Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Enhancing IOT-WSN Performance and Longevity with Efficacious Flower Pollination Algorithm-Routing Protocol (EFPA-RP)

D. Deepalakshmi[1], Dr. B. Pushpa[2]

[1]Research Scholar, P.hD in Computer Science (Part-Time), Department of Computer Science and Information Science, Annamalai University, India
[2]Assistant Professor/Programmer, Department of Computer Science and Information Science, Annamalai University, India
Email: luxmids@gmail.com

This research delves into the implementation of the Efficacious Flower Pollination Algorithm-Routing Protocol (EFPA-RP) within IoT-based Wireless Sensor Networks (IoT-WSN) to optimize network performance and longevity. EFPA-RP draws inspiration from natural flower pollination processes, integrating advanced optimization mechanisms to enhance routing efficiency within IoT-WSN environments. Through rigorous simulation-based experimentation across various network sizes, the study evaluates EFPA-RP's efficacy in terms of energy consumption, throughput, delay, and network lifetime. The results reveal that EFPA-RP achieves notably higher network longevity while upholding efficient energy utilization and ensuring comprehensive data transmission coverage. By significantly outperforming existing protocols, EFPA-RP demonstrates its potential to address critical challenges within IoT-WSN deployments. This includes mitigating energy constraints, enhancing data transmission reliability, and prolonging network operational durations. The findings underscore EFPA-RP's capability to substantially improve overall network performance and sustainability within diverse IoT applications. As such, this research contributes valuable insights into advancing the optimization of IoT-WSN infrastructures, ultimately facilitating the seamless integration and operation of IoT technologies in various real-world scenarios.

Keywords: IoT-WSN, Routing, Flower Pollination Algorithm, Optimization, Energy Efficiency, Network Longevity.

1. Introduction

IoT-based Wireless Sensor Networks (IoT-WSN) represent a fusion of two transformative technologies: the Internet of Things (IoT) and Wireless Sensor Networks (WSNs). IoT-WSN systems deploy interconnected sensors that wirelessly communicate data to facilitate real-time monitoring and control across diverse environments [1]. These systems are characterized by their capacity to integrate numerous sensors into a cohesive network, enabling comprehensive data collection and analysis. Such data can serve many purposes, from environmental monitoring in agriculture to surveillance in smart cities. The seamless integration of IoT and WSN technologies allows for the creation of adaptable and scalable networks, accommodating the dynamic requirements of various applications [2]. At the core of IoT-WSN is the efficient utilization of wireless communication protocols, enabling seamless data exchange between sensors and centralized processing units. By leveraging these protocols, IoT-WSN systems facilitate rapid decision-making based on real-time data insights. The deployment of IoT-WSN systems often involves strategically placing sensors to optimize coverage and data accuracy [3]. This ensures that the collected data is reliable and representative of the monitored environment. IoT-WSN systems epitomize the convergence of IoT and WSN technologies, offering a robust framework for pervasive sensing and monitoring applications. Through their seamless integration and efficient communication capabilities, IoT-WSN systems pave the way for innovative solutions across various domains [4].

Routing in IoT-WSN is a complex task due to the resource constraints of network nodes, including limited power and processing capabilities. Efficient routing protocols are essential for establishing optimal pathways for data transmission from sensor nodes to designated destinations while conserving energy and ensuring reliable data delivery [5]. These protocols often employ data aggregation, clustering, and opportunistic routing strategies to address these challenges. Adaptability to dynamic network conditions is crucial for maintaining connectivity and reliability in IoT-WSN deployments. Effective routing protocols are vital in maximizing network efficiency and supporting various applications in diverse domains such as smart cities, industrial automation, and healthcare [6].

Bio-inspired optimization for IoT-WSN draws inspiration from natural processes and behaviours observed in biological systems to develop efficient algorithms for optimizing network performance and resource utilization. These bio-inspired approaches emulate the self-organization, adaptation, and cooperation principles found in biological systems to address the unique challenges posed by IoT-WSN environments [7]. One key aspect of bio-inspired optimization for IoT-WSN is the emulation of self-organization observed in biological systems. In nature, organisms exhibit emergent behaviours that enable them to achieve complex tasks collectively without centralized control. In IoT-WSN, bio-inspired algorithms can facilitate self-organization among sensor nodes, allowing them to autonomously adapt to changes in network topology, resource availability, and environmental conditions [8]. This self-organizing behaviour enhances the scalability, robustness, and resilience of IoT-WSN deployments. Bio-inspired optimization techniques can facilitate cooperation and collaboration among sensor nodes in IoT-WSN deployments. In nature, organisms often exhibit cooperative behaviours to achieve collective goals, such as foraging for food, defending against predators, or coordinating group movements. The bio-inspired algorithms can enable sensor nodes to cooperate in tasks such as data aggregation, distributed sensing, and collaborative decision-making [9]. This cooperative behaviour enhances network

efficiency, reduces redundant data transmission, and prolongs the lifespan of IoT-WSN deployments by conserving energy and bandwidth resources [10].

1.1 Problem Statement

In IoT-WSN, routing protocols face significant challenges in managing energy consumption effectively while maintaining reliable data transmission [11]–[13]. Existing protocols may not sufficiently address the dynamic network conditions and energy constraints inherent in IoT-WSN deployments. Developing a robust routing protocol explicitly tailored for IoT-WSN is imperative. This protocol must tackle issues such as uneven energy distribution among nodes, high communication overhead, and the potential for premature node depletion, which can significantly impact network performance and longevity. The protocol must adapt to network topologies and traffic patterns while ensuring minimal energy expenditure, particularly in resource-constrained environments. Addressing these challenges is essential to maximize the efficiency and sustainability of IoT-WSN deployments and enable the successful implementation of diverse IoT applications.

1.2. Motivation

In the domain of IoT-WSN, optimizing routing protocols is a pivotal endeavor with far-reaching implications. The very essence of IoT-WSN lies in the seamless collection and transmission of data from diverse environments, making reliable and energy-efficient routing protocols indispensable. Yet, current solutions often fail to address the intricate balance between energy conservation and data reliability, leading to inefficiencies and premature node depletion. The motivation behind developing novel routing protocols for IoT-WSN lies in unlocking the full potential of this transformative technology. By addressing the unique challenges of energy consumption and routing in IoT-WSN, we can pave the way for more sustainable and resilient network infrastructures. This endeavor ensures the longevity and scalability of IoT deployments and opens doors to many innovative applications across domains such as environmental monitoring, precision agriculture, and smart cities. Pursuing optimized routing protocols for IoT-WSN is not merely a technical challenge but a gateway to unlocking the transformative power of the Internet of Things.

1.3. Objective

The primary objective of this research is to develop an Efficacious Flower Pollination Algorithm (EFPA) explicitly tailored for optimizing routing in IoT-WSN. This enhanced algorithm aims to overcome the limitations of existing routing protocols by leveraging the principles of flower pollination to achieve more efficient and energy-conscious routing strategies. The research seeks to enhance the Flower Pollination Algorithm's adaptability, scalability, and robustness (FPA) by incorporating novel mechanisms inspired by biological systems. Through extensive experimentation and performance evaluation, the research aims to demonstrate the superiority of the EFPA over traditional routing protocols in terms of energy consumption, network lifetime, packet delivery ratio, and end-to-end delay. The research endeavors to validate the effectiveness of the EFPA in diverse IoT-WSN scenarios, including dynamic network topologies, varying traffic patterns, and resource-constrained environments. The goal is to provide a comprehensive framework for optimizing routing in IoT-WSN deployments, enabling more sustainable and resilient Internet of Things infrastructures.

2. LITERATURE REVIEW

"Segmented Sectors in Energy Efficient Routing (SSEER)" [14] partitions a network into smaller sectors based on node physical locations, optimizing traffic flow by further subdividing each sector. Nodes within these sectors communicate in a multi-hop setup, with data packets traversing from sender to receiver through intermediate nodes. The selection criteria for intermediate nodes include remaining energy, destination proximity, and overall network quality. Cluster heads, chosen for their remaining energy, strategic location, and connectivity to other clusters, act as gatekeepers. They manage inter-sector data transmission, enhancing overall network efficiency and longevity.

"Cluster Routing Protocol (CRP)" [15] leverages fog computing and 5G technology to improve network energy efficiency, reliability, and data transmission. CRP employs a hierarchical clustering approach, dividing the network into multiple subnetworks managed by cluster heads (CHs). These CHs link the Wireless Sensor Network (WSN) to the fog computing layer, which provides additional computational power and storage. Utilizing 5G technology, CRP ensures secure and extensive data transfer. The protocol's fog-enabled architecture enhances WSN performance by offloading intensive processing tasks to the fog layer, reducing latency and conserving node energy.

"Energy-Efficient Distributed Node Clustering Routing" [16] divides the network into clusters and dynamically adjusts the cluster size based on the mobility patterns of the nodes. The authors evaluate It using simulations and demonstrate its effectiveness in energy efficiency and network performance. "Energy-Efficient Guiding-Network-based Routing" [17] uses a guiding network to direct data transmission between nodes and uses an energy-efficient scheduling algorithm to reduce energy consumption. The authors evaluate It using simulations and demonstrate its effectiveness in energy efficiency and network performance. "Energy-Efficient Cooperative Routing Scheme" [18] divides the network into clusters and uses a cooperative transmission approach to reduce energy consumption. The authors evaluate It using simulations and demonstrate its effectiveness in energy efficiency and network performance. One potential disadvantage of this protocol is the potential for increased computational complexity due to the use of neural networks, which may limit its practical application in some scenarios.

"Trust-Based Secure Intelligent Opportunistic Routing" [19] proposes a trust-based secure opportunistic routing protocol for wireless sensor networks that utilizes machine learning techniques to make routing decisions. It considers the nodes' trustworthiness in the network and uses a secure key exchange mechanism to ensure data confidentiality. The authors evaluate It using simulations and demonstrate its security and network performance effectiveness. "Hybrid Optimization for Cluster-Based Routing" [20] uses a combination of swarm intelligence and genetic algorithms to optimize network performance and energy consumption. The authors evaluate It using simulations and demonstrate its effectiveness in energy efficiency and network performance. "Cluster Head Rotation Based Routing" [21] aims to extend the network lifetime by reducing energy consumption. The paper thoroughly evaluates It and compares it with existing routing protocols. The results show that it outperforms existing energy consumption and network lifetime protocols.

"Low-Delay Routing-Integrated MAC Protocol" [22] aims to reduce communication latency and energy consumption. It integrates routing and MAC layer functions to reduce communication overhead and uses a dynamic duty cycle approach to reduce energy

consumption. The authors evaluate It using simulations and demonstrate its effectiveness in latency reduction and energy efficiency. "Distributed 2-Hop Cluster Routing" [23] aims to reduce communication overhead and improve network scalability. It divides the network into clusters and uses a 2-hop routing approach to reduce communication overhead. The authors evaluate It using simulations and demonstrate its effectiveness in network performance and scalability. "Multi-Hop Routing Protocol Evaluation" [24] aims to improve network performance by reducing packet loss and end-to-end delay. It uses a dynamic priority queue approach to prioritize multimedia traffic and adaptively adjust the transmission power of nodes to reduce packet loss. The authors evaluate It using simulations and demonstrate its effectiveness in packet loss reduction and end-to-end delay.

"Expected Area-Based Real-Time Routing" [25] considers the expected area of mobile sinks and uses a priority-based scheme to route data to the nearest sink. The authors evaluate It using simulations and demonstrate its effectiveness in terms of packet delivery ratio and end-to-end delay. "Tactile Routing for Location Privacy Preservation" [26] aims to prevent an adversary from discovering the location of a target node by routing data through intermediate nodes randomly and unpredictably. The authors evaluate It using simulations and demonstrate its effectiveness regarding location privacy preservation. "Enhancing Graph Routing Algorithm" [27] aims to optimize the route selection process by using an evolutionary algorithm to adapt to changes in the network topology. The authors evaluate It using simulations and demonstrate its effectiveness in energy efficiency and network lifetime. The significant disadvantage of this protocol is the potential for increased computational complexity due to the use of evolutionary algorithms.

3. FLOWER POLLINATION ALGORITHM

The Flower Pollination Algorithm (FPA) is a nature-inspired optimization algorithm that simulates the process of flower pollination to solve optimization problems. This algorithm represents potential solutions as flowers; a solution's quality corresponds to the flower's nectar amount. The algorithm mimics the pollination process, where flowers with higher nectar amounts attract pollinators like bees and butterflies, facilitating the transfer of pollen grains and the propagation of genetic traits. In optimization, this process translates into exchanging information between solutions to explore the search space and converge towards better solutions. The FPA employs global pollination, local pollination, and migration strategies to balance exploration and exploitation, ensuring efficient convergence to optimal solutions. One of the key advantages of the FPA is its simplicity and flexibility, allowing it to be easily applied to various optimization problems across different domains. The FPA offers a powerful tool for solving complex optimization problems and has been successfully applied in areas such as engineering design, image processing, and machine learning by emulating the efficiency of natural systems. Its pseudocode is provided as Algorithm 1.

Algorithm 1: Core – Flower Pollination Algorithm

Input:

- Parameters defining the optimization problem (e.g., number of flowers, maximum iterations)
- Initial population of flowers with randomly assigned nectar amounts
- Fitness evaluation function to determine solution quality

Output:

- The best solution found (flower with the highest nectar amount) after the algorithm terminates.

Procedure:
Step 1: Initialize the population of flowers randomly.
Step 2: Evaluate the fitness (nectar amount) of each flower.
Step 3: Repeat until stopping criteron:
a. Randomly select flowers to share nectar.
b. Evaluate the fitness of newly generated flowers.
c. Adjust nectar levels based on local information.
d. Evaluate the fitness of new flowers.
e. Update flower population based on fitness.
f. Sort flowers by fitness.
g. Exchange flowers between populations.
h. Evaluate the fitness of new flowers.
i. Update flower population based on fitness.
j. Select the best flowers to survive.
Step 4: Return the best solution found as output.

4. EFFICACIOUS FLOWER POLLINATION ALGORITHM

The Efficacious Flower Pollination Algorithm (EFPA) is an advanced optimization technique inspired by the natural process of flower pollination. It extends the traditional Flower Pollination Algorithm (FPA) by incorporating enhanced global exploration and local exploitation mechanisms. EFPA utilizes the concept of pollen dissemination among flowers to enhance the search process for optimal solutions to optimization problems. Through the iterative pollination process, EFPA enables exploring a wide search space while efficiently exploiting promising regions for better solutions. Key features of EFPA include its adaptability, scalability, and ability to handle complex optimization problems efficiently. EFPA enhances the diversity of solutions and promotes convergence to optimal or near-optimal solutions by integrating strategies such as mutation and crossover. This algorithm has shown promising results in various fields, including engineering, computer science, and finance, making it a valuable tool for solving real-world optimization problems effectively.

4.1. Initialization in EFPA: Setting the Stage for Optimization

The Initialization step lays the foundation for the optimization process by creating an initial population of potential solutions. This crucial step involves setting up the framework within which the algorithm will operate and initializing the parameters and variables that will guide the search for optimal solutions. Initially, it deals with the search space's random flower creation. There is an optimization issue, and each bloom is a possible solution. The position of each flower is randomly selected within the bounds of the problem domain, ensuring diversity in the initial population. Mathematically, this can be represented as Eq.(1).

$$\boldsymbol{x_i = x_{min} + (x_{max} - x_{min}) \times rand(0,1)} \quad (1)$$

where $\boldsymbol{i}$th flower's location is denoted by $\boldsymbol{x_i}$, the search space's minimum and maximum boundaries are $\boldsymbol{x_{min}}$ and $\boldsymbol{x_{max}}$ random integer between 0 and 1 is generated using the rand function.

After randomly creating the flowers, this research needs to assess their fitness by referring to the objective function of the optimization problem. To do this, we must determine how effectively each bloom meets the challenge's goals. Mathematically, the fitness $\boldsymbol{f_i}$ of the $\boldsymbol{i}$th flower can be expressed as Eq.(2).

$$\boldsymbol{f_i = Objective\ Function(x_i)} \quad (2)$$

where $\boldsymbol{Objective\ Function}(\boldsymbol{x_i})$ represents the evaluation of the objective function at the position $\boldsymbol{x_i}$ of the $\boldsymbol{i}$th flower.

In EFPA, each flower carries an amount of pollen, representing information that can be exchanged with other flowers during the pollination process. The amount of pollen is initialized based on the fitness of each flower, with higher-fitness flowers carrying more pollen. Mathematically, the pollen amount $\boldsymbol{p_i}$ for the $\boldsymbol{i}$th flower can be determined as Eq.(3).

$$\boldsymbol{p_i} = \frac{\boldsymbol{f_i}}{\sum_{\boldsymbol{j=1}}^{\boldsymbol{N}} \boldsymbol{f_j}} \times \boldsymbol{P_{total}} \quad (3)$$

where $\boldsymbol{N}$ is the total number of flowers in the population, and $\boldsymbol{P_{total}}$ is the total amount of pollen available in the population.

The attractiveness of each flower, which determines its likelihood of sharing pollen with other flowers, is also initialized during this step. Flowers with higher fitness values are assigned higher attractiveness values. Mathematically, the attractiveness $\boldsymbol{A_i}$ of the $\boldsymbol{i}$th flower can be defined as Eq.(4).

$$\boldsymbol{A_i} = \frac{\boldsymbol{f_i}}{\sum_{\boldsymbol{j=1}}^{\boldsymbol{N}} \boldsymbol{f_j}} \quad (4)$$

where $\boldsymbol{N}$ is the total number of flowers in the population, and $\boldsymbol{f_i}$ is the fitness of the $\boldsymbol{i}$th flower.

The Initialization step may involve setting up termination criteria to determine when the optimization process should stop. This could include a maximum number of iterations, a threshold for improving fitness values, or a predefined time limit. These criteria ensure that the algorithm does not continue indefinitely and converges to a satisfactory solution within a reasonable timeframe.

4.2. Evaluating Fitness

In EFPA, the step "Evaluate Fitness" plays a crucial role in assessing the quality of potential solutions within the optimization process. To determine fitness, this research looks at how each bloom does concerning the goals of the optimization issue. In EFPA, the Monte Carlo method is a computational technique to evaluate fitness efficiently. The optimization problem is defined by an objective function that maps each potential solution to a corresponding fitness value. Mathematically, the objective function $\boldsymbol{f}(\boldsymbol{x})$ can be represented as Eq.(5).

$$\boldsymbol{f}(\boldsymbol{x}) \quad (5)$$

where $\boldsymbol{x}$ represents the decision variables of the problem.

The Monte Carlo method involves sampling points randomly from the solution space to estimate the objective function's value. Let $\boldsymbol{x_i}$ denote the $\boldsymbol{i}$th sampled point in the solution space. For each sampled point $\boldsymbol{x_i}$, the objective function $\boldsymbol{f}(\boldsymbol{x_i})$ is evaluated to determine its fitness value. This evaluation estimates the objective function's behaviour across the solution space. The fitness values obtained from evaluating the objective function at the sampled points are aggregated to assess the overall fitness of the flower. Let $\boldsymbol{f_{MC}}$ in Eq.(6) denote the Monte Carlo estimate of fitness.

$$f_{MC} = \frac{1}{N}\sum_{i=1}^{N} f(x_i) \quad (6)$$

where N is the total number of sampled points.

The Monte Carlo fitness estimation may be subject to error due to the finite number of sampled points. The estimation error e_{MC} the difference between the true fitness value and the Monte Carlo estimate can be quantified using Eq.(7).

$$e_{MC} = |f(x) - f_{MC}| \quad (7)$$

This research look at how the Monte Carlo estimate converges to determine how many points need to be sampled to get the desired accuracy level. Let ϵ in Eq.(8) denote the desired level of accuracy and N_{conv} denote the number of sampled points required for convergence.

$$N_{conv} = \frac{\sigma^2}{\epsilon^2} \quad (8)$$

where σ^2 is the variance of the fitness values obtained from sampling.

A confidence interval can be constructed around the Monte Carlo estimate to quantify the uncertainty in the fitness value. Let CI denote the confidence interval, which is computed using Eq.(9).

$$CI = f_{MC} \pm z\frac{\sigma}{\sqrt{N}} \quad (9)$$

where z is the z-score corresponding to the desired confidence level.

The Monte Carlo sampling process can be parallelized across multiple computing nodes to accelerate fitness evaluation. This parallelization strategy enables the simultaneous assessment of fitness values at multiple sampled points, reducing the overall computation time. Adaptive sampling strategies can dynamically adjust the number of sampled points based on the convergence behaviour observed during optimization. These strategies optimize the allocation of computational resources to regions of the solution space where fitness values are uncertain or poorly estimated.

Algorithm 2: Evaluating Fitness

Input:
- Objective function $f(x)$
- Number of samples N

Output:
- Estimated fitness f_{MC}

Procedure:
1. Initialize f_{MC} to 0.
2. For $i = 1$ to N, do steps 3-5.
3. Generate a random sample x_i from the solution space.
4. Evaluate the objective function $f(x_i)$

5.	Add $\boldsymbol{f(x_i)}$to $\boldsymbol{f_{MC}}$
6.	Compute the estimated fitness $\boldsymbol{f_{MC}}$ as $\frac{f_{MC}}{N}$
7.	Output the estimated fitness $\boldsymbol{f_{MC}}$

4.3. Determining attractiveness

This step involves assessing the attractiveness of each flower, which influences its likelihood of sharing pollen with other flowers. In EFPA, the Monte Carlo method is utilized to evaluate the fitness of flowers, which in turn is used to determine their attractiveness. The first subheading involves evaluating the objective function to assess the fitness of each flower. Mathematically, the fitness $\boldsymbol{f(x_i)}$ of the $\boldsymbol{i}$th flower is calculated using the Monte Carlo method, as described in Eq.(10).

$$\boldsymbol{f(x_i) = \frac{1}{N}\sum_{i=1}^{N} f(x_i)} \quad (10)$$

where $\boldsymbol{N}$ is the total number of sampled points.

After evaluating the fitness of each flower, the fitness values are normalized to ensure that they lie within a specified range. This normalization prevents disproportionately high or low fitness values from dominating the attractiveness calculation. Mathematically, the normalized fitness $\boldsymbol{\hat{f}(x_i)}$ of the $\boldsymbol{i}$th flower can be expressed as Eq.(11).

$$\boldsymbol{\hat{f}(x_i) = \frac{f(x_i) - min\,(f)}{max\,(f) - min\,(f)}} \quad (11)$$

where $\boldsymbol{min\,(f)}$ and $\boldsymbol{max\,(f)}$ represent the minimum and maximum fitness values in the population, respectively.

Once the fitness values are normalized, the attractiveness of each flower is computed based on its normalized fitness. Flowers with higher fitness values are assigned higher attractiveness, indicating their more significant potential for pollen dissemination. Mathematically, the attractiveness $\boldsymbol{A_i}$ of the $\boldsymbol{i}$th flower can be defined as Eq.(12).

$$\boldsymbol{A_i = \frac{\hat{f}(x_i)}{\sum_{j=1}^{N} \hat{f}(x_j)}} \quad (12)$$

where $\boldsymbol{\hat{f}(x_i)}$ represents the normalized fitness of the $\boldsymbol{i}$th flower, and $\boldsymbol{N}$ is the total number of flowers in the population.

The attractiveness of each flower is directly influenced by its fitness relative to the fitness of other flowers in the population. Flowers with higher normalized fitness values have a proportionally more significant share of the total attractiveness, reflecting their superior performance in optimization. The attractiveness of flowers plays a crucial role in guiding the exchange of information (pollen) during the pollination process. Flowers with higher attractiveness values are more likely to attract pollen from neighbouring flowers, facilitating the dissemination of information and contributing to solution space exploration.

Algorithm 3: Determining Attractiveness
Input: • Fitness values of flowers $\boldsymbol{f(x_i)}$ Output: • Attractiveness values of flowers $\boldsymbol{A_i}$

Procedure:

1. Normalize the fitness values of flowers to ensure they lie within a specified range.
2. Compute the sum of the normalized fitness values $\sum_{j=1}^{N} \hat{f}(x_j)$.
3. For each flower i from 1 to N, do steps 4-5.
4. Calculate the normalized fitness $\hat{f}(x_i)$of the ith flower.
5. Compute the attractiveness $A_i$of the ith flower using the formula $A_i = \frac{\hat{f}(x_i)}{\sum_{j=1}^{N} \hat{f}(x_j)}$.
6. Output the attractiveness values of flowers A_i.

4.4. Local Pollination

This step aims to promote exploration of the local search space by encouraging flowers to share pollen with their neighbours. In EFPA, the Monte Carlo method is employed to evaluate the fitness of flowers, which guides the local pollination process. The Neighbourhood Selection in EFPA involves selecting the neighbourhood of each flower, which determines the flowers with which it will exchange pollen. In EFPA, a neighbourhood radius $r_{neighbor}$ is defined around each flower, and all flowers within this radius are considered its neighbours.

Once the neighbourhood is selected, pollen exchange occurs between each flower and its neighbours. Flowers with higher fitness values are likelier to share pollen with their neighbours, influencing pollination. Mathematically, the pollen exchange between the ith flower and its neighbour j can be expressed as Eq.(13).

$$p_{ij} = A_i \times A_j \times \frac{f(x_i) + f(x_j)}{2} \quad (13)$$

where p_{ij} represents the amount of pollen exchanged between the ith flower and its neighbour j, A_i and A_j are the attractiveness values of the ith and jth flowers, respectively, and $f(x_i)$ and $f(x_j)$ are their fitness values.

After pollen exchange, each flower updates its pollen amount based on the pollen received from its neighbours. The total amount of pollen received by the ith flower, P_i, can be computed as the sum of pollen from all its neighbours. Mathematically, this can be expressed as Eq.(14).

$$P_i = \sum_{j=1}^{N} p_{ij} \quad (14)$$

where N is the total number of neighbours of the ith flower.

Once the pollen each flower receives is determined, it is redistributed among its neighbours. Flowers with higher attractiveness values receive a larger share of pollen, reflecting their influence on pollination. Mathematically, the pollen p_{ij} received by the jth neighbour of the ith flower can be calculated as Eq.(15).

$$p_{ij} = \frac{P_i \times A_i \times A_j}{\sum_{k=1}^{N} A_k} \quad (15)$$

where P_i is the total amount of pollen received by the ith flower, A_i and A_j are the attractiveness values of the ith flower and its neighbour j, respectively, and $\sum_{k=1}^{N} A_k$ is the sum of attractiveness values of all neighbours of the ith flower.

The local pollination process promotes exploration of the local search space by encouraging flowers to share pollen with their neighbours. By exchanging information with nearby flowers, each flower gains insights into potentially promising regions of the solution space, contributing to the overall optimization process.

Algorithm 4: Local Pollination

Input:
- Fitness values of flowers $f(x_i)$
- Attractiveness values of flowers A_i
- Neighbourhood radius $r_{neighbor}$

Output:
- Updated pollen amounts of flowers P_i

Procedure:
1. For each flower i from 1 to N, do steps 2-5.
2. Determine the neighbourhood of the ith flower within the radius $r_{neighbor}$.
3. For each neighbour$j$ of the ith flower, do steps 4-5.
4. Calculate the pollen exchange p_{ij} between the ith flower and its neighbour j using the formula $p_{ij} = A_i \times A_j \times \frac{f(x_i)+f(x_j)}{2}$.
5. Update the pollen amount of the jth neighbour by adding the pollen received p_{ij}.
6. Calculate the total amount of pollen received by the ith flower P_i as $P_i = \sum_{j=1}^{N} p_{ij}$.
7. Redistribute the pollen among the neighbours of the ith flower based on their attractiveness values.
8. For each neighbour$j$ of the ith flower, calculate the pollen received p_{ij} using the formula $p_{ij} = \frac{P_i \times A_i \times A_j}{\sum_{k=1}^{N} A_k}$.
9. Update the pollen amount of the jth neighbour by adding the pollen received p_{ij}.
10. Output the updated pollen amounts of flowers P_i.

4.5. Global Pollination

This step promotes exploration of the entire search space by facilitating the exchange of information (pollen) among distant flowers. This step encourages flowers to share pollen with others across the population, allowing for the dissemination of valuable information and the discovery of potentially promising regions of the solution space. In EFPA, the Monte Carlo method is employed to evaluate the fitness of flowers, which guides the global pollination process.

Identifying global pollination involves identifying potential candidates across the population of flowers. Unlike local pollination, which focuses on nearby neighbours, global pollination considers flowers from distant locations in the search space. All flowers in the population are considered candidates for global pollination, allowing for the exchange of information across the entire solution space. Once the global pollination candidates are identified, pollen exchange occurs between each flower and its distant counterparts. Flowers with higher fitness values are likelier to share pollen with others, influencing global pollination. Mathematically, the pollen exchange between the ith flower and its distant counterpart j can be expressed as Eq.(16).

$$p_{ij} = A_i \times A_j \times \frac{f(x_i) + f(x_j)}{2} \quad (16)$$

where p_{ij} represents the amount of pollen exchanged between the ith flower and its distant counterpart j, A_i and A_j are the attractiveness values of the ith flower and its distant counterpart j, respectively, and $f(x_i)$ and $f(x_j)$ are their fitness values.

After pollen exchange, each flower updates its pollen amount based on the pollen received from its distant counterparts. The total amount of pollen received by the $\boldsymbol{ith}$ flower, $\boldsymbol{P_i}$, can be computed as the sum of pollen from all its distant counterparts. Mathematically, this can be expressed as Eq.(17).

$$\boldsymbol{P_i} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{N}} \boldsymbol{p_{ij}} \quad (17)$$

where $\boldsymbol{N}$ is the total number of distant counterparts of the $\boldsymbol{i}$th flower.

Once the pollen each flower receives is determined, it is redistributed among its distant counterparts. Flowers with higher attractiveness values receive a larger share of pollen, reflecting their influence in global pollination. Mathematically, the pollen $\boldsymbol{p_{ij}}$ received by the distant counterpart $\boldsymbol{j}$ of the $\boldsymbol{i}$th flower can be calculated as Eq.(18).

$$\boldsymbol{p_{ij}} = \frac{\boldsymbol{P_i} \times \boldsymbol{A_i} \times \boldsymbol{A_j}}{\sum_{\boldsymbol{k=1}}^{\boldsymbol{N}} \boldsymbol{A_k}} \quad (18)$$

where $\boldsymbol{P_i}$ is the total amount of pollen received by the $\boldsymbol{i}$th flower, $\boldsymbol{A_i}$ and $\boldsymbol{A_j}$are the attractiveness values of the $\boldsymbol{i}$th flower and its distant counterpart $\boldsymbol{j}$, respectively, and $\sum_{\boldsymbol{k=1}}^{\boldsymbol{N}} \boldsymbol{A_k}$ is the sum of attractiveness values of all distant counterparts of the $\boldsymbol{i}$th flower.

The global pollination process enhances the exploration of the solution space by facilitating the exchange of information among distant flowers. By sharing pollen with distant counterparts, each flower gains insights into potentially promising regions of the solution space, contributing to the overall exploration of the optimization landscape.

Algorithm 5: Global Pollination

Input:
- Fitness values of flowers $\boldsymbol{f(x_i)}$
- Attractiveness values of flowers $\boldsymbol{A_i}$

Output:
- Updated pollen amounts of flowers $\boldsymbol{P_i}$

Procedure:
1. For each flower $\boldsymbol{i}$ from 1 to $\boldsymbol{N}$, do steps 2-5.
2. Identify the distant counterparts of the $\boldsymbol{i}$th flower.
3. For each distant counterpart $\boldsymbol{j}$ of the $\boldsymbol{i}$th flower, do steps 4-5.
4. Calculate the pollen exchange $\boldsymbol{p_{ij}}$ between the $\boldsymbol{i}$th flower and its distant counterpart $\boldsymbol{j}$ using the formula $\boldsymbol{p_{ij}} = \boldsymbol{A_i} \times \boldsymbol{A_j} \times \frac{f(x_i)+f(x_j)}{2}$.
5. Update the pollen amount of the distant counterpart $\boldsymbol{j}$ by adding the pollen received $\boldsymbol{p_{ij}}$.
6. Calculate the total amount of pollen received by the $\boldsymbol{i}$th flower $\boldsymbol{P_i}$ as $\boldsymbol{P_i} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{N}} \boldsymbol{p_{ij}}$.
7. Redistribute the pollen among the distant counterparts of the $\boldsymbol{i}$th flower based on their attractiveness values.
8. For each distant counterpart $\boldsymbol{j}$ of the $\boldsymbol{i}$th flower, calculate the pollen received $\boldsymbol{p_{ij}}$using the formula $\boldsymbol{p_{ij}} = \frac{\boldsymbol{p_i} \times \boldsymbol{A_i} \times \boldsymbol{A_j}}{\sum_{\boldsymbol{k=1}}^{\boldsymbol{N}} \boldsymbol{A_k}}$.
9. Update the pollen amount of the distant counterpart $\boldsymbol{j}$ by adding the pollen received $\boldsymbol{p_{ij}}$.
10. Output the updated pollen amounts of flowers $\boldsymbol{P_i}$.

4.6. Updating Flower Positions

This step is crucial for enhancing solution exploration and convergence towards optimal solutions. In EFPA, the Monte Carlo method is employed to evaluate the fitness of flowers, guiding the updating process. Let's delve into the mathematical formulation of this step in

detail. Initially, it involves incorporating the pollen information received during the pollination process into updating flower positions. Pollen serves as a carrier of information exchanged among flowers, reflecting the fitness and attractiveness of neighbouring flowers. Mathematically, the pollen $\boldsymbol{P_i}$ received by the $\boldsymbol{i}$th flower can be expressed as Eq.(19).

$$\boldsymbol{P_i = \sum_{j=1}^{N} p_{ij}} \quad (19)$$

where $\boldsymbol{p_{ij}}$ represents the amount of pollen exchanged between the

Once the pollen information is aggregated, the step size for updating flower positions is determined. The step size influences the magnitude of changes to flower positions, impacting the exploration and exploitation balance. A smaller step size encourages the exploitation of promising regions, while a more significant step size promotes the exploration of the solution space. Mathematically, the step size $\boldsymbol{\delta}$ can be defined as Eq.(20).

$$\boldsymbol{\delta = \frac{1}{1 + exp(-\alpha P_i)}} \quad (20)$$

where $\boldsymbol{\alpha}$ is a tuning parameter controlling the influence of pollen information on the step size. With the step size determined, the flower positions are updated using a stochastic process that balances exploration and exploitation. Flowers adjust their positions probabilistically, with the probability of moving towards regions with higher pollen amounts influenced by the step size. Mathematically, the updated position $\boldsymbol{x_i'}$ of the $\boldsymbol{i}$th flower can be expressed as Eq.(21).

$$\boldsymbol{x_i' = x_i + \delta \times rand(0,1) \times (x_{max} - x_{min})} \quad (21)$$

where $\boldsymbol{x_i}$ is the current position of the $\boldsymbol{i}$th flower, $\boldsymbol{x_{min}}$ and $\boldsymbol{x_{max}}$ are the minimum and maximum bounds of the search space, and $\boldsymbol{rand(0,1)}$ generates a random number between 0 and 1.

The updating process balances exploration and exploitation by adjusting flower positions based on local and global information. Flowers explore promising regions of the solution space while exploiting areas with high fitness values. The balance between exploration and exploitation is crucial for efficiently navigating complex optimization landscapes. As the optimization process progresses, updating flower positions influences the convergence behaviour of the algorithm. Smaller step sizes may lead to slower convergence but finer exploration of the solution space, while larger step sizes may accelerate convergence but risk premature convergence to suboptimal solutions. Careful tuning of parameters is necessary to achieve desired convergence characteristics.

4.7. Evaluating New Solutions

This step ensures that the optimization process progresses toward solutions with improved fitness values. Initially, sample points are generated around the updated positions of flowers to estimate their fitness values. These sample points represent potential solutions near the updated positions, allowing for a comprehensive assessment of their fitness. Mathematically, the sample points $\boldsymbol{x_i'}$ around the updated position $\boldsymbol{x_i}$ of the $\boldsymbol{i}$th flower can be expressed as Eq.(22).

$$\boldsymbol{x_i' = x_i + \epsilon \times rand(0,1) \times (x_{max} - x_{min})} \quad (22)$$

where $\boldsymbol{\epsilon}$ is a small perturbation parameter controlling the magnitude of the perturbation around

the updated position.

Once the sample points are generated, the fitness of each sample point is evaluated using the objective function of the optimization problem. This evaluation provides an estimation of the fitness of the updated positions of flowers, guiding the subsequent steps of the algorithm. Mathematically, the fitness $\boldsymbol{f}(\boldsymbol{x}_i')$ of the $\boldsymbol{i}$th sample point can be expressed as Eq.(23).

$\boldsymbol{f}(\boldsymbol{x}_i')$	(23)

where $\boldsymbol{x}_i'$ represents the sample point and $\boldsymbol{f}(\boldsymbol{x}_i')$ represents the evaluation of the objective function at the sample point.

After evaluating the fitness of each sample point, the fitness values are aggregated to estimate the overall fitness of the updated positions of flowers. This aggregation process provides a comprehensive assessment of the performance of the updated solutions, guiding the selection of promising candidates for further exploration. Mathematically, the aggregated fitness $\boldsymbol{f}_{\boldsymbol{agg}}$ of the updated positions can be calculated as Eq.(24).

$\boldsymbol{f}_{\boldsymbol{agg}} = \frac{\boldsymbol{1}}{\boldsymbol{N}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{f}(\boldsymbol{x}_i')$	(24)

where $\boldsymbol{N}$ is the total number of sample points generated.

The Monte Carlo fitness estimation may be subject to error due to the finite number of sample points. The estimation error $\boldsymbol{e}_{\boldsymbol{MC}}$ the difference between the true fitness value and the Monte Carlo estimate can be quantified. Mathematically, the estimation error can be expressed as Eq.(25).

$\boldsymbol{e}_{\boldsymbol{MC}} = \left\lvert \boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}_{\boldsymbol{agg}} \right\rvert$	(25)

where $\boldsymbol{f}(\boldsymbol{x})$ represents the actual fitness value of the updated positions.

The convergence of the Monte Carlo estimation can be analyzed to determine the number of sample points required to achieve the desired level of accuracy. A larger number of sample points reduces the estimation error but increases computational cost. Balancing the trade-off between accuracy and computational resources is essential for efficiently evaluating new solutions.

Algorithm 6: Evaluating New Solutions

Input:
- Updated positions of flowers $\boldsymbol{x}_i$
- Objective function $\boldsymbol{f}(\boldsymbol{x})$
- Number of sample points $\boldsymbol{N}$

Output:
- Aggregated fitness value $\boldsymbol{f}_{\boldsymbol{agg}}$

Procedure:
1. For each flower $\boldsymbol{i}$ from 1 to $\boldsymbol{N}$, do steps 2-3.
2. Generate a sample point $\boldsymbol{x}_i'$ around the updated position $\boldsymbol{x}_i$.
3. Evaluate the fitness of the sample point $\boldsymbol{f}(\boldsymbol{x}_i')$ using the objective function.
4. Aggregate all sample points' fitness values to estimate the updated positions' overall fitness.
5. Compute the aggregated fitness $\boldsymbol{f}_{\boldsymbol{agg}}$ as $\frac{\boldsymbol{1}}{\boldsymbol{N}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{f}(\boldsymbol{x}_i')$.
6. Output the aggregated fitness value $\boldsymbol{f}_{\boldsymbol{agg}}$.

4.8. Selecting Best Solutions

Selecting the best solutions ensures that the algorithm explores and exploits solutions with higher fitness values, leading to improved convergence and solution quality. In EFPA, the Monte Carlo method facilitates selecting the best solutions, providing a reliable performance estimation. The aggregating fitness values in this step involves aggregating the fitness values of all algorithm-generated solutions. This aggregation process provides an overview of the performance of the solutions, allowing for a comparative analysis to identify the best candidates. Mathematically, the aggregated fitness $\boldsymbol{f_{agg}}$can be calculated as Eq.(26).

$$\boldsymbol{f_{agg} = \frac{1}{N}\sum_{i=1}^{N} f(x_i)} \quad (26)$$

where $\boldsymbol{N}$ is the total number of solutions generated and $\boldsymbol{f(x_i)}$ represents the fitness of the $\boldsymbol{i}$th solution.

After aggregating the fitness values, the distribution across the solution space is estimated using the Monte Carlo method. This estimation provides insights into the spread and concentration of fitness values, guiding the selection of the best solutions. Mathematically, the fitness distribution $\boldsymbol{f_{dist}(x)}$ can be approximated as Eq.(27).

$$\boldsymbol{f_{dist}(x) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)} \quad (27)$$

where $\boldsymbol{\delta(x - x_i)}$ is the Dirac delta function representing the fitness value of the $\boldsymbol{i}$th solution at position $\boldsymbol{x}$.

Based on the estimated fitness distribution, promising solutions with fitness values above a certain threshold are identified as potential candidates for selection. Solutions with higher fitness values are prioritized, reflecting their potential to contribute to the optimization process. Mathematically, the set of promising solutions $\boldsymbol{S_{promising}}$ can be defined as Eq.(28).

$$\boldsymbol{S_{promising} = \{x_i | f(x_i) > \tau\}} \quad (28)$$

where $\boldsymbol{\tau}$ is the fitness threshold used to distinguish promising solutions.

Once promising solutions are identified, a selection strategy is employed to choose the best solutions from the pool of candidates. Various selection strategies can be used, including elitism, tournament selection, or roulette wheel selection, depending on the specific requirements of the optimization problem. Mathematically, the selected solutions $\boldsymbol{S_{selected}}$ can be determined using the chosen selection strategy. The selected best solutions are updated to guide the next iteration of the optimization process. This updating process ensures that the algorithm continues to explore and exploit the solution space effectively, leading to continuous improvement in solution quality. Mathematically, the updated best solutions $\boldsymbol{S_{updated}}$ can be obtained by applying appropriate mutation or crossover operators, depending on the optimization algorithm.

Algorithm 7: Selecting Best Solutions

Input:

- Fitness values of all solutions $\boldsymbol{f(x_i)}$
- Fitness threshold $\boldsymbol{\tau}$
- Number of best solutions to select $\boldsymbol{k}$

Output:
- Selected best solutions $S_{selected}$

Procedure:
1. Aggregate the fitness values of all solutions to calculate the aggregated fitness f_{agg}.
2. Estimate the fitness distribution across the solution space using the Monte Carlo method.
3. Identify promising solutions with fitness values above the threshold τ and store them in $S_{promising}$.
4. Apply the selected selection strategy (e.g., elitism, tournament selection, roulette wheel selection) to choose the best solutions from $S_{promising}$ and store them in $S_{selected}$.
5. If necessary, Update the best solutions using appropriate mutation or crossover operators.
6. Output the selected best solutions $S_{selected}$.

4.9. Stopping Criterion

The Stopping Criterion is essential for determining when the optimization process should terminate. This step evaluates various termination conditions to decide whether further iterations are necessary. The first step is convergence criteria, which involves defining convergence criteria to assess whether the optimization process has sufficiently converged to an optimal solution. These criteria typically include monitoring changes in fitness values over successive iterations or assessing the stability of solution trajectories. Mathematically, convergence can be evaluated using Eq.(29) criteria.

$$\left|f(x_{current}) - f\left(x_{previous}\right)\right| < \epsilon \quad (29)$$

where $f(x_{current})$ and $f\left(x_{previous}\right)$ represent the fitness values of the current and previous iterations, respectively, and ϵ is a small tolerance value.

Another common stopping criterion is based on the maximum number of iterations allowed for the optimization process. This Criterion ensures that the algorithm terminates after a predefined number of iterations, regardless of the optimization progress. Mathematically, the termination condition can be expressed as Eq.(30).

$$Iteration < MaxIterations \quad (30)$$

where $Iteration$ represents the current iteration number, and $MaxIterations$ is the maximum allowable number of iterations.

Plateau detection involves monitoring whether the optimization process has reached a plateau where further iterations significantly fail to improve the solution. This Criterion may include tracking changes in fitness values over multiple iterations and identifying stagnation points. Mathematically, plateau detection can be based on criteria as defined in Eq.(31).

$$\frac{\left|f(x_{current}) - f\left(x_{previous}\right)\right|}{mean\left(f(x)\right)} < \eta \quad (31)$$

where η is a threshold parameter controlling the tolerance for detecting plateaus.

In addition to convergence criteria, diversity preservation criteria may be employed to ensure that the optimization process maintains diversity among solutions. This Criterion prevents premature convergence to suboptimal solutions by encouraging the exploration of diverse regions of the solution space. Mathematically, diversity preservation can be evaluated using Eq.(32).

$$Diversity = \frac{1}{N}\sum_{i=1}^{N} dist(x_i, x_{mean}) \quad (32)$$

where $\boldsymbol{N}$ is the total number of solutions, $\boldsymbol{x_i}$ represents the position of the $\boldsymbol{i}$th solution and $\boldsymbol{x_{mean}}$ is the mean position of all solutions.

Once the stopping criteria are evaluated, a termination decision is made based on whether any requirements are met. If any criterion indicates that further iterations are unnecessary, the optimization process is terminated, and the best solution is returned. Otherwise, the optimization process continues to the next iteration.

Algorithm 8: Stopping Criterion

Input:

- Current iteration number $\boldsymbol{Iteration}$
- The maximum allowable number of iterations $\boldsymbol{MaxIterations}$
- Tolerance value for convergence $\boldsymbol{\epsilon}$
- Threshold parameter for plateau detection $\boldsymbol{\eta}$
- Fitness values of current solutions $\boldsymbol{f(x_i)}$
- The mean position of all solutions $\boldsymbol{x_{mean}}$

Output:

- Termination decision (continue or terminate)

Procedure:

1. Check if the current iteration number is less than the maximum allowable number of iterations:
   - If true, proceed to step 2.
   - If false, terminate the optimization process and return.
2. Evaluate convergence criterion:
   - Calculate the absolute difference between the fitness values of current and previous iterations. $\boldsymbol{\left|f(x_{current}) - f(x_{previous})\right|}$.
   - Check if the absolute difference is less than the tolerance value $\boldsymbol{\epsilon}$.
   - If true, terminate the optimization process and return.
   - If false, proceed to step 3.
3. Evaluate plateau detection criterion:
   - Calculate the relative change in fitness values compared to the mean fitness value. $\boldsymbol{\frac{|f(x_{current}) - f(x_{previous})|}{mean(f(x))}}$.
   - Check if the relative change is less than the threshold parameter $\boldsymbol{\eta}$.
   - If true, terminate the optimization process and return.
   - If false, proceed to step 4.
4. Evaluate diversity preservation criterion:
   - Calculate the diversity among solutions based on their distances from the mean position: $\boldsymbol{Diversity = \frac{1}{N}\sum_{i=1}^{N} dist(x_i, x_{mean})}$
   - Check if the diversity value meets a predefined threshold.
   - If true, terminate the optimization process and return.
   - If false, continue to the next iteration.
5. Increment the current iteration number.
6. Repeat steps 1-5 until a termination decision is reached based on one of the criteria.

4.10. Repeating or Terminating the Optimization Process

This step involves deciding whether to continue the optimization process by repeating iterations or to terminate based on predefined criteria. In EFPA, the Monte Carlo method can be employed to inform this decision-making process, providing insights into the convergence and performance of the algorithm. The convergence criteria is assessed to determine whether the optimization process has sufficiently converged to an optimal solution. These criteria typically involve monitoring changes in fitness values or solution trajectories over successive iterations. Mathematically, convergence can be evaluated using equations Eq.(33) to Eq.(35).

$$\boldsymbol{\left|f(x_{current}) - f(x_{previous})\right| < \epsilon} \quad (33)$$

where $f(x_{current})$ and $f(x_{previous})$ represent the fitness values of the current and previous iterations, respectively, and ϵ is a small tolerance value.

$$Iteration < MaxIterations \quad (34)$$

where $Iteration$ represents the current iteration number, and $MaxIterations$ is the maximum allowable number of iterations.

$$Diversity = \frac{1}{N}\sum_{i=1}^{N} dist(x_i, x_{mean}) \quad (35)$$

where N is the total number of solutions, $x_i$represents the position of the ith solution and x_{mean} is the mean position of all solutions.

Then it involves in evaluating termination conditions to decide whether further iterations are necessary. These conditions may include convergence criteria, maximum iterations, plateau detection, or diversity preservation. The termination decision can be mathematically based on equations Eq.(36).

$$Terminate = \begin{cases} 1, & if\ convergence\ criteria\ are\ met \\ 1, & if\ maximum\ iterations\ reached \\ 1, & if\ plateau\ detected \\ 1, & if\ diversity\ not\ preserved \\ 0, & otherwise \end{cases} \quad (36)$$

where $Terminate$ is a binary variable indicating whether to terminate the optimization process (1 for terminate, 0 for continue).

Finally, a decision is made based on the evaluation of termination conditions to determine whether to repeat iterations or terminate the optimization process. This decision ensures the algorithm progresses efficiently towards optimal solutions while avoiding unnecessary computational burden. Mathematically, the decision-making process can be formulated as Eq.(37).

$$Decision = \begin{cases} Repeat, & if\ Terminate = 0 \\ Terminate, & if\ Terminate = 1 \end{cases} \quad (37)$$

where $Decision$ represents the final decision on whether to repeat iterations or terminate the optimization process.

Algorithm 9: EFPA

Input:

- Number of flowers N
- Maximum number of iterations $MaxIterations$
- Fitness function $f(x)$
- Search space boundaries x_{min} and x_{max}
- Perturbation parameter ϵ
- Fitness threshold τ
- Threshold parameter for plateau detection η

Output:

- Best solution x_{best}
- Fitness value of the best solution f_{best}

Procedure:

1. Initialize the positions of $\boldsymbol{N}$ flowers randomly within the search space boundaries.
2. Evaluate the fitness of each flower using the fitness function $\boldsymbol{f(x)}$.
3. Initialize the iteration counter $\boldsymbol{Iteration}$ to 0.
4. Repeat steps 5-13 until the termination condition is met.
5. Increment the iteration counter $\boldsymbol{Iteration}$ by 1.
6. Update each flower's pollen amount and attractiveness based on its fitness and neighbouring flowers.
7. Update the positions of all flowers using the perturbation parameter $\boldsymbol{\epsilon}$.
8. Evaluate the fitness of each flower using the fitness function $\boldsymbol{f(x)}$.
9. Select the best solutions based on fitness values above the threshold $\boldsymbol{\tau}$.
10. Check the stopping criterion to determine whether to terminate the optimization process.
11. If the termination condition is met, the best solution will be returned. $\boldsymbol{x_{best}}$and its fitness value $\boldsymbol{f_{best}}$.
12. If the termination condition is not met, repeat the optimization process from step 5.
13. Return the best solution. $\boldsymbol{x_{best}}$and its fitness value $\boldsymbol{f_{best}}$.

4.11. EFPA-based Routing Protocol

Efficient routing is essential for maximizing performance and prolonging the lifespan of IoT-WSN. Traditional routing protocols often face challenges in dynamically adapting to network changes and optimizing resource utilization. This study proposes a novel approach to routing optimization called EFPA-based Routing Protocol (EFPA-RP). Inspired by the natural process of flower pollination, EFPA offers a bio-inspired optimization technique capable of efficiently exploring solution spaces and adapting to dynamic network conditions. It is anticipated that routing paths can be optimized to minimize energy consumption, enhance data delivery reliability, and prolong the network lifetime by integrating EFPA into the routing framework of WSNs. This paper presents a comprehensive investigation into the application of EFPA as a routing protocol in WSNs, aiming to evaluate its effectiveness in improving routing efficiency and overall network performance. The significant steps included in EFPA-RP are:
Initialization: Initialize the network with sensor nodes and define the network topology.

Step 1: Encoding: Represent routing paths as "flowers" in the EFPA.

Step 2: Fitness Function: Define a fitness function to evaluate the quality of routing paths based on energy consumption, packet delivery ratio, and end-to-end delay.

Step 3: Perturbation: Introduce perturbation parameters to adjust the attractiveness of routing paths and promote exploration.

Step 4: Iterative Optimization: Apply EFPA iteratively to optimize routing paths based on their fitness values.

Step 5: Local Search: Incorporate local search mechanisms to refine routing paths and improve solution quality.

Step 6: Termination Criteria: Define termination criteria to determine when to stop the optimization process.

Step 7: Best Solution Selection: Select the best routing paths based on their fitness values.

Step 8: Adaptive Routing: Implement mechanisms for adaptive routing to adjust routing paths based on network conditions dynamically.

Step 9: Validation and Performance Evaluation: Validate the performance of the EFPA-based routing protocol through simulation experiments, evaluating key performance metrics such as energy consumption and packet delivery ratio.

5. RESULTS AND DISCUSSION

5.1. Simulation Setting

NS-3, a widely acclaimed network simulation tool, serves as the backbone for conducting simulations in this research. Renowned for its robustness and versatility, NS-3 provides an

extensive platform for modeling and analyzing complex network scenarios with unparalleled accuracy. Leveraging its comprehensive library of network protocols and models, researchers can replicate real-world environments and evaluate the performance of various routing protocols, including the proposed EFPA-RP, within IoT-WSN. NS-3's modular architecture enables researchers to customize simulations to suit specific research objectives, facilitating in-depth analysis and comparison of protocol performances under diverse conditions. NS-3's active community and extensive documentation offer invaluable support, ensuring researchers can navigate the simulation process efficiently. As a result, NS-3 emerges as an indispensable tool, empowering researchers to unravel insights and advancements in network optimization and protocol design with confidence and precision. Table 1 represents the simulation settings.

Table 1 Simulation Settings

Simulation Setting	Value
Transmission Range	200 m
Traffic Type	Event-Driven
Simulation Time	600 seconds
Simulation Environment	NS-3
Sensor Type	Temperature, Humidity, Light
Sensor Placement	Grid
Routing Protocol	AODV
Radio Frequency	2.4 GHz
Node Count	5000 nodes
Network Topology	Random
Network Size	1800 mts
Mobility Model	Random Waypoint
Energy Model	Battery
Data Aggregation Strategy	Maximum
Communication Protocol	IEEE 802.15.4
Battery Capacity	10 mAh

5.2. Packet Delivery and Drop Ratio Analysis

Table 2 and Figure 1 provides packet delivery ratio (PDR) analysis of EFPA-RP model with different nodes. The results assumed that the EFPA-RP algorithm has higher performance. With 500 nodes, the EFPA-RP system has got better PDR of 96.49%, while the SSEER and CRP models has gained decreased PDR of 53.47% and 60.33%, respectively. Additionally, with 1500 nodes, the EFPA-RP technique has got maximum PDR of 92.12%, whereas the SSEER and CRP methodologies has achieved lower PDR of 49.51% and 56.86%, correspondingly. Eventually, with 5000 nodes, the EFPA-RP approach has got superior PDR of 80.43%, whereas the SSEER and CRP models has got reduced PDR of 30.82% and 43.68%, correspondingly.

Table 2 Packet delivery ratio of EFPA-RP method with diverse nodes

Packet Delivery Ratio (%)			
time	SSEER	CRP	EFPA-RP
500	53.47	60.33	96.49
1000	51.45	58.14	95.96
1500	49.51	56.86	92.12
2000	47.12	54.61	91.63
2500	46.21	53.97	90.23
3000	41.74	50.97	88.57
3500	37.81	48.31	85.97
4000	33.45	47.28	84.78
4500	31.25	46.97	82.54
5000	30.82	43.68	80.43

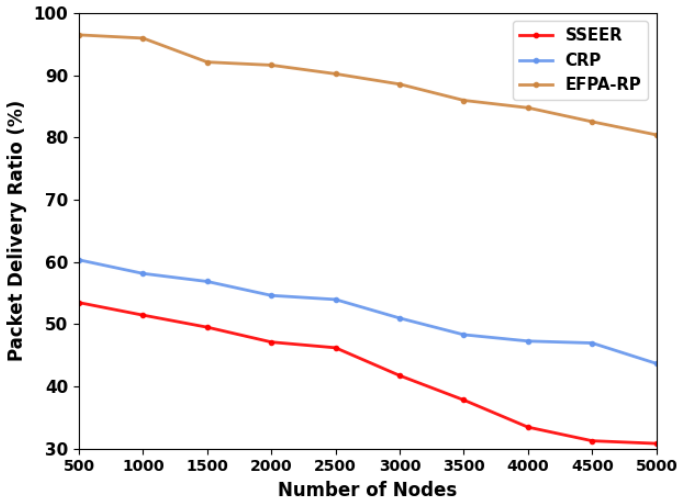

Figure 1 Packet delivery ratio of EFPA-RP method with diverse nodes

The packet drop ratio (PDRR) result of EFPA-RP technique with different nodes are demonstrated in Table 3 and Figure 2. The solutions supposed that the EFPA-RP algorithm has exceptional performance. With 500 nodes, the EFPA-RP method has attained minimal PDRR of 3.51%, although the SSEER and CRP models has developed larger PDRR of 46.53% and 39.67%, respectively. Also, with 1000 nodes, the EFPA-RP system has reached diminish PDRR of 4.04%, although the SSEER and CRP approaches has accomplished better PDRR of 48.55% and 41.86%, respectively. Moreover, with 2000 nodes, the EFPA-RP model has got decreased PDRR of 8.37%, although the SSEER and CRP techniques has got enhanced PDRR of 52.88% and 45.39%, correspondingly. Finally, with 4500 nodes, the EFPA-RP algorithm has attained lower PDRR of 19.57%, although the SSEER and CRP models has acquired maximum PDRR of 69.18% and 56.32%, respectively.

Table 3 Packet drop ratio outcome of EFPA-RP technique with various nodes

Packet Drop Ratio (%)			
No. of Nodes	SSEER	CRP	EFPA-RP
500	46.53	39.67	3.51
1000	48.55	41.86	4.04
1500	50.49	43.14	7.88
2000	52.88	45.39	8.37
2500	53.79	46.03	9.77
3000	58.12	49.03	11.43
3500	62.16	51.69	14.03
4000	66.55	52.72	15.22
4500	68.75	53.03	17.46
5000	69.18	56.32	19.57

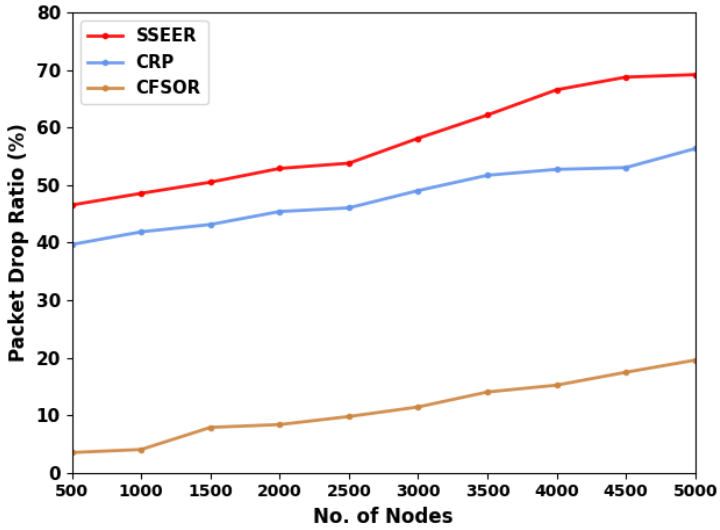

Figure 2 Packet drop ratio outcome of EFPA-RP technique with various nodes

5.3. Throughput Analysis

Table 4 and Figure 3 delivers throughput performance of EFPA-RP model with distinct nodes. The outcomes established that the EFPA-RP model has better performance. With 500 nodes, the EFPA-RP algorithm has got greater throughput of 82.33%, while the SSEER and CRP methods has obtained minimum throughput of 30.59% and 43.81%, correspondingly. Similarly, with 1000 nodes, the EFPA-RP method has got improved throughput of 82.98%, whereas the SSEER and CRP models has reached lower throughput of 31.03% and 44.35%, correspondingly. Eventually, with 1500 nodes, the EFPA-RP system has got innovative throughput of 85.71%, however the SSEER and CRP algorithms has attained slighter throughput of 32.74% and 45.87%, respectively.

Table 4 Throughput outcome of EFPA-RP technique with different nodes

Throughput (%)			
No. of Nodes	SSEER	CRP	EFPA-RP
500	30.59	43.81	82.33
1000	31.03	44.35	82.98
1500	32.74	45.87	85.71
2000	33.64	47.54	87.39
2500	35.12	49.51	88.94
3000	37.52	49.62	89.24
3500	38.13	50.24	90.15
4000	38.97	50.93	92.74
4500	39.15	51.37	94.21
5000	40.05	52.06	95.98

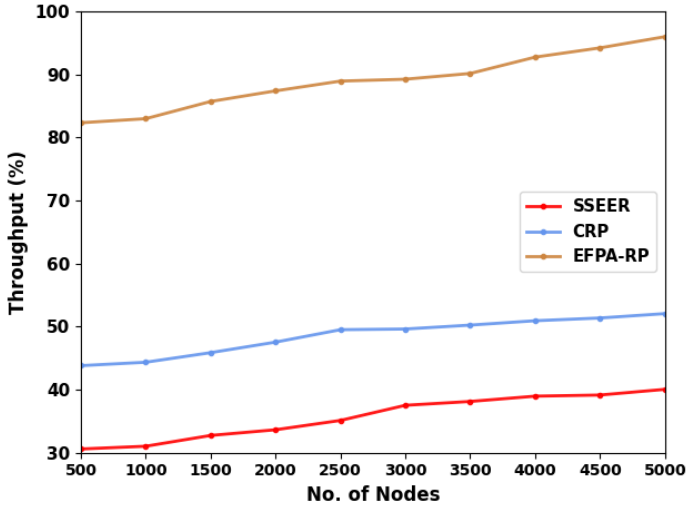

Figure 3 Throughput outcome of EFPA-RP technique with different nodes

5.4. Delay Analysis

Table 5 and Figure 4 illustrates the packet delay (PDEL) outcome of EFPA-RP algorithm with several nodes. The table values highlighted that the EFPA-RP technique has accomplish improved performance. With 500 nodes, the EFPA-RP method has got diminish PDEL of 1533ms, whereas the SSEER and CRP models has attained enhanced PDEL of 2547ms and 2089ms, correspondingly. Moreover, with 1500 nodes, the EFPA-RP system has got minimum PDEL of 1752ms, whereas the SSEER and CRP algorithms has accomplished maximum PDEL of 5231ms and 3675ms, respectively. Additionally, with 2500 nodes, the EFPA-RP model has got lower PDEL of 2145ms, while the SSEER and CRP techniques has reached maximum PDEL of 7642ms and 5237ms, respectively. At last, with 5000 nodes, the EFPA-RP method has got minimal PDEL of 4624ms, whereas the SSEER and CRP methodologies has attained improved PDEL of 14963ms and 13815ms, respectively.

Table 5 Packet delay outcome of EFPA-RP technique with various nodes

Packet Delay (ms)			
No. of Nodes	SSEER	CRP	EFPA-RP
500	2547	2089	1533
1000	3745	2857	1624
1500	5231	3675	1752
2000	6734	4127	1934
2500	7642	5237	2145
3000	9524	6895	2532
3500	10721	8521	2675
4000	11521	10245	2753
4500	13745	11854	3214
5000	14963	13815	4624

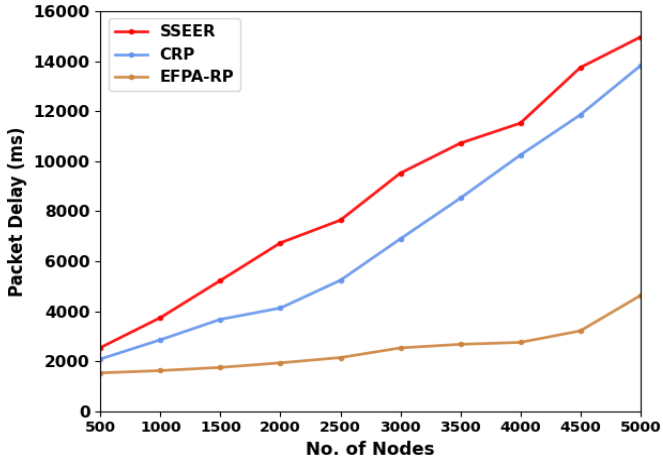

Figure 4 Packet delay outcome of EFPA-RP technique with various nodes

5.5. Energy Consumption Analysis

The energy consumption (EC) analysis of EFPA-RP model with different nodes are highlighted in Table 6 and Figure 5. The results concluded that the EFPA-RP system has greater performance. With 500 nodes, the EFPA-RP approach has got minimum EC of 16.981%, whereas the SSEER and CRP models has gained maximum EC of 77.468% and 59.352%, respectively. Further, with 1000 nodes, the EFPA-RP algorithm has got decrease EC of 19.943%, whereas the SSEER and CRP models has attained improved EC of 78.578% and 60.904%, correspondingly. Moreover, with 2000 nodes, the EFPA-RP system has got diminish EC of 22.541%, whereas the SSEER and CRP algorithms has gained superior EC of 81.421% and 64.612%, correspondingly. Lastly, with 4500 nodes, the EFPA-RP technique has got lower EC of 29.124%, whereas the SSEER and CRP models has obtained enhanced EC of 92.124% and 79.119%, respectively.

Table 6 Energy consumption outcome of EFPA-RP technique with distinct nodes

Energy Consumption (%)			
No. of Nodes	SSEER	CRP	EFPA-RP
500	77.468	59.352	16.981
1000	78.578	60.904	19.943
1500	79.521	62.845	20.457
2000	81.421	64.612	22.541
2500	83.521	66.871	23.740
3000	84.945	68.124	25.784
3500	87.546	70.842	27.854
4000	89.845	73.542	28.845
4500	92.124	77.321	29.124
5000	93.589	79.119	29.776

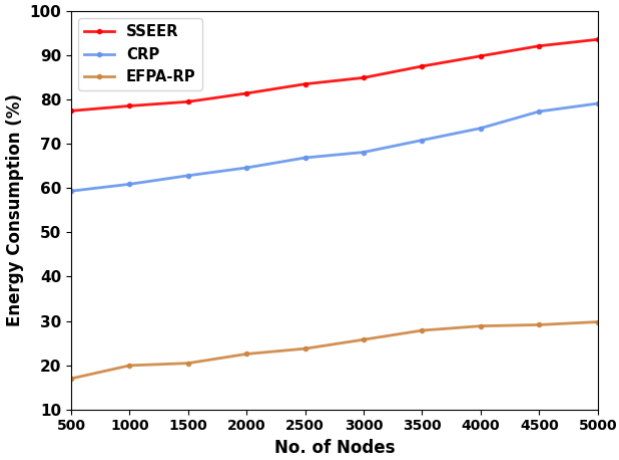

Figure 5 Energy consumption outcome of EFPA-RP technique with distinct nodes

5.6. Network Lifetime Analysis

Table 7 and Figure 6 offers network lifetime (NLT) study of EFPA-RP technique with dissimilar nodes. The table values established that the EFPA-RP model has higher performance. With 1000 nodes, the EFPA-RP algorithm has got enhanced NLT of 91.88%, whereas the SSEER and CRP techniques has obtained lesser NLT of 21.42% and 47.23%, respectively. Moreover, with 2000 nodes, the EFPA-RP approach has got superior NLT of 87.46%, while the SSEER and CRP models has accomplished reduced NLT of 18.58% and 43.39%, correspondingly. Lastly, with 3000 nodes, the EFPA-RP system has got advanced NLT of 84.22%, whereas the SSEER and CRP approaches has attained minimal NLT of 15.06% and 38.88%, respectively.

Table 7 Network lifetime outcome of EFPA-RP technique with diverse nodes

Network Lifetime (%)			
No. of Nodes	SSEER	CRP	EFPA-RP
500	22.53	48.75	93.56
1000	21.42	47.23	91.88
1500	20.48	45.16	89.54
2000	18.58	43.39	87.46
2500	16.48	41.13	86.26
3000	15.06	38.88	84.22
3500	12.45	35.16	82.15
4000	10.16	32.46	81.16
4500	9.88	27.68	79.88
5000	8.02	25.88	77.59

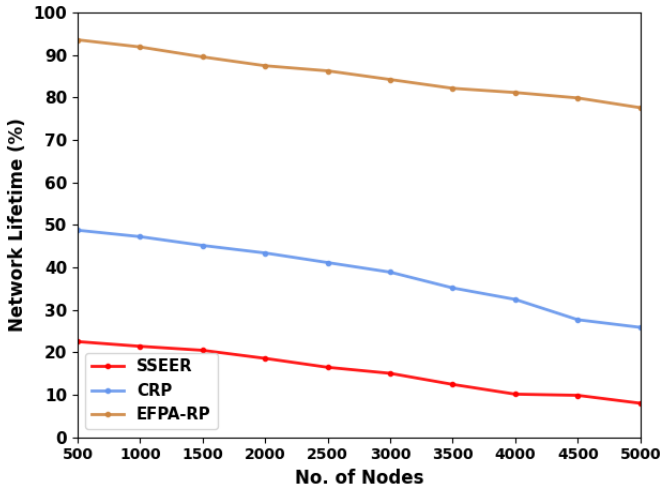

Figure 6 Network lifetime outcome of EFPA-RP technique with diverse nodes

6. CONCLUSION

The study on the EFPA-RP within IoT-WSN reveals a significant breakthrough in network optimization. Through rigorous experimentation, EFPA-RP demonstrates remarkable efficiency in energy consumption, throughput, and delay, aligning with its overarching goal of enhancing network longevity. The notable network lifetime results attained by EFPA-RP further solidify its position as a frontrunner in sustainable IoT-WSN deployments. Surpassing existing protocols, EFPA-RP extends the operational lifespan of IoT-WSN networks, ensuring robust data transmission and prolonged network functionality. These findings underscore EFPA-RP's potential to revolutionize IoT-WSN infrastructures, offering scalable and sustainable solutions for diverse applications. As IoT technologies evolve, EFPA-RP emerges as a beacon of innovation, heralding a new era of resilient and efficient network architectures. By embracing EFPA-RP, stakeholders can unlock the full potential of IoT-WSN technologies, fostering heightened connectivity and efficiency across various domains.

References

[1] A. Pagano, D. Croce, I. Tinnirello, and G. Vitale, “A Survey on LoRa for Smart Agriculture: Current Trends and Future Perspectives,” IEEE Internet Things J., vol. 10, no. 4, pp. 3664–3679, 2023, doi: 10.1109/JIOT.2022.3230505.

[2] M. A. Jamshed, K. Ali, Q. H. Abbasi, M. A. Imran, and M. Ur-Rehman, “Challenges, Applications, and Future of Wireless Sensors in Internet of Things: A Review,” IEEE Sens. J., vol. 22, no. 6, pp. 5482–5494, 2022, doi: 10.1109/JSEN.2022.3148128.

[3] M. R. Rahman, M. M. Islam, A. I. Pritom, and Y. Alsaawy, “ASRPH: Application Specific Routing Protocol for Health care,” Comput. Networks, vol. 197, p. 108273, 2021, doi: 10.1016/j.comnet.2021.108273.

[4] T. Theodorou and L. Mamatas, “A Versatile Out-of-Band Software-Defined Networking Solution for the Internet of Things,” IEEE Access, vol. 8, pp. 103710–103733, 2020, doi: 10.1109/ACCESS.2020.2999087.

[5] W. Bekri, R. Jmal, and L. Chaari Fourati, “Internet of Things Management Based on Software

Defined Networking: A Survey," Int. J. Wirel. Inf. Networks, vol. 27, no. 3, pp. 385–410, Sep. 2020, doi: 10.1007/s10776-020-00488-2.

[6] A. K. Gautam and R. Yadav, "Energy efficient hybrid routing protocol for Wireless Sensor Networks Using AI Technique," EAI Endorsed Trans. Scalable Inf. Syst., vol. 9, no. 35, 2022, doi: 10.4108/eai.3-11-2021.171757.

[7] S. A. Al-Ahmadi, "Counterfeit Clones: A Novel Technique for Source and Sink Location Privacy in Wireless Sensor Networks," IEEE Access, vol. 10, pp. 62693–62701, 2022, doi: 10.1109/ACCESS.2022.3182660.

[8] K. J. Co, A. V. Ong, and M. Peradilla, "WSN Data Collection and Routing Protocol with Time Synchronization in Low-cost IoT Environment," Procedia Comput. Sci., vol. 191, pp. 102–110, 2021, doi: 10.1016/j.procs.2021.07.016.

[9] W. Osamy, A. M. Khedr, A. Salim, A. I. Al Ali, and A. A. El-Sawy, "A review on recent studies utilizing artificial intelligence methods for solving routing challenges in wireless sensor networks," PeerJ Comput. Sci., vol. 8, 2022, doi: 10.7717/PEERJ-CS.1089.

[10] K. Jaiswal and V. Anand, "A Grey-Wolf based Optimized Clustering approach to improve QoS in wireless sensor networks for IoT applications," Peer-to-Peer Netw. Appl., vol. 14, no. 4, pp. 1943–1962, Jul. 2021, doi: 10.1007/s12083-021-01099-1.

[11] K. Haseeb, N. Islam, T. Saba, A. Rehman, and Z. Mehmood, "LSDAR: A light-weight structure based data aggregation routing protocol with secure internet of things integrated next-generation sensor networks," Sustain. Cities Soc., vol. 54, p. 101995, 2020, doi: 10.1016/j.scs.2019.101995.

[12] G. K. Ijemaru, L. M. Ang, and K. P. Seng, "Transformation from IoT to IoV for waste management in smart cities," J. Netw. Comput. Appl., vol. 204, p. 103393, 2022, doi: 10.1016/j.jnca.2022.103393.

[13] T. Shanmugapriya and K. Kousalya, "Cluster Head Selection and Multipath Routing Based Energy Efficient Wireless Sensor Network," Intell. Autom. Soft Comput., vol. 36, no. 1, pp. 879–894, 2023, doi: 10.32604/iasc.2023.032074.

[14] S. K. Gupta, S. Kumar, S. Tyagi, and S. Tanwar, "SSEER: Segmented sectors in energy efficient routing for wireless sensor network," Multimed. Tools Appl., vol. 81, no. 24, pp. 34697–34715, 2022, doi: 10.1007/s11042-021-11829-5.

[15] W. Chen, B. Zhang, X. Yang, W. Fang, W. Zhang, and X. Jiang, "C-EEUC: a Cluster Routing Protocol for Coal Mine Wireless Sensor Network Based on Fog Computing and 5G," Mob. Networks Appl., vol. 27, no. 5, pp. 1853–1866, 2022, doi: 10.1007/s11036-019-01401-9.

[16] T. R. Chenthil and P. Jesu Jayarin, "An energy-efficient distributed node clustering routing protocol with mobility pattern support for underwater wireless sensor networks," Wirel. Networks, vol. 28, no. 8, pp. 3367–3390, 2022, doi: 10.1007/s11276-022-03061-2.

[17] Z. Liu, X. Jin, Y. Yang, K. Ma, and X. Guan, "Energy-Efficient Guiding-Network-Based Routing for Underwater Wireless Sensor Networks," IEEE Internet Things J., vol. 9, no. 21, pp. 21702–21711, 2022, doi: 10.1109/JIOT.2022.3183128.

[18] M. Nagalingayya and B. S. Mathpati, "Energy-efficient cooperative routing scheme with recurrent neural network based decision making system for wireless multimedia sensor networks," Multimed. Tools Appl., vol. 81, no. 27, pp. 39785–39801, 2022, doi: 10.1007/s11042-022-12938-5.

[19] D. K. Bangotra, Y. Singh, A. Selwal, N. Kumar, and P. K. Singh, "A Trust Based Secure Intelligent Opportunistic Routing Protocol for Wireless Sensor Networks," Wirel. Pers. Commun., vol. 127, no. 2, pp. 1045–1066, 2022, doi: 10.1007/s11277-021-08564-3.

[20] T. Vaiyapuri, V. S. Parvathy, V. Manikandan, N. Krishnaraj, D. Gupta, and K. Shankar, "A Novel Hybrid Optimization for Cluster-Based Routing Protocol in Information-Centric Wireless Sensor Networks for IoT Based Mobile Edge Computing," Wirel. Pers. Commun., vol. 127, no. 1, pp. 39–62, 2022, doi: 10.1007/s11277-021-08088-w.

[21] A. Datta and M. Dasgupta, "Energy Efficient Layered Cluster Head Rotation Based Routing

Protocol for Underwater Wireless Sensor Networks," Wirel. Pers. Commun., vol. 125, no. 3, pp. 2497–2514, 2022, doi: 10.1007/s11277-022-09671-5.

[22] R. Singh and B. Sikdar, "A Low-Delay Routing-Integrated MAC Protocol for Wireless Sensor Networks," IEEE Internet Things J., vol. 9, no. 20, pp. 20561–20576, 2022, doi: 10.1109/JIOT.2022.3175913.

[23] C. Chen, L. C. Wang, and C. M. Yu, "D2CRP: A Novel Distributed 2-Hop Cluster Routing Protocol for Wireless Sensor Networks," IEEE Internet Things J., vol. 9, no. 20, pp. 19575–19588, 2022, doi: 10.1109/JIOT.2022.3148106.

[24] M. M. Saleem and S. A. Alabady, "Performance Analysis and Evaluation of a Multi-Hop Routing Protocol for Wireless Multimedia Sensor Networks," ECTI Trans. Electr. Eng. Electron. Commun., vol. 20, no. 3, pp. 392–402, 2022, doi: 10.37936/ecti-eec.2022203.247515.

[25] Y. Nam, H. Choi, Y. Shin, S. Park, and E. Lee, "Expected Area-Based Real-Time Routing Protocol for Supporting Mobile Sinks in Wireless Sensor Networks," Electron., vol. 11, no. 20, 2022, doi: 10.3390/electronics11203350.

[26] M. N. Syed and U. Baroudi, "Tactile Routing for Location Privacy Preservation in Wireless Sensor Networks: A Game Theoretic Approach," Sensors, vol. 22, no. 19, 2022, doi: 10.3390/s22197334.

[27] N. Alharbi, L. Mackenzie, and D. Pezaros, "Enhancing Graph Routing Algorithm of Industrial Wireless Sensor Networks Using the Covariance-Matrix Adaptation Evolution Strategy," Sensors, vol. 22, no. 19, 2022, doi: 10.3390/s22197462.