ISFO-CS: An Improved Sailfish Optimization Algorithm for Controller Selection in SDWSN

Manisha Kumari Rajoriya (manishakumarirajoriya23@gmail.com)
Rajasthan Technical University
Chandra Prakash Gupta
Rajasthan Technical University

Research Article

Keywords: Software Defined Wireless Sensor Network (SDWSN), Controller Selection Problem (CSP), Controller Nodes (CNs), clustering algorithm, data transmission, multi-objective fitness function

Posted Date: January 15th, 2024

DOI: https://doi.org/10.21203/rs.3.rs-3849743/v1

Additional Declarations: No competing interests reported.

ISFO-CS: An Improved Sailfish Optimization Algorithm for Controller Selection in SDWSN

[*1]Manisha Kumari Rajoriya, [2]Chandra Prakash Gupta

[*1]Research Scholar, Department of Computer Science & Engineering,
Rajasthan Technical University, Kota,
Rajasthan - 324010, India.

[2]Professor, Department of Computer Science & Engineering,
Rajasthan Technical University, Kota,
Rajasthan - 324010, India.
*Email: manishakumarirajoriya23@gmail.com

Abstract: Software-defined wireless sensor networks (SDWSNs) have recently been added to networking, increasing scalability and performance. Choosing SDN controllers is a critical issue for network administrators in SDNs. The control plane in SDN is a separate procedure that operates on the control layer. In terms of applications and services, the controller provides a comprehensive view of the entire network. The three key factors examined when selecting a controller are open source, campus network, and productivity. An appropriate device for the prompt processing of all switch requests is required for SDN to function properly and the network to behave properly. To determine the optimum controller for the given parameters, decision logic that allows controller comparison must be developed. As a result, an improved Sailfish Optimization (ISFO) method is proposed in this study to ideally take the best controller node (CN) from a group of sensor nodes. The ISFO selects the best CN by considering a multi-objective fitness function incorporating distance, residual energy, node coverage, and sensor node communication cost. After selecting CN, the Fuzzy C-means (FCM) Clustering algorithm executes the subsequent data transmission process. The Matlab program is used to execute the simulation, and the performance of the proposed methodology is assessed using several performance criteria. The proposed model accomplishes a 0.95% packet delivery ratio for 500 rounds and takes 1.8s for 500 rounds of node to controller latency.

Index terms: Software Defined Wireless Sensor Network (SDWSN), Controller Selection Problem (CSP), Controller Nodes (CNs), clustering algorithm, data transmission, multi-objective fitness function.

1. Introduction

In response to the multiple basic challenges that WSNs face today, many academics have proposed SDWSNs and SDN into WSNs [1]. SDWSNs are employed in a wide range of applications, including 5G networks, smart homes, smart grids, and more. They are capable of doing flexible network administration. The forwarding and control layers are kept apart in the SDWSN paradigm, and a logically centralized Controller oversees the management of the sensor nodes. Since the IoT is anticipated to play a substantial role in the sector, the objective of using SDN in WSN is to enhance usability and applicability [2]. While delivering flexibility, latent efficiency, innovations, and other benefits, this new networking paradigm was developed to tackle the inherent challenges and complexities of WSN [3]. Inefficient energy consumption, network architecture and management, scalability, security concerns, routing, mobility, and localization are a few of the difficulties [4].

These days, a single controller design or a distributed/multiple controller design might make up an SDN control plane. These control architectures can also be further divided into two categories: physically or theoretically centralized control architectures and distributed control architectures, each of which has a range of controller kinds [5]. However, in order to truly apply the SDN concept, an appropriate controller must be chosen. This can be a difficult decision because the number of controllers is growing, and it may be difficult to identify the right metrics. Despite these benefits, having several controllers introduces a number of challenges, one of which is the optimum controller placement problem (CPP) [6]. For example, adding more controllers or distributing them randomly around the network would not provide scalability because acceptable performance will not be achieved. This means that in order to satisfy varied needs, several controllers must be strategically placed in relevant regions [7].

In terms of scalability, the controller selection problem (CSP) and CPP are just as important as SDN for efficiently utilizing SDWSN services on a wide scale [8]. But because of the nature of the WSN, other objectives like node mobility and energy efficiency will also need to be considered by the CPP technique for the SDWSN [9]. In other words, while SDN-based CPP procedures are critical for installing SDWSN controllers, the methods may not entirely address all SDWSN-related issues. Furthermore, some placements are executed dynamically, while others are performed statically [10]. As a result, a number of different dynamic controller placement approaches, including virtualization, will be necessary for an SDWSN-based CPP to guarantee the CPP strategy's success. CSP is treated as the controllers' global point for the entire model in this study.

Furthermore, the controllers are in charge of computing activities such as resource allocation, traffic management, QoS, and route calculation [11]. The single controller cannot balance between networks and switches in large networks. To overcome this, multi-controllers with better scalability, reduced latency, dependability, and resilience are deployed [12]. Appropriate controller load balancing optimizes network evaluation. As more controllers are required to boost network availability and performance, CSP aims to select the finest possible collection of controllers for SDWSN. The following are the research's main contributions:

- ❖ To increase the performance of the SDWSN, multiple controllers are distributed and optimally selected for performing the data transmission process.
- ❖ To select the optimal number of controller nodes from the sensor nodes in SDWSN using an Improved Sailfish Optimization (ISFO) algorithm.
- ❖ To create a multi-objective-based fitness function including node coverage, residual energy, communication cost and node distance for performing the controller selection.

The research paper's following sections are arranged as follows: A few recent studies that are relevant to the suggested methodology are reviewed in Section 2. The multi-objective-based improved optimization technique and its operation are explained in Section 3. The performance of the suggested methodology is examined using several performance parameters in Section 4, along with the results of the simulation. Section 5 concludes the research article by summarizing its overall findings and outlining how this research will be enhanced in future.

2. Literature Review

This review covers some of the material that has been written about the suggested methodology.

Tahmasebi *et al.* [13] introduced the Cuckoo algorithm (CA) for the CPP. This algorithm was used for optimizing the latency of the network on the basis of the synchronization overhead. The work focused on reducing synchronization overhead and latency while improving the reliability of SDWSN. CA outperformed the conventional optimizations by minimizing the mean distance between controllers and sensors up to 9% and 13%. Samarji and Salamah [14] modelled CPP using a genetic algorithm (GA) and greedy-randomized adaptive search problem (G-RASP). Based on the controller's response time, the quantity of active nodes, the controller load, and the percentage of successfully received packets, the authors concluded that three SDN controllers are adequate. The flow of controllers increased load balancing between controllers, increased network longevity, and lowered latency. However, the authors did not address the issue of energy and controller overhead.

Two optimization algorithms were described by Firouz et al. [15]. Manta Ray Foraging (MRF) and Salp Swarm Optimization (SSO) were utilized to achieve the best CPP in SDN. The shortest path was determined using Dijkstra's method. To determine the separation between the switches, the Haversine formula was applied. The performance was tested by altering the number of controllers in six different types of networks. Shiny and Murugan [16] presented a T-SDN (threshold-SDN) protocol-based architecture to decrease control and data messages by considering the threshold. The threshold was calculated automatically using data from the server. PDR, controller reaction time, control message overhead, network lifetime time, and EC were all taken into account when analyzing the results. The fundamental disadvantage of a centralized controller was its limited bandwidth.

Jurado-Lasso et al. [17] presented an energy aware routing (EAR) paradigm to minimize controller interface and control overhead by utilizing the checksum function. The primary goal of the previous study was to reduce control overhead and extend network longevity. The primary function that drained energy was discovered, and the aggregation of data packets was decreased. The network's lifetime was increased by 6.5%, while overhead was reduced by 12%. However, because the controller was placed in one of the sensor nodes, its performance suffered.

Selecting the best controller in SDWSN is difficult. Manisha Kumari Rajoriya and Chandra Prakash Gupta [18] looked into an energy-efficient multi-objective meta-heuristic-based routing technique to find out which nodes to use as CNs. The method to solve the Controller Selection and Placement (CSPP) issue makes use of fitness, distance, and residual energy functions. The SFO technique is used in the MATLAB program to treat the problem as a multi-objective function. Both energy usage and transmission distance (delay) are decreased by the SFO-CS approach. Simulation experiments demonstrate that SFO-CS performs better than previous approaches in terms of packet delivery ratio (PDR), energy consumption, latency, and overall network lifetime.

In this work, Elham Hajian et al. [19] created a novel SDN architecture with the goal of improving durability and decreasing load distribution. The architecture comprises many components, including virtual routing, links, BS and controller discovery, and topology. Consequently, a unique virtualization and SDN-based load-balancing routing solution is suggested. In many IoT uses, the OpenFlow protocol implementation may ascertain load-balancing routing for every flow by directly analyzing connection load statistics and network performance. There are numerous ways in which a base station can accept flows from different resource applications. This technique reduces the amount of relevant data that is exchanged, like network status. For each Internet of Things (IoT) application, virtual routing chooses the optimal node by pre-weighing variables. The proposed method outperforms the LEACH, modified LEACH, and LEACH-C algorithms in terms of network lifetime and energy consumption balance, according to the simulation findings.

Sathyamoorthy Malathy et al. [20] provide a successful load-balancing and clustering method based on Q-learning and an improved K-Means algorithm. It is divided into two phases: node balancing and clustering. The suggested method employs the Q-learning algorithm to select a CH and place sensor nodes in relevant clusters. The clustering method involves grouping the nodes together according to the average values that have been determined. Separating the cluster into 'k' pieces follows the proper placement of sensors within each cluster. Taking into account throughput, end-to-end delay, packet delivery ratio, and energy consumption, the suggested clustering algorithm based on Q-Learning maximizes the reward. Finally, compare and contrast the state-of-the-art k-means-based clustering methodology with the Q-learning-based clustering approach to determine its effectiveness. A 1.56% improvement in packet delivery ratio, a 3.34% increase in network lifespan, an 8.23% drop in end-to-end latency, and a 2.34% rise in throughput are all results of the proposed strategy.

Xiao Yan et al. [21] developed an energy-efficient clustering approach (GEC) for WSNs based on game theory, in which each sensor node represents a player in the game. By keeping track of how much time it spends listening in its active state before opting to sleep or not, the sensor node can determine its optimal techniques. To prevent sensor nodes from acting selfishly in the future, a punishment system can be set up to encourage them to work together. According to the results of the simulation, using game theory to the sensor network may increase data transmission throughout it while decreasing energy consumption, achieving the objective of prolonging its lifespan.

Shirin Tahmasebi et al. [22] characterized the CPP as a multi-objective optimization problem and developed the Cuckoo optimization (CO) population-based meta-heuristic technique. This algorithm mimics the brood parasitism of different species of cuckoo birds in an attempt to find the global optimum. Finally, a comparison with a range of other techniques is used to assess the suggested methodology. The results of the studies show that the recommended technique beats the most modern methods in terms of synchronization cost and performance, notably Quantum Annealing (QA) and Simulated Annealing (SA). Furthermore, the proposed method is much more scalable than ILP (integer linear programming).

2.1 Problem description and motivation

The SDWSN paradigm employs network programmability to ensure network flexibility, efficiency, and innovation, thereby overcoming the core challenges of traditional networks [3]. However, when it comes to the deployment of many or dispersed controllers, there are challenges with the networking paradigm. There are several significant concerns to consider when designing an effective and dependable distributed SDN controller platform, including scalability, consistency, reliability, and interoperability [13-15]. SDN scalability is also affected by a number of other characteristics, including flow setup latency, consistency, controller placement, and controller failure. Among these challenges, controller location posed a considerable impediment to meeting other critical network requirements and SDN scalability. This is known as the CSP, and numerous approaches for large-scale SDWSN networks have been developed or proposed in this context. Several studies were conducted and several methodologies were found, such as heuristic-based techniques, cluster-based approaches, bio-inspired techniques [18], greedy algorithms, and others. Network portioning or clustering approaches dominate the CPP method. These are followed by heuristic-based, mathematical model-based, biologically inspired Linear Programming (LP) strategies. In addition, each approach optimizes its performance metrics through reduction or maximization in order to get the desired Quality of Service (QoS) and network performance. While some CPP techniques took into account single objectives like cost, latency, reliability, etc., others took into account multi-objectives in their optimization model.

Furthermore, the placement and choice of controller greatly minimize load and latency on the network. No published research has been done on the analysis of SDWSNs with several controllers distributed throughout the network using multi-objective optimization approaches. For instance, the authors in [18] adopt the SFO approach for CSP with the node distance and residual energy as their two main objectives. Additionally, in order to conduct controller placement in SDWSN, authors in [22] devised a fitness function with only two objectives: synchronization cost and node distance. This encourages the development of a better multi-objective optimization method to choose the best nodes to be CNs.

3. Proposed Methodology

A novel network architecture called SDWSN was created to tackle the challenges that arise when managing conventional networks like SDN and WSN. The architecture offers two deployment options: either a single controller or several controllers. While the former is not appropriate for extensive networks, the latter faces a Constraint Satisfaction Problem (CSP) and a Constraint Programming Problem (CPP) in the context of a large-scale network. The difficulty with CPP and CSP is finding out how to divide up and choose a network's controller count wisely in order to meet conflicting performance demands such computation time, latency, load balancing, dependability, and energy economy. In CSP, it may be impossible to find a single optimum or random alternative; consequently, extensive planning is required to determine the best trade-off between the metrics. An ISFO algorithm is designed to pick ideal nodes as CNs from a large number of sensor nodes. In addition, a multi-objective fitness function with node distance, communication cost, residual energy, and node coverage is designed to pick the node as a CN optimally. The FCM method is then used to execute the clustering and data transfer processes. Figure 1 depicts the workflow of the research paper.

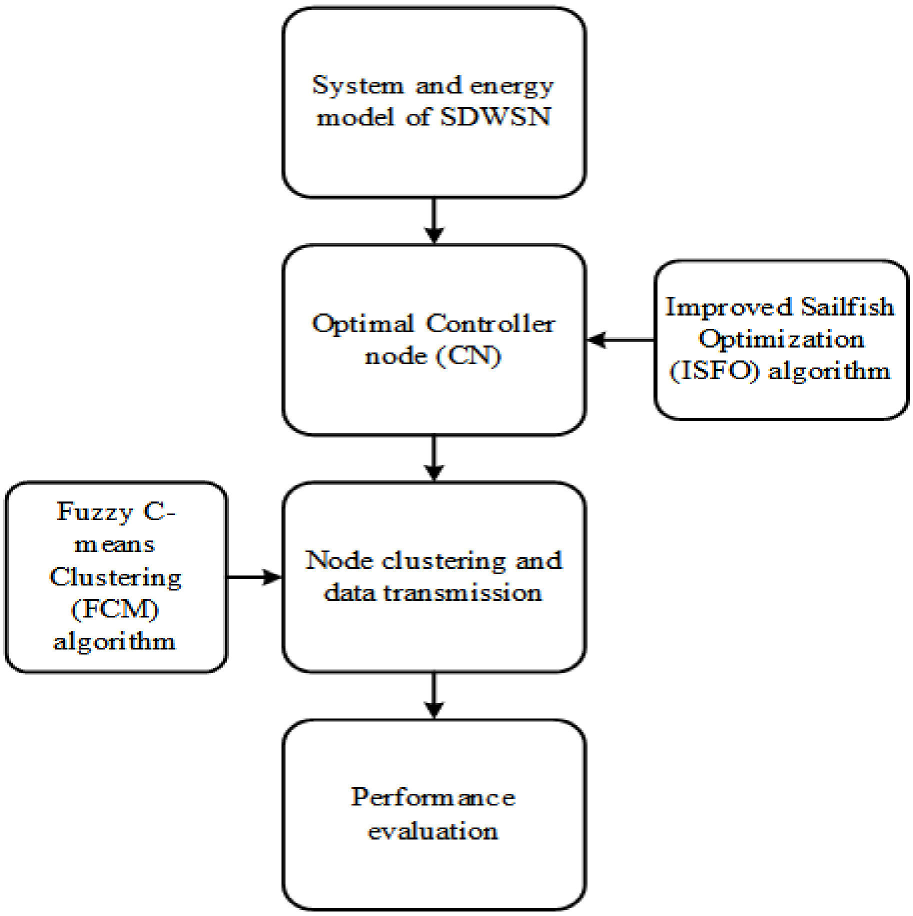


Figure 1: Workflow model of the proposed methodology for CSP

3.1 System model and energy model

In an operational WSN architecture, an SDWSN model is seen as a digraph $Gn = (V, L)$. The vertex set in the model is designated by V and consists of control nodes (CNs), sink nodes, and software-defined WSNs (common nodes), as well as control servers (CSs). They are dispersed at random within the designated monitoring region. The directed transmission collection or communication connection that is dedicated to transmitting the gathered data from common nodes to the CN is indicated by the L in the graph.

Energy dissipation: Based on the route loss idea, the most popular data transmission model under consideration is SDWSN, which includes channel usage for multipath (E_{mp}), fading (d^4 power loss), and free space (E_{fs}) (d^2 power loss). The model's energy consumption is a function of the separation (d) between two entities. The coordinates for the transmitter are (Xi, Yi), and the coordinates for the receiver are (Xj, Yj). The Euclidean distance formula, which may be used to compute the distance, is as follows:

$$(Xj, Yj) = \sqrt{(Xj - Xi)^2 + (Yj - Yi)^2} \tag{1}$$

When the estimated d is below the threshold (d_0), this work employs the power control mechanism and compensates for the route loss concept using either the free space model or the multipath model. The following equation can be used to determine the energy dissipation to transmit data for the common SDWSN node ($ETXN_{SDWSN}$) in order to convey a l-bit message over a d -distance:

$$E_{TXN_{SDWSN(l,d)}} = \begin{cases} k \times E_{elec} + k \times E_{fs} \times d^2 d \leq d_0 \\ k \times E_{elec} + k \times E_{mp} \times d^4 d > d_0 \end{cases} \tag{2}$$

The energy transmission for the control node of l bit data packet is as follows:

$$E_{TXN_{CN(l,d)}} = \begin{cases} k \times (E_{elec} + E_{DA}) + k \times E_{fs} \times d^2 d \leq d_0 \\ k \times (E_{elec} + E_{DA}) + k \times E_{mp} \times d^4 d > d_0 \end{cases} \tag{3}$$

The $ETXN$ is recognized for the energy needed for transmission; the control node uses energy EDA for data aggregation. On the other hand, d defines the distance between two SDWSN sensor units or between nodes and the control server. The energy required to run a transmitter or receiver circuit is denoted by the symbol E_{elec}. It is dependent on a number of factors, including signal dispersion, modulation, source coding, and filtering. E_{fs} plus $E_{mp}$are dependent on the model of transmitter amp. Here, (d_0), the transmission distance threshold, is commonly stated as follows, and l indicates the size of the packet to be sent.

$$d_0 = \sqrt{\frac{E_{fs}}{E_{mp}}} \tag{4}$$

To carry out the data transmission procedure, the radio transmitter uses the following energy: $E_{TXN}(l) = l * E_{elec}$ [23]. The symbols and their meanings utilized in the proposed model are listed in Table 1.

Table 1: Used symbols and its explanations

Symbols	Explanation
Gn	Diagraph
V	Vertex set
L	Communication link
E_{fs}	Free space model
d^4	Fading channel
E_{mp}	Energy model for multipath
d	Distance or dimension of the variable
i	Transmitter
j	Receiver
$ETXN_{SDWSN}$	Energy dissipation to transmit the data
l	Data packet
$E_{TXN_{CN(l,d)}}$	The energy required for transmission
d_0	Threshold value for transmission distance
m	Number of sailfish

t	Number of the current iteration
$F_{SF}^{(t)}$	Fitness matrix of the sailfish
$F'^{(t)}_{i}$	Fitness value of the i^{th} sailfish
$F_{S}^{(t)}$	Fitness matrix of the sardine
$X_{elite}^{(t)}$	Position of elite sailfish
$X'^{(t)}_{injured}$	Position of injured sardine
$X_{elite}^{(t-1)}$	Elite sailfish in the $t-1$ iteration
$X'^{(t)}_{injured}$	Sardine, who was seriously injured in the $t-1$ iteration
$X_{i}^{(t-1)}$	i^{th} sailfish in the $t-1$ iteration
$X_{i}^{(t)}$	i^{th} sailfish in the t iteration
$rand$	Random value
λ_i	Update coefficient
$M_{SF}^{(t)}$	Number of sailfish in the t iteration
$M_{S}^{(t)}$	Number of sardines in the t iteration
A and ξ	Attack strength coefficients
γ	Number of updated sardines
η	Number of variables
u	Weight inertia
u_{min} and u_{max}	Lower and upper limits of weight inertia u
r	Random variable
S_{Levy}	The step size of the Levy flight
τ	Constant value
Γ	Gamma function
RE	Residual energy
e_{sn}	The total energy of sensor node
e_{co}	Energy used for data collection
e_{recp}	Energy used for data transmission
NC	Node coverage
$R(X_y)$	Radius of sensor node
NP	Node proximity
$N_n(x,y)$	Distance
CC	Communication cost
d_{avg}^2	The average distance of the node
d_r^2	Average distance of the nearest node
w_i	Weight value
c	Clusters
u_{ij}	Node j's degree of belonging to cluster i
d_{ij}	Distance between node j and the centre point of cluster i
m	Fuzzified number

3.2 Optimal CN selection using ISFO algorithm

This section discusses the CSP using the IFSO algorithm in detail. The following subsection briefly explains the working process of the proposed methodology in detail.

3.2.1 Basics of SFO algorithm

According to Shadravan et al. (2019) [24], SFO is a brand-new, hyper-heuristic optimization technique that draws inspiration from nature. The approach is composed of two populations: one that diversifies the search space and improves search capabilities, and the other that boosts search skills.

- ***Initialization of the location of the sardine and sailfish populations***

SFO considers sailfish as a solution. The sardine population is specified by many solutions. Sailfish and sardine locations in the search space solve the controller selection problem. This generates a random population in solution space. Initialized sailfish position matrix:

$$X_{SF}^{(u)} = \begin{bmatrix} X_{1,1}^{(u)} & X_{1,2}^{(u)} & X_{1,d}^{(u)} \\ X_{2,1}^{(u)} & X_{2,2}^{(u)} & X_{2,d}^{(u)} \\ \vdots & \vdots & \vdots \\ X_{m,1}^{(u)} & X_{m,2}^{(u)} & X_{m,d}^{(u)} \end{bmatrix} \tag{5}$$

where $X_{ij}^{(u)}$ represents the value of the j^{th} dimension of the i^{th} sailfish, u is the number of current iterations, d is the variable's dimension, and m denotes the quantity of sailfish in population. Following the sardine position initialization, the matrix is displayed in the equation below.

$$X_{S}^{(u)} = \left[X_{i,j}^{(u)}\right]_{nxd} = \begin{bmatrix} X_{1,1}^{(u)} & X_{1,2}^{(u)} & X_{1,d}^{(u)} \\ X_{2,1}^{(u)} & X_{2,2}^{(u)} & X_{2,d}^{(u)} \\ \vdots & \vdots & \vdots \\ X_{n,1}^{(u)} & X_{n,2}^{(u)} & X_{n,d}^{(u)} \end{bmatrix} \tag{6}$$

There are n sardines in the population, with $X_{i,j}^{(u)}$ representing the j^{th} dimension of the i^{th} sardine. The algorithm's fitness function F is determined by solving the sailfish and sardine solution's fitness with the SDWSN problem's objective function to assess fish and sardine quality. The results are stored as a matrix, and the sailfish fitness matrix is displayed in the equation below.

$$F_{SF}^{(u)} = \begin{bmatrix} F_{1}^{(u)} \\ F_{2}^{(u)} \\ \vdots \\ F_{m}^{(u)} \end{bmatrix} = \begin{bmatrix} F(X_{1,1}^{(u)} & X_{1,2}^{(u)} & X_{1,d}^{(u)}) \\ F(X_{2,1}^{(u)} & X_{2,2}^{(u)} & X_{2,d}^{(u)}) \\ \vdots & \vdots & \vdots \\ F(X_{m,1}^{(u)} & X_{m,2}^{(u)} & X_{m,d}^{(u)}) \end{bmatrix} \tag{7}$$

where m is the sailfish quantity, $F'^{(u)}_{i}$ is the fitness value of the i^{th} sailfish, and $F_{SF}^{(u)}$ is the sailfish's fitness matrix. Sardines' fitness matrix is displayed as follows:

$$F_{S}^{(u)} = \begin{bmatrix} F_{1}^{(u)} \\ F_{2}^{(u)} \\ \vdots \\ F_{n}^{(u)} \end{bmatrix} = \begin{bmatrix} F(X_{1,1}^{(u)} & X_{1,2}^{(u)} & X_{1,d}^{(u)}) \\ F(X_{2,1}^{(u)} & X_{2,2}^{(u)} & X_{2,d}^{(u)}) \\ \vdots & \vdots & \vdots \\ F(X_{n,1}^{(u)} & X_{n,2}^{(u)} & X_{n,d}^{(u)}) \end{bmatrix} \tag{8}$$

Where n is the number of sardines, $F_{i}^{\prime(u)}$ is the fitness value of the i^{th} sardine, and $F_{S}^{(u)}$ is the sardine fitness matrix.

- ***Elite selection strategy***

The elite sailfish in the SFO algorithm's iterative phase is the one with the lowest fitness, and its location is noted as $X_{elite}^{(u)}$. Sardines are also susceptible to injury by sailfish; badly damaged sardines are located at $X'^{(u)}_{injured}$, and their fitness is the lowest in the sardine population.

- ***Alternate attack strategy***

When hunting, sailfish change their location relative to other sailfish that are near sardines. The following formula is displayed to adjust the Sailfish position:

$$X_i^{(u)} = X_{elite}^{(u-1)} - \lambda_i^{(u-1)}\left(rand * \left(\frac{X_{elite}^{(u-1)} - X'^{(u)}_{injured}}{2}\right) - X_i^{(u-1)}\right) \tag{9}$$

where $X_{elite}^{(u-1)}$ represents the elite sailfish in the $u-1$ iteration, $X'^{(u)}_{injured}$, symbolizes the sardine gravely hurt in $u-1$, $X_i^{(u-1)}$ represents the ith sailfish in the $u-1$iteration, and $X_i^{(u)}$ represents the i^{th} sailfish in the u iteration. The update coefficient is represented as follows:

$$\begin{cases} \lambda_i^{(u)} = 2 * rand * D^{(u)} - D^{(u)} \\ D^{(u)} = 1 - \left(\frac{M_{SF}^{(u)}}{M_{SF}^{(u)} + M_S^{(u)}}\right) \end{cases} \tag{10}$$

where $M_{SF}^{(u)}$ is the sailfish quantity at the iteration of t and the sardines quantity is given as $M_S^{(u)}$ for the u iteration.

- ***Evasion strategy***

Sardines will move to avoid the elite sailfish by assessing its location and attack ferocity with each repetition. The sardine position update equation is:

$$\begin{cases} X'^{(u)}_i = rand(X_{elite}^{(u-1)} - X_i^{(u-1)} + Q^{(u-1)}) \\ Q^{(u-1)} = A * -(1 - (2(u-1)\xi) \end{cases} \tag{11}$$

In the current iteration, u is the number of iterations, $X'^{(u-1)}_i$ is the i^{th} sardine in the $u-1$ iteration, $X'^{(u)}_i$is the i^{th} sardine in the u iteration, $Q^{(u-1)}$ is the attack strength of sailfish in the $u-1$ iteration, A and ξ are coefficients with values of 4 and 0.001, and $Q^{(u-1)}$ is the sailfish attack strength The assault strength of sailfish affects both the quantity of updated sardines γ and variables η.

$$\begin{cases} \gamma^{(u)} = M_s^{(u)} * Q^{(u)} \\ \eta^{(u)} = d_i^{(u)} * Q^{(u)} \end{cases} \tag{12}$$

In this case, $d_i^{(u)}$ indicates the number of the i^{th} sardine variables in the u iteration, while $M_s^{(u)}$ indicates the number of sardines in the u iteration.

- ***Hunting strategy***

When the sardine's fitness is lower than the sailfish's, the sailfish's position replaces the captured sardine's, as shown by equation (13).

$$X_i^{(u)} = X_i^{(u)} \quad if \;\; F\left(X_i^{(u)}\right) > F'\left(X'^{(u)}_i\right) \tag{13}$$

In this case, the positions of the sailfish and sardine in the u iteration are represented by the variables $X_i^{(u)}$ and $X'^{(u)}_i$, respectively.

3.2.2 ISFO

In this research, the SFO algorithm is improved by adopting three different concepts, including Inertia weight, global search formula and Levy flight strategy.

- ***Inertia weight***

Weight inertia iw is included in the SFO algorithm's alternating attack and pursuit phase to improve the local search efficiency of sardines and sailfish. Based on this, the location update equation is displayed as follows:

$$\begin{cases} X_i^{(u)} = iw(u-1) * X_{elite}^{(u-1)} - \lambda_i^{(u-1)} \left(rand * \left(\frac{X_{elite}^{(u-1)} - X'^{(u)}_{injured}}{2} \right) - X_i^{(u-1)} \right) \\ X'^{(t)}_i = rand(iw(u-1) * X_{elite}^{(u-1)} - X'^{(u-1)}_i + Q^{(u-1)}) \end{cases} \quad (14)$$

where the variable $rand$ denotes random values, which span from 0 to 1. The following is the expression for weight inertia:

$$iw(t) = iw_{max} + (iw_{max} - iw_{min}) * exp\left(-25 * \left(\frac{u}{T}\right)\right)^3 \quad (15)$$

where t and T stand for the current and maximum number of iterations, and iw and iw_{max} for the lower and upper bounds of weight inertia iw ($iw = 0.4$, $iw_{max} = 0.9$.).

- ***Global search formula***

Throughout the population location update process, the global search formula is employed to enhance the global search performance of the SFO algorithm while preventing it from transitioning into local optimization. The global search can be stated numerically as:

$$\begin{cases} X_i^{(u+1)} = X_i^{(u)} + (X_{elite}^{(u)} - X_i^{(u)}) * \exp\left(F\left(X_{elite}^{(u-1)}\right) - F\left(X_i^{(u)}\right)\right) r \\ X'^{(u+1)}_i = X'^{(u)}_i + (X_{injured}^{(u)} - X'^{(u)}_i) * \exp\left(F'\left(X'^{(u)}_{injured}\right) - F'\left(X'^{(u)}_i\right)\right) r \end{cases} \quad (16)$$

where r is a random value between 0 and 1.

- ***Levy flight strategy***

The Levy flight method is a random search technique that adheres to the Levy distribution and broadens the population of algorithms. The equation given below represents the Levy flight technique.

$$x_i^{(u+1)} = x_i^{(u)} + S_{Levy} \quad (17)$$

where the sailfish or sardine's position in the u iteration is denoted by $x_i^{(u)}$. The Levy flight's step size, or S_{Levy}, is represented in the equation below.

$$\begin{cases} S_{Levy} = \frac{rand * \zeta}{|rand|^{1/\tau}} \\ \zeta = \left(\frac{\Gamma(1+\tau) * \sin\left(\tau\pi/2\right)}{\Gamma\left(\frac{1+\tau}{2}\right) * \tau * 2^{\left(\frac{\tau-1}{2}\right)}} \right)^{1/\tau} \end{cases} \quad (18)$$

where τ is a constant and the value is 1.5, Γ is the gamma function. Algorithm 1 depicts the steps involved in the selection of optimal CNs using the ISFO algorithm.

Algorithm 1: Selection of CN using multi-objective-based ISFO algorithm
Input: N number of SNs, initial population and max iterations. **Output:** Selection of optimal node as 'CN'
Initialize all the algorithm parameters Randomly initialize the sardine and sailfish population. ***for*** i = 1 to N ***do*** calculate the fitness values for sailfish and choose severe sardine to record the global optimal solution. ***While*** (Selecting the CN)

if rand>0.5

Update the location of sailfish using the inertia weight formula in equation (14).

else

Update the location of sailfish using the global search formula in equation (16).

if Q<0.5

Update the location of the sardine using the inertia weight formula in equation (14).

else

Update the location of sailfish using the global search formula in equation (16).

end if

end if

Perform levy flight operation for updated sailfish and sardine using equation (17).

Compute the fitness value using equation (27)

Sort the fitness function

Update the elite fish and seriously injured sardines to record the global optimal solution.

end for

if max iteration reached (T<u)

Output the global optimal sailfish (CN)

else

Repeat the process until the optimal solution is obtained.

end if

Return optimal CNs

3.2.3 Fitness function calculation

Using a number of variables, including research into communication costs, distance, node coverage, and residual energy, the fitness function selects the sensor nodes that are most suited to serve as the CN. The sensor node's residual energy is the initial goal, which is determined using the equation below.

$$RE = e_{sn} - (e_{co} + e_{trans} + e_{recp}) \tag{19}$$

Here, e_{sn}, e_{co}, e_{trans} and e_{recp} signifies the total energy of the sensor node, energy used for data collection, energy utilized for data transmission and energy used for data reception, respectively. Finally, this is mentioned in the equation given below.

$$Max\ (F_1(x)) = \frac{1}{SN_t} \sum_{y=1}^{x} RE(X_y) \tag{20}$$

Here, SN_t mentions the number of sensor nodes. The second objective function is node coverage, which is calculated based on the equation given below.

$$NC = R(X_y) \tag{21}$$

Here, $R(X_y)$ is the radius of the sensor node, which is mentioned as the second objective, and it is mathematically expressed as follows:

$$Max\ (F_2(x)) = \frac{1}{SN_t} \sum_{y=1}^{x} NC(X_y) \tag{22}$$

The third objective function is distance, which is also known as node proximity, and it is computed based on the equation given below.

$$NP = \frac{1}{SN_t} \sum_{y=1}^{1-SN_t} N_n(x, y) \tag{23}$$

Here, the distance between the nodes is mentioned as $N_n(x, y)$ and the third objective function is defined in the given equation below.

$$Min\ (F_3(x)) = \frac{1}{SN_t} \sum_{y=1}^{x} NP(X_y) \tag{24}$$

Finally, communication costs are considered the fourth objective function. This function calculates the cost of the node used to communicate with the neighboring node.

$$CC = \frac{d_{avg}^2}{d_r^2} \tag{25}$$

The node's average distance is represented by d_{avg}^2 , while the nearest node's average distance is represented by d_r^2. Based on this, the fourth objective function is computed based on the equation given below.

$$Min\ (F_4(x)) = \frac{1}{SN_t} \sum_{y=1}^{x} CC(X_y) \tag{26}$$

The preceding equations show that the first two aims are in contradiction with the next two. Because two are minimal functions, and two are maximum functions that should be avoided. For that, the objection function should be either maximum or minimum. In this research, the final objective function is to be a maximum value, and based on that, the final fitness function is calculated using equation (27).

$$F(x) = max\ \frac{(w_1F_1(x)+w_2F_2(x))}{(w_3F_3(x)+w_4F_4(x))} \tag{27}$$

Since the user is expected to supply the weights, the approach is known as the priori approach here, where the value of $\sum w_i = 1$. The suggested ISFO algorithm's flowchart for the best CN selection in SDWSN is shown in Figure 2.

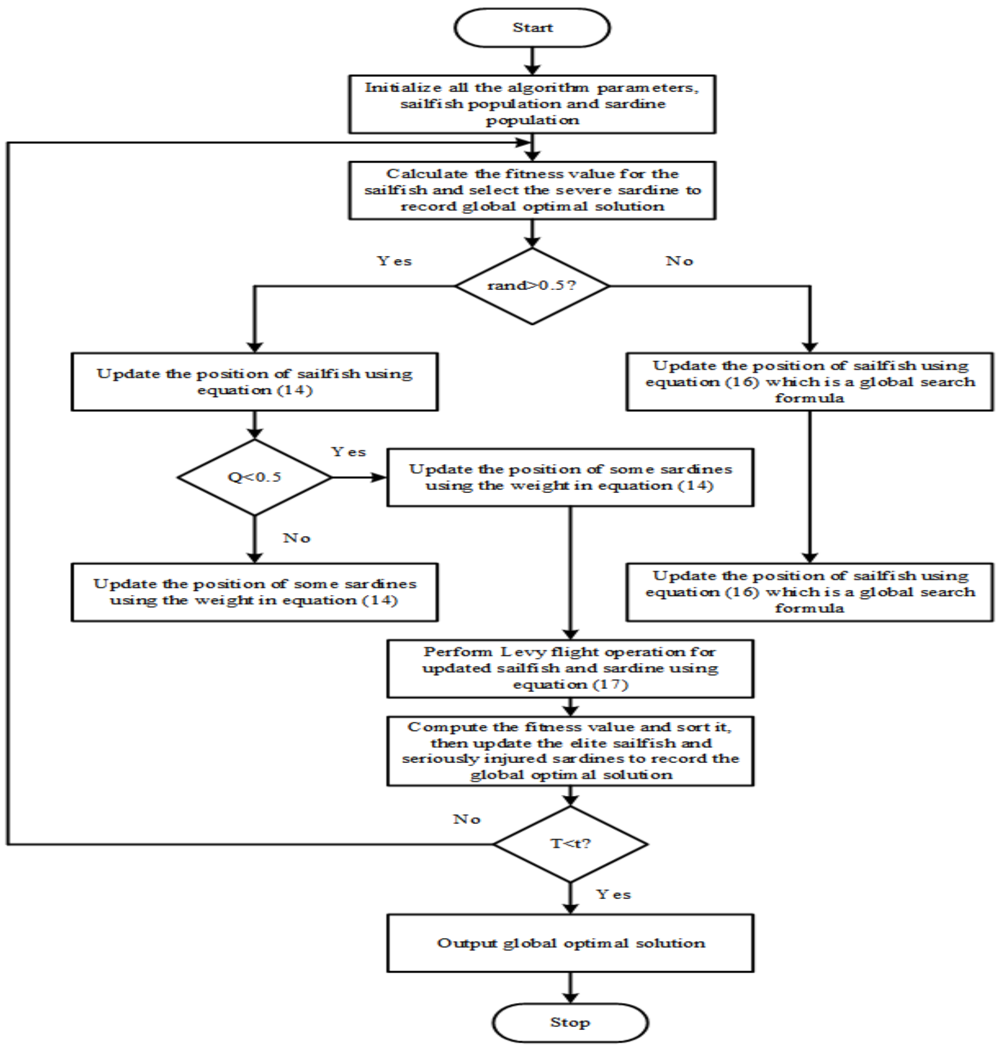


Figure 2: Flowchart of the proposed model for CNs selection in SDWSN

3.3 FCM algorithm for node clustering

The clustering technique has two stages: setup and "data transmission." The tasks performed during the setup phase include cluster construction and CH selection. The sensor nodes in each cluster supply the CHs with data during the data transmission stage. The CHs then combine and fuse the data, compress it, and send it to the BS. The cluster in this study is constructed using the FCM methodology. A centralized clustering mechanism is the FCM clustering protocol. After the BS computes and clusters sensor nodes according to their geographic coordinates, the sensor node with the highest residual energy inside the cluster is assigned the CH position. N sensor nodes arranged in a network into c clusters: $C_1, C_2, \ldots C_c$. The following objective function was reduced using the cluster creation technique.

$$J_m = \sum_{i=1}^{c} \sum_{j=1}^{N} u_{ij}^{m} d_{ij}^{2} \tag{28}$$

where m is the fuzzified parameter, d_{ij} is the distance between node j and the cluster i center point, and $u_{ij}$is the node j's degree of cluster i membership. The following formula computes and fuzzifies the degree u_{ij} of node j with regard to cluster i using the real number $m > 1$.

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}}\right)^{2/(m-1)}} \tag{29}$$

Sensor node to center point distance is Euclidean distance. Limiting geographic distance optimizes node cluster dispersion and energy balance.

Phase1: Cluster formation

In one application scenario, $M \times M \;\; m_m$ is covered by N sensor nodes that are haphazardly placed around the field. These sensor nodes transmit an advertising message to the base station (BS) at the beginning of this phase, containing their position and remaining energy. The BS assigns sensor nodes to clusters and identifies cluster centers based on data from the sensor nodes using the FCM approach. Each node is not fully part of a single cluster; instead, it is given a level of cluster belongingness. So, nodes that are closer to the edge of a cluster may join it with a degree that is close to the degree of belonging to the next cluster. After the grouping process, the BS chooses the CH node as the one with the most remaining energy in each cluster.

Phase 2: Data transmission

After setting the transmission plan, all nodes get the assignment message. Sensor nodes will gather and communicate data to CHs. The FCM algorithm maximizes cluster member node transmission power by finding the shortest path to CHs. Time-scheduled cluster member nodes only turn on their radio component when sending data packets and off when sending packet. The BS only gets compressed data bits; the CH level is where data fusion and aggregation happen. The effect is that less information is sent across the network, which means less energy is used. [25].

4. Simulation results and analysis

This part explains the result of the proposed method, which demonstrates the effectiveness of the introduced system compared to other methods. The simulations are carried out using the MATLAB tool. The experiments are performed using the Matlab tool installed in the personal computer (PC) with 12GB RAM. The tests were kept running on an Intel(R) Core (TM) i7-8700CPU@3.20GHz processor with the 64-bit operating system, x64-based processor. This display does not support pen or touch input. The proposed methodology's effectiveness is proved through an evaluation and comparison to competing techniques in terms of several performance measures. These metrics include total energy consumption, node-to-controller latency, remaining energy, average energy consumption, number of alive nodes, packet delivery ratio, synchronization cost, network lifetime, and average latency. Tables 2 and 3 list the system parameters and simulation parameters of the proposed method.

Table 2: System parameters of the proposed method

Scenarios	No. of sensors	Deployment area (m^2)	Optimal controllers
Scenario 1	100	100x100	3
Scenario 2	200	300x300	6
Scenario 3	300	500x500	8
Scenario 4	400	700x700	10
Scenario 5	500	1000x1000	12

Table 3: Simulation parameters of the proposed method

Parameters	Values
Area of deployment	100×100 (m^2) to 1000×1000 (m^2)
Initial energy utilized	1-2 Joules
Number of sink nodes	1
Number of sensor nodes	100-500
Data packets size	512x512
Packet length	250
Transmission distance	75m

Table 4: Algorithmic parameters and their values of existing and proposed methods

Methods	Parameters	Values
GA	Population size	50
	Chromosome length	16
	Number of iterations	200
	Selection and crossover rate	0.8
	Mutation rate	0.2
SA	Population size	50
	Hydration rate	0.8
	Number of iterations	500
	Initial temperature	1000
	Termination temperature	0.01
ABP-CO	Number of iterations	100
	Initial population	100
	Step length	0.01
	Levy distribution parameter	1.5
SFO-CS	Number of initial populations	50
	Maximum iteration	50
	A	4
	ε	0.01
ISFO (Proposed)	Number of initial populations	50
	Maximum iteration	50
	$rand$	0 to 1
	A	4
	ε	0.001
		0.4 and 0.9

	Inertia weight (u_{min}) and (u_{max}) r τ Step size of the Levy flight	0 to 1 1.5 0.01

Table 4 lists the existing and proposed algorithm's parameters and their values. Based on the system parameters, the network model with optimal controller selection is displayed in Figure 3. Figure 3 represents the node visualization for 100 nodes.

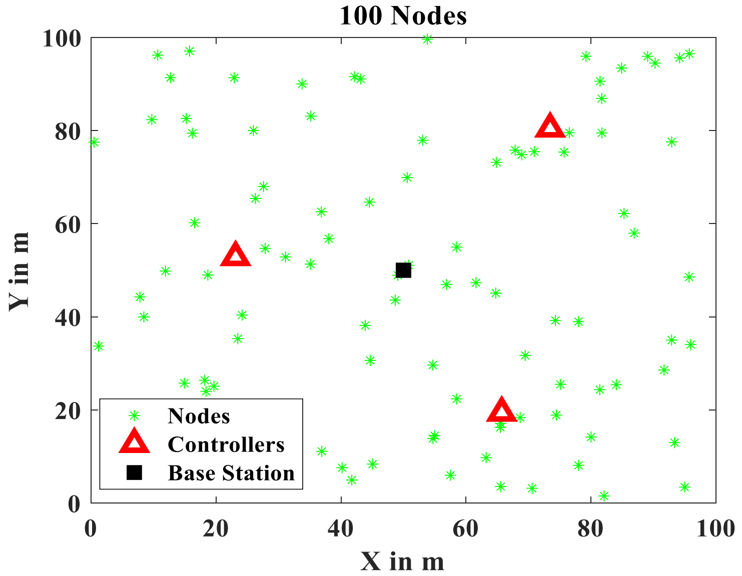


Figure 3: Visualization of controller selection in SDWSN for 100 nodes

4.1 Performance analysis with existing methods

The performance of the suggested method was briefly discussed in this section using a variety of performance parameters, including convergence analysis, average energy consumption, packet delivery ratio vs. rounds, latency vs. rounds, number of alive nodes vs. rounds, synchronization cost vs. nodes, total energy consumption, average latency vs. nodes, rounds until first node dies vs. number of rounds, and energy consumption analysis vs. nodes.

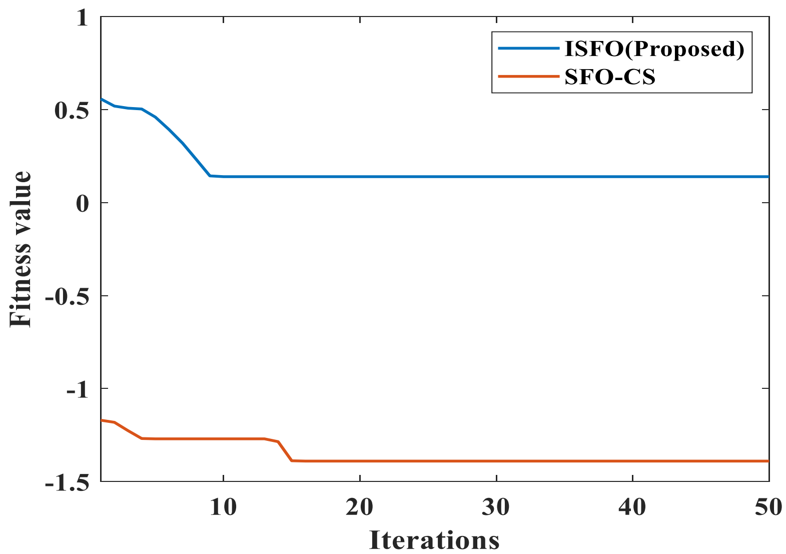


Figure 4: Convergence analysis

The convergence study of the suggested approach with the current methodologies is shown in Figure 4. The suggested approach converges at the tenth iteration in this case, while the current SFO-CS algorithm converges at the fifteenth iteration after a total of fifty iterations. This is due to the advantage of the IFSO algorithm, which improves the solution speed as well as search performance.

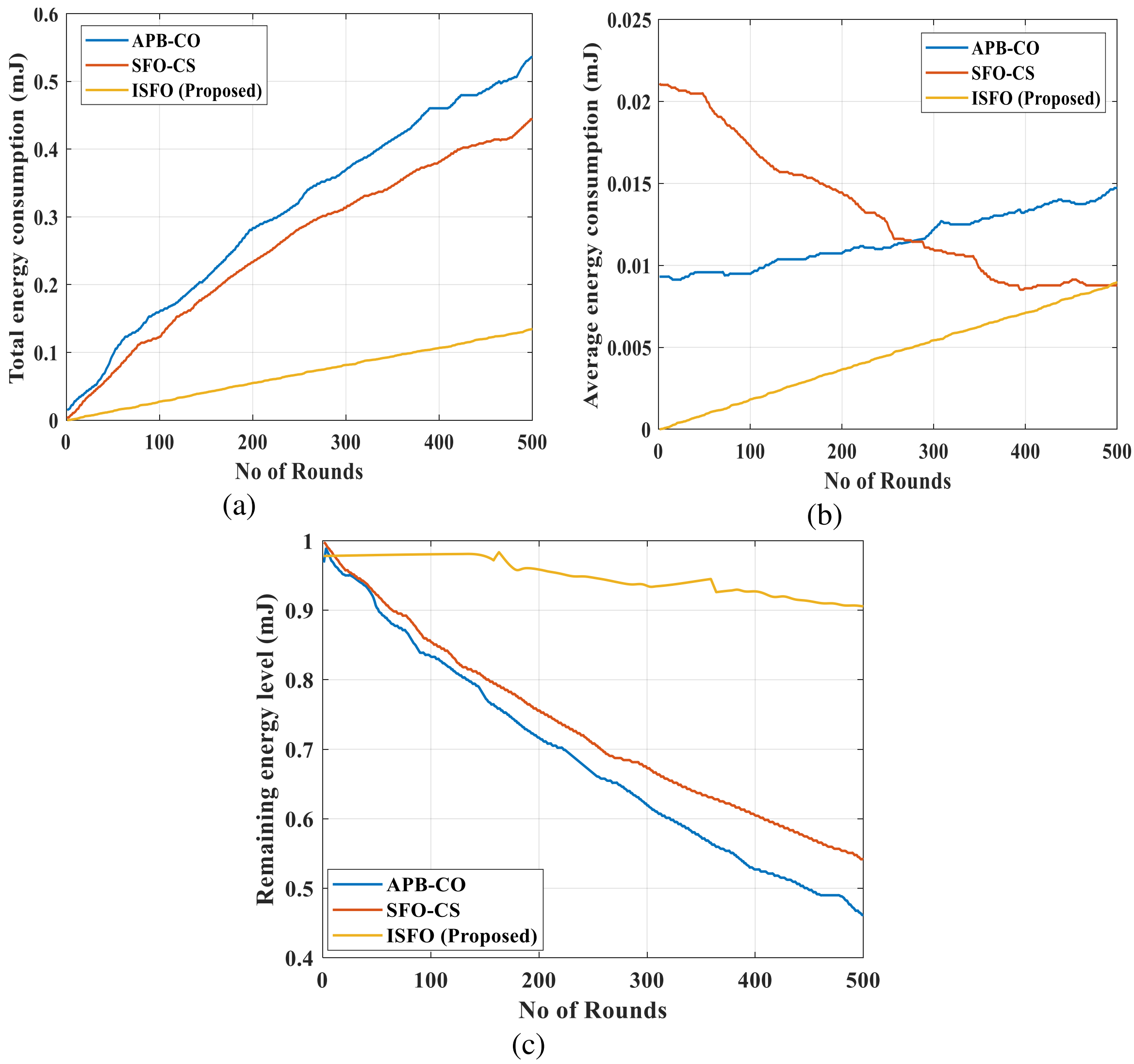


Figure 5: Energy consumption analysis

The performance of the suggested model is tested against current techniques such as Adaptive Population-Based Cuckoo Optimization (APB-CO) for optimal controller placement [26] and SFO-CS [18] algorithms. Figure 5 shows the energy consumption analysis against the number of rounds. The average and total energy consumption of the suggested methods are shown in Figures 5(a) and 5(b). The investigation shows that energy use is far lower than current models. Similarly, the remaining energy level is shown in figure 5(c), which is far higher than the models that are currently in usage.

The amount of rounds is used to figure out the packet delivery ratio, which is shown in Figure 6. It is the number of packets that were delivered out of all the packets that were sent from the source node to the target node in the network. This analysis was also compared with the APB-CO and SFO-CS techniques. The study of the graph shows that the suggested method has a much higher packet delivery rate than the current methods.

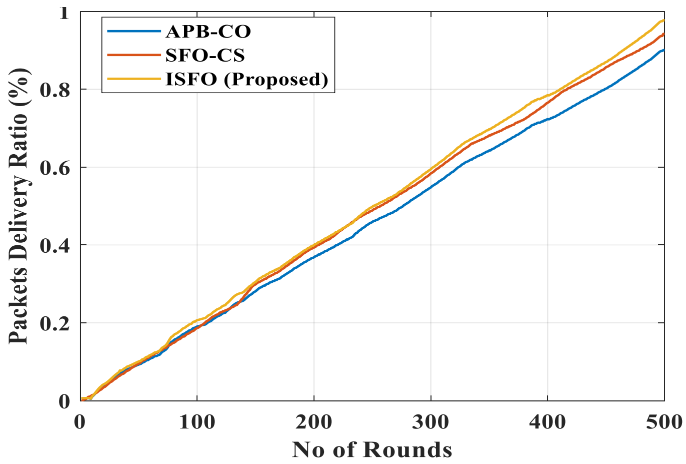


Figure 6: Performance analysis of packet delivery ratio

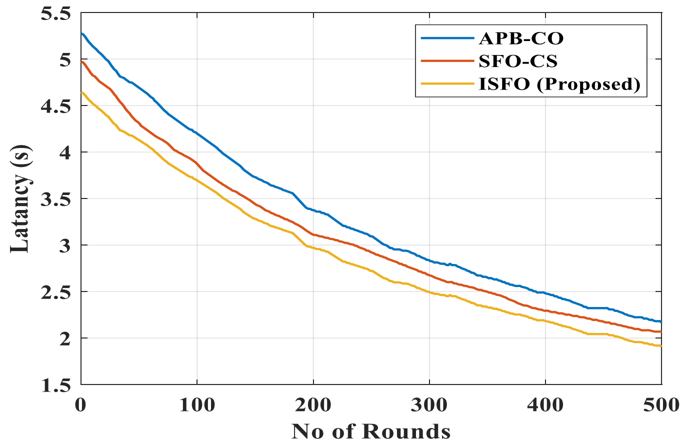


Figure 7: Performance analysis of node-to-controller latency (s) vs number of rounds

Figure 7 displays the node-to-controller latency analysis of the proposed methodology against the number of rounds. Worst-case latency is defined as the maximum propagation delay from node to controller. The number of rounds varies from 50 to 500 in this analysis. Because there are fewer forwarding nodes available, latency increases with a decrease in the number of nodes. Similarly, there is less of a gap between nodes. Packet delivery time's decrease as the number of nodes rises since there are more forwarding nodes available and the distance is shorter. The investigation demonstrates that the suggested model's latency is significantly less than that of the current APB-CO and SFO-CS techniques.

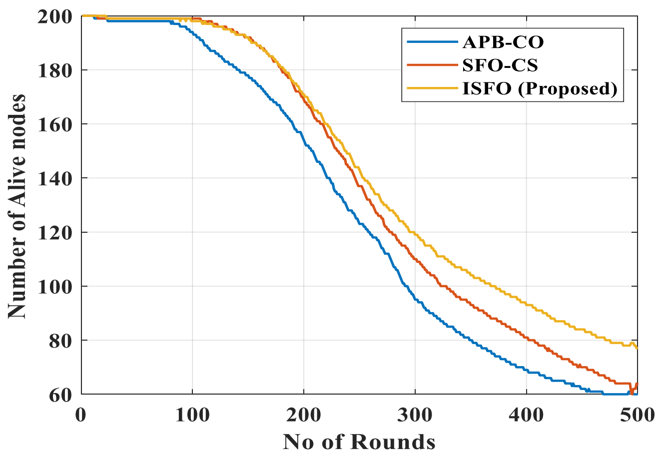


Figure 8: Number of alive nodes vs number of rounds

Figure 8 shows the number of alive nodes compared to the number of rounds. The suggested approach outperforms the current approaches, such as APB-CO and SFO-CS, during the 2500 rounds of analysis.

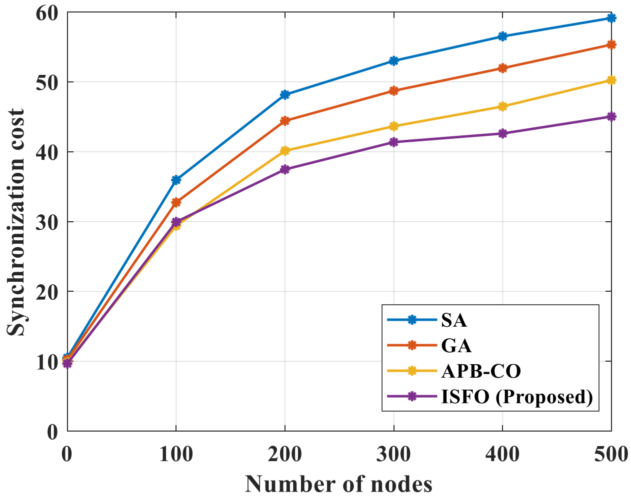


Figure 9: Synchronization cost vs number of nodes

Figure 9 displays the performance analysis of synchronization cost against number of nodes. Generally, the synchronization cost is calculated based on two parameters: synchronization delay and synchronized data transmission. The synchronization delay is the time it takes for one controller to learn about an event that another controller set off. It also includes the amount of synchronization data that is sent between controllers. The range of nodes in this instance is 100 to 500. Also, the proposed method is compared against existing methods like Simulated Annealing (SA), Greedy Approach (GA), and ABP-CO. For 100 nodes, the synchronization cost of the proposed methodology is 30; for 200 nodes, it is 38; and for 500 nodes, the synchronization cost is 45. The investigation indicates that the suggested methodology's synchronization cost is significantly less than that of the current models.

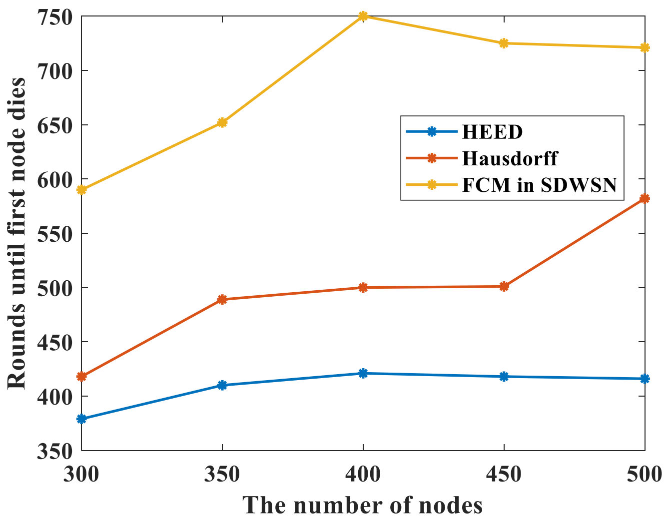


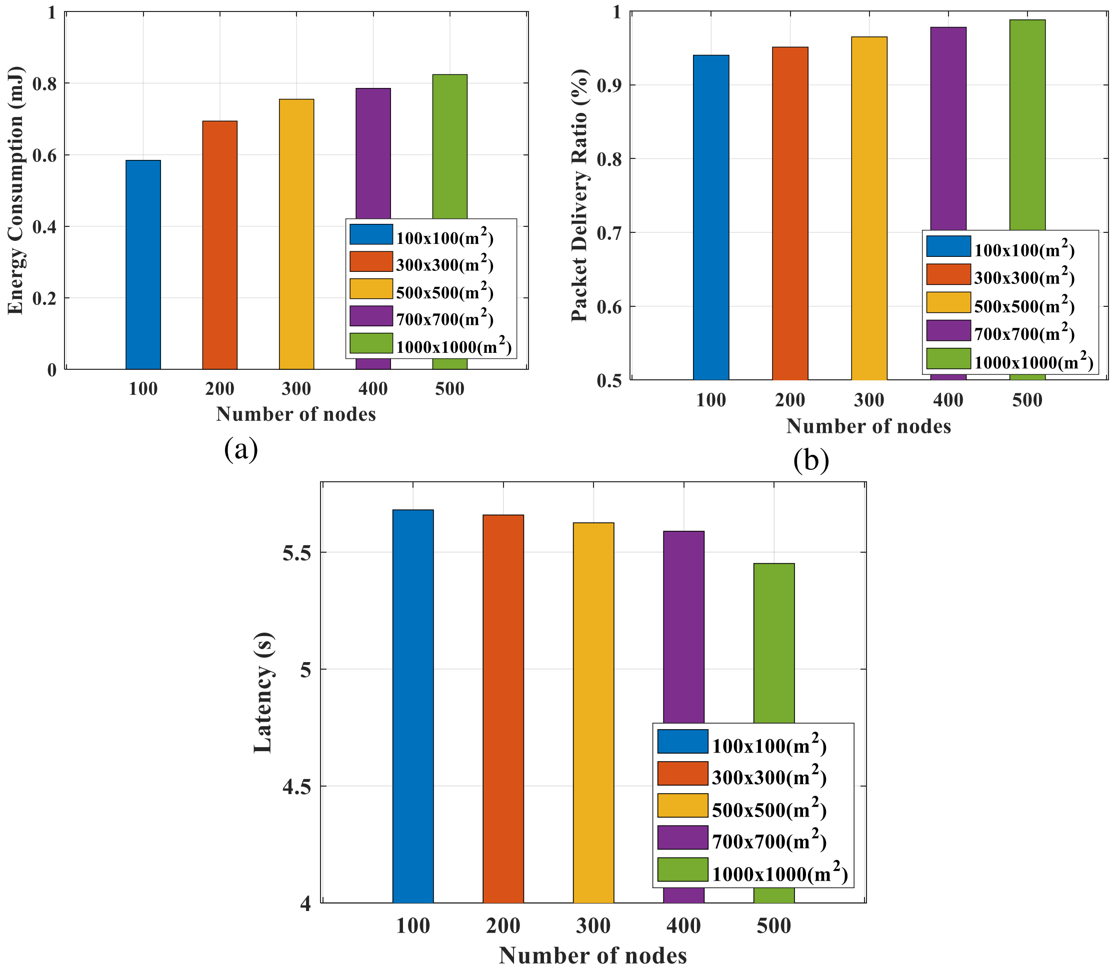
Figure 10: Network lifetime comparison vs number of nodes

A comparison of the network lifetime and node count is shown in Figure 10. It is obvious that node density must be used to increase the lifespan of networks. The present analysis employs the FCM clustering technique to carry out the node clustering procedure, whereby the nodes' minimalized transmission and residual energy are identical to those of HEED and

Hausdorff clustering [27]. All of the nodes use less energy because there is less space between the CNs and the sensor nodes. Consequently, the suggested approach's lifetime is increased in comparison to the current clustering process approaches.

4.2 Parametric study for the proposed methodology

The parametric study of the suggested methodology utilizing several performance metrics, such as energy usage, packet delivery ratio, and delay, is briefly covered in this part. Also, this analysis is performed by varying network size from 100 to 500 nodes, varying deployment area from 100x100 to 1000x1000 and varying number of rounds from 500 to 2500 rounds. In addition, the performance analysis is taken for varying numbers of controllers against packet delivery ratio, energy consumption and latency analysis. The primary purpose of these analyses is to demonstrate how well the suggested methodology works with various performance metrics.



(c)

Figure 11: Performance analysis of proposed methodology vs number of nodes

Figure 11 shows the results of the performance analysis of the suggested methodology utilizing several factors, including (a) energy usage, (b) packet delivery ratio, and (c) delay. Here, the number of nodes varied from 100 to 500, and the network area varied from $100x100m^2$ to $1000x1000m^2$. Figure 11(a) shows the energy consumption analysis, which shows that as the number of nodes increases, so does the energy consumption. The packet delivery ratio vs the number of nodes for various region deployment ranges is shown in Figure 11(b). This study shows that the suggested model's packet delivery rate goes up as the area size and sensor node range get bigger. As the number of nodes and deployment area go up, so does the packet delivery ratio. This is because as the number of nodes goes up, the network gets faster. Lastly, the latency analysis is displayed in Figure 11(c), which means that as the number of nodes goes up, the delay goes down. This is because the rate at which packets are sent slows down as the number of nodes increases, which causes the variable number of nodes to experience less latency.

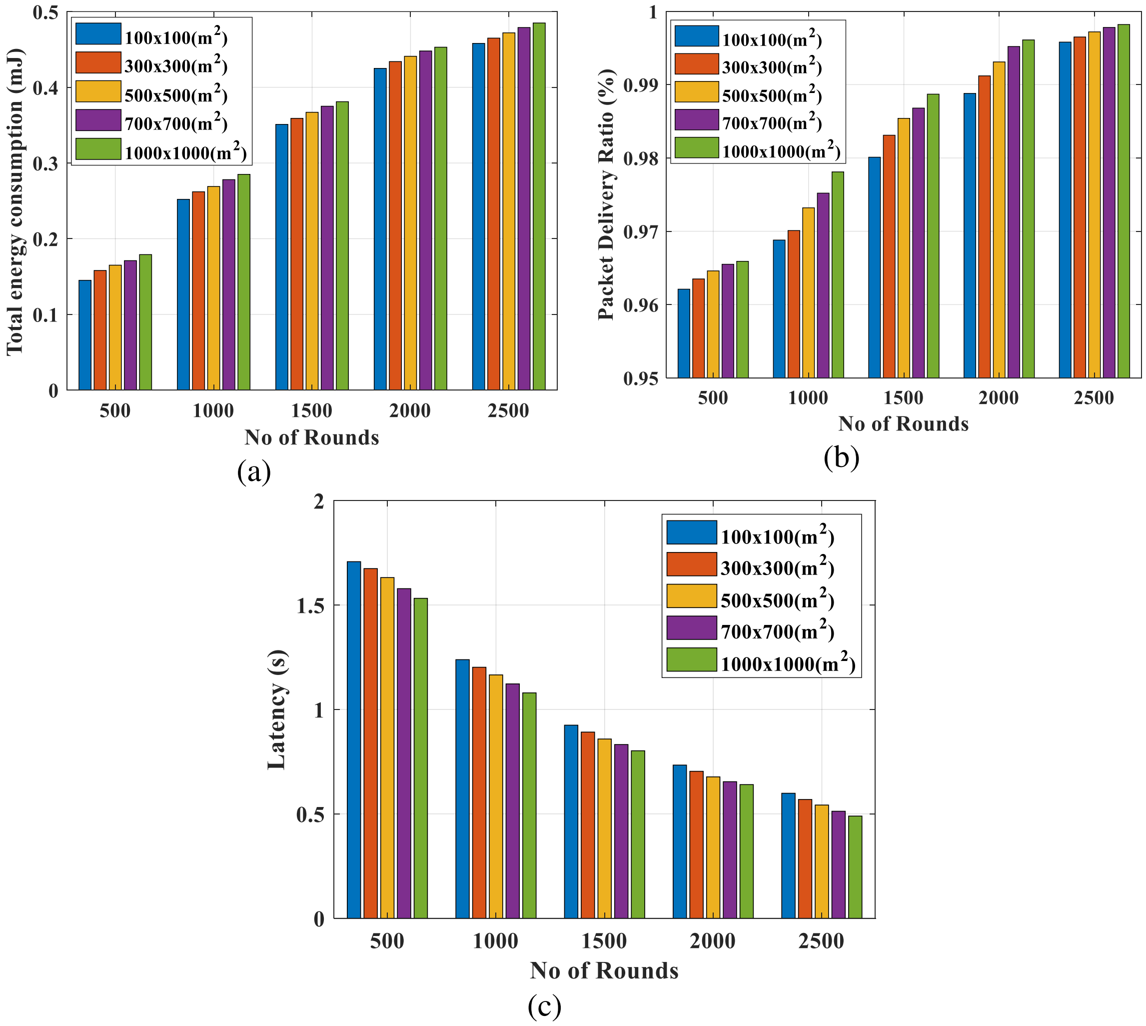


Figure 12: Performance analysis of proposed methodology vs number of nodes

The performance analysis of the suggested methodology by different numbers of rounds is shown in Figure 12. The number of rounds varies from 500 to 2500 in this instance. The suggested approach uses more energy as the number of network rounds goes up, as shown in Figure 12(a) of this study. Figure 12(b) shows the packet delivery ratio versus the number of rounds. It shows that the suggested model's delivery ratio is better as the number of rounds moves upwards. Finally, latency analysis is shown in Figure 12(c), which shows that latency decreases with increasing number of rounds. This is due to node density, which reduces the distance between each node. As a result, the latency of the proposed model becomes lesser for the increasing number of network rounds.

Figure 13 displays the performance analysis of selecting optimal CNs from varying network sizes. The analysis shows that there are 3 CNs that are optimally selected for 100 nodes. Likewise, 6 CNs, 8 CNs, 10 CNs and 12 CNs are optimally selected for 200, 300, 400 and 500 nodes, respectively. The selection of CNs in each network size is enough for efficient data transmission, and this is proven by the performance analysis graphs in Figure 14.

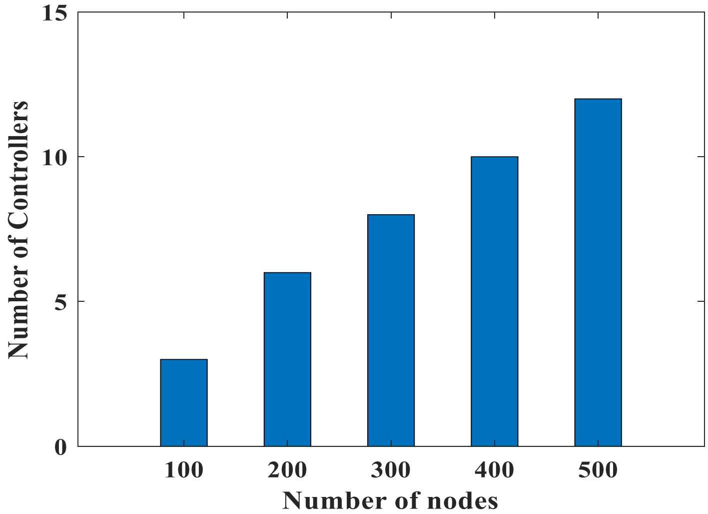


Figure 13: Number of nodes vs number of controllers

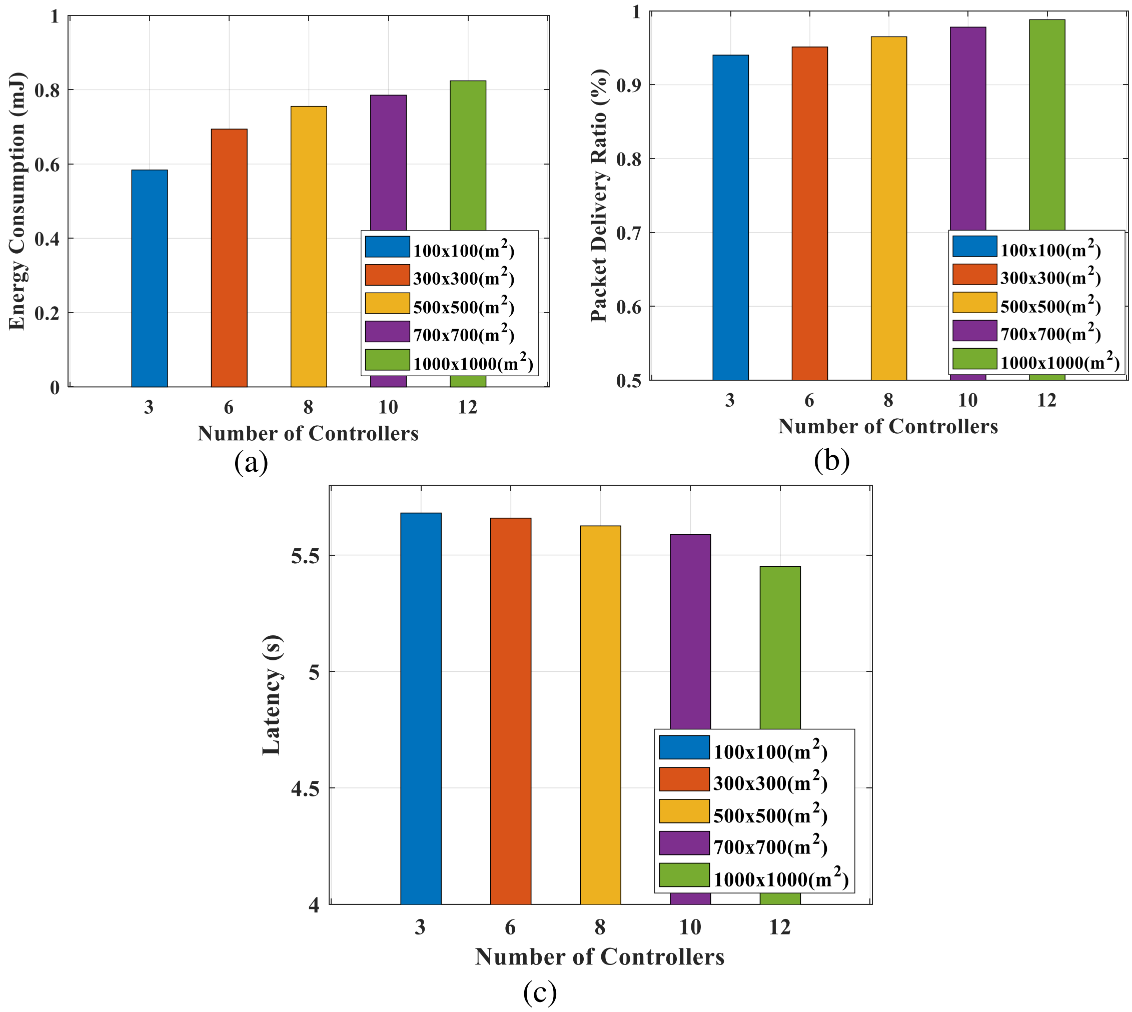


Figure 14: Performance analysis of proposed methodology vs number of controllers

Figure 14 shows the results of the performance analysis of the suggested methodology using a variety of factors, including (a) energy usage, (b) packet delivery ratio, and (c) latency. This analysis is taken for the optimally selected CNs from the sensor nodes. The energy consumption analysis, as shown in Figure 14(a), reveals that the power consumption rises in direct proportion to the number of controllers. The packet delivery ratio against a few controllers for different area deployment ranges is shown in Figure 14(b). This analysis shows that as the range of controllers and area size rise, so does the packet delivery ratio of the suggested model. Finally, the latency analysis is displayed in Figure 14(c), which indicates that the latency decreases with the increasing number of nodes. This is due to the fact that the increasing node density decreases the packet transmission rate, which results in less latency for the varying number of controllers in each network size.

5. Conclusion

This study proposed an ISFO method to select the best CNs from a large number of sensor nodes. To determine the optimal CN, a multi-objective function was designed that combines node coverage, communication cost, node distance, and residual energy of the nodes. The FCM clustering technique was used to accomplish both the clustering and data transfer processes after the optimal CN was completed. Furthermore, the performance of the suggested methodology was assessed using several performance characteristics such as energy consumption, synchronization cost, average latency, and network lifetime. From the analysis, it was known that the proposed ISFO method consumes 0.13mJ, which was comparatively lesser than the existing SFO-CS and APB-CO algorithms as they consume 0.42mJ and 0.53mJ for 500 nodes, respectively. In a similar vein, both the average latency of the suggested approach and the packet delivery ratio of the proposed model were considerably lower than those of the present approaches. In the future, the proposed methodology will be evaluated on the different scenarios and versions of the network with multiple mobile sink nodes.

Compliance with Ethical Standards

Funding: No funding is provided for the preparation of manuscript.
Conflict of Interest: Authors declare that they have no conflict of interest.
Ethical Approval: This article does not contain any studies with human participants or animals performed by any of the authors.
Consent to participate: All the authors involved have agreed to participate in this submitted article.
Consent to Publish: All the authors involved in this manuscript give full consent for publication of this submitted article.
Authors Contributions: All authors have equal contributions in this work.
Data Availability Statement: Data sharing not applicable to this article.

References

[1] Vishnu, V. M., & Manjunath, P. (2019). SeC-SDWSN: Secure cluster-based SDWSN environment for QoS guaranteed routing in three-tier architecture. *International Journal of Communication Systems*, *32*(14), e4020.
[2] Cui, X., Huang, X., Ma, Y., & Meng, Q. (2019). A load balancing routing mechanism based on SDWSN in smart city. *Electronics*, *8*(3), 273.
[3] Letswamotse, B. B., Malekian, R., Chen, C. Y., & Modieginyane, K. M. (2018). Software defined wireless sensor networks (SDWSN): a review on efficient resources, applications and technologies. *Journal of Internet Technology*, *19*(5), 1303-1313.
[4] Alves, R. C., Oliveira, D. A., Pereira, G. C., Albertini, B. C., & Margi, C. B. (2018). WS3N: wireless secure SDN-based communication for sensor networks. *Security and Communication Networks*, *2018*.

[5] Abdou, A., Van Oorschot, P. C., & Wan, T. (2018). Comparative analysis of control plane security of SDN and conventional networks. *IEEE Communications Surveys & Tutorials*, *20*(4), 3542-3559.
[6] Kgogo, T., Isong, B., Lugayizi, F., & Abu-Mahfouz, A. M. (2021). A survey of resource allocation and controller placement problem in SDN-SDWSN. In *2021 3rd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*, 1-8. IEEE.
[7] Kobo, H. I. (2018). *An efficient distributed control system for software-defined wireless sensor networks* (Doctoral dissertation, University of Pretoria).
[8] Shiny, S. S. G., Priya, S. S., & Murugan, K. (2021). Repeated game theory-based reducer selection strategy for energy management in SDWSN. *Computer Networks*, *193*, 108094.
[9] Wang, R., Zhang, Z., Zhang, Z., & Jia, Z. (2018). ETMRM: An energy-efficient trust management and routing mechanism for SDWSNs. *Computer Networks*, *139*, 119-135.
[10] Ali, J., & Roh, B. H. (2021). Quality of service improvement with optimal software-defined networking controller and control plane clustering. *Comput. Mater. Contin*, *67*, 849-875.
[11] Ali, J., Roh, B. H., & Lee, S. (2019). QoS improvement with an optimum controller selection for software-defined networks. *Plos one*, *14*(5), e0217631.
[12] Letswamotse, B. B., Malekian, R., Chen, C. Y., & Modieginyane, K. M. (2018). Software defined wireless sensor networks and efficient congestion control. *IET Networks*, *7*(6), 460-464.
[13] Tahmasebi, S., Safi, M., Zolfi, S., Maghsoudi, M. R., Faragardi, H. R., & Fotouhi, H. (2020). Cuckoo-PC: an evolutionary synchronization-aware placement of SDN controllers for optimizing the network performance in WSNs. *Sensors*, *20*(11), 3231.
[14] Samarji, N., & Salamah, M. (2021). A fault tolerance metaheuristic-based scheme for controller placement problem in wireless software-defined networks. *International Journal of Communication Systems*, *34*(4), e4624.
[15] Firouz, N., Masdari, M., Sangar, A. B., & Majidzadeh, K. (2021). A novel controller placement algorithm based on network portioning concept and a hybrid discrete optimization algorithm for multi-controller software-defined networks. *Cluster Computing*, *24*, 2511-2544.
[16] Shiny, S. S. G., & Murugan, K. (2021). TSDN-WISE: Automatic threshold-based low control-flow communication protocol for SDWSN. *IEEE Sensors Journal*, *21*(17), 19560-19569.
[17] Jurado-Lasso, F. F., Clarke, K., Cadavid, A. N., & Nirmalathas, A. (2021). Energy-aware routing for software-defined multihop wireless sensor networks. *IEEE Sensors Journal*, *21*(8), 10174-10182.
[18] Rajoriya, M. K., & Gupta, C. P. (2023). Sailfish optimization-based controller selection (SFO-CS) for energy-aware multi-hop routing in software defined wireless sensor network (SDWSN). *International Journal of Information Technology*, *15*(7), 3935-3948.
[19] Hajian, E., Khayyambashi, M. R., & Movahhedinia, N. (2022). A mechanism for load balancing routing and virtualization based on SDWSN for IoT applications. *IEEE Access*, *10*, 37457-37476.
[20] Sathyamoorthy, M., Kuppusamy, S., Dhanaraj, R. K., & Ravi, V. (2022). Improved K-means based q learning algorithm for optimal clustering and node balancing in WSN. *Wireless Personal Communications*, *122*(3), 2745-2766.
[21] Yan, X., Huang, C., Gan, J., & Wu, X. (2022). Game theory-based energy-efficient clustering algorithm for wireless sensor networks. *Sensors*, *22*(2), 478.

[22] Tahmasebi, S., Rasouli, N., Kashefi, A. H., Rezabeyk, E., & Faragardi, H. R. (2021). SYNCOP: An evolutionary multi-objective placement of SDN controllers for optimizing cost and network performance in WSNs. *Computer Networks*, *185*, 107727.
[23] Rahimifar, A., Seifi Kavian, Y., Kaabi, H., & Soroosh, M. (2021). Predicting the energy consumption in software defined wireless sensor networks: a probabilistic Markov model approach. *Journal of Ambient Intelligence and Humanized Computing*, *12*, 9053-9066.
[24] Shadravan, S., Naji, H. R., & Bardsiri, V. K. (2019). The Sailfish Optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems. *Engineering Applications of Artificial Intelligence*, *80*, 20-34.
[25] Abdulzahra, A. M. K., & Al-Qurabat, A. K. M. (2022). A clustering approach based on fuzzy C-means in wireless sensor networks for IoT applications. *Karbala International Journal of Modern Science*, *8*(4), 579-595.
[26] Qaffas, A. A., Kamal, S., Sayeed, F., Dutta, P., Joshi, S., & Alhassan, I. (2023). Adaptive population-based multi-objective optimization in SDN controllers for cost optimization. *Physical Communication*, *58*, 102006.
[27] Xiang, W., Wang, N., & Zhou, Y. (2016). An energy-efficient routing algorithm for software-defined wireless sensor networks. *IEEE Sensors Journal*, *16*(20), 7393-7400.