2I-CSO: A Novel Intelligent and Interoperable Cat Swarm Optimizer Approach to Optimal Cluster Head Selection and Self-Termination Search

Oumaima Hassan*, Mohammed Essaid Riffi
Faculty of Sciences-Department of Computer Science, Chouaib Doukkali University, El Jadida, Morocco

***Abstract*—Wireless sensor networks (WSNs), fundamental building block of IoT, are subject to several constraints because of the finite non-rechargeable energy resources available in the nodes. The selection of Cluster Head (CH) plays a critical role in determining the energy balance in a network. Conventional methods such as the LEACH algorithm choose CH randomly with a probability mechanism that might lead to choosing weak nodes as CHs and thereby fail prematurely. The biologically - inspired optimization methods, such as PSO and CSO, help to enhance CH selection using a global approach. However, these methods suffer from the following three major shortcomings: 1) random switching between the exploration and exploitation stages, 2) lack of intelligence during the formation of clusters, and 3) growing exponentially complex search space of CHs.This study proposes an Intelligent and Interoperable Cat Swarm Optimizer (2I-CSO), a protocol designed to address these limitations simultaneously. 2I-CSO also introduces an interoperable configuration mechanism based on LEACH's hierarchical architecture, where the Base Station maintains a centralized energy configuration table shared with Cluster Heads and member nodes, ensuring network-wide parameter consistency and enabling the intelligent stopping mechanism. Experiments conducted on five well-known TSPLIB test cases and WSN simulations demonstrate that 2I-CSO outperforms individual metaheuristics. Simulation results on a custom web-based platform further show that 2I-CSO achieves faster convergence, lower computational cost, and competitive network lifetime compared to standard CSO and the Emperor Penguin Optimizer (EPO). To the best of our knowledge, the proposed intelligent stopping condition is the first introduced for bio-inspired WSN clustering protocols.**

Keywords*—Wireless sensor networks; Cat Swarm Optimization; Cluster Head Selection; energy efficiency; bio-inspired optimization; configuration space reduction; intelligent stopping condition; Emperor Penguin Optimizer; network lifetime

I. Introduction

WSN refers to distributed computing systems consisting of sensor nodes (SNs), which have the capability to sense physical phenomena, perform computations on sensed data, and communicate information in a wireless manner to the central Base Station (BS), where information is processed [1]. Because of their adaptability, scalability, and low-cost, WSNs have become essential elements of IoT systems and are applied extensively in various fields like environmental monitoring, healthcare, industrial automation, and smart farming systems. Nevertheless, SNs function in constrained hardware environments especially with limited energy budget of batteries, making energy efficiency the major concern in WSN architecture [2]. Given that energy drain during wireless communications takes up the majority of the node's energy budget, energy imbalance often arises, leading to premature failure of overloaded nodes, thereby decreasing network lifetime. Fig. 1 presents the LEACH protocol architecture for WSN.

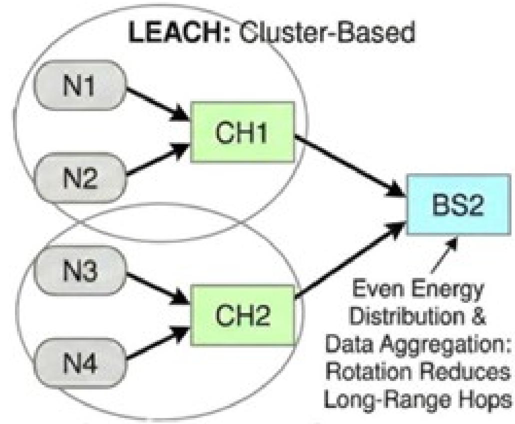


Fig. 1. LEACH protocol architecture for WSN.

Hierarchical clustering has, therefore, proven to be a reliable approach towards improving communication among nodes [3]. WSNs employing hierarchical clustering techniques are divided into clusters whereby each cluster has a leader known as a cluster head (CH). The task of choosing proper CHs determines the performance of the system in terms of lifetime, energy balance, and communications efficiency [2]. Optimal selection of CHs is an NP-hard optimization problem, because the number of combinations increases exponentially as the size of the network grows. Deterministic and heuristic clustering techniques such as those that follow LEACH, fail to achieve optimal clustering because of their poor search ability [3]. Due to this problem, bio-inspired metaheuristic algorithms, providing a reliable approach to perform the global search, have been employed to solve hard combinatorial problems [7]. Some of the clustering techniques that use PSO have proven effective in energy balance and quality improvement [5]. Among other algorithms, the relatively new Emperor Penguin Optimizer (EPO) has demonstrated good results in engineering optimization and WSN/IoT clustering because of its population migration approach [4], [8], [9]. However, among more recent swarm intelligence optimization algorithms, the focus here is on Cat Swarm Optimization (CSO) because of its distinct dual search approach: seeking mode and tracing mode. In

*Corresponding author

benchmark comparisons performed recently, CSO has proved to be among top-performing metaheuristic algorithms when tested on standard optimization benchmark problems [12]. At the same time, recent comprehensive reviews show that the algorithm has been applied successfully to various engineering problems, including WSN optimization [13]. While there are numerous benefits of using the CSO algorithm, it also has a number of disadvantages when used for Cluster Head selection. Firstly, it is plagued by the issue of configuration space explosion, which means that many redundant CH solutions are analyzed. Secondly, the conventional CSO does not have an intelligent stopping condition, meaning that the searching process continues even though the optimal solution is already found. Finally, the algorithm's random mode switching strategy neglects the optimization process [14][15]. In order to mitigate the constraints outlined above, this study suggests 2I-CSO (Intelligent and Interoperable Cat Swarm Optimization) – an innovative enhancement to the clustering process that enables optimal cluster heads election in a WSN. Three distinct enhancements are introduced by the suggested framework. Firstly, configuration space is reduced through search space reduction technique, which converts the search problem from ordered permutation into unique combination without any additional computational cost. Secondly, an intelligence stopping criterion monitors the searched configurations and halts the process when all unique and feasible solutions are examined. Finally, an adaptive mode switching technique, based on unified exploration and exploitation like Sand Cat Swarm Optimization (SCSO) algorithm, changes the optimization mode depending on global best performance [21]. Moreover, 2I-CSO provides an interoperable parameter standardization approach, in which the Base Station keeps and disseminates a centralized parameter configuration of energy parameters for the whole network. Parameter consistency is achieved in all nodes, ensuring the validity of the proposed stopping condition.

The key contributions of this study are briefly described below:

- Complete Modeling of Standard CSO: A thorough analysis of the application of the standard CSO in WSN clustering, including the network model, energy model, and optimization process.
- Improved 2I-CSO Protocol: A novel protocol based on the standard CSO algorithm integrating:
    - configuration space reduction,
    - intelligent stopping,
    - adaptive mode switching,
    - interoperable parameter standardization.
- Experimental Comparison of the Approaches: A structured comparison of Standard CSO [2], [3], 2I-CSO, and EPO [8], evaluating convergence speed, computational complexity, energy consumption, and network lifetime.

Simulation results show that the **2I-CSO** protocol performs better than existing protocols in terms of convergence rate, computational cost, and network lifetime.

The work in this study is organized as follows: Section II contains a literature review on clustering algorithms and bio-inspired optimization in wireless sensor networks (WSNs). Section III describes the system and energy models used in our analysis. Interoperable Configuration Model is introduced in Section IV. In Section V, the canonical CSO algorithm and its shortcomings are described. Section VI contains our novel approach named 2I-CSO protocol, as well as simulation experiments.Section VII provides a simulation and implementation,proceeded by a conclusion and future work in Section VIII.

II. Related Work

This section reviews the main clustering and optimization strategies related to WSN energy efficiency and CH selection. It first discusses classical heuristic protocols that establish the basis for hierarchical routing, then examines bio-inspired approaches that improve global search capability, and finally identifies the research gaps that motivate the proposed 2I-CSO framework.

A. Heuristic Protocols

Traditional heuristic routing protocols established early frameworks in randomized CH rotation, chain-based topologies, and hybrid energy-aware elections, as comprehensively surveyed in [1], [3]. Fuzzy C-means clustering has also been used for WSN grouping and CH-related partitioning decisions [6]. In parallel, penguin-search optimization and fuzzy ant-colony CH selection further improve WSN routing by combining residual-energy awareness with population-based search [10], [11]. However, these approaches rely mainly on local or heuristic decision-making, which limits explicit control over redundant CH-configuration exploration.

B. Bio-Inspired Approaches

Recent years have seen numerous CSO-based approaches for WSN optimization. Kumar and Singh [14] proposed Improved CSO (ICSO) with enhanced tracing mode equations and local search for better clustering convergence. Singh and Kumar [15] introduced neighborhood-based search strategies to mitigate CSO's local optima problem.Aramuthakannan et al. [16] have used fuzzy logic CH selection along with adaptive CSO routing, giving an energy saving of 43.4%. Wategaonkar and Reshmi [17] have used CSO for chain-based sectoring with sector head selection based on RSSI values. Kong et al. [18] presented Enhanced Parallel CSO (EPCSO) for balanced WSN routing, saving more than 35% power consumption. In addition, recent works include Taylor series along with Spotted Cat Optimization by Kalburgi and Manimozhi [19] for CH selection based on trust awareness, and Basheri and Ragab [20] extended CSO with quantum computing principles for IoT clustering and intrusion detection.

In addition to basic CSO, Sand Cat Swarm Optimization (SCSO) [21] developed a combined exploration and exploitation strategy based on the nature of sand cats' hunting behavior. The WSN-oriented derivatives of SCSO included: Gowda and Ramalingappa [22] used multi-objective SCSO to optimize underwater WSN CH selection with 99.32% packet transmission, and Sindian et al. [23] presented a hybrid dynamic SCSO using trust-based security in IoT resource allocation. However, none of these methods solves all three problems related to CSO simultaneously: configuration space explosion, lack of

TABLE I. RESEARCH GAPS ADDRESSED BY 2I-CSO

Gap	Impact
Configuration Space Explosion (N^k redundant configs)	Wasted computation
No Intelligent Stopping (fixed round count)	Resources wasted after optimum found
Random Mode Switching (ignores optimization state)	Missed exploitation or wasted tracing

intelligent stopping, and random mode switching, which is motivating 2I-CSO.

C. Research Gap

Previous enhancements to CSO [14][15] contribute to improving the performance of CSO but ignore the issue of configuration space explosion and lack any stopping mechanism. The efficiency of CSO on various NP-hard problems, such as job scheduling [24], to WSN clustering [12], serves as another justification for our approach to overcoming these structural gaps (see Table I).

III. SYSTEM MODEL

This section presents the basic modeling assumptions used in the suggested CH selection framework. First, we introduce the network deployment and clustering assumption. Second, we present the assumed radio energy model for evaluating communication costs. Finally, formulates the objective function that links network topology, energy consumption, and optimization-based CH selection.

A. Network Model

We consider $\boldsymbol{N}$ sensor nodes deployed randomly in an $\boldsymbol{M} \times \boldsymbol{M}$ m^2 field, organized into k clusters, each managed by a single CH. **Assumptions:** stationary nodes/BS; homogeneous initial energy E_0; unlimited BS energy; known positions; variable TX power; fixed packet size l bits (see Fig. 2).

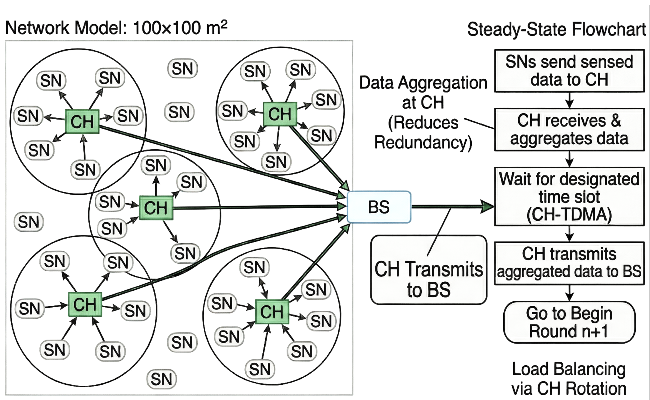


Fig. 2. Network model: N sensor nodes in k clusters, each with a CH that forwards aggregated data to the BS.

B. Energy Model

We adopt the first-order radio energy model described and analyzed in WSN surveys [19][20]. The energy consumed to transmit an l-bit data packet over distance d (see Fig. 3):

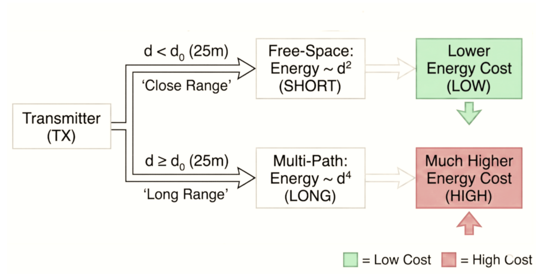


Fig. 3. Transmission cost depends on distance relative to threshold d_0.

- If $d < d_0$ (short distance, free-space model):

$$E_{Tx}(l, d) = l \cdot E_{elec} + l \cdot \varepsilon_{fs} \cdot d^2 \tag{1}$$

- If $d \geq d_0$ (long distance, multi-path fading model):

$$E_{Tx}(l, d) = l \cdot E_{elec} + l \cdot \varepsilon_{mp} \cdot d^4 \tag{2}$$

where, $d_0 = \sqrt{\varepsilon_{fs}/\varepsilon_{mp}}$. Reception: $E_{Rx}(l) = l \cdot E_{elec}$. A CH received from N/k members, aggregates, and transmits to BS consuming more energy than a member, making CH selection the most critical factor. Table II presents the simulation parameters.

TABLE II. SIMULATION PARAMETERS

Parameter	Symbol	Value
Field / Nodes / BS	—	100×100m^2, N=100, (50,50)
Initial energy	E_0	1 J
Electronics / DA	E_{elec} / E_{DA}	50 nJ/bit / 5 nJ/bit/msg
Amplification (fs/mp)	ε_{fs} / ε_{mp}	10 pJ/bit/m^2 / 0.0013 pJ/bit/m^4
Threshold / Packet	d_0 / l	25 m / 4000 bits

C. Objective Function Formulation

The CH selection problem is formulated as a minimization problem using a composite objective function that simultaneously balances two competing criteria.

The Composite Fitness Function:

$$f_{obj} = \alpha \cdot f_1 + (1 - \alpha) \cdot f_2 \tag{3}$$

where, α is a weight parameter (typically 0.5–0.7) that controls the relative priority between distance efficiency and energy protection.

Intra-Cluster Communication Distance (Minimize): f_1 represents the total Euclidean distance between all member sensor nodes and their assigned Cluster Heads:

$$f_1 = \sum_{i=1}^{k} \sum_{j \in c_i} d(SN_j, CH_i) \tag{4}$$

Rationale: Since transmission energy scales with d^2 or d^4, minimizing the physical distances between sensors and their CHs directly reduces the dominant source of energy consumption.

Residual Energy Ratio (Minimize): f_2 measures the energy quality of the elected CHs relative to the overall network:

$$f_2 = \frac{\sum_{j=1}^{N} E_{residual}(SN_j)}{\sum_{i=1}^{k} E_{residual}(CH_i)} \tag{5}$$

Rationale: A low f_2 value indicates that the selected CHs possess a large share of the network's total residual energy meaning they are "strong" nodes capable of sustaining the heavy CH workload.

IV. INTEROPERABLE CONFIGURATION STANDARDIZATION

The "Interoperable" component of 2I-CSO addresses a practical yet often overlooked problem in real-world WSN deployments: how to ensure that all sensor nodes operate under identical energy parameters the very condition required for the intelligent stopping condition (Section VI-C) to be valid.

In reality, WSN implementation may consist of sensor nodes by diverse manufacturers or even from different production batches or firmware revisions. Differences in the energy consumption of radio electronics ($Eelec$), amplifiers (εfs, εmp), and data aggregation (EDA) may occur because of manufacturing tolerances even in one production batch. In the long run, some of the sensor nodes may receive new firmware while others may not, resulting in the emergence of configuration diversity – each node having individual energy parameters due to which the whole system becomes heterogeneous.

Our solution is based on standardizing the configuration process of LEACH's natural communication hierarchy: **BS → CH → SN**.

The central point of our solution is the Configuration Table stored in the Base Station. Table III consists of the standard energy values of the whole WSN.

TABLE III. BS CONFIGURATION TABLE: THE SINGLE SOURCE OF TRUTH FOR NETWORK ENERGY PARAMETERS

Parameter	**Symbol**	**Standardized Value**	**Unit**
Electronics energy	E_{elec}	50	nJ/bit
Free-space amplification	ε_{fs}	10	pJ/bit/m^2
Multi-path amplification	ε_{mp}	0.0013	pJ/bit/m^4
Data aggregation energy	E_{DA}	5	nJ/bit/message
Threshold distance	d_0	25	m
Packet size	l	4000	bits
Config version	V_{config}	v1.0	—

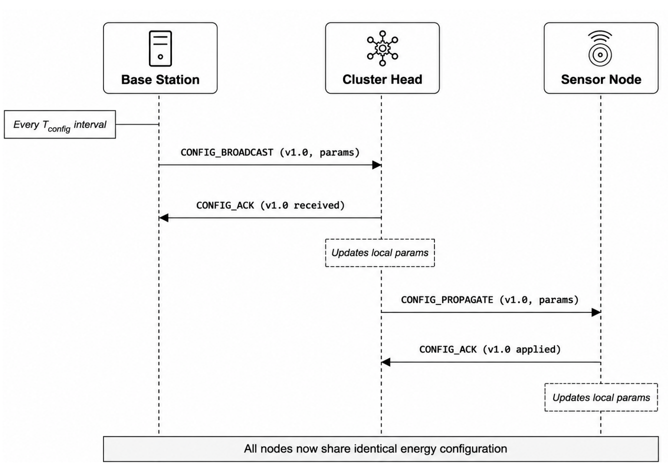


Fig. 4. Sequence diagram: scheduled interoperable configuration standardization.

Fig. 4 summarizes the scheduled configuration synchronization process, in which the BS periodically broadcasts the standardized configuration table to active CHs, each CH stores the received version and forwards it to its member SNs during intra-cluster communication, and every node uses the included version number to avoid unnecessary updates while preserving network-wide energy-parameter homogeneity.

V. CAT SWARM OPTIMIZATION (CSO)

This section introduces the standard Cat Swarm Optimization algorithm as the baseline mechanism for CH selection. It first presents the biological inspiration behind CSO, then defines its main control parameters, explains the seeking and tracing search modes, and finally summarizes the standard workflow and limitations that motivate the proposed 2I-CSO enhancements.

A. Biological Inspiration

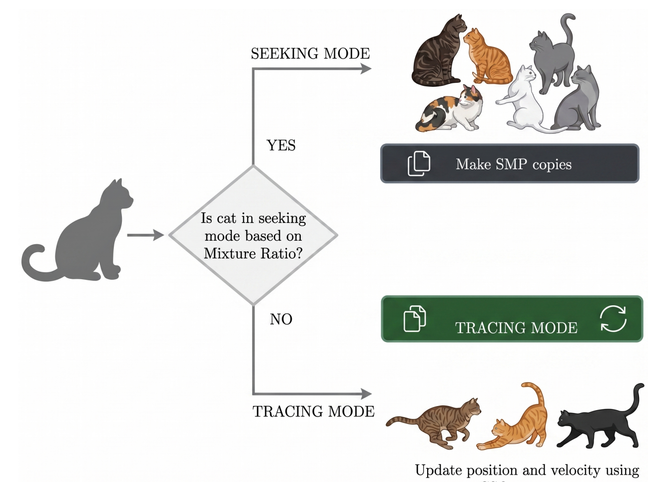


Fig. 5. CSO seeking and tracing modes: nature inspiration.

CSO, introduced in 2006 and reviewed in later CSO surveys and performance studies [12], [13], relates the cat's rest state to "Seeking mode" (local search) and the hunting state to "Tracing mode" (global search). An individual cat can be viewed as an instance of CH structure configuration such as [N2, N7]. The whole process converges towards finding the best possible configuration (gbest) for the group of cats. Benchmark studies of CSO report competitive performance against several metaheuristic algorithms on standard optimization functions [12] (see Fig. 5).

B. Parameters

The standard CSO algorithm is controlled by a small set of parameters that balance exploration and exploitation. MR determines how many cats follow the current best solution, while SMP, SRD, and CDC define how each seeking cat generates and modifies local candidate solutions. The acceleration coefficient c controls the strength of movement toward the global best configuration (see Table IV).

TABLE IV. CSO PARAMETERS

Parameter	Definition	Value
MR	Fraction in tracing mode	0.7
SMP	Number of copies generated in seeking mode	5
SRD	Mutation radius applied to selected dimensions	20%
CDC	Percentage of dimensions changed in each copy	80%
c	Tracing aggressiveness toward gbest	2.0

C. Seeking Mode: Local Exploration

A seeking cat examines nearby alternatives and moves to a superior neighbor through a structured local search:

- Create SMP copies of the current position vector.
- For each copy, randomly select CDC% of dimensions and mutate each by $\pm$SRD%.
- Evaluate the fitness f_{obj} for all copies.
- Compute selection probability: $P_i = \frac{|f_i - f_b|}{f_{max} - f_{min}}$, where f_b is the current best fitness.
- The cat is positioned where the copy chosen through the roulette wheel selection according to P_i.n

Through this process, there will be a detailed exploration of the environment around the promising solution in the manner a cat would explore its surroundings for subtle environmental changes when taking a rest.

D. Tracing Mode: Global Exploitation

The tracing cat rapidly moves towards the present global optimal solution (x_{best}):

$$v_{id} = v_{id} + r \cdot c \cdot (x_{best,d} - x_{id}) \tag{6}$$

$$x_{id} = x_{id} + v_{id} \tag{7}$$

where, $r \in [0, 1]$ is a uniformly distributed random number and c is the acceleration constant. In case of clustering of WSN nodes, the positions are converted to IDs by the rounding process: nodeID $= \text{round}(x_{id}) \mod N$.

With MR=0.7, at each iteration 70% of cats chase (exploitation of the present optimal solution) and 30% explore (searching for alternative solutions). This dual-mode architecture prevents premature convergence to local optima while maintaining exploitation pressure toward the global optimum.

E. Standard CSO Workflow

The standard CSO procedure for WSN clustering is summarized in Algorithm 1.

Algorithm 1. Standard CSO Algorithm for WSN Clustering

Input: Sensor nodes N, number of CHs k, swarm size P, maximum iterations T, MR, SMP, SRD, CDC, c
Output: Best CH configuration $gbest$
Initialize P cats with random CH configurations $[CH_1, \ldots, CH_k]$
Evaluate each cat using $f_{obj} = \alpha f_1 + (1 - \alpha) f_2$
Set the best initial cat as $gbest$
for $t = 1$ to T **do**
    for each cat i in the swarm **do**
        Assign non-CH nodes to the nearest CH using Euclidean distance
        Generate a random value $r \in [0, 1]$
        if $r <$ MR **then**
            Apply Tracing Mode toward $gbest$
        else
            Apply Seeking Mode using SMP, SRD, and CDC
        end if
        Map updated positions to valid sensor node IDs
        Recompute f_{obj} for cat i
        if $f_{obj}(i) < f_{obj}(gbest)$ **then**
            Update $gbest \leftarrow i$
        end if
    end for
end for
return $gbest$

F. Limitations of Standard CSO

These three limitations motivate the enhancements proposed in Section VI.

1) Problem 1: Random mode switching: In standard CSO, each cat's mode is assigned randomly at every iteration according to the Mixture Ratio. A random value $r \in [0, 1]$ is generated; if $r < MR$, the cat enters tracing mode, otherwise it enters seeking mode. This mechanism is context-blind because it ignores the actual optimization state.

For example, when $gbest$ has just changed, cats should preferably trace the new optimum to exploit it. However, random assignment may still place some cats in seeking mode, causing a missed exploitation opportunity. Conversely, when $gbest$ has remained unchanged for many iterations, more cats should seek alternative regions, but standard CSO may continue assigning them to tracing mode and waste computation around a known position. Near convergence, where most cats become similar, the fixed 70% tracing and 30% seeking split can also lead to redundant movements toward the same $gbest$.

This limitation contrasts with the Emperor Penguin Optimizer, where all penguins follow a unified movement rule whose behavior changes dynamically with temperature. There is no artificial random split between modes, which makes the search behavior more adaptive.

2) Problem 2: No intelligent stopping condition: Standard CSO has no intelligent stopping condition for `chSelection()` beyond a fixed round or iteration limit. This creates two forms of computational waste. First, the optimal CH configuration may be discovered early, for example at round 20, while the algorithm continues evaluating candidates until round 2000. Second, the search may exhaust all meaningful alternatives by an early round but still continue generating redundant configurations until the fixed limit is reached.

The key insight is that existing bio-inspired WSN clustering protocols, including CSO and EPO, do not explicitly detect when CH selection has become redundant. Addressing this gap is the primary novel contribution of the proposed 2I-CSO protocol.

3) Problem 3: Configuration space explosion: Each cat in each iteration produces a CH configuration that requires cluster formation and fitness evaluation, which are among the most expensive operations in the protocol. With an ordered representation, the number of possible configurations grows as N^k, where N is the number of sensor nodes and k is the number of selected CHs. Table V presents the combinatorial explosion of CH configuration space.

TABLE V. COMBINATORIAL EXPLOSION OF CH CONFIGURATION SPACE

Network	k	Ordered Configs	Problem
4 nodes	2	$4^2 = 16$	Manageable
10 nodes	2	$10^2 = 100$	Feasible
50 nodes	5	$50^5 \approx 3.1 \times 10^8$	Enormous
100 nodes	5	$100^5 = 10^{10}$	Intractable

Many of these configurations are redundant because the order of CHs does not change the resulting clusters. For instance, $[N_2, N_7]$ and $[N_7, N_2]$ represent the same CH set and produce identical cluster formation. Standard CSO has no mechanism to detect these repeated configurations or avoid evaluating them again.

VI. PROPOSED ENHANCEMENT: 2I-CSO

This section presents the proposed Intelligent and Interoperable Cat Swarm Optimization (2I-CSO) protocol. The objective of 2I-CSO is to improve the standard CSO search process for WSN clustering by reducing redundant CH configurations, avoiding unnecessary iterations, and adapting the swarm behavior to the current convergence state. Unlike the standard CSO, which relies on random mode assignment and a fixed stopping budget, 2I-CSO introduces a controlled search mechanism that combines configuration-space reduction, intelligent termination, adaptive mode switching, fast configuration hashing, and dynamic seeking-memory adjustment. The improvements are made to retain the optimization power of CSO, while at the same time reducing computational overhead and improving CH selection stability.

A. Overview

2I-CSO is a refined version of CSO that aims at optimizing CH selection in WSNs. Maintains the biological architecture of standard CSO (Seeking Mode for local searches and Tracing Mode for convergence) but changes the search control strategy such that it avoids redundancy and is sensitive to the state of convergence. The proposed design addresses three weaknesses of standard CSO: redundant CH configurations, fixed-budget computation, and static mode distribution.

The updated implementation introduces three main enhancement layers. First, configuration vectors are converted into canonical CH-set keys so that equivalent configurations such as $[N_2, N_7]$ and $[N_7, N_2]$ are treated as the same solution. Second, intelligent stopping is extended beyond full configuration exhaustion by adding a practical stagnation detector based on the change in global best fitness. Third, adaptive mode switching is combined with dynamic SMP decay so that 2I-CSO explores aggressively early in the search and progressively reduces expensive seeking evaluations near convergence.

B. Enhancement 1: Configuration Space Reduction

In a WSN with N active nodes and k selected CHs, a cat position represents a list of CH node IDs. Standard CSO handles this list as an ordered vector during position updates, although physically the order of CHs does not affect cluster formation. For example, $\{N_1, N_5, N_{12}\}$ and $\{N_{12}, N_1, N_5\}$ describe the same CH set and, therefore, produce the same clustering result.

The theoretical ordered search space is N^k. 2I-CSO reduces redundant tracking by converting each position into a canonical unordered configuration. In the baseline formulation, this key is obtained by sorting rounded node IDs:

$$\text{Key} = \text{tuple}\left(\text{sorted}\left(\lfloor x_i \rceil \mid x_i \in \text{position}\right)\right). \tag{8}$$

The number of unique valid configurations is then bounded by:

$$\text{Max Unique Configs} = \binom{N+k-1}{k} - N \tag{9}$$

where, the $-N$ term removes illegal configurations in which all selected CHs collapse to the same single node (see Table VI).

TABLE VI. CONFIGURATION SPACE REDUCTION EXAMPLES

Network	k	Ordered N^k	Unique Valid
4 nodes	2	16	6
10 nodes	2	100	45
100 nodes	5	10^{10}	96,560,546

C. Enhancement 2: Intelligent Stopping Condition

2I-CSO stores explored canonical configurations in a hash set $\mathcal{S}$. When the number of explored keys reaches the theoretical maximum, the CH selection engine stops because any further configuration must be redundant:

$$|\mathcal{S}| \geq \binom{N+k-1}{k} - N \Rightarrow \text{stop } \texttt{chSelection()}. \tag{10}$$

This creates a provable termination point for homogeneous search spaces and separates the expensive `chSelection()` phase from the lightweight `transmitData()` phase, which can continue until the last node dies.

In static benchmark experiments, full combinatorial exhaustion may still be difficult to reach because the valid configuration space remains large. For example, for $N = 29$ active nodes and $k = 5$ CHs, the stopping boundary is:

$$\binom{29+5-1}{5} - 29 = \binom{33}{5} - 29 = 237{,}307. \tag{11}$$

With a population of 30 cats, 500 iterations, and $SMP = 5$, the search can evaluate at most:

$$30 \times 500 \times 5 = 75{,}000 \tag{12}$$

seeking candidates. Since $75{,}000 < 237{,}307$, the complete-space stopping rule is unlikely to trigger in this benchmark setting.

To make stopping practical in benchmark and simulation settings, the updated 2I-CSO adds a delta-gbest stagnation rule:

$$\Delta f_{gbest} = |f_{gbest}^{(t-1)} - f_{gbest}^{(t)}|. \tag{13}$$

If $\Delta f_{gbest} < 10^{-6}$ in 50 successive iterations, the swarm is considered stagnant, `chSelection()` is deactivated, and the termination iteration is determined. This allows maintaining the original stopping criteria in the whole search space, with an extra option to terminate the search in large spaces.

D. Enhancement 3: Adaptive Mode Switching

Standard CSO uses random selection or tracing by setting a a fixed Mixture Ratio. In contrast, 2I-CSO selects the mode based on the condition of $gbest$:

- Active search phase: if $gbest$ changed in the previous iteration, all cats enter Tracing Mode to exploit the newly discovered promising region.
- Stable phase: if $gbest$ remains unchanged, all cats enter Seeking Mode to explore nearby alternatives and escape local stagnation.
- Initial phase: before a stable previous $gbest$ exists, cats begin with Seeking Mode to generate local diversity.

This context-aware rule keeps CSO's explicit dual-mode architecture while adopting the adaptive behavior observed in unified-motion optimizers such as EPO and SCSO [21]. Table VII presents the mode selection in Standard CSO and 2I-CSO.

TABLE VII. MODE SELECTION IN STANDARD CSO AND 2I-CSO

Optimization State	Standard CSO	2I-CSO
First iteration	Random MR split	Seeking
$gbest$ changed	Random MR split	Tracing
$gbest$ stable	Random MR split	Seeking

E. Enhancement 4: Fast Bitmask Hashing

The original canonical key based on sorting and tuple conversion is correct, but it adds overhead when called repeatedly during seeking mode. The updated implementation replaces it with an $\mathcal{O}(k)$ bitmask hash:

$$\text{Hash} = \sum_{x \in \text{position}} 2^{\lfloor x \rceil}. \tag{14}$$

For example, the CH set $\{3, 7, 12\}$ is encoded as:

$$2^3 + 2^7 + 2^{12} = 8 + 128 + 4096 = 4232. \tag{15}$$

Because addition is commutative, $[3, 7, 12]$ and $[12, 3, 7]$ generate the same key. The representation is collision-free for unique node sets because each node ID corresponds to one active bit in the integer. This avoids repeated sorting and tuple creation and improves configuration lookup efficiency.

F. Enhancement 5: Dynamic SMP Decay

The computational complexity in adaptive mode switching could be increased by switching into Seeking Mode due to the stability of $gbest$ for all cats. Given that the value of $SMP = 5$ remains fixed, this could result in numerous extra function evaluations. To avoid the above, 2I-CSO decreases the Seeking Memory Pool with respect to the iteration limit:

$$SMP_c = \max\left(2, \lfloor SMP_0 - (SMP_0 - 2)\rho \rceil\right), \quad \rho = \frac{t}{T_{max}}. \tag{16}$$

where, SMP_0 denotes the initial seeking memory pool, SMP_c the number of seeking copies currently, while ρ indicates the normalized search progress.

G. Enhancement 6: Adaptive Hybrid Mixture Ratio

The first adaptation rule had an absolute switch in that all cats were tracked whenever $gbest$ became better, and all cats searched if $gbest$ remained unchanged. Although this technique optimizes exploration or exploitation strategies, it loses the attraction aspect of the swarm when the algorithm becomes stagnated since all the cats may perform random searches without being attracted to the present gbest cat.

To preserve swarm cohesion, the optimized 2I-CSO uses an adaptive hybrid Mixture Ratio. The tracing probability is selected from the convergence state:

$$MR(t) = \begin{cases} 0.80, & \text{if } gbest \text{ changed,} \\ 0.35, & \text{if } gbest \text{ is stable} \end{cases} \tag{17}$$

In the active search phase, 80% of cats trace the newly improved leader and 20% remain in Seeking Mode for diversity. In the stagnation phase, 35% of cats maintain tracing pull toward $gbest$, while 65% perform high-pressure neighborhood search. This hybrid assignment keeps the swarm structurally aligned with the best solution while preserving the local exploration needed to escape stagnation.

H. Enhancement 7: Fitness Cache and Selective Perturbation

Because WSN fitness computation is dominated by repeated distance-energy calculations, the optimized implementation stores evaluated configurations in an exploration cache:

$$\mathcal{C} = \{\text{Hash}(\mathbf{x}) \to f(\mathbf{x})\} \tag{18}$$

Before evaluating a parent cat or a seeking copy, 2I-CSO first queries $\mathcal{C}$. If the bitmask key already exists, the cached fitness is reused in $\mathcal{O}(1)$ time; otherwise, the fitness is computed and inserted. This converts the explored-configuration structure from a pure membership set into a reusable fitness map, reducing redundant WSN evaluations during convergence plateaus.

2I-CSO also introduces selective swarm perturbation when stagnation persists for a fixed interval. The swarm is sorted by fitness, the best 50% of cats and the current $gbest$ are preserved, and only the worst 50% are relocated by randomized velocities and partial CH-node scattering. This keeps the exploitation core intact while turning weaker cats into exploration scouts that search for alternative CH basins.

I. Enhancement 8: Stagnation-Driven SMP Scaling

The linear SMP decay in Enhancement 5 reduces the seeking cost over the iteration budget. The latest implementation further adapts SMP to the stagnation depth. During active progress, SMP remains at its maximum value to preserve local-search resolution. Once a plateau is detected, the algorithm starts with a compact neighborhood and increases it only if stagnation deepens:

$$SMP_{dyn} = \max\left(2, \min\left(SMP_0, 2 + \left\lfloor \frac{s}{10} \right\rfloor\right)\right), \tag{19}$$

where, s is the consecutive stagnation counter. Thus, early plateaus are handled with a low-cost $SMP = 2$, while long stagnation gradually expands the local neighborhood up to SMP_0 to increase escape pressure.

J. Integrated Benchmarking Controls

The simulation platform provides the early termination stopping criterion as one of the benchmarks. It is off by default when there is the need to conduct an exact round-by-round comparison between CSO, EPO, and PSO, but turned on when the need is to cut down the runtime of the operations. In cases where early stopping is used, the backend stores the stop round and extends the rest of the convergence plot to ensure consistency with the sequence lengths.

The benchmark simulations will be automatically stored in the historical simulation database and can then be analyzed using the multi-run analysis interface. Thus, the proposed 2I-CSO becomes ideal for two distinct research methods – fixed-budget benchmarking and adaptive termination.

K. Performance Diagnostic

The updated implementation clarifies an important execution-time paradox. In theory, 2I-CSO should be faster than standard CSO because it can stop redundant searches. In static benchmarks, however, the full configuration boundary is often too large to reach, so the algorithm may still run for the maximum iteration budget.

Moreover, standard CSO with $MR = 0.7$ evaluates all parent cats plus only the seeking subset. With population $P = 30$ and $SMP = 5$, its approximate evaluation count per iteration is:

$$Eval_{CSO} = P + P(1 - MR)SMP = 30 + 30(0.3)(5) = 75. \tag{20}$$

When 2I-CSO enters a stable phase, all cats seek. Dynamic SMP decay controls this cost by reducing the number of copies as the iteration budget is consumed, as shown in Table VIII.

TABLE VIII. DYNAMIC SMP DECAY AND STABLE-PHASE EVALUATIONS ($P = 30, T_{max} = 500$).

Iter.	SMP_c	Copies	Eval.
0–83	5	150	180
84–250	4	120	150
251–416	3	90	120
417–500	2	60	90

Thus, 2I-CSO applies maximum local search when stagnation appears early, but progressively limits the seeking workload as convergence advances. The optimized version addresses the execution-time overhead through delta-gbest stopping, bitmask hashing, and dynamic SMP decay.

L. Complete 2I-CSO Algorithm

The complete 2I-CSO procedure is summarized in Algorithm 2. The algorithm begins by generating an initial population of candidate CH configurations and by creating an empty cache for previously evaluated solutions. Each candidate configuration is converted into a canonical bitmask key, which removes duplicate CH permutations and ensures that the same cluster-head set is not evaluated more than once. During each iteration, the algorithm evaluates the current population, updates the global best solution, and measures the change in best fitness to detect whether the search is still improving or has entered stagnation. If improvement is detected, the adaptive mixture ratio favors tracing mode to guide cats toward the current best solution; otherwise, it increases seeking mode and local perturbation to intensify exploration around promising regions. The process stops either when the maximum iteration budget is reached, when stagnation exceeds the allowed threshold, or when the cache confirms that all feasible unique CH configurations have been explored. The final global best configuration is then used for cluster formation and data transmission.

Algorithm 2. Optimized 2I-CSO Protocol Pseudocode

Input: Active nodes N, CH count k, population P, maximum iterations T_{max}
Initialize swarm cats with random CH positions
Initialize fitness cache $\mathcal{C} \leftarrow \emptyset$, $gbest \leftarrow \emptyset$, and stagnation counter $s \leftarrow 0$
Compute $max\_configs = \binom{N+k-1}{k} - N$
for $t = 1$ to T_{max} **do**
    if $|\mathcal{C}| \geq max\_configs$ or $s \geq 50$ **then**
        Deactivate `chSelection()` and record stop iteration
        break
    end if
    Evaluate all cats using cached bitmask fitness values and update $gbest$
    Compute $\Delta f_{gbest} = |f_{gbest}^{(t-1)} - f_{gbest}^{(t)}|$
    if $\Delta f_{gbest} < 10^{-6}$ **then**
        $s \leftarrow s + 1$
    else
        $s \leftarrow 0$
    end if
    Compute adaptive $MR(t)$ from the $gbest$ state
    Compute $SMP_{current}$ using iteration decay and stagnation-driven scaling
    if $s > 0$ and s is divisible by 30 **then**
        Preserve the best 50% of cats and perturb the worst 50% as exploration scouts
    end if
    for each cat c in the swarm **do**
        if $rand() < MR(t)$ **then**
            Apply Tracing Mode toward $gbest$ to maintain swarm pull
        else
            Apply Seeking Mode using $SMP_{current}$, SRD, CDC, and cached copy fitness
        end if
        Convert updated CH position to a bitmask key and store/reuse it in $\mathcal{C}$
    end for
end for
Use final $gbest$ for cluster formation and data transmission

M. Real-World WSN Operation

In static TSPLIB-style benchmarks, the active node set is fixed and the search space remains very large. In a real WSN, CH selection is executed round by round while nodes consume energy and die. Therefore, the active candidate pool N_{alive} decreases over time:

$$N_{alive}^{(r)} \leq N_{alive}^{(r-1)}. \tag{21}$$

As N_{alive} shrinks, the maximum configuration space $\binom{N_{alive}+k-1}{k} - N_{alive}$ also shrinks rapidly. For example, with $k = 5$, $N_{alive} = 15$ gives $\binom{19}{5} - 15 = 11{,}613$ unique valid configurations, while $N_{alive} = 8$ gives $\binom{12}{5} - 8 = 784$. In late network rounds, the explored set can therefore reach the complete search boundary quickly, and the intelligent stopping condition becomes increasingly effective.

The operational implication is that 2I-CSO becomes faster as the network ages. Early rounds perform deeper heuristic search when many CH choices exist, while late rounds transition into short-circuited configuration scanning, reducing the computational energy spent on CH selection and supporting longer WSN lifetime.

VII. Simulation and Implementation

This section describes the experimental environment and benchmarking protocol used to assess the proposed 2I-CSO algorithm. The evaluation is conducted through a dedicated TSPLIB benchmarking module integrated into the WSN-Simulation application. The goal is to provide a controlled and reproducible comparison between 2I-CSO and the baseline optimization algorithms under identical execution conditions.

A. Experimental Setup

All experiments were executed on a personal computer running a 64-bit Windows 10 operating system. The hardware platform was equipped with 32 GB of RAM and an Intel Core i5 14th-generation processor with a maximum clock frequency of 5.2 GHz. The same machine was used for all TSPLIB and WSN simulation runs to ensure that runtime comparisons among CSO, 2I-CSO, EPO, PSO, and LEACH-C were performed under identical computational conditions.

B. Application Platform

To support reproducible experimentation, the WSN-Simulation application was implemented using a Next.js frontend and a Python Flask backend. The frontend provides the interface for dataset selection, parameter configuration, execution control, and visual analytics, while the Flask backend implements the optimization algorithms and returns structured benchmark results. A separate TSPLIB simulation page was added to the application in order to evaluate the proposed 2I-CSO algorithm on standardized traveling-salesman benchmark instances independently from the WSN simulation modules.

C. Rationale for Using TSPLIB in WSN Clustering Validation

Despite the fact that TSPLIB was initially created as a benchmark library for TSP problems, it could be used as a spatial optimization testing ground in this research, but not as a WSN lifetime benchmark [29]. Each TSPLIB problem presents two dimensional points for nodes together with a distance matrix for those nodes. The described characteristics could be applied to the geometric aspect of the WSN clustering, where physical deployment of sensors plays an important role as well as the dependence of communication energy from the transmission distance. As a result, TSPLIB problem could be interpreted as a static sensor deployment, where cities correspond to sensors and the chosen cities serve as candidates for CHs.

On this approach, the fitness function based on TSPLIB does not compute a TSP tour. Instead, it measures how good the clustering algorithm is at forming clusters with minimized distance clustering cost. This is useful because the typical energy models used for WSN radios are functions of distance, which can be expressed in either quadratic (d^2) or quartic (d^4) form depending on the channel conditions. Hence, clustering that minimizes average/total cluster member distance to the CH will decrease communication energy cost, make clusters more spatially compact, and allow comparison between CSO, 2I-CSO, EPO, PSO, and LEACH-C algorithms using similar node placement scenarios.

TABLE IX. Energy Depletion Benchmark for 29 Nodes, 100×100 m^2, $k = 2$

Protocol	FND	HND	LND	vs. LEACH-C (%)
CSO	608	2219	**3393**	-72.8%
2I-CSO	558	2238	3374	-75.1%
EPO	847	**2524**	3241	-62.2%
PSO	962	2220	3348	-57.0%
LEACH-C	**2239**	2246	2320	0.0%

TABLE X. FND Comparison for the Current Base Station Position.

BS Position	PSO	CSO	2I-CSO	EPO	LEACH-C
Current $(50, 50)$	962	608	558	847	**2239**

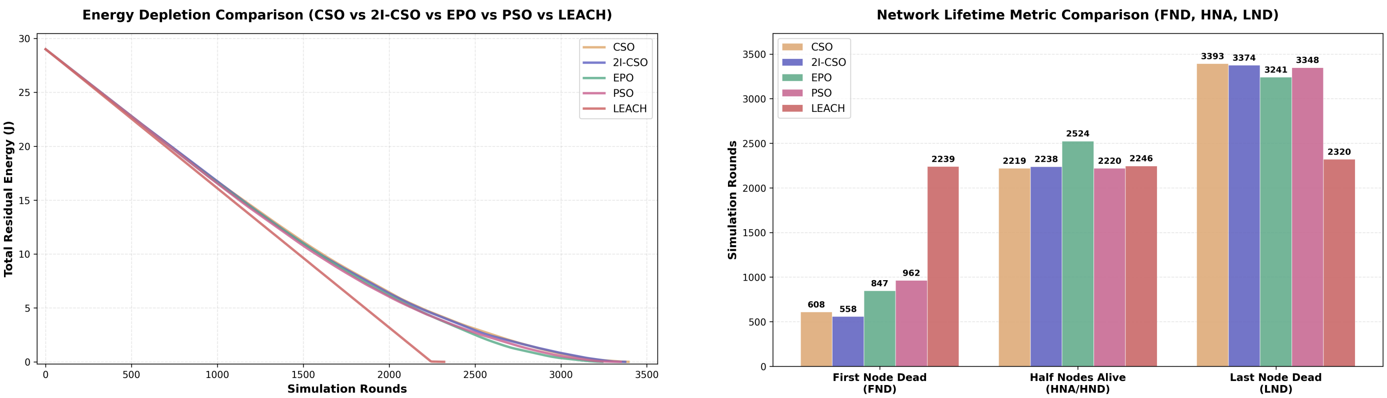
Still, TSPLIB findings need to be considered carefully. They prove that the optimizer is capable of searching large, repeatable, and distance-based CH selection spaces and of saving computation cost, convergence time, and spatial clustering fitness. Nevertheless, they are not by themselves evidence for network lifetime improvement since the TSPLIB problems lack factors such as battery depletion dynamics, traffic, CH turnover, radio interference, and node death. This is the reason why TSPLIB benchmarking is done separately as an algorithmic and geometric proof step, whereas the claims on FND, HND, LND, and lifetime are proven by the other benchmark, which follows in the next subsection.

D. WSN Energy Depletion Benchmark

Before the TSPLIB benchmark, an additional WSN-specific energy depletion test was conducted to evaluate network lifetime behavior under a direct sensor-network scenario. The test uses 29 sensor nodes deployed in a 100×100 m^2 field, with $k = 2$ cluster heads and the Base Station located at $(50, 50)$. Three standard lifetime indicators are reported: first node death (FND), half node death (HND), and last node death (LND). The percentage column compares FND against the LEACH-C baseline (see Table IX).

Interpretation: The energy depletion results show that LEACH-C achieves the latest FND, meaning that it delays the first node failure more effectively in this deployment (see Fig. 6). However, the metaheuristic protocols maintain longer post-FND operation, with CSO and 2I-CSO producing the largest LND values. This indicates a trade-off between early stability and tail lifetime: LEACH-C is stronger when the objective is to postpone the first failure, while CSO-family methods can keep the network operating for more rounds after initial depletion begins. The 2I-CSO result remains close to CSO in LND while preserving the runtime advantages shown in the TSPLIB benchmark.

The TSPLIB webpage compares 2I-CSO against CSO, PSO, EPO, and LEACH-C in an equivalent experimentation setup (see Table X). This comparison is aimed at assessing the solution quality, convergence speed, stagnation recovery, runtime efficiency, and scalability. The experiment is carried out in five runs, 1000 iterations per run, with maximum iterations



(a) Energy depletion milestones.

(b) Network lifetime metric comparison.

Fig. 6. Grouped visualization of energy depletion and network lifetime behavior.

as the stopping criterion, similar hardware/software configurations, and similar random seeds used for each algorithm. In the case of the TSPLIB benchmark test mode, the early stagnation detection stopping rule is disabled for 2I-CSO, although the program can automatically terminate the execution when the number of consecutive stagnant runs exceeds 50. Dynamic SMP (self-multiplying population) decrement is also disabled in 2I-CSO in order to compare all algorithms on the same iteration budget without any differences in stopping rules affecting the result. This makes sure that the difference in execution time reflects only the difference in algorithmic update cost per iteration among algorithms. Population scaling factor is adaptively adjusted depending on benchmark complexity: 30 agents for small instances, 40 agents for moderate instances, 50 agents for large instances, and 60 agents for extra-large instances. This scaling preserves fairness because all algorithms use the same population size within each category (see Table XI).

The benchmark suite contains the reported TSPLIB instances distributed across three groups. The small group contains att48, bayg29, bays29, burma14, dantzig42, ulysses16, and two ulysses22 benchmark runs (see Table XII). The moderate group contains berlin52, eil51, eil76, gr96, kroa100, and krob100 (see Table XIII). The large group contains bier127, ch130, ch150, d198, eil101, and gr120 (see Table XIV). Overall, the reported experimental campaign consists of:

$$20 \times 5 \text{ protocols} \times 5 \text{ runs} = 500 \text{ executions} \tag{22}$$

The recorded metrics include best fitness, execution time, convergence behavior, and Wilcoxon signed-rank significance results. The Wilcoxon test is applied at a level of significance of $p < 0.05$ to determine whether the observed best-fitness differences between protocols are statistically meaningful. Fig. 7 to Fig. 9 presents the TSPLIB convergence and multi-criteria radar comparison of three scales: small, moderate and large.

Table indicators have the following meanings. Best fitness corresponds to the best, or lowest, tour fitness obtained across the independent runs and measures optimization precision. Mean fitness stands for the average quality of solution, which is calculated as $\mu_f = (1/R)\sum_{i=1}^{R} f_i$ and implies expected behavior in the typical run. Standard deviation estimates the stability of the search process by measuring the variation of solution quality and is low for the most consistent optimization. Execution time shows the average CPU time per trial in seconds and measures computational efficiency. Stagnation indicators report the number and duration of non-improving intervals, while escape ratio measures the ability of the algorithm to resume improvement after stagnation.In the instance-level tables, each protocol column reports the final measured value, and the Winner column identifies the protocol with the lowest best fitness for that instance.

TABLE XI. TSPLIB BENCHMARK CONFIGURATION

Category	Instances	Population	Runs	Iterations
Small	8	30	5	1000
Moderate	6	40	5	1000
Large	6	50	5	1000

1) Final TSPLIB benchmark results: The final TSPLIB comparison evaluates five protocols: CSO, 2I-CSO, EPO, PSO, and LEACH-C. LEACH-C is included as a centralized clustering baseline because its BS-assisted CH selection is closer to the centralized configuration assumed by the proposed interoperable 2I-CSO model. For all tables, lower best-fitness values indicate better clustering cost, and execution time is reported in seconds per run. The last column reports the speedup of 2I-CSO relative to standard CSO.

Interpretation: The updated results show that 2I-CSO consistently reduces runtime relative to standard CSO across all reported TSPLIB groups, with speedups ranging from 3.2× to 56.6×. In terms of best fitness, 2I-CSO remains competitive with CSO and improves over EPO and PSO on several instances, especially in the large group. However, LEACH-C obtains the best fitness on all moderate and large instances. Therefore, the revised conclusion is that 2I-CSO offers a strong runtime–quality trade-off and improves the CSO)family computationally, while LEACH-C remains a strong centralized baseline for the TSPLIB-derived spatial clustering objective (see Table XV to Table XVII).

2) Wilcoxon signed-rank test results: The Wilcoxon signed-rank test was applied to the best-fitness values to evalu-

TABLE XII. SMALL-SCALE TSPLIB BEST-FITNESS RESULTS LOWER VALUES ARE BETTER

Instance	n	k	CSO	2I-CSO	EPO	PSO	LEACH-C	Winner
att48	48	6	**14420.8692**	14543.3432	14860.7903	14580.6611	14430.5470	CSO
bayg29	29	5	**3162.6316**	**3162.6316**	3215.8889	3185.4538	**3162.6316**	Tie
bays29	29	5	**3162.6316**	**3162.6316**	3224.5043	**3162.6316**	**3162.6316**	Tie
burma14	14	3	**10.5464**	**10.5464**	**10.5464**	**10.5464**	**10.5464**	Tie
dantzig42	42	6	288.9417	**287.1232**	296.3814	290.1005	288.7553	2I-CSO
ulysses16	16	4	**14.7519**	**14.7519**	**14.7519**	**14.7519**	14.8482	Tie
ulysses22	22	4	**19.8681**	**19.8681**	20.7483	**19.8681**	20.7483	Tie
ulysses22	22	4	**19.8681**	**19.8681**	**19.8681**	**19.8681**	20.7483	Tie

TABLE XIII. MODERATE-SCALE TSPLIB BEST-FITNESS RESULTS LOWER VALUES ARE BETTER

Instance	n	k	CSO	2I-CSO	EPO	PSO	LEACH-C	Winner
berlin52	52	7	2926.8931	2909.0199	3005.9378	3060.2780	**2905.1055**	LEACH-C
eil51	51	7	187.5013	188.8639	188.2857	190.5318	**183.1374**	LEACH-C
eil76	76	8	275.0143	275.9129	274.5026	276.2674	**265.0979**	LEACH-C
gr96	96	9	293.7943	291.7287	301.8480	312.0413	**291.0895**	LEACH-C
kroa100	100	10	12757.3741	12988.0102	13048.1546	13122.0496	**12385.5907**	LEACH-C
krob100	100	10	12857.5093	12887.3798	12860.5270	12972.6605	**12409.6372**	LEACH-C

TABLE XIV. LARGE-SCALE TSPLIB BEST-FITNESS RESULTS LOWER VALUES ARE BETTER

Instance	n	k	CSO	2I-CSO	EPO	PSO	LEACH-C	Winner
bier127	127	11	62254.5134	61863.7028	62015.0273	62191.2565	**60977.6240**	LEACH-C
ch130	130	11	4012.6637	3976.1627	4136.8095	4206.1308	**3785.5199**	LEACH-C
ch150	150	12	4543.9215	4542.9249	4500.0578	4551.2479	**4179.4261**	LEACH-C
d198	198	14	8005.7409	7735.9442	7794.9243	7749.7510	**7089.1803**	LEACH-C
eil101	101	10	331.0526	325.0657	334.2789	335.4407	**314.9213**	LEACH-C
gr120	120	10	932.9801	924.3486	960.8591	958.8654	**876.7315**	LEACH-C

TABLE XV. SMALL-SCALE TSPLIB EXECUTION TIME

Instance	CSO	2I-CSO	EPO	PSO	LEACH-C	Speedup
att48	18.514	2.954	2.383	3.299	11.555	6.3×
bayg29	10.179	2.263	1.373	1.859	10.491	4.5×
bays29	10.929	2.202	1.440	1.810	10.420	5.0×
burma14	3.234	0.997	0.607	0.772	9.708	3.2×
dantzig42	16.323	2.967	2.125	2.966	10.870	5.5×
ulysses16	4.917	1.149	0.759	1.106	10.164	4.3×
ulysses22	6.159	1.147	0.903	1.296	10.332	5.4×
ulysses22	6.812	1.365	1.018	1.412	10.202	5.0×

TABLE XVI. MODERATE-SCALE TSPLIB EXECUTION TIME

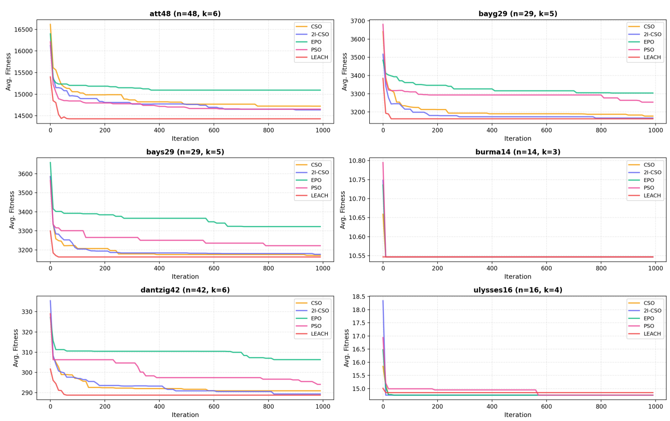
Instance	CSO	2I-CSO	EPO	PSO	LEACH-C	Speedup
berlin52	30.653	3.951	3.455	5.007	13.188	7.8×
eil51	29.213	3.693	3.305	4.711	11.981	7.9×
eil76	51.421	4.326	5.718	7.764	11.399	11.9×
gr96	73.865	4.772	7.391	10.742	12.064	15.5×
kroa100	115.141	5.183	8.766	12.184	12.226	22.2×
krob100	109.532	5.197	8.671	12.353	12.253	21.1×

TABLE XVII. LARGE-SCALE TSPLIB EXECUTION TIME

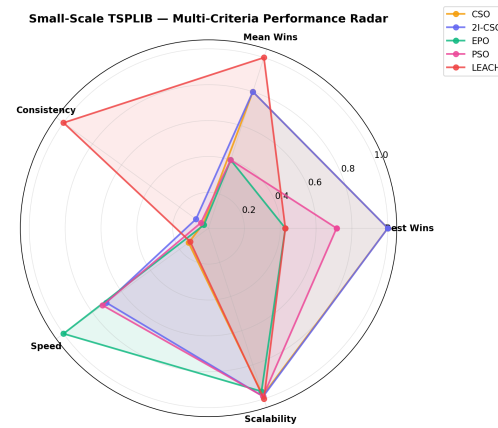
Instance	CSO	2I-CSO	EPO	PSO	LEACH-C	Speedup
bier127	144.643	7.358	14.359	20.979	29.509	19.7×
ch130	147.230	7.669	15.125	21.305	29.978	19.2×
ch150	191.401	8.517	19.370	27.815	31.345	22.5×
d198	304.949	10.689	79.614	130.372	34.434	28.5×
eil101	434.995	7.679	12.515	17.119	28.426	56.6×
gr120	142.955	7.935	14.252	19.907	28.788	18.0×

TABLE XVIII. WILCOXON SIGNED-RANK TEST RESULTS FOR TSPLIB BEST FITNESS SIGNIFICANCE THRESHOLD: $p < 0.05$

Group	**Comparison**	W	p**-value**	**Significant**	**Outcome / Winner**
Small	CSO vs 2I-CSO	2.0	0.37500	No	Statistically equivalent
Small	EPO vs 2I-CSO	0.0	0.03125	Yes	2I-CSO superior
Small	LEACH-C vs 2I-CSO	8.0	0.35938	No	Statistically equivalent
Small	PSO vs 2I-CSO	0.0	0.03125	Yes	2I-CSO superior
Moderate	2I-CSO vs CSO	9.0	0.84375	No	Statistically equivalent
Moderate	EPO vs CSO	0.0	0.03125	Yes	CSO superior
Moderate	LEACH-C vs CSO	0.0	0.03125	Yes	LEACH-C superior
Moderate	PSO vs CSO	1.0	0.06250	No	Statistically equivalent
Large	CSO vs 2I-CSO	3.0	0.15625	No	Statistically equivalent
Large	EPO vs 2I-CSO	0.0	0.03125	Yes	2I-CSO superior
Large	LEACH-C vs 2I-CSO	0.0	0.03125	Yes	LEACH-C superior
Large	PSO vs 2I-CSO	0.0	0.03125	Yes	2I-CSO superior

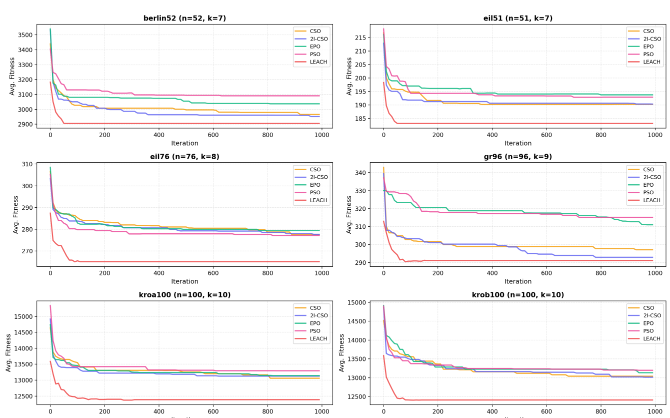


(a) Small-scale TSPLIB convergence curves: average fitness over iterations.

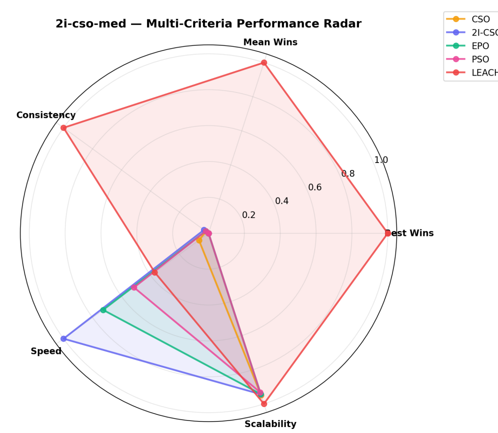


(b) Small-scale TSPLIB multi-criteria radar.

Fig. 7. Small-scale TSPLIB convergence and multi-criteria radar comparison.



(a) Moderate-scale TSPLIB convergence curves: average fitness over iterations.

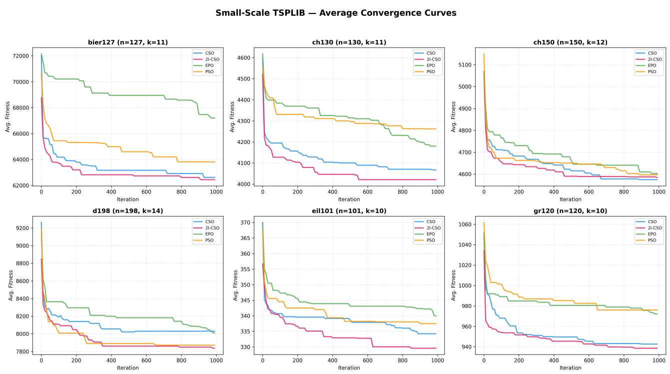


(b) Moderate-scale TSPLIB multi-criteria radar.

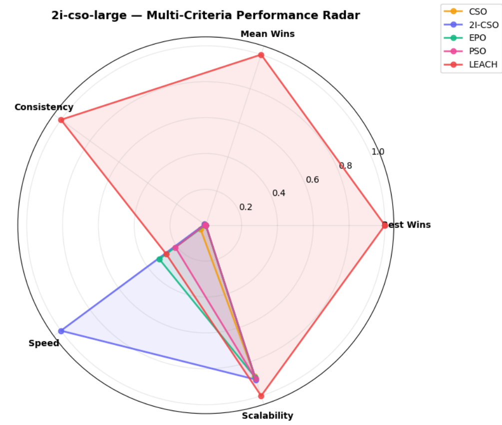
Fig. 8. Moderate-scale TSPLIB convergence and multi-criteria radar comparison.

ate whether the observed differences are statistically significant at $p < 0.05$. Table XVIII reports the final results for the small, moderate, and large TSPLIB groups. The small group uses 2I-CSO as the reference protocol. In the moderate group, the reported comparison follows the CSO-based reference used in the benchmark output. In the large group, 2I-CSO is again used as the reference protocol.

Statistical interpretation: On small instances, 2I-CSO is statistically equivalent to CSO and LEACH-C, but significantly



(a) Large-scale TSPLIB convergence curves: average fitness over iterations.



(b) Large-scale TSPLIB multi-criteria radar.

Fig. 9. Large-scale TSPLIB convergence and multi-criteria radar comparison.

better than EPO and PSO. On moderate instances, CSO and 2I-CSO are statistically equivalent, while LEACH-C is significantly better than CSO. On large instances, 2I-CSO is statistically equivalent to CSO and significantly better than EPO and PSO, but LEACH-C is significantly better than 2I-CSO. These results support a conservative interpretation: 2I-CSO improves runtime and remains competitive in fitness, but it should not be claimed to dominate LEACH-C on all TSPLIB-derived spatial benchmarks.

E. Discussion

1) Novelty of the stopping condition: The intelligent stopping criterion, as far as we know, is the first completely proven terminating approach in mathematically sound way for bio-inspired clustering of wireless sensor networks. The reason behind this is that whereas heuristic stopping criteria such as "stop if no improvement in T iterations", provide no guarantees of achieving the optimal solution, set-based monitoring guarantees that the best global solution achieved is the *true* global solution of the configuration space.

2) Synergy of enhancements: The three enhancements are not independent improvements, but form a synergistic system: Configuration Space Reduction defines the finite boundary that makes stopping possible; the Stopping Condition exploits this boundary for provable termination; and Adaptive Mode Switching accelerates convergence to reach the boundary faster. Without space reduction, the stopping threshold would be impractically large. Without adaptive switching, reaching the threshold would take longer.

3) Broader applicability: CSO has demonstrated broad applicability beyond CH selection including balanced WSN routing [18], sink node placement [25], node localization [26], underwater network partition recovery [27], target tracking [28], and NP-hard scheduling [24]. The structural nature of 2I-CSO's enhancements—operating on the search space topology rather than domain-specific fitness functions—supports their generalizability to the entire CSO family and other population-based metaheuristics.

4) Limitations:

a) Energy homogeneity requirement: The stopping condition assumes stable fitness rankings, which holds only in homogeneous networks. Heterogeneous deployments require the dynamic energy exception pathway.

b) CH candidate assumption: The formula assumes all N nodes are valid CH candidates. If certain nodes are excluded (e.g., due to location constraints), the formula must be adjusted with a reduced candidate pool $N' < N$.

c) Simulation-based validation: Results are obtained from a custom web-based simulator. Validation on a physical testbed (e.g., TelosB or MICAz motes) is necessary to confirm real-world applicability.

d) Memory scalability: When dealing with very large networks ($N > 200$, $k > 10$), Set data structure scales combinatorially and might need memory-efficient approaches (like Bloom filters, which can accept false positives).

VIII. Conclusion and Future Work

The study introduced 2I-CSO, an Intelligent and Interoperable Cat Swarm Optimization algorithm for energy-efficient Cluster Head Selection in wireless sensor network. The new approach solves many shortcomings of CSO by minimizing repetitive CH formations, incorporating a set-based intelligent termination condition, adopting a mode switching approach and unifying energy-related parameters using interoperable Base Station configuration model. The TSPLIB test cases represent a replicable spatial basis for the evaluation of searching ability, convergence characteristics, and efficiency, whereas the WSN energy exhaustion test case directly evaluates lifetime parameters such as FND, HND, and LND. The findings indicate that 2I-CSO has significantly lowered run-time relative to standard CSO and is still comparable in clustering performance,Although it appears that the lifetime study indicates that CSO-based approaches can operate for extended periods post-FND even when the delay introduced by LEACH-C in the death of the first node is greater than that achieved by 2I-CSO in the considered network. Thus, the principal advantage of the 2I-CSO approach does not consist in being superior to all existing approaches unconditionally, but rather in a balance between runtime performance and quality based on the existence of the termination criterion and a better distinction between geometric optimizer testing and network lifetime estimation.

References

[1] X. Liu, "A Survey on Clustering Routing Protocols in Wireless Sensor Networks," *Sensors*, vol. 12, no. 8, pp. 11113–11153, 2012.

[2] M. M. Afsar and M.-H. Tayarani-N, "Clustering in Sensor Networks: A Literature Survey," *J. Netw. Comput. Appl.*, vol. 46, pp. 198–226, 2014.

[3] S. K. Singh, P. Kumar, and J. P. Singh, "A Survey on Successors of LEACH Protocol," *IEEE Access*, vol. 5, pp. 4298–4328, 2017.

[4] S. Subha and H. Anandakumar, "Improved EPOA for WSN," *J. Electr. Eng. Technol.*, vol. 17, pp. 2499–2511, 2022.

[5] B. Singh and D. K. Lobiyal, "PSO-based CH selection for IoT-enabled WSN," *Future Gener. Comput. Syst.*, vol. 92, pp. 76–86, 2019.

[6] D. C. Hoang et al., "Fuzzy C-Means Clustering for WSNs," in *IEEE ISIE*, 2010.

[7] R. Elhabyan et al., "Pareto Optimization for WSN Clustering," *J. Netw. Comput. Appl.*, vol. 114, pp. 57–69, 2019.

[8] S. Arora and S. Singh, "Emperor Penguin Optimizer," *Knowl. Based Syst.*, vol. 164, pp. 334–345, 2019.

[9] G. Dhiman and V. Kumar, "EPO for Engineering Problems," *Knowl. Based Syst.*, vol. 159, pp. 20–50, 2018.

[10] Y. Gheraibia et al., "PeSOA," *Int. Arab J. Inf. Technol.*, vol. 16, no. 1, pp. 123–135, 2019.

[11] N. Kulkarni and P. Malathi, "PFuzzyACO for WSN," *J. Internet Technol.*, vol. 20, no. 4, pp. 1245–1254, 2019.

[12] A. M. Ahmed, T. A. Rashid, and S. A. M. Saeed, "Cat Swarm Optimization Algorithm: A Survey and Performance Evaluation," *Comput. Intell.*, vol. 36, no. 1, pp. 221–263, 2020.

[13] R. R. Ihsan et al., "A Survey on Cat Swarm Optimization Algorithm," *Asian J. Res. Comput. Sci.*, vol. 10, no. 2, pp. 22–32, 2021.

[14] Y. Kumar and P. K. Singh, "Improved Cat Swarm Optimization for Global Optimization and Clustering," *Appl. Intell.*, vol. 48, pp. 2681–2697, 2018.

[15] H. Singh and Y. Kumar, "Neighborhood Search Based Cat Swarm Optimization for Clustering," *Evol. Intell.*, vol. 15, pp. 1–23, 2020.

[16] S. Aramuthakannan et al., "Enhanced CH Selection Using Fuzzy Logic and Adaptive CSO," *Int. J. Intell. Eng. Syst.*, vol. 17, no. 1, pp. 336–349, 2024.

[17] D. Wategaonkar and T. R. Reshmi, "CSO-Based Energy-Efficient Reliable Sectoring Scheme in WSN," *J. Inf. Sci. Eng.*, vol. 35, pp. 1249–1268, 2019.

[18] L. Kong et al., "Balanced Power Consumption via Enhanced Parallel CSO for WSN," *Sci. Program.*, vol. 2015, 913502, 2015.

[19] S. Kalburgi and M. Manimozhi, "Taylor-SCO for Energy-Efficient CH Selection with Trust Factor in WSN," *Hindawi*, 2024.

[20] M. Basheri and M. Ragab, "Quantum CSO Clustering with Intrusion Detection for IoT," *Comput. Syst. Sci. Eng.*, vol. 46, pp. 2625–2640, 2023.

[21] A. Seyyedabbasi and F. Kiani, "Sand Cat Swarm Optimization," *Eng. Comput.*, vol. 39, pp. 2627–2651, 2022.

[22] S. S. Gowda and A. Ramalingappa, "Energy Optimized CH Selection Based on M-SCSO in UWSN," *Int. J. Intell. Eng. Syst.*, vol. 17, no. 2, pp. 304–317, 2024.

[23] S. Sindian et al., "Hybrid Dynamic Sand Cat Swarm Optimisation for WSN Resource Allocation," *IET Wirel. Sens. Syst.*, 2026.

[24] A. Bouzidi, M. E. Riffi, and M. Barkatou, "CSO for Open Shop Scheduling," *J. Ind. Manag. Optim.*, vol. 15, no. 2, pp. 733–753, 2019.

[25] V. Snasel et al., "Sink Node Placement Strategies Based on CSO," *J. Netw. Intell.*, vol. 1, no. 2, pp. 52–60, 2016.

[26] J.-S. Pan et al., "CSO-Based WSN Node Localization Algorithm," *J. Internet Technol.*, vol. 23, no. 5, pp. 949–957, 2022.

[27] M. Rao and N. K. Kamila, "CSO-Based Autonomous Recovery from Network Partitioning in UWSN," *Int. J. Syst. Assur. Eng. Manag.*, vol. 12, pp. 1125–1137, 2021.

[28] S. Tabatabaei, "CSO-Based Energy Efficient Target Tracking in WSN," *Wirel. Pers. Commun.*, vol. 137, pp. 1–24, 2024.

[29] G. Reinelt, "TSPLIB—A Traveling Salesman Problem Library," *ORSA Journal on Computing*, vol. 3, no. 4, pp. 376–384, 1991.