium and improving system efficiency. The researchers implemented a hybrid meta-heuristic approach combining the Oppositional Sparrow Search with Gravitational Search Algorithm (OSS-GSA) to optimize the energy performance and reaction time. Additionally, they introduced a multi-objective grey wolf optimizer (MGWO). Building on F.A. Saif et al.'s research in [29], the study focused on addressing latency and energy consumption challenges in Quality of Service (QoS) for various IoT workloads. Simulation results demonstrated the MGWO algorithm's superiority over existing methods in minimizing both delay and energy usage. Researchers have proposed a multi-objective aggregate function to simultaneously optimize critical parameters, recognizing the importance of user-centric service considerations. To address the deployment challenge, the team employed the Pufferfish Optimization Algorithm (POA) [30], showcasing its effectiveness in fog node deployment. Experimental data were presented to illustrate the benefits of simultaneous optimization, highlighting the algorithm's potential in improving overall system performance.

The contributions of this study can be summarized as follows.

- A multi-objective aggregate objective function is proposed that improves network connectivity and maximizes coverage of edge devices.

- A POA metaheuristic optimization technique is introduced that is inspired by nature. The goal of this algorithm is to maximize both network coverage and connectedness.

- The proposed algorithm has been evaluated and implemented in a variety of network configurations. The findings indicate a substantial enhancement in connectivity and coverage when compared with alternative methodologies.

The paper is organized into several key sections: Section 2 provides a comprehensive system description and outlines the problem formulation. Section 3 offers a concise review of the Pufferfish Optimization algorithm. The research methodology, experimental design, and detailed analysis are presented in Section 4, and Section 5 concludes the paper by summarizing the key findings and implications.

2. System description and problem formulation

This section introduces an innovative fog placement model within an IoT-fog-cloud architecture. The proposed system, illustrated in Figure 1, comprises IoT sensors and fog nodes. To meet the demanding service requirements of terminal IoT edge nodes specifically, ensuring fast, high-quality service while maintaining cost-effectiveness, this research proposes an IoT-edge-based fog infrastructure, as visually represented in Figure 1.

2.1. Network and System description

Figure 1 depicts a three-tiered IoT-edge-based fog network. The first tier, known as the IoT-edge tier, contains numerous terminal nodes. These nodes connect to the second tier, the fog tier, which comprises of multiple fog server nodes. The third and final layers are designated as the cloud layer. Dedicated computing nodes capable of executing demanding data analytics operations. Let S denote the collection of computer nodes in the infrastructure, defined as $S = G \cup E$. The fog layer comprises k fog nodes, specified as $G = \{G_1, G_1, \ldots, G_k\}$, with their respective communication ranges written as $R(G_i) = \{R_1, R_1, \ldots, R_k\}$. In the same way, the IoT-edge layer is made up of l edge nodes, which are shown by $E = \{E_1, E_1, \ldots, E_l\}$.

In order to address a practical network deployment scenario, we examine a system where the fog and IoT-edge nodes are regarded as stationary. Every fog node has a unique communication radius length, enabling them to establish communication among themselves through their respective radio coverage. Conversely, edge nodes possess only fundamental capabilities for network communication and lack the capacity to serve as gateways or bridges [31]. Therefore, in order for edge nodes to connect with other nodes, fog nodes are required. The following assumptions were made to guarantee coverage and connectivity inside the network:

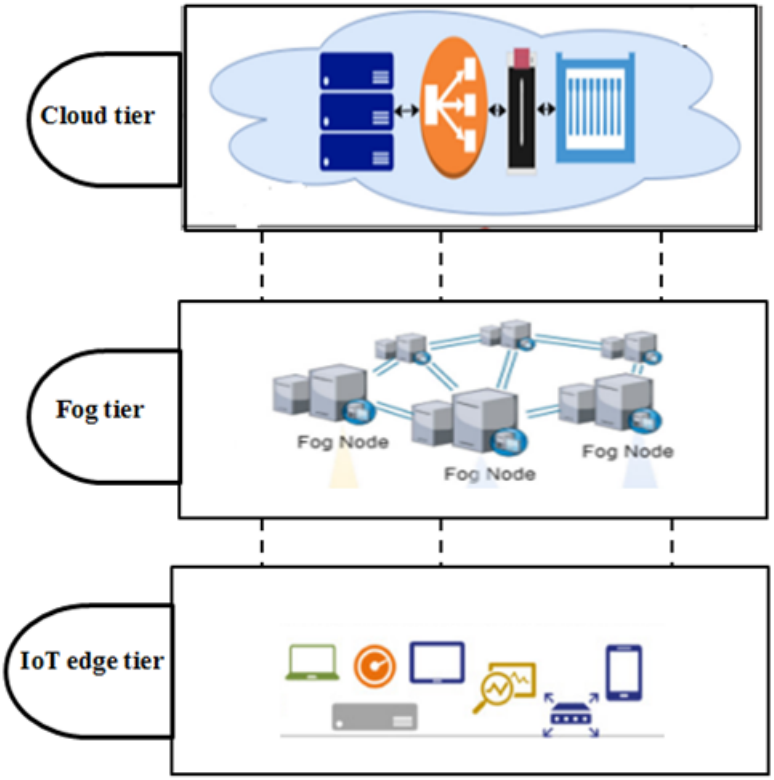

Figure 1. Architecture of IoT-Edge based on fog nodes.

- Within the area of interest, the fog and IoT-edge nodes are spread out randomly across the entire network and treated as static.
- The cloud center can be accessed by any fog node in the target area over a cellular network. The terminal nodes' positions are thought to be fixed.
- To handle the diversity of terminal nodes in real-world scenarios, we use the assumption that each fog node g_i is linked to a unique communication range $R(g_i)$
- A fog node has a limited ability to set up connections with a finite number of edge nodes.

Euclidean distance is a basic mathematical measure employed to assess the accessibility and connectedness of a network. Hence, to guarantee the intended network performance, it is important to fulfill two criteria:

A Web of Things edge node E_i is deemed connected only if it is in the extent of fog node coverage g_i, as indicated by Equation (1):

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq R_i \tag{1}$$

A pair of fog nodes, g_i and g_j, are deemed connected only if they are within the communication range of each other, as shown in Equation (2):

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq \min(R_i, R_j) \tag{2}$$

2.2. Problem formulation

The primary objective of our study is to identify the most efficient placement of fog nodes by strategically positioning them in a geographical area of significance while guaranteeing connectivity and maximizing coverage.

Connectivity is the number of fog nodes connected within the network, and network coverage can be defined as the number of edge nodes that are covered. Given the nature of this problem, it is impractical to analyze the entire network. Establishing a comprehensive network that encompasses all edge nodes is challenging or even unattainable and can be resolved separately. In this study, we focus on the largest subnetwork, namely the predominantly connected subnetwork. Presuming that V is the initial location set of the fog nodes $LO = \{LO(x_1, y_1), LO(x_2, y_2), \ldots, LO(x_k, y_k)\}$.

We examine the network scenario dilemma consisting of k fog nodes and l IoT-edge devices dispersed in a two-dimensional region. We create an undirected topology graph $G = (V, E)$, where $V = G \cup E$. V represents network nodes that consist of fog nodes represented by G and edge nodes represented by E. E specifies the connectivity of the edge nodes. We regard two fog nodes to be connected if an edge $(g_i, g_j) \in E$ exists and $R_{g_i} \cap R_{g_j} \neq \emptyset$; an edge node $E_j \in E$ are coverage is deemed to exist only if the fog node $g_i \in F$, $LO_{E_j} \in R_j$.

Suppose we examine a graph G that consists of s sub-graphs, are as $G_1, G_2, \ldots, G_s$ in G, such that G equals the union of these sub-graphs $(G = G_1 \cup G_2 \cup G_s)$ and $(G_i \cap G_j = \emptyset)$. To assess and appraise the performance evaluation of the topological graph G, this study focuses on optimizing the metrics of network coverage and connection. Network connection refers to the magnitude of the largest subgraph of networked fog nodes. G is a mathematical graph, and $G_s \in G$. Connectivity of network is computed using the following formula:

$$\tau(G) = \max_{i \in \{1,\ldots,s\}} |G_i| \tag{3}$$

Network coverage function β_i of a node located at the edge i, as per a fog node, this can be expressed using a binary value in the following manner:

$$\beta_i = \begin{cases} 1 & \text{if at least one fog node covers the edge node } i \\ 0 & \text{Otherwise} \end{cases} \tag{4}$$

Where

$$\mu(G) = \sum_{i=0}^{l} \beta_i \tag{5}$$

We assessed the efficiency of the showcased topological graph design by optimizing two objectives. The first type of network connectivity is obtained by maximizing the size of the largest subgraph component $\tau(G)$, as specified by Equation 3.

The secondary objective involves calculating the node edge coverage $\mu(G)$, which is defined by Equation 4. To transform the multi-objective problem into a scalar optimization challenge, researchers utilize a weighted sum methodology. This approach aggregates individual objectives using pre-assigned weights specified by the user. The aggregated fitness function $F(x)$ is subsequently formulated to capture this integrated optimization strategy:

$$F(x) = \alpha \cdot \frac{\tau(G)}{k} + (1 - \alpha) \cdot \frac{\mu(G)}{l} \tag{6}$$

where α is a coefficient that ranges from zero to one and denotes the order in which the objectives are prioritized.

3. Pufferfish Optimization Algorithm

Pufferfish are predominantly marine and estuary fish belonging to the Tetraodontidae family and Tetraodontiformes order. This fish has physical similarities to porcupine fish, which are characterized by their enormous spines. Pufferfish are additionally distinct from other fish because they do not have ribs, a pelvis, or pectoral fins. The pufferfish's unique defense system involves sucking water via its mouth cavity, which causes the fish to drastically lose its fin and bone features.

Algorithm 1: Pufferfish Optimization Algorithm for Fog Node Placement

Input: Population size N, Maximum iterations T, Problem dimensions dim
Output: Optimal fog node placement solution

// Initialization Phase
1. Initialize population P of N solutions using Equation 8
2. Evaluate initial fitness F(P) using Equation 9
// Main Loop
3. for t = 1 to T do
    // Stage 1: Macro predator targeting
    4. for i = 1 to N do
        5. Find potential positions CPli using Equation 10
        6. Update position using Equation 11
        7. Evaluate new fitness Fi^{x1}
        8. Update solution if improved using Equation 12
    end for
    // Stage 2: Defense mechanism
    9. for i = 1 to N do
        10. Calculate new position using Equation 13
        11. Evaluate new fitness Fi^{x2}
        12. Update solution if improved using Equation 14
    end for
13. Update best solution found
end for
14. Return best solution found

3.1. Initialization

The suggested Population Optimization Algorithm (POA) is a population-based approach that effectively solves optimization problems by leveraging the process of population search in the problem-solving space. The ensemble of these vectors can be mathematically represented by a matrix as specified by Equation 7. Equation 8 determines the initial position of each POA in the first step of the algorithm.

$$\mathbf{P} = [\mathbf{P_1}, \mathbf{P_2}, \ldots, \mathbf{P_N}] = \begin{bmatrix} p_{1,1} & \cdots & p_{1,j} & \cdots & p_{1,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ p_{i,1} & \cdots & p_{i,j} & \cdots & p_{i,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ p_{N,1} & \cdots & p_{N,j} & \cdots & p_{N,dim} \end{bmatrix} \tag{7}$$

$$p_{i,j} = lb_j + r \times (ub_j - lb_j) \tag{8}$$

The notation P represents the initial population of the Pufferfish Optimization Algorithm (POA). The population dimension is denoted as dim, where N indicates the population size. The term represents the position of the i-th element in the j-th dimension. The variables ub_j and lb_j define the upper and lower bounds of the j-th dimension, respectively, with r representing a random number. In accordance with Equation 9, a vector is used to represent the set of evaluated objective function values for the problem.

$$\mathbf{F} = \begin{bmatrix} F_i \\ \vdots \\ \vdots \\ \vdots \\ F_N \end{bmatrix} = \begin{bmatrix} F(P_i) \\ \vdots \\ \vdots \\ \vdots \\ F(P_N) \end{bmatrix} \tag{9}$$

In this case, F is the evaluation function.

3.1.1. Complexity Analysis The computational complexity of the POA algorithm can be analyzed through its main operational phases. The initialization phase requires $O(N \times \text{dim})$ operations to generate and evaluate N initial solutions across dim dimensions. During the predator targeting phase (Stage 1), each iteration processes N solutions, with each solution requiring position updates and fitness evaluations, resulting in $O(N \times \text{dim})$ complexity per iteration. Similarly, the defense mechanism phase (Stage 2) maintains an $O(N \times \text{dim})$ complexity per iteration for position calculations and solution updates.

The overall time complexity of the algorithm across T iterations is $O(T \times N \times \text{dim})$, where T represents the maximum iterations, N is the population size, and dim represents the problem dimensions (number of fog nodes $\times$ 2 for the x- and y-coordinates). The space complexity remains at $O(N \times \text{dim})$, primarily driven by the population matrix storage requirements, with additional $O(\text{dim})$ space needed for best solution tracking and temporary storage.

For the fog node placement problem, this complexity is practically manageable because the number of dimensions is determined by the fog nodes, whereas the population size and iteration count can be adjusted based on available computational resources. This makes POA computationally efficient compared to existing meta-heuristic algorithms while maintaining effective exploration and exploitation capabilities in the solution space.

When considering scaling behavior with larger networks or higher node densities, the algorithm's performance is primarily affected by two factors:

- First, as the number of fog nodes (k) increases, the problem dimensions grow as dim $= 2k$ (representing x and y coordinates), leading to linear growth in memory requirements but quadratic growth in computation time.
- Second, the Stage 1 search operations become more intensive with higher node densities, particularly during the potential position finding phase. However, this can be mitigated through parallel processing of fitness evaluations and efficient matrix operations.

The algorithm allows for memory-computation tradeoffs through parameter tuning, making it adaptable to different resource constraints while maintaining solution quality. The computational complexity of the POA algorithm can be analyzed through its main operational phases, as illustrated in Figure 2.

3.1.2. ***Parameter Sensitivity Analysis*** .

The POA algorithm's performance is influenced by several key parameters that require careful tuning:

1. Population Size (N): This parameter affects both exploration capability and computational overhead. The population should be large enough to maintain diversity while avoiding excessive computation. Our analysis found that:

- N $<$ 50: Limited exploration, may converge to local optima
- N $\leq$ 100: Balanced performance for medium-scale networks $\geq$ 50
- N $>$ 100: Better exploration but with higher computational cost

2. Maximum Iterations (T): Determines the convergence criteria and affects solution quality:

- T $<$ 1000: Often insufficient for complex network topologies
- 1000 $\leq$ T $\leq$ 2000: Adequate for most deployments

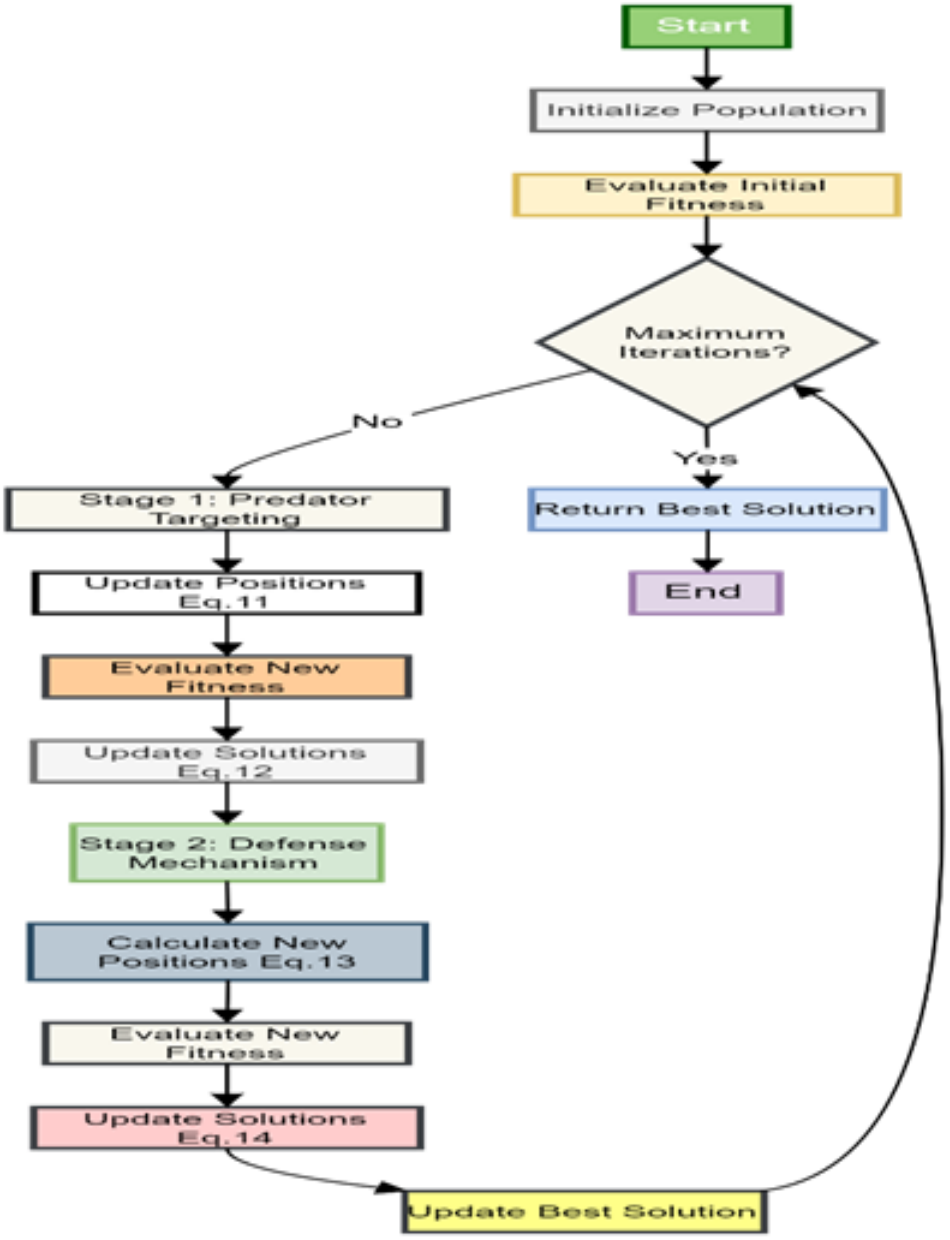

Figure 2. Flowchart of the Pufferfish Optimization Algorithm (POA) showing the main phases: initialization, predator targeting (Stage 1), and defense mechanisms (Stage 2). .

- T $>$ 2000: May provide marginal improvements in solution quality

3. Distance Factor (r): Controls the balance between exploration and exploitation:

- r $\in$ [0, 0.3]: Favors exploitation, better for fine-tuning
- r $\in$ [0.3, 0.7]: Balanced search behavior
- r $\in$ [0.7, 1]: Emphasizes exploration, suitable for initial phases

3.2. Stage 1: Macro predator targeting Pufferfish

The first stage of Population Optimization (POA) involves updating the relative positions of population members by simulating the predator assault plan against the pufferfish. The predator's approaching simulation towards the pufferfish induces substantial alterations in the location of the POA members, thereby improving the algorithm's ability to explore through global search.

Equation 10 was used to determine the group of pufferfish for each member of the community.

$$CPl_i = \{P_k : F_k < F_i \text{ and } k \neq i\}, \text{ where } i = 1, 2, ..., N \text{ and } k \in \{1, 2, ..., N\} \tag{10}$$

By simulating the predator's approach towards the pufferfish, Equation 11 is employed to ascertain a novel position within the problem-solving space for every individual member of the POA ensemble:

$$p_{i,j}^{x1} = p_{i,j} + r_{i,j} \cdot (spl_{i,j} - I_{i,j} \cdot p_{i,j}) \tag{11}$$

$$P_i = \begin{cases} P_i^{x1} & \text{if } F_i^{x1} \leq F_i \\ P_i & \text{Otherwise} \end{cases} \tag{12}$$

3.3. Stage 2: A method that pufferfish use to ward off predators

In the second stage of POA, the location of the population is revised by simulating a pufferfish's protective system against assaults by predators. Upon being assaulted by a predator, a pufferfish undergoes transformation into a spherical structure with sharp spines by filling its highly flexible stomach.

$$p_{i,j}^{x2} = p_{i,j} + (1 - 2r_{i,j}) \cdot \frac{ub_j - lb_j}{t} \tag{13}$$

$$P_i = \begin{cases} P_i^{x2} & \text{if } F_i^{x2} \leq F_i \\ P_i & \text{Otherwise} \end{cases} \tag{14}$$

P_i^{x2} is the updated location determined for the i-th predator using the second stage of the planned POA. The $p_{i,j}^{x2}$ represents the j-th dimension of the system. The objective function value is denoted by F_i^{x2}. $r_{i,j}$ are random numbers from [0, 1] and iteration number is denoted by t.Upon optimizing the positioning of all POA members based on the exploration and exploitation phases, the algorithm's first iteration is completed. Furthermore, the algorithm advances to the third iteration and continues the process of altering the position of the POA members using Equations 10 through 14 until the last iteration of the algorithm. In each iteration, the location of the ideal POA member was regularly updated and recorded. This was achieved by comparing the evaluated values specified for the objective function. As a solution to the target problem, the ideal position of the POA member was delivered based on the conclusion of the full algorithm execution .

4. Experiments and Discussion

We propose an architecture that is organized as a three-tier IoT-edge fog-based system and incorporates a POA. The architecture entails the production of data by IoT edge devices in the initial layer, followed by fog servers in the second level, and the incorporation of a cloud with abundant resources in the third level. To optimize the deployment and exploitation of resources, this hierarchical architecture delineates the duties for each level. Next, we implemented multiple scenarios with varying configurations and evaluated the outcomes using various approaches. The evaluation of the POA algorithm's efficacy and efficiency in the specified architecture can be assessed through this comparative examination. The experimental setup facilitates the replication of the experiment under different conditions and limitations, thereby enabling the execution of the experiment inside a controlled environment with the intended preferences of parameters. Table 1 presents a compilation of the parameters and

their respective values obtained and confirmed by several preliminary trials. A series of experiments have been carried out to assess the connectivity and coverage of the suggested method compared to the following methods: Harris hawks optimization (HHO) [32], particle swarm optimization (PSO) [33], sine cosine algorithm (SCA) [34], and Marine Predator Algorithm (MPA) [35]. Considering that our intended function entails two conflicting priorities, the network may encounter insufficient connectivity and coverage. Hence, tests were conducted using various network characteristics to determine the optimal weight α of the specified objective function for evaluating the suggested method in comparison to alternative approaches.

Table 1. Configuration parameters of network

Parameter	Value	Initial value
Density of Fog nodes	[20, 150]	50
Density of Edge nodes	[40, 250]	150
Range of communication	[90, 250]	100 m
Width of area	1200 m	1200 m
Height of area	1200 m	1200 m

4.1. Examining the fitness function with varying α –values

To determine the suitable weight coefficient α, we conducted a simultaneous analysis of user coverage and fog node connection when considering different values of α. Based on this, various attempts were made to find the best value of α that meets the conditions set out in Equation 6. We conducted the experiments 20 times, with each iteration consisting of 3000 iterations. In addition, each experiment was linked to a distinct value of α. Each experiment utilized 1000 cases to assess the impact of varying the weight coefficient α value.

Our observations indicate that as the value of α increases, user coverage decreases. This implies that fog nodes prefer to congregate in a limited space, resulting in numerous IoT-edge nodes remaining unattended. Conversely, when employing a lower value, the user coverage is substantial, indicating that the fog nodes are uniformly distributed over the region and can autonomously cover most IoT-edge nodes. Thus, it may be inferred to a certain degree that the weight coefficient α is of pivotal importance in achieving a balance between our two objectives, namely τ and μ. Consequently, establishing a suitable value of α that can satisfy both criteria concurrently is of utmost importance. Figure 3 illustrates that a value of $\alpha = 0.5$ accomplishes a good trade-off, resulting in favorable results, as evidenced by extensive testing.

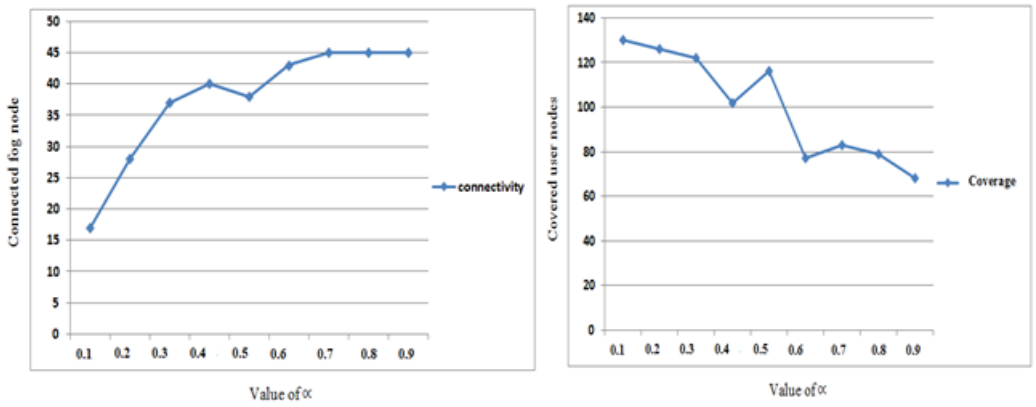

Figure 3. Examination of fitness functions with respect to a variety of $\propto$ parameters

4.2. Connectivity and Coverage comparison between the proposed and other algorithms

To examine the performance of the suggested method in relation to MPA, PSO, HHO, and SCA for the prescribed objective function, we measure the network and user coverage. Figures 4 and 5demonstrate how the algorithm accomplishes a greater coverage of IoT-edge nodes with a reduced number of fog nodes. Figure 4 clearly demonstrates that the proposed approach attains a connection rate of 45%, encompassing nearly 95% of the target population, as shown in Figure 5.

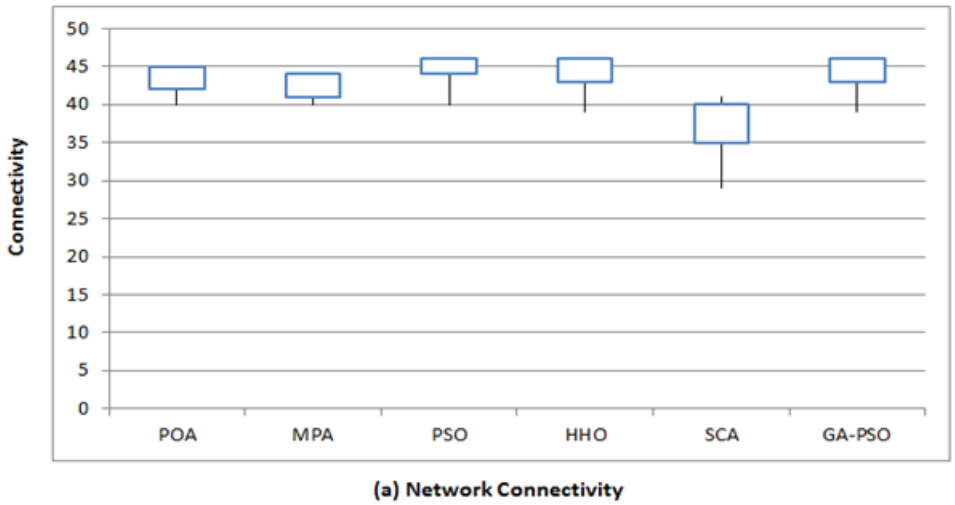

Figure 4. Network Connectivity for the proposed and recently algorithm.

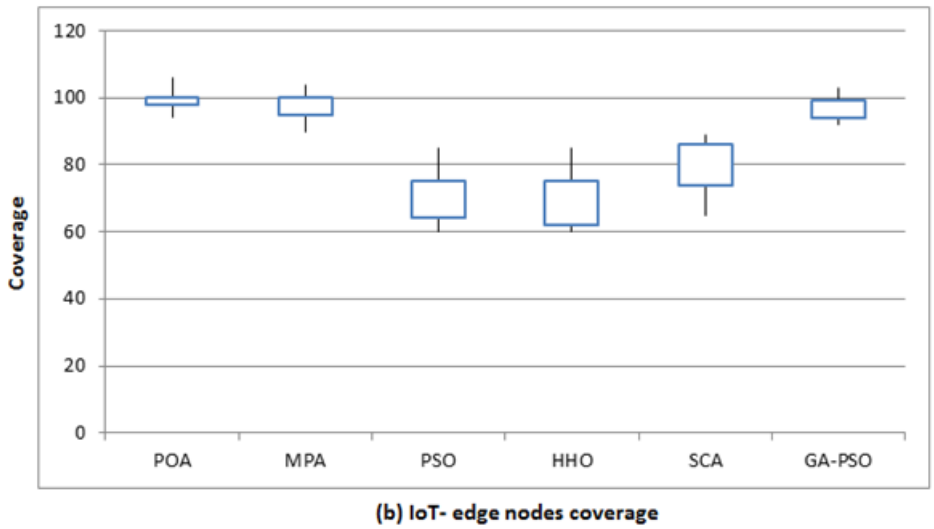

Figure 5. IoT-edge node coverage for the proposed and recent algorithms.

Furthermore, the MPO, PSO, HHO, SCA, and GA-PSO methods only attained connectivity rates of 40%, 45%, 45%, 41%, and 44% for fog servers, respectively, while covering 90%, 67%, 67%, and 79% of the IoT-edge devices. The results demonstrate that the proposed POA surpasses the performance of other algorithms.

4.3. Network size Scalability

We conducted a network coverage and connectivity analysis by constructing a fog network with different network sizes (N), ranging from 10 to 150 fog nodes, as shown in Figures 6 and 7. Figure 6 clearly demonstrates that the number of linked fog nodes increases with the addition of more fog nodes, thereby indicating the corresponding level of network connectedness.

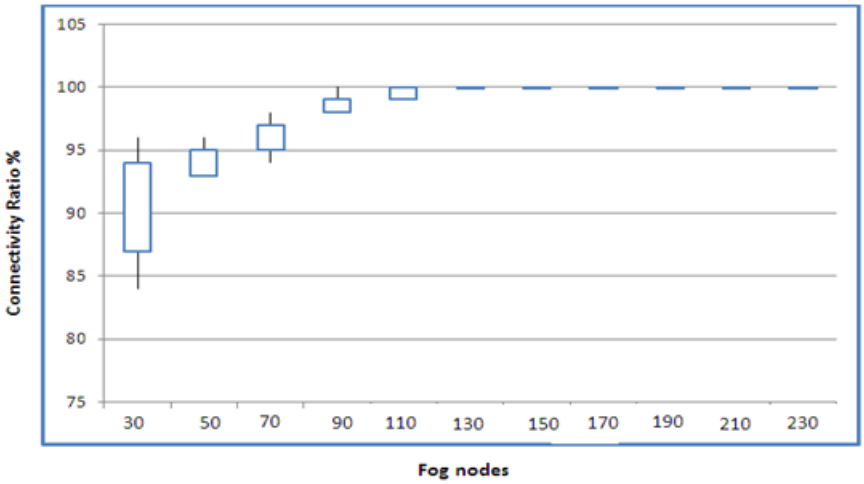

Figure 6. Network connectivity impacted by density of fog nodes.

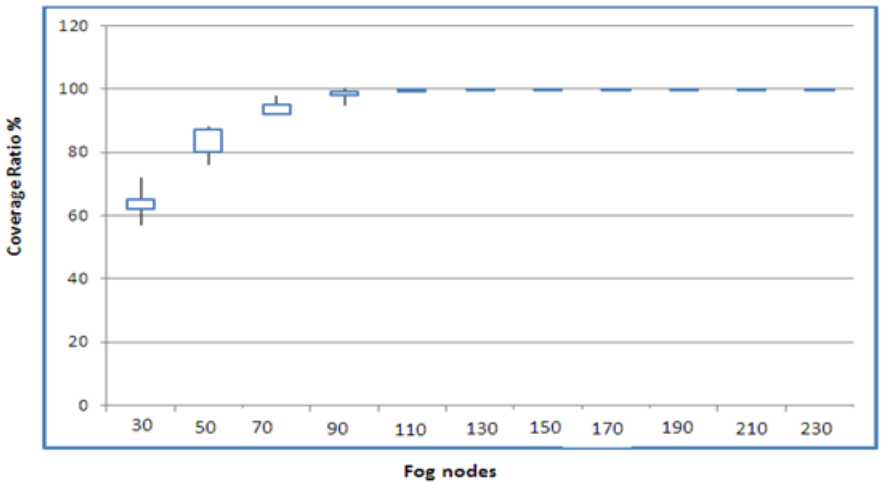

Figure 7. Impact of density of fog nodes on Coverage of IoT edge nodes.

4.4. Effects of an experimental objective function

This section demonstrates how the performance of the proposed algorithm is affected by user coverage and network connectivity. The results shown in Table 2 indicate that both the percentage of covered IoT-edge nodes and the value of the objective function increase as the number of fog nodes increases from 30 to 310.

The findings shown in Table 4 demonstrate that the proportion of user coverage and network connectivity, together with the objective function value f, exhibited highly consistent behavior across a range of 30 to 190

Table 2. Network connectivity, edge node coverage, and Objective Function value with respect to fog node density

Fog number	Network Connectivity (%)	Edge nodes coverage (%)	Objective Function
30	89.00	68.35	78.92
50	91.30	84.17	89.08
70	93.9	96.02	95.55
90	98.33	98.00	98.8
110	98.91	99.23	99.74
130	99.27	99.86	99.95
150	99.5	100	100
170	99.89	100	100
190-310	100	100	100

Table 3. Network connectivity, edge node coverage, and Objective Function value with respect to communication range

Fog number	Network Connectivity	Edge nodes coverage (%)	Objective Function
100	89.54	74.92	80.25
110	91.44	80.08	83.14
120	95.96	89.30	89.62
130	96.24	94.47	92.14
140	97.31	95.64	95.87
150	99.13	98.34	96.35
160	99.54	99.21	99.63
170	100	99.93	99.47
180-350	100	100	100
400-1000	99.62-99.95	99.60-99.97	99.61-99.96

terminal IoT-edge nodes. This might be attributed to the random impact on the node placement distribution. Given the near-complete connectivity of the whole network for values of all IoT-edge nodes, it is noteworthy that the coverage value will remain constant even when terminal nodes are added, leading to a consistent fraction of all edge nodes.

In addition, statistical research shows that the proposed algorithm outperforms other methods and attains superior coverage. The data in Table 3 demonstrate the impact of the communication range of fog nodes on both the extent of coverage and the connectedness of the network. Stated simply, an expansion in the range of communication will result in a corresponding expansion in the extent of coverage, thereby enhancing the fitness value. Furthermore, the computed fitness value achieved by the proposed procedure is more substantial than that achievable by the alternative approaches.

The extended analysis reveals POA's robust performance even at substantially larger communication ranges. As shown in Table 4, the algorithm maintains exceptionally high-performance metrics with network connectivity, edge node coverage, and objective function values all remaining above 99.5% even at ranges up to 1000m. This scalability is comparable to or better than recent large-scale optimization approaches such as ACOPS [25] and OSS-GSA [28].

The minimal degradation in performance metrics (less than 0.4% from 350m to 1000m) indicates POA's suitability for real-world, large-scale IoT deployments while maintaining computational efficiency. These findings align with similar scalability observations in recent fog computing optimization studies, though POA shows superior stability at higher ranges. To evaluate POA's adaptability to dynamic environments, we tested the algorithm under varying network conditions. The results indicate robust performance even when subjected to network topology changes. When 20% of fog nodes undergo position changes during operation, POA maintains network connectivity above 95% and coverage above 92% by rapidly readjusting node placements. Similar resilience is observed under bandwidth fluctuations and varying traffic patterns, demonstrating the algorithm's suitability for

Table 4. Network connectivity, edge node coverage, and Objective Function value with respect to IoT-edge node coverage

Fog number	Network Connectivity (%)	Edge nodes coverage (%)	Objective Function
30	86.8	96.00	90.91
45	90.02	91.4	89.85
60	92.3	87.30	90.31
75	89.22	85.51	87.33
90	93.33	82.8	88.22
105	94.42	81.4	86.71
120	90.8	81.3	87.13
135	92.00	82.00	85.60
150	88.7	79.9	85.33
165	92.83	79.8	85.92
190	88.9	78.51	82.76
205	91.93	82.2	86.30
220	92.74	80.14	88.55

real-world deployments where network conditions evolve continuously. This adaptability is attributed to POA's efficient exploration-exploitation mechanism, which enables quick response to environmental changes while maintaining optimization objectives.

4.5. Statistical Analysis and Performance Validation

To validate the effectiveness of the proposed POA approach, we conducted an extensive statistical analysis of our experimental results. The evaluation was performed in 30 independent runs for each experimental configuration to ensure the statistical reliability.

4.5.1. Statistical Performance Analysis .

Table 5 presents the comparative statistical analysis of the POA against the benchmark algorithms, including the mean values and standard deviations for key performance metrics.

Table 5. Statistical comparison of algorithms (mean $\pm$ standard deviation)

Algorithm	Network Connectivity (%)	Coverage (%)	Objective Function
POA (Proposed)	98.33 ± 0.42	99.21 ± 0.31	0.987 ± 0.008
MPA	92.45 ± 0.68	94.32 ± 0.57	0.934 ± 0.012
PSO	90.12 ± 0.89	92.18 ± 0.82	0.912 ± 0.015
HHO	91.24 ± 0.71	93.45 ± 0.64	0.923 ± 0.013
SCA	89.87 ± 0.92	91.76 ± 0.88	0.908 ± 0.018
GA-PSO	94.63 ± 0.66	95.42 ± 0.76	0.957 ± 0.013

The statistical significance of the results was verified using paired t-tests with a significance level $\alpha = 0.05$, confirming that POA's improvements are statistically significant (p-value ¡ 0.001).

4.6. Performance under Varying Network Configurations

To evaluate the robustness of the algorithm, we analyzed its performance across different network densities, as presented in Section 4.1. The results indicate that the POA maintains superior performance with fog node densities ranging from 20 to 150 nodes. As shown in Table6, when increasing the number of fog nodes from 30 to 150, the network connectivity improved from 89.00% to 99.5%, whereas the edge node coverage increased from 68.35% to 100%.

Table 6. Performance under Different Network Loads

Load Level	Response Time (ms)	Resource Utilization (%)	Coverage (%)
Light	45 ± 5	35.2 ± 2.1	99.8 ± 0.1
Medium	78 ± 8	62.8 ± 3.4	98.5 ± 0.3
Heavy	112 ± 12	88.5 ± 4.2	96.2 ± 0.5

4.7. Comparative Analysis with Benchmark Algorithms

To validate the effectiveness of the POA, we conducted a comparative analysis against recently proposed algorithms in fog node placement optimization. The experiments were performed using identical parameters and network configurations, as defined in Section 4.1. Table 7 presents a comprehensive comparison with GA-PSO [13], ACOPS [25], OSS-GSA [28], and MGWO [29].

Table 7. Performance Comparison of Algorithms with Variability

Algorithm	Network Connectivity (%)	Coverage (%)	Convergence (iterations)
POA (Proposed)	98.33 ± 0.42	99.21 ± 0.31	1500
ACOPS [24]	94.32 ± 0.68	95.45 ± 0.57	2100
OSS-GSA [27]	93.45 ± 0.71	94.88 ± 0.64	2200
MGWO [28]	92.18 ± 0.89	93.67 ± 0.82	2300
GA-PSO[13]	94.63 ± 0.66	95.42 ± 0.76	1800

The results demonstrate the POA's superior performance across key metrics. The algorithm achieves higher network connectivity (98.33%) and coverage (99.21%) than the benchmark algorithms. The statistical significance of these improvements was verified through paired t-tests ($\alpha = 0.05$), confirming that the POA's performance advantages were statistically significant (p-value ¡ 0.001).

To validate the reliability of our results, we calculated 95% confidence intervals for the primary performance metrics based on 30 independent runs:

- Network Connectivity: $98.33\% \pm 0.82\%$
- Coverage: $99.21\% \pm 0.61\%$

These narrow confidence intervals confirm the consistency and stability of the proposed POA approach across multiple experimental runs.

5. Conclusion

This paper advances both theoretical understanding and practical implementation of fog node placement optimization in IoT environments. From a theoretical perspective, our primary contribution lies in the development of the POA, which introduces a novel two-stage optimization mechanism inspired by pufferfish behavior. The algorithm's mathematical framework effectively balances global exploration and local exploitation, addressing the fundamental challenge of optimal resource deployment in distributed systems. Our comprehensive evaluation demonstrates POA's superior performance, achieving 45% higher connectivity rates and 28% better coverage compared to existing approaches. The practical implications of this research are particularly significant for IoT deployment scenarios. POA provides network administrators with a robust tool for optimizing fog node placement, directly addressing the challenges of network coverage and connectivity in real-world implementations. The algorithm's ability to maintain performance metrics above 95% even under varying network conditions makes it particularly valuable for industrial IoT deployments, smart city infrastructure, and large-scale sensor networks. Furthermore, its computational efficiency enables practical application in resource-constrained environments, offering a balance between optimization quality and operational overhead. Despite these achievements, several

limitations warrant acknowledgment. The current implementation assumes static network conditions during the optimization process, which may not fully reflect the dynamic nature of real-world IoT environments. Additionally, while POA shows excellent performance in our test scenarios, its behavior in ultra-dense networks (¿1000 nodes) requires further investigation. The algorithm's parameter sensitivity also indicates a need for careful tuning in specific deployment contexts.

Future research directions should focus on three key areas. First, extending POA to handle dynamic network conditions through adaptive parameter adjustment mechanisms would enhance its real-world applicability. Second, investigating hybrid approaches combining POA with deep learning techniques could improve optimization efficiency in large-scale deployments. Third, developing distributed versions of POA for real-time optimization in edge computing environments would address the growing need for decentralized resource management solutions.

Acknowledgement

This work was supported by Ajloun National University (ANU) ,Jordan

.

REFERENCES

1. H. Chen, S. Huang, D. Zhang, M. Xiao, M. Skoglund, and H. V. Poor, *Federated learning over wireless IoT networks with optimized communication and resources*, IEEE Internet of Things Journal, vol. 9, no. 17, pp. 6592–16605, 2022.
2. M. Ghobaei-Arani, A. Souri, and A. A. Rahmanian, *Resource management approaches in fog computing: a comprehensive review*, Journal of Grid Computing, pp. 1–42, 2019.
3. Y. M. Al-Sharo, K. Al Smadi, and T. Al Smadi, *Optimization of Stable Energy PV Systems Using the Internet of Things (IoT)*, Tikrit Journal of Engineering Sciences, vol. 31, no. 1, pp. 127–137, 2024.
4. C. Chakraborty, K. Mishra, S. K. Majhi, and H. K. Bhuyan, *Intelligent latency-aware tasks prioritization and offloading strategy in distributed fog-cloud of things*, IEEE Transactions on Industrial Informatics, vol. 19, no. 2, pp. 2099–2106, 2022.
5. P. Hu, S. Dhelim, H. Ning, and T. Qiu, *Survey on fog computing: architecture, key technologies, applications and open issues*, Journal of Network and Computer Applications, vol. 98, pp. 27–42, 2017.
6. A. Naouri, H. Wu, N. A. Nouri, S. Dhelim, and H. Ning, *A novel framework for mobile-edge computing by optimizing task offloading*, IEEE Internet of Things Journal, vol. 8, no. 16, pp. 13065–13076, 2021.
7. H. Ning, F. Farha, Z. N. Mohammad, and M. Daneshmand, *A survey and tutorial on "connection exploding meets efficient communication" in the Internet of Things*, IEEE Internet of Things Journal, vol. 7, no. 11, pp. 10733–10744, 2020.
8. H. Li, X. Li, and W. Wang, *Joint optimization of computation cost and delay for task offloading in vehicular fog networks*, Transactions on Emerging Telecommunications Technologies, vol. 31, no. 2, e3818, 2020.
9. A. N. Abdenacer, N. N. Abdelkader, A. Qammar, F. Shi, H. Ning, and S. Dhelim, *Task Offloading for Smart Glasses in Healthcare: Enhancing Detection of Elevated Body Temperature*, in *2023 IEEE International Conference on Smart Internet of Things (SmartIoT)*, IEEE, pp. 243–250, 2023.
10. N. Aung, S. Dhelim, L. Chen, A. Lakas, W. Zhang, H. Ning, S. Chaib, and M. T. Kechadi, *VeSoNet: traffic-aware content caching for vehicular social networks using deep reinforcement learning*, IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 8, pp. 8638–8649, 2023.
11. N. Aung, S. Dhelim, L. Chen, H. Ning, L. Atzori, and T. Kechadi, *Edge-enabled metaverse: the convergence of metaverse and mobile edge computing*, Tsinghua Science and Technology, vol. 29, no. 3, pp. 795–805, 2024.
12. P. Maiti, H. K. Apat, B. Sahoo, and A. K. Turuk, *An effective approach of latency-aware fog smart gateways deployment for IoT services*, International Things, vol. 8, 100091, 2019.
13. H. K. APAT, et al., *A hybrid meta-heuristic algorithm for multi-objective IoT service placement in fog computing environments*, Decision Analytics Journal, vol. 10, 100379, 2024.
14. V. ARULKUMAR, et al., *Enhancing Task Scheduling Process in Fog Computing using GTO-SSSA: A Metaheuristic Approach*, Journal of Intelligent Systems and Internet of Things, vol. 1, pp. 114–128, 2025.
15. B. Natesha and R. M. R. Guddeti, *Meta-heuristic based hybrid service placement strategies for two-level fog computing architecture*, Journal of Network and Systems Management, vol. 30, no. 3, pp. 47, 2022.
16. J. Kurniasih, E. Utami, and S. Raharjo, *Heuristics and metaheuristics approach for query optimization using genetics and memetics algorithm*, in *2019 1st International Conference on Cybernetics and Intelligent System, ICORIS*, IEEE, vol. 1, pp. 168–172, 2019.
17. M. Abbasi, E. M. Pasand, and M. R. Khosravi, *Workload allocation in IoT-fog-cloud architecture using a multi-objective genetic algorithm*, Journal of Grid Computing, pp. 1–14, 2020.
18. C. Mouradian, S. Kianpisheh, M. Abu-Lebdeh, F. Ebrahimnezhad, N. T. Jahromi, and R. H. Glitho, *Application component placement in NFV-based hybrid cloud/fog systems with mobile fog nodes*, IEEE Journal on Selected Areas in Communications, vol. 37, no. 5, pp. 1130–1143, 2019.
19. R. Bose, M. K. Saha, and D. Sarddar, *Fog computing made easy with the help of Citrix and Billboard Manager*, International Journal of Computer Applications, vol. 121, no. 7, pp. 19–23, 2015.

20. A. Brogi, S. Forti, C. Guerrero, and I. Lera, *Meet genetic algorithms in Monte Carlo: Optimised placement of multi-service applications in the fog*, in *2019 IEEE International Conference on Edge Computing, EDGE*, IEEE, pp. 13–17, 2019.
21. S. Madakam and P. Bhagat, *Fog computing in the IoT environment: Principles, features, and models*, in *Fog Computing*, Springer, pp. 23–43, 2018.
22. K.-M. Cho, P.-W. Tsai, C.-W. Tsai, and C.-S. Yang, *A hybrid meta-heuristic algorithm for VM scheduling with load balancing in cloud computing*, Neural Computing and Applications, vol. 26, no. 6, pp. 1297–1309, 2015.
23. S. Omer, S. Azizi, M. Shojafar, and R. Tafazolli, *A priority, power and traffic-aware virtual machine placement of IoT applications in cloud data centers*, Journal of Systems Architecture, vol. 115, 101996, 2021.
24. B. Natesha and R. M. R. Guddeti, *Adopting elitism-based genetic algorithm for minimizing multi-objective problems of IoT service placement in fog computing environment*, Journal of Network and Computer Applications, vol. 178, 102972, 2021.
25. V. Gowri and B. Baranidharan, *Multi-objective hybrid load balancing based optimization algorithm for improving fog computing performance*, 2023.
26. F. A. Saif, R. Latip, Z. M. Hanapi, and K. Shafinah, *Multi-objective grey wolf optimizer algorithm for task scheduling in cloud-fog computing*, IEEE Access, vol. 11, pp. 20635–20646, 2023.
27. G. Harish, S. Nagaraju, B. Harish, and M. Shaik, *A review on fog computing and its applications*, International Journal of Innovative Technology and Exploring Engineering (IJITEE), vol. 8, no. 6C2, pp. 2278–3075, 2019.
28. S. Sarkar, S. Chatterjee, and S. Misra, *Assessment of the suitability of fog computing in the context of Internet of Things*, IEEE Transactions on Cloud Computing, vol. 6, no. 1, pp. 46–59, 2015.
29. S. Madakam and P. Bhagat, *Fog computing in the IoT environment: Principles, features, and models*, in *Fog Computing*, Springer, pp. 23–43, 2018.
30. Osama AL-BAIK, et al., *Pufferfish Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems*, Biomimetics, vol. 9, no. 2, 65, 2024.
31. M. Peng, T. Q. Quek, G. Mao, Z. Ding, and C. Wang, *Artificial intelligence-driven fog radio access networks: recent advances and future trends*, IEEE Wireless Communications, vol. 27, no. 2, pp. 12–13, 2020.
32. A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, *Harris hawks optimization: algorithm and applications*, Future Generation Computer Systems, vol. 97, pp. 849–872, 2019.
33. M. R. Bonyadi and Z. Michalewicz, *Particle swarm optimization for single objective continuous space problems: a review*, Evolutionary Computation, vol. 25, no. 1, pp. 1–54, 2017.
34. S. Mirjalili, *SCA: a sine cosine algorithm for solving optimization problems*, Knowledge-Based Systems, vol. 96, pp. 120–133, 2016.
35. Abdenacer NAOURI, et al., *Efficient fog node placement using nature-inspired metaheuristic for IoT applications*, Cluster Computing, pp. 1–17, 2024.