STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **16**, December 2026, pp 5320–5342.
Published online in International Academic Press (www.IAPress.org)

A Coverage-Preserving Dual-Graph Neural Sleep Scheduler for Wireless Sensor Networks

Ahmida Djediai [1,*], Samir Balbal [2], Amine Khaldi [1]

[1] *Laboratory of Artificial Intelligence and Information Technology (LINATI), Kasdi Merbah University, Ouargla, Algeria*
[2] *Artificial Intelligence Laboratory, Department of Computer Science, Ferhat Abbas University Setif-1, Setif, Algeria*

Abstract Sleep scheduling is a direct means of extending the lifetime of dense wireless sensor networks, but independently classifying each node as active or sleeping does not guarantee that the resulting subset preserves both sensing coverage and communication connectivity. The Dual-Graph Neural Sleep Scheduler (DGNSS) is proposed as a coverage-preserving method for dynamic sleep scheduling. The method represents a deployment by two complementary graphs: a communication graph encoding radio reachability and a coverage-overlap graph encoding sensing redundancy. Separate Graph Sample and Aggregate (GraphSAGE) encoders process the two structures, and a gated fusion module produces node-level keep scores from structural, energy, coverage-witness, articulation, and temporal duty-cycle features. Instead of applying a fixed probability threshold, the neural scores guide a deterministic feasibility-by-construction decoder. The decoder combines multi-start pruning, constructive scheduling, exact grid-coverage and sink-connectivity tests, two-for-one exchanges, temporal energy rotation, and a safe-improvement guardrail. Experiments use reproducible energy-stress scenarios with 50 and 100 nodes and compare the proposed method with Always Active, random and threshold scheduling, an energy-aware greedy scheduler, coverage-redundancy and connected-dominating-set heuristics, a Low-Energy Adaptive Clustering Hierarchy (LEACH)-like policy, a greedy oracle, the former independent-threshold decoder, and method ablations. In the 50-node scenario, DGNSS saves 50.87% energy, maintains the coverage requirement for all 140 rounds, and delays first-node death by 4.33 rounds relative to the strongest energy-aware greedy baseline. In the 100-node scenario, it saves 24.47% energy and extends coverage lifetime by one round relative to the same baseline. Additional ten-topology robustness runs show a statistically supported energy-saving gain over Energy-Aware Greedy at 50 nodes and a smaller, non-significant energy-saving trend at 100 nodes. A feature-only multilayer perceptron ranker is also competitive under the same decoder, indicating that the deterministic connected-cover decoder and coverage-aware features account for a substantial part of the observed gains. These results show that learned ranking and exact constrained decoding are complementary: the graph model supplies transferable structural preferences, whereas the decoder prevents infeasible schedules and improves the energy–lifetime trade-off.

Keywords wireless sensor networks, sleep scheduling, graph neural networks, coverage preservation, connectivity, energy efficiency, GraphSAGE, duty cycling

AMS 2010 subject classifications 68M10, 68T07, 90C27

DOI: 10.19139/soic-2310-5070-4245

1. Introduction

Wireless sensor networks (WSNs) are composed of spatially distributed, battery-powered nodes that sense physical phenomena and communicate observations to a sink. Their practical lifetime is constrained by the energy required for sensing, idle listening, reception, and packet transmission. Battery replacement may be expensive or infeasible in environmental monitoring, infrastructure inspection, industrial supervision, and remote deployments [1, 2].

*Correspondence to: Ahmida Djediai (Email: ahmida.djediai@univ-ouargla.dz). Laboratory of Artificial Intelligence and Information Technology (LINATI), Kasdi Merbah University, Ouargla, Algeria.

ISSN 2310-5070 (online) ISSN 2311-004X (print)
Copyright © 2026 International Academic Press

Energy-aware communication protocols reduce this burden, but dense sensing deployments contain an additional source of efficiency: neighboring nodes frequently provide redundant sensing and forwarding capability.

Sleep scheduling exploits this redundancy by activating only a subset of nodes during a communication round. The scheduling problem is nevertheless more restrictive than ordinary binary classification. A low-cardinality subset is useful only when it continues to cover the monitored field and when every active sensor can communicate with the sink, directly or through active relays. Coverage, connectivity, energy, and switching stability must therefore be considered jointly. Classical protocols such as Low-Energy Adaptive Clustering Hierarchy (LEACH) [3], Hybrid Energy-Efficient Distributed clustering (HEED) [4], and Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [5] reduce energy consumption through clustering or chain formation, whereas coverage-oriented schedulers explicitly reason about redundant sensors [7, 9]. These approaches remain effective baselines, but their priorities are usually expressed by fixed rules and are difficult to adapt across heterogeneous graph structures and evolving residual-energy states.

Graph neural networks (GNNs) provide a natural representation for wireless systems because their message-passing operations respect node permutation and exploit local interactions. Graph convolutional networks, Graph Sample and Aggregate (GraphSAGE), and graph attention networks have demonstrated that node representations can be learned directly from graph topology and attributes [10, 11, 12]. GNNs have also been applied to wireless resource allocation because their parameter count can remain independent of network size and their computations can transfer across related graph instances [13, 14]. However, a neural output alone does not satisfy a combinatorial coverage-and-connectivity constraint. The initial implementation investigated in this work illustrates this limitation: independent thresholding caused nearly all node probabilities to fall on the active side of the decision boundary, making the learned method behaviorally equivalent to an Always Active policy.

The proposed method addresses that failure by separating *ranking* from *feasibility*. The Dual-Graph Neural Sleep Scheduler (DGNSS) uses neural inference to rank the utility of retaining each node, while a deterministic decoder constructs and improves an executable connected-cover subset. Two graphs are used because communication adjacency and sensing redundancy represent different physical relations. The communication graph identifies potential forwarding paths; the coverage-overlap graph identifies sensors whose sensing disks cover similar regions. Their embeddings are fused with residual-energy and temporal duty-cycle information before constrained decoding.

The principal contributions are summarized as follows:

- A dual-graph representation that distinguishes communication reachability from sensing overlap and augments conventional energy and degree features with coverage-witness, articulation-risk, sink-hop, duty-cycle, and active-streak descriptors.
- A gated dual-GraphSAGE architecture and a constraint-aware training objective that combines supervised oracle imitation with differentiable coverage, pairwise ranking, cardinality calibration, energy, active-ratio, and switching terms.
- A feasibility-by-construction decoder integrating multi-start pruning, energy-aware constructive candidates, exact connected-coverage verification, two-for-one exchanges, temporal energy rotation, and a safe-improvement guardrail.
- A reproducible evaluation framework that exports checkpoints, Open Neural Network Exchange (ONNX) and TorchScript models, topology and schedule visualizations, round-level and node-level data, confidence intervals, paired comparisons, ablations, and offline Objective Modular Network Testbed in C++ (OMNeT++) decision files.
- An empirical assessment showing that DGNSS improves the energy–lifetime trade-off over a strong energy-aware greedy baseline at 50 and 100 nodes, while the independent-threshold version remains substantially weaker.

The remainder is organized as follows. Section 2 reviews energy-aware sleep scheduling and graph learning for wireless systems. Section 3 defines the network model and optimization objective. Section 4 presents DGNSS. Section 5 describes the experimental protocol, and Section 6 reports the results and ablations. Section 7 discusses limitations and threats to validity, followed by the conclusion in Section 8.

2. Related work

Research on sleep scheduling for WSNs lies at the intersection of energy-aware communication, coverage-preserving activation, joint coverage–connectivity maintenance, and learning-based graph decision making. Classical clustering protocols established the importance of energy balancing. LEACH introduced randomized cluster-head rotation to distribute long-range communication costs among nodes [3]. HEED subsequently selected cluster heads using residual energy as the primary criterion and communication cost as a secondary criterion [4]. PEGASIS organized nodes into a chain so that each sensor communicated with a nearby neighbor while the chain leader rotated over time [5]. Later LEACH variants, such as T-LEACH, further adapted cluster-head selection to residual energy and threshold-sensitive operation [15], while surveys of LEACH-based protocols show that most improvements target cluster formation, cluster-head rotation, or intra-cluster communication efficiency [16]. These protocols reduce communication energy, but they do not directly decide which redundant sensors can sleep while preserving area coverage.

Recent studies further underline the same energy-management pressure in IoT and WSN systems. Bouarourou et al. optimized WSN routing with a whale-optimization pathfinding model and reported gains in lifetime, active-node count, delivery, and delay [42]. Ouaarouss et al. studied data offloading in IoT-based WSNs, combining resource constraints, clustering, LSTM prediction, and KNN classification to reduce residual-energy depletion and sensor failures [43]. These works address routing and offloading rather than coverage-preserving sleep scheduling, but they reinforce the need to couple energy use with network-level service constraints.

Coverage-preserving sleep scheduling addresses this complementary problem by activating only a subset of nodes that is sufficient to monitor the region of interest. Wen and Chen proposed dynamic hierarchical sleep scheduling to exploit node redundancy while maintaining sensing and communication requirements [7]. Early local schemes detect redundancy from geometric or probabilistic criteria. Yufu and HongJun used a coverage-preserving node-scheduling procedure with a local coverage test and collision-avoidance mechanism to prevent holes during state transitions [17]. Qu et al. introduced an energy-efficient coverage-preserving scheduler in which local wake-up decisions are triggered by energy-related conditions [18]. Bulut and Korpeoglu estimated expected common coverage between neighboring sensing disks and used this estimate to guide distributed sleep decisions under coverage and connectivity requirements [19]. Other approaches formulate activation as an optimization problem. Dineshkumar et al. modeled sleep scheduling as an energy-aware coverage problem with disjoint set covers and solved it using evolutionary metaheuristics [20]. Wu et al. combined node utility and critical-target prioritization to balance residual energy with coverage contribution [21]. Geometric and redundancy-elimination schemes similarly use local positions, overlap, and inter-node distance to identify nodes that can sleep without degrading coverage [22, 23]. Although these methods are lightweight and often distributed, their decisions are primarily local; therefore, full coverage and sink connectivity are usually guaranteed only under restrictive geometric assumptions or by additional repair mechanisms.

A second line of work treats coverage and connectivity as coupled constraints. Tian and Georganas showed that, in a connected deployment, coverage preservation can imply connectivity when the communication radius is at least twice the sensing radius, and they extended this reasoning to k-coverage and connectivity [24]. Cardei and Cardei formalized the connected set cover problem for WSNs, where each activation set must cover the monitored targets while keeping active nodes connected to the base station, and showed that the problem is computationally hard [25]. He et al. considered duty-cycled event-monitoring networks in which link availability is transient and jointly addressed coverage and connectivity under sleep scheduling [26]. These studies motivate the constraint model adopted in DGNSS: a feasible schedule should be judged not only by the number of active nodes or their residual energy, but also by whether every monitored location remains covered and whether active sensors remain connected to the sink.

Several structural heuristics are closely related to this feasibility requirement. Connected Dominating Set (CDS) methods construct a small active backbone such that each inactive node is adjacent to at least one active node and the active nodes remain connected. Chandra et al. proposed a distributed target-coverage CDS method based on prime-numbered node identifiers and adjustable sensing radius [27]. Ingelrest improved classical two-hop-neighbor CDS construction by producing smaller connected dominating sets while preserving the required

domination and connectivity properties [28]. A complementary family of k-coverage redundancy algorithms identifies sensors whose sensing regions are already sufficiently covered by neighbors. Chenait et al. proposed a linear-time redundancy model for homogeneous WSNs [29] and later extended the idea to heterogeneous IoT sensor deployments using locus-based redundancy determination [30]. In the present work, CDS and coverage-redundancy rules are useful baselines and provide interpretable structural signals, but DGNSS differs by combining such structural information with learned ranking and exact feasibility checks.

Learning-based sleep scheduling has been explored to improve adaptation under dynamic energy, traffic, and topology conditions. Sinde et al. combined clustering, Deep Reinforcement Learning (DRL)-based duty cycling, and routing optimization to improve WSN lifetime, throughput, and delay [31]. Verma et al. modeled each node as a reinforcement-learning agent that selects among transmit, listen, and sleep actions instead of following a fixed duty cycle [32]. Vinutha used a Deep Q-Network scheduler to learn energy-aware sleep decisions [33]. Xing et al. applied DRL to agricultural WSNs by predicting suitable wake-up times from historical sensing data [34]. In industrial IoT, Jurado-Lasso et al. used hierarchical reinforcement learning for multi-objective TSCH scheduling [35]. These methods demonstrate the value of learned scheduling, but they generally rely on interaction-intensive training and probabilistic action selection. More importantly, a learned node action does not automatically ensure that the resulting active set forms a connected cover at every round.

GNNs provide a natural learning model for WSNs because sensor deployments are graphs with variable size, local interactions, and permutation-equivariant structure. The graph convolutional network of Kipf and Welling [10], the inductive neighborhood aggregation mechanism of GraphSAGE [11], and masked graph attention [12] are widely used message-passing architectures, and PyTorch Geometric provides efficient sparse implementations for such models [36]. A recent study by Abdulghani and Shukur used graph convolutional networks for spatio-temporal climate imputation, showing the value of graph structure when observations are tied by spatial relations rather than being independent samples [44]. In wireless optimization, Eisen and Ribeiro used random-edge GNNs for wireless resource allocation and showed transfer across network instances [13], while Shen et al. studied scalable GNN architectures for radio-resource management [14]. More specific wireless scheduling and routing studies also support the relevance of graph learning: Vo et al. applied graph attention to IoT data-aggregation scheduling [37]; Xu et al. embedded GNN representations in a DRL routing policy for WSN topologies affected by node failures [38]; and Lu et al. used GraphSAGE to score multipath routes in wireless mesh networks [39]. These works show that GNNs can encode wireless interaction structure, but they mainly target link scheduling, routing, aggregation, resource allocation, or sink management rather than coverage-preserving sleep scheduling.

The design of DGNSS is also related to graph-based learning for combinatorial optimization. Khalil et al. showed how graph embeddings and reinforcement learning can guide greedy construction for problems such as minimum vertex cover, maximum cut, and the traveling salesperson problem [40]. Darvariu et al. surveyed graph reinforcement learning for combinatorial optimization and emphasized the general pattern of using neural models to rank graph elements while a constructive algorithm commits to feasible decisions [41]. DGNSS follows this safer learn-to-rank philosophy. It uses dual GraphSAGE encoders to separately process the communication graph and the sensing-overlap graph, fuses the two structural views into node-level keep scores, and then uses these scores only as priorities inside a deterministic constrained decoder. The decoder performs exact grid-coverage and sink-connectivity checks, applies multi-start pruning and two-for-one exchanges, and incorporates temporal energy balancing through a safe-improvement guardrail. Therefore, the neural model is not trusted to output a feasible binary schedule directly. This separates DGNSS from local coverage heuristics, clustering-based energy balancing, unconstrained RL schedulers, and single-graph GNN wireless optimizers. The contribution is the integrated methodology: dual structural learning for ranking, followed by deterministic connected-cover construction and energy-aware temporal rotation.

3. System model and problem formulation

This section formalizes the sensing environment, the two graph structures, the coverage and connectivity requirements, and the round-level energy dynamics considered in this study. The network is modeled as a

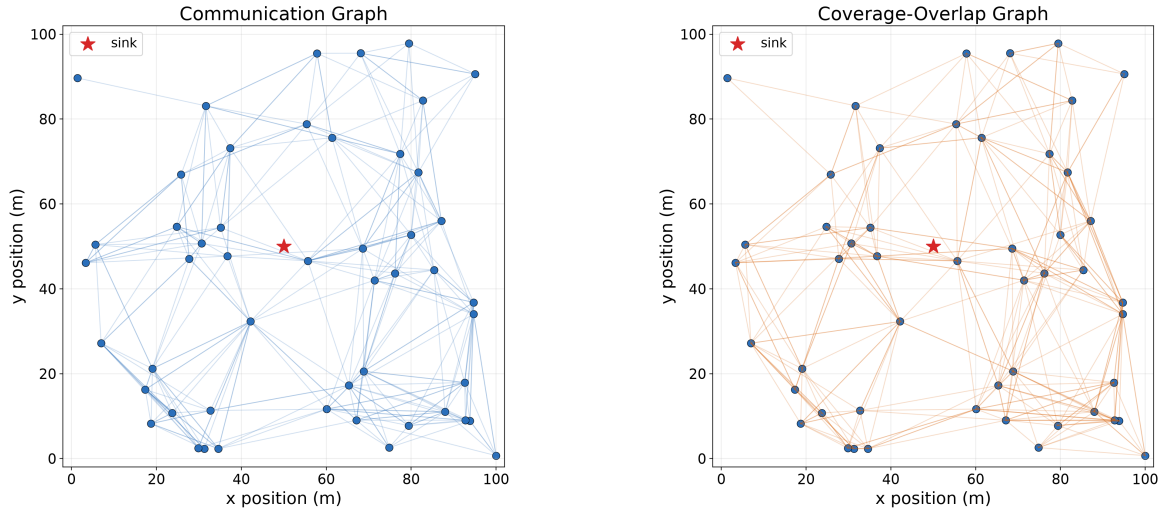


Figure 1. Communication and sensing-overlap graphs.

static battery-powered deployment in which a centralized scheduler selects an active subset at each round. The resulting subset must provide the prescribed sensing coverage while maintaining a communication path from every active node to the sink. On this basis, the sleep-scheduling task is formulated as a constrained energy-balancing problem whose solution must account for spatial redundancy, forwarding importance, residual energy, and temporal activation history.

3.1. Deployment and dual graphs

Let $\mathcal{V} = \{1, \dots, N\}$ denote a set of static sensor nodes deployed in a rectangular region $\mathcal{A} \subset \mathbb{R}^2$. Node i has position $\mathbf{p}_i = (x_i, y_i)$, initial energy E_i^0, and residual energy $E_i(t)$ at round t. A fixed sink s is located at $\mathbf{p}_s$. The sensing and communication radii are denoted by R_s and R_c, respectively.

The communication graph is

$$\mathcal{G}_c = (\mathcal{V}, \mathcal{E}_c), \qquad (i,j) \in \mathcal{E}_c \iff \|\mathbf{p}_i - \mathbf{p}_j\|_2 \le R_c. \tag{1}$$

An edge is associated with normalized distance and a distance-based link-quality proxy,

$$a_{ij}^c = \left[\frac{d_{ij}}{R_c}, \exp\left(-\left(\frac{d_{ij}}{R_c} \right)^2 \right) \right]. \tag{2}$$

The coverage-overlap graph is

$$\mathcal{G}_o = (\mathcal{V}, \mathcal{E}_o), \qquad (i,j) \in \mathcal{E}_o \iff \|\mathbf{p}_i - \mathbf{p}_j\|_2 \le 2R_s. \tag{3}$$

Its edge attributes are normalized distance and the intersection-over-union ratio of the two sensing disks. Figure 1 illustrates that the graphs encode related but non-identical neighborhoods. The sink is marked by a red star in both panels.

3.2. Coverage and connectivity constraints

The monitored region is discretized into a set of grid points $\mathcal{Q} = \{1, \dots, M\}$. The binary sensing matrix is

$$C_{iq} = \begin{cases} 1, & \|\mathbf{p}_i - \mathbf{q}\|_2 \le R_s, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Let $z_i(t) \in \{0,1\}$ denote the state of node i, where one represents active and zero represents sleeping. The achieved coverage ratio is

$$\rho_{\text{cov}}(t) = \frac{1}{M} \sum_{q=1}^{M} \mathbb{I}\left[\sum_{i=1}^{N} z_i(t) C_{iq} > 0 \right]. \tag{5}$$

For a prescribed threshold η, a valid schedule must satisfy

$$\rho_{\text{cov}}(t) \geq \eta. \tag{6}$$

Connectivity is evaluated on the subgraph induced by the active nodes, augmented with the sink and sink–sensor edges whose lengths do not exceed R_c. If $\mathcal{R}_s(t)$ denotes the set of active nodes reachable from the sink, the connectivity ratio is

$$\rho_{\text{con}}(t) = \frac{|\mathcal{R}_s(t)|}{\max(1, \sum_i z_i(t))}. \tag{7}$$

The connected-cover requirement is $\rho_{\text{con}}(t) = 1$.

3.3. Round-level energy model

For a packet of L bits, the active energy of node i is modeled as

$$e_i^{\text{act}}(t) = e_{\text{sense}} + e_{\text{idle}} + e_{\text{rx}} + L\left(e_{\text{elec}} + e_{\text{amp}} d_{i,s}^{\alpha}\right), \tag{8}$$

where the transmission distance is capped by R_c in the simulator, and α is the path-loss exponent. Sleeping nodes consume e_{sleep}. Residual energy evolves as

$$E_i(t+1) = \max\left(0, E_i(t) - z_i(t) e_i^{\text{act}}(t) - (1 - z_i(t)) e_{\text{sleep}}\right). \tag{9}$$

3.4. Scheduling objective

At each round, the scheduler seeks a feasible active set that is small, energy balanced, and temporally stable. A representative instantaneous objective is

$$\begin{aligned} \min_{\mathbf{z}(t)} \quad & \sum_i z_i(t) + \lambda_E \sum_i z_i(t) \left(1 - \frac{E_i(t)}{E_i^0}\right) \\ & + \lambda_D \sum_i z_i(t) D_i(t) + \lambda_S \sum_i |z_i(t) - z_i(t-1)|, \end{aligned} \tag{10}$$

subject to Equation (6) and $\rho_{\text{con}}(t) = 1$. Here $D_i(t)$ denotes the historical duty fraction. The exact binary optimization is combinatorial; DGNSS learns a structural ranking and then solves a constrained approximation through deterministic search.

4. Proposed DGNSS method

Figure 2 presents the complete method. The topology and current round state are transformed into the two graphs and a node-feature matrix. Two GraphSAGE encoders produce relation-specific embeddings. Gated fusion generates node keep scores, after which a multi-stage decoder constructs the schedule. Exact coverage and sink-connectivity checks are applied throughout decoding rather than only after thresholding.

4.1. Node features

For every node, the model uses a 14-dimensional feature vector. Table 1 groups these features by purpose.

The unique-coverage feature is particularly important. A grid point is a *coverage witness* of node i when i is currently its only covering sensor. Removing a node with many witnesses has a high probability of violating Eq. (6). Similarly, articulation-point and sink-hop features identify nodes with potential forwarding significance.

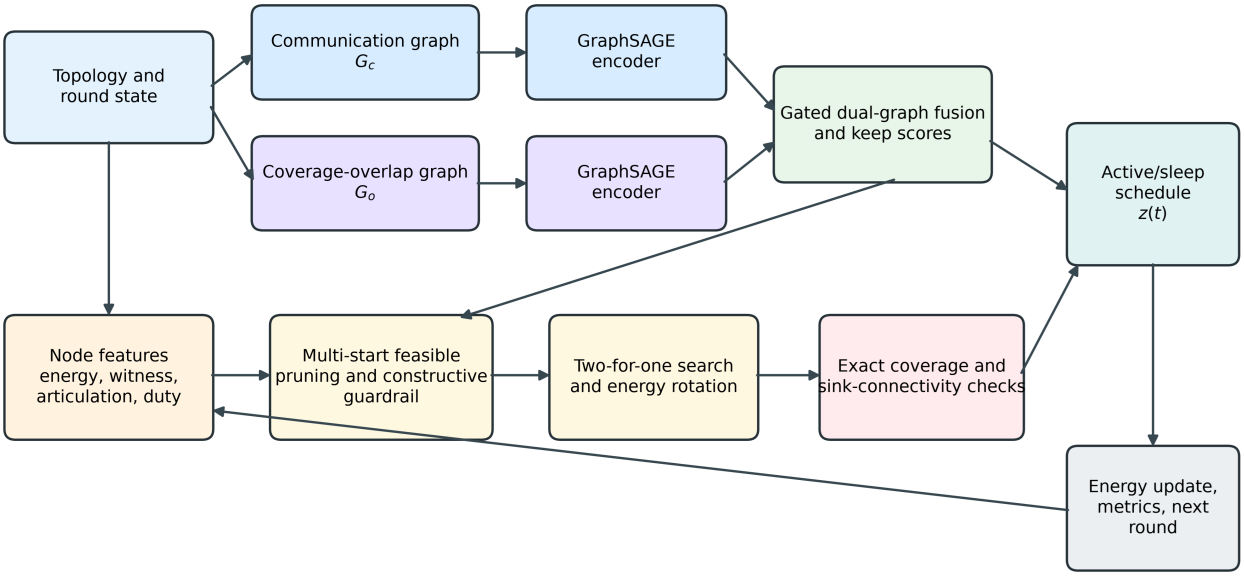


Figure 2. DGNSS scheduling architecture.

Table 1. DGNSS node features.

Feature group	Components
Energy	normalized residual energy; residual-energy percentile
Communication structure	normalized communication degree; normalized sink distance; normalized sink-hop distance; mean link quality; articulation-point indicator
Coverage structure	normalized overlap degree; sensing redundancy; sensing contribution; unique-coverage witness contribution
Temporal state	previous active state; cumulative duty cycle; normalized consecutive-active streak

Table 2. Neural and decoder dimensions.

Stage	Input	Output
Node features	node state	$N \times 14$
Communication encoder	$N \times 14$, $\mathcal{E}_c$	$N \times H$
Overlap encoder	$N \times 14$, $\mathcal{E}_o$	$N \times H$
Graph gate	$N \times (2H + 14)$	$N \times 2$
Fusion concatenation	gated embeddings, difference, features	$N \times (3H + 14)$
Fusion multilayer perceptron	$N \times (3H + 14)$	$N \times 1$
Sigmoid layer	$N \times 1$ logits	N keep scores
Constrained decoder	scores, topology, coverage state	N binary states

4.2. Dual-graph encoding and gated fusion

Figure 3 details the neural input–output pathway and the tensor dimensions used by the model. The communication and coverage-overlap branches share the same node-feature matrix but use different edge sets and independent trainable parameters. Table 2 summarizes the corresponding dimensional transformations.

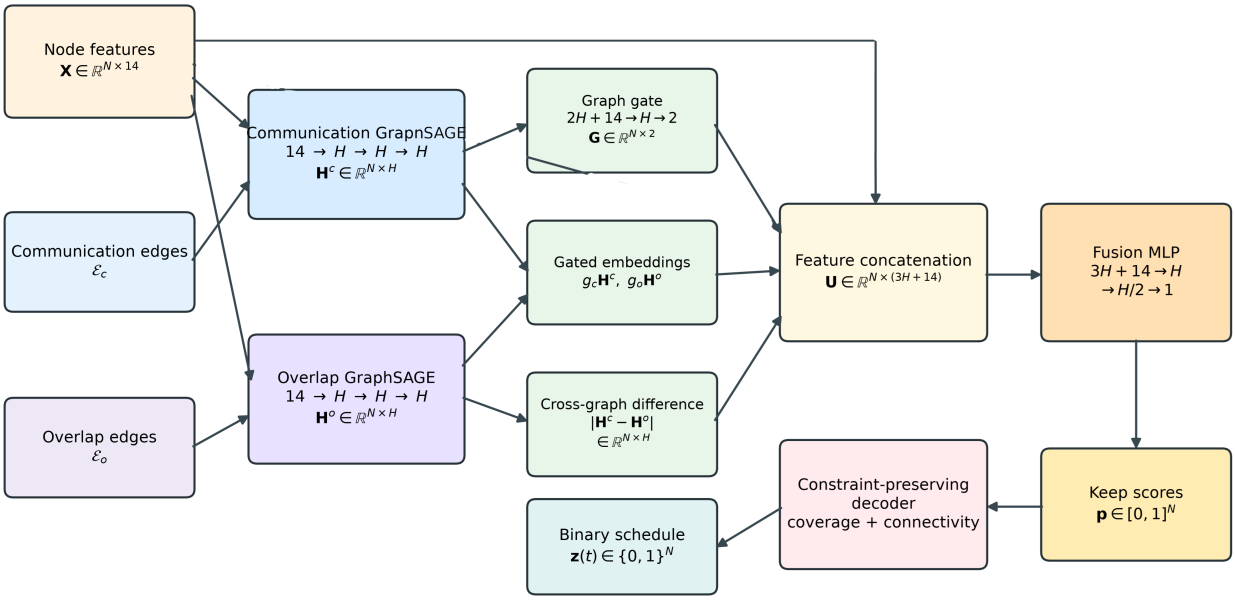


Figure 3. DGNSS neural architecture.

Let $\mathbf{X} \in \mathbb{R}^{N \times F}$ denote the node-feature matrix, where $F = 14$. Separate GraphSAGE encoders process the two graphs:

$$\mathbf{H}^c = \mathrm{SAGE}_c(\mathbf{X}, \mathcal{E}_c), \qquad \mathbf{H}^o = \mathrm{SAGE}_o(\mathbf{X}, \mathcal{E}_o). \tag{11}$$

For node i, a learned gate is computed as

$$\mathbf{g}_i = \sigma\left(\mathbf{W}_g[\mathbf{h}_i^c \| \mathbf{h}_i^o \| \mathbf{x}_i] + \mathbf{b}_g\right), \tag{12}$$

where $\mathbf{g}_i \in (0,1)^2$. The fused vector is

$$\mathbf{u}_i = \left[g_{i,c}\mathbf{h}_i^c \| g_{i,o}\mathbf{h}_i^o \| |\mathbf{h}_i^c - \mathbf{h}_i^o| \| \mathbf{x}_i\right]. \tag{13}$$

A multilayer perceptron (MLP) maps $\mathbf{u}_i$ to a logit ℓ_i, and $p_i = \sigma(\ell_i)$ is interpreted as a *keep score*. It is not interpreted as an independently feasible binary decision.

4.3. Oracle supervision and constraint-aware loss

Training labels are generated by a greedy connected-cover oracle. Starting with all alive nodes, the oracle orders removal candidates using sensing redundancy, energy, sink distance, sensing contribution, and betweenness centrality. A removal is accepted only when both coverage and connectivity remain feasible.

The oracle is heuristic rather than optimal. Its sleep score is

$$s_i^{\mathrm{sleep}} = 2.0\, r_i + 0.8(1 - e_i) + 0.2\, d_i - 1.4\, c_i - 1.2\, b_i, \tag{14}$$

where r_i is normalized sensing redundancy, e_i is normalized residual energy, d_i is normalized sink distance, c_i is normalized sensing contribution, and b_i is normalized betweenness centrality in the alive-node communication graph. The coefficients were fixed as an energy-aware removal heuristic: high redundancy, low residual energy, and larger sink distance increase the tendency to sleep, whereas high sensing contribution and communication centrality protect nodes that are likely to be coverage witnesses or relays. Because every proposed removal is still checked by exact coverage and connectivity tests, these weights affect candidate order but cannot create infeasible labels.

The supervised objective combines binary cross-entropy (BCE) with coverage, activity, energy, switching, ranking, and cardinality terms:

$$\begin{aligned}\mathcal{L} = \mathcal{L}_{\mathrm{BCE}} + \lambda_{\mathrm{cov}}\mathcal{L}_{\mathrm{cov}} + \lambda_{\mathrm{act}}\mathcal{L}_{\mathrm{act}} + \lambda_{\mathrm{eng}}\mathcal{L}_{\mathrm{eng}} \\ + \lambda_{\mathrm{sw}}\mathcal{L}_{\mathrm{sw}} + \lambda_{\mathrm{rank}}\mathcal{L}_{\mathrm{rank}} + \lambda_{\mathrm{card}}\mathcal{L}_{\mathrm{card}}.\end{aligned} \tag{15}$$

The differentiable coverage estimate is

$$\widetilde{\rho}_{\mathrm{cov}} = \frac{1}{M}\sum_{q=1}^{M}\left(1 - \prod_{i=1}^{N}(1 - p_i C_{iq})\right), \tag{16}$$

and

$$\mathcal{L}_{\mathrm{cov}} = \max(0, \eta - \widetilde{\rho}_{\mathrm{cov}}). \tag{17}$$

The active loss is the mean keep probability, the energy loss penalizes selecting depleted nodes, and the switching loss penalizes deviation from the previous state. Pairwise ranking encourages oracle-active nodes to receive higher logits than oracle-sleeping nodes:

$$\mathcal{L}_{\mathrm{rank}} = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} \log\left(1 + \exp[-(\ell_i - \ell_j)]\right), \tag{18}$$

where i is oracle-active and j is oracle-sleeping. Cardinality calibration aligns the mean probability with the oracle active fraction.

4.4. Feasibility-by-construction decoder

The decoder begins with a learned removal priority that combines low keep probability, low residual energy, sensing redundancy, accumulated duty, active streak, prior state, and witness protection. Multiple candidate schedules are then generated:

1. learned, energy, redundancy, hybrid, and perturbed removal orders;
2. energy-aware constructive schedules that add nodes according to uncovered-grid gain and residual energy;
3. a model-independent constructive guardrail matching the strongest greedy baseline;
4. a warm start from the previous round between full re-optimization rounds.

Every deletion is checked using exact grid coverage and graph reachability. Feasible candidates are ranked first by cardinality, then by coverage margin and temporal energy objective. A two-for-one local search replaces two active nodes with one sleeping node whenever feasibility is retained. Equal-cardinality one-for-one exchanges rotate the burden toward nodes with higher energy and lower historical duty. Finally, the learned candidate is compared with the constructive guardrail. It is accepted only if it uses no more nodes and does not lose more than a configured coverage-margin tolerance.

Algorithm 1 summarizes the process.

4.5. Computational considerations

Graph construction is $O(N^2)$ for the direct distance implementation, while a fixed-depth GNN forward pass is $O(|\mathcal{E}_c| + |\mathcal{E}_o|)$. Candidate pruning performs repeated coverage updates and connectivity tests. In the worst case, repeated graph traversals yield approximately $O(KN(|\mathcal{E}_c| + M))$ for K starts. The local exchange phase is restricted to small candidate lists, preventing an exhaustive combinatorial search. This design favors deterministic feasibility and reproducibility over minimum inference latency; spatial indexing and incremental connectivity structures remain possible optimizations.

Algorithm 1 DGNSS decoder.

Require: keep scores $\mathbf{p}$, topology, coverage matrix $\mathbf{C}$, previous state, duty history
Ensure: active/sleep vector $\mathbf{z}(t)$
compute learned removal priorities
generate multi-start pruning and constructive candidates
for each candidate **do**
    accept a node deletion only if coverage $\geq \eta$ and all active nodes reach the sink
end for
include the energy-aware constructive guardrail
select the minimum-cardinality feasible candidate
perform feasible two-for-one exchanges
perform feasible energy-rotation exchanges
if candidate is larger than the guardrail or loses excessive coverage margin **then**
    replace candidate by guardrail
end if
return the verified schedule

5. Experimental methodology

The evaluation protocol is designed to test whether learned ranking and exact constrained decoding improve the energy–coverage trade-off under reproducible WSN settings. This section specifies the implementation, simulation regimes, training configuration, baselines, metrics, and OMNeT++ schedule-replay path.

5.1. Implementation and reproducibility

The implementation uses Python, NumPy, pandas, NetworkX, PyTorch, and PyTorch Geometric [36]. GraphSAGE is the principal encoder; native PyTorch implementations of a graph convolutional network (GCN) and a graph attention network (GAT) are also provided as dependency-light fallbacks. The experiment pipeline generates topologies, builds both graphs, constructs oracle labels, trains the model, evaluates baselines and ablations, exports publication figures and tables, and writes round–node decision files for offline OMNeT++ replay.

For every publication figure, the numerical values are exported to a same-name comma-separated values (CSV) file. Additional artifacts include training histories, validation predictions, round metrics, node-state time series, lifetime events, duty-cycle statistics, paired-improvement tables, decoder traces, model metadata, PyTorch checkpoints, TorchScript models, ONNX models, topology coordinates, graph edges, and artifact manifests.

5.2. Scenarios and hyperparameters

Table 3 reports the two energy-stress scenarios. The initial energies were selected so that node deaths and coverage degradation occur within a tractable simulation horizon. The scenarios therefore evaluate scheduling quality rather than representing a specific commercial sensor platform.

The energy parameters common to both scenarios are: packet length 4096 bits, sensing energy 0.0010 J, idle energy 0.0008 J, reception energy 0.0006 J, sleep energy 0.00008 J, electronic transmission energy 50 nJ/bit, amplifier energy 100 pJ/bit/m^2, and path-loss exponent two.

Additional robustness experiments were run without replacing the primary 50-node and 100-node scenarios. The trained 50-node and 100-node GraphSAGE checkpoints were re-evaluated on ten independent test topologies using the same topology-seed sequence extended beyond the original three and two deployments. The robustness runs report paired t-test and Wilcoxon signed-rank p-values for matched topology seeds. A feature-only multilayer perceptron (MLP) ranker was trained with the same node features and loss terms, then passed to the same constrained decoder, to separate the contribution of graph message passing from feature engineering and deterministic decoding. A limited 50-node loss-weight sensitivity check varied $\lambda_{\mathrm{cov}} \in \{0.60, 1.80\}$ and $\lambda_{\mathrm{rank}} \in$

Table 3. Experimental settings.

Parameter	50-node scenario	100-node scenario
Deployment area	100×100 m^2	200×200 m^2
Deployment	random	random
Initial energy	0.28 J	0.35 J
Energy jitter	10%	10%
Sensing radius R_s	18 m	25 m
Communication radius R_c	35 m	55 m
Coverage grid	26×26	28×28
Coverage threshold η	0.90	0.92
Training/validation topologies	32/8	20/5
Test topologies	3	2
Simulation rounds	140	160
GraphSAGE hidden dimension	64	80
GraphSAGE layers	3	3
Dropout	0.15	0.15
Training epochs	12	10
Decoder multi-starts	8	8
Re-optimization period	8 rounds	10 rounds

$\{0.15, 0.45\}$ over five test topologies. A controlled 200-node run used the 100-node area, radii, energy, threshold, training budget, and horizon while changing only the number of sensors to 200; this run tests computational feasibility in a denser deployment and is not used to claim broad large-scale generalization.

5.3. Training and optimization details

Table 4 reports the model-training and decoder hyperparameters used in the primary experiments. Both models were trained with the Adaptive Moment Estimation (Adam) optimizer and gradient-norm clipping. The same constraint-aware loss structure was used at both scales, while the hidden dimension, number of training topologies, number of validation topologies, number of epochs, and decoder re-optimization period were adjusted to the network size. Model selection used the validation F1 score, defined as the harmonic mean of precision and recall.

5.4. Baselines and ablations

The evaluation includes the following baselines:

- **Always Active**: every alive node remains active.
- **Random Sleep**: nodes sleep probabilistically, followed by constraint repair.
- **Energy Threshold**: sufficiently charged nodes are selected, followed by repair.
- **Energy-Aware Greedy**: nodes are added by uncovered-area gain, energy, and sink distance, followed by connected-coverage repair.
- **Coverage Redundancy**: highly redundant nodes are removed while preserving constraints.
- **Connected Dominating Set (CDS) Approximation**: a connected-dominating-set-inspired score constructs a communication backbone, followed by coverage repair.
- **LEACH-like**: energy-biased randomized head selection followed by connected-coverage repair.
- **Greedy Oracle**: the same connected-cover heuristic used to produce supervised labels, recomputed from the current state.

The legacy method independently thresholds GNN probabilities and applies the former repair stage. Ablations remove local exchange search, temporal duty features and energy rotation, or coverage-witness protection. The no-witness variant is available for the 50-node analysis.

Table 4. Training and decoder settings.

Parameter	50-node scenario	100-node scenario
Optimizer	Adam	Adam
Learning rate	1×10^{-3}	1×10^{-3}
Weight decay	1×10^{-5}	1×10^{-5}
Gradient-norm clipping	5.0	5.0
Training epochs	12	10
Training topologies	32	20
Validation topologies	8	5
Model selection criterion	validation F1 score	validation F1 score
Graph encoder	GraphSAGE	GraphSAGE
GraphSAGE layers per branch	3	3
Hidden dimension H	64	80
Dropout probability	0.15	0.15
Output activation	sigmoid	sigmoid
Coverage-loss weight λ_{cov}	1.20	1.20
Active-ratio weight λ_{act}	0.10	0.10
Energy-loss weight λ_{eng}	0.08	0.08
Switching-loss weight λ_{sw}	0.03	0.03
Ranking-loss weight λ_{rank}	0.30	0.30
Cardinality-loss weight λ_{card}	0.25	0.25
Positive-class weighting	disabled	disabled
Decoder multi-starts	8	8
Full re-optimization period	8 rounds	10 rounds
Two-for-one exchange rounds	4	3
Exchange candidate limit	16	14
Energy-rotation candidate limit	16	14
Guardrail coverage-margin tolerance	0.0025	0.0

Two additional structural ablations are included in the robustness runs. The no-communication-graph variant disables message passing on the radio-reachability graph, while the no-coverage-graph variant disables message passing on the sensing-overlap graph. The MLP ranker keeps the full feature vector but removes graph convolutions completely; its output still enters the same connected-cover decoder.

5.5. Evaluation metrics

The reported metrics are mean coverage ratio, mean connectivity ratio, mean active-node ratio, mean packet-delivery ratio (PDR), energy saving relative to Always Active (AA), first-node-death (FND) round, coverage-lifetime round, and final alive-node count. Energy saving is

$$S_E = 100\frac{E_{\mathrm{AA}} - E_{\mathrm{method}}}{E_{\mathrm{AA}}}, \tag{19}$$

where E_{AA} is the energy consumed by Always Active on the same topology. Coverage lifetime is the first round for which $\rho_{\mathrm{cov}} < \eta$. The tables report means and 95% confidence intervals when applicable. Because only three 50-node and two 100-node test topologies are used, confidence intervals describe observed dispersion but are not treated as conclusive significance tests. For the ten-topology robustness runs, paired tests are computed from topology-wise differences between DGNSS and each baseline. The paired t-test checks whether the mean difference is nonzero under an approximately symmetric difference distribution; the Wilcoxon signed-rank test gives a non-parametric matched-sample check. These tests are reported as supporting evidence rather than as a

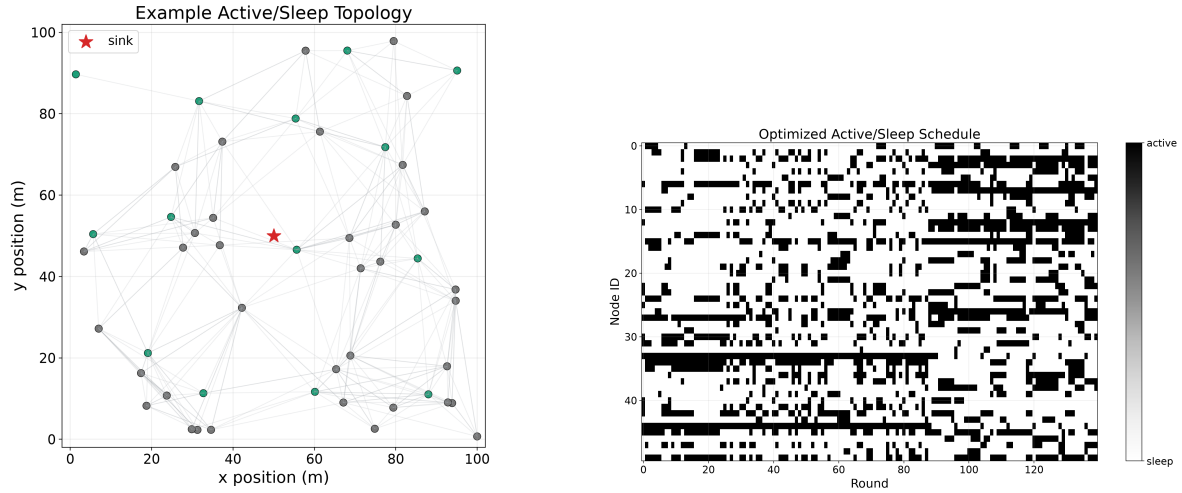


Figure 4. Representative 50-node schedule.

substitute for much larger deployment ensembles. Runtime tables report topology generation, graph construction, coverage-matrix construction, feature extraction, model inference, decoder time, and total decision time measured with Python wall-clock timers.

5.6. OMNeT++ schedule replay

The Python pipeline exports schedules with columns `round`, `node_id`, and `state`. The OMNeT++ scheduler reads this file, applies states at fixed round intervals, and records active nodes, sleeping nodes, residual energy, and packet events. This path verifies schedule serialization and execution. The present OMNeT++ model is intentionally lightweight and uses simplified direct packet delivery; the main quantitative comparison therefore comes from the common Python simulator so that every method is evaluated under identical coverage, connectivity, and energy calculations.

6. Results and discussion

The results are organized from direct behavioral evidence to aggregate comparisons and additional robustness checks. The first part examines the 50-node schedule and its round-level evolution. The second part compares 50-node and 100-node performance, followed by paired improvements, ablations, sensitivity tests, a controlled 200-node check, and runtime measurements.

6.1. Qualitative connected-cover behavior

Figure 4 shows a representative 50-node active/sleep topology and its schedule over time. The active nodes form a sparse spatial structure rather than a compact cluster around the sink. The heatmap shows that activation responsibility rotates among nodes; persistent activations remain for structurally critical regions and forwarding positions. Green nodes are active and gray nodes sleep in the topology panel, while black heatmap cells denote active states.

The proposed scheduler uses, on average, only 29.37% of the 50 nodes while maintaining mean coverage 0.9059 and full connectivity. This confirms that the decoder does more than repair arbitrary neural outputs: it constructs a sparse connected cover from the ranking.

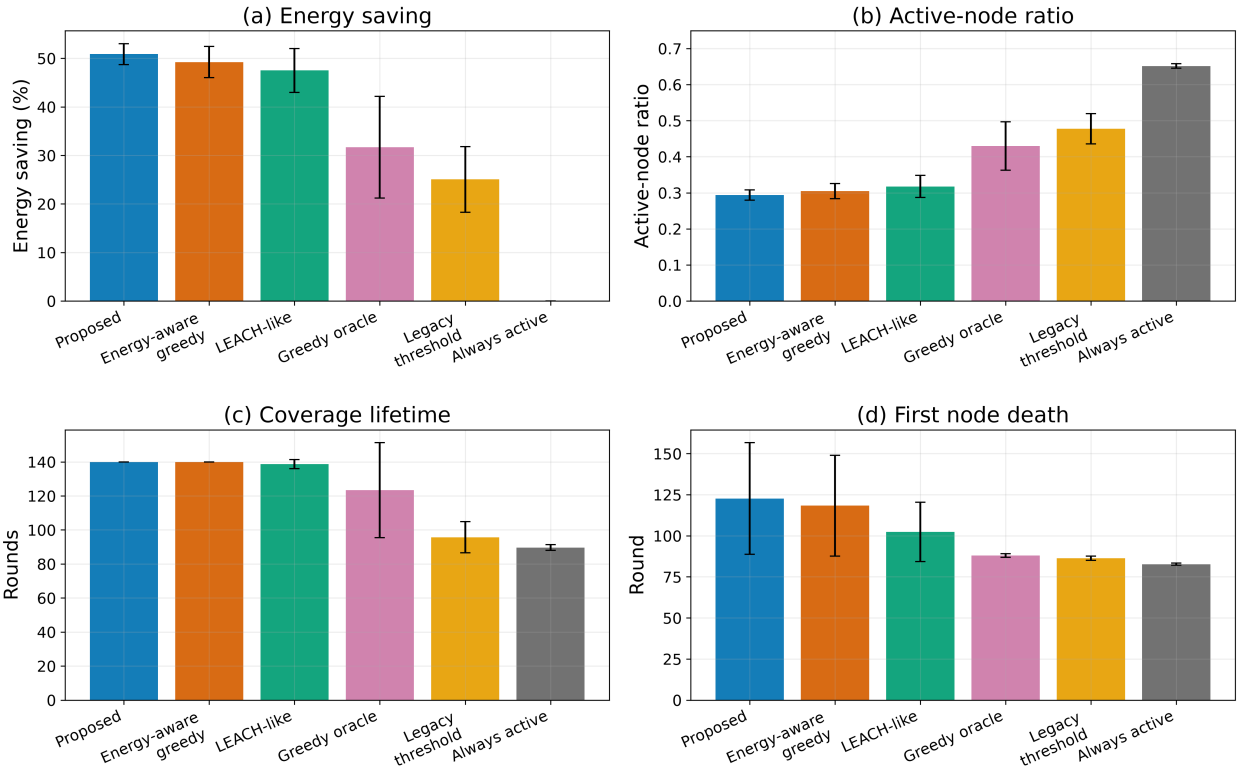


Figure 5. 50-node comparison.

6.2. *Fifty-node comparison*

Figure 5 and Table 5 summarize the 50-node results. DGNSS achieves 50.87% energy saving and preserves the coverage requirement for the full 140-round horizon. The strongest non-neural baseline, Energy-Aware Greedy, achieves 49.22% energy saving and the same coverage lifetime. DGNSS nevertheless uses 1.07 percentage points fewer active nodes and delays FND from 118.33 to 122.67 rounds. Figure error bars denote 95% confidence intervals over three test topologies; Table 5 reports mean values over the corresponding test topologies.

The LEACH-like baseline is competitive in energy saving (47.50%) but reaches FND at round 102.33 and loses coverage slightly before the horizon. The greedy oracle activates 42.96% of the nodes and saves 31.69%, indicating that the static oracle used for label generation is not itself an optimal temporal scheduler. This result is not contradictory: the neural model learns structural preferences from many oracle solutions, while constrained temporal decoding exploits current energy and duty history.

The legacy independent-threshold decoder demonstrates the importance of the proposed decoding strategy. It activates 47.73% of nodes, saves only 25.04%, and reaches a coverage lifetime of 95.67 rounds. Always Active consumes the reference energy and loses coverage at 89.67 rounds because widespread simultaneous activation accelerates energy depletion.

6.3. *Round-level evolution*

Figure 6 reports round-level averages for the 50-node scenario over three test topologies. The proposed method maintains coverage close to the required threshold while operating with a substantially smaller active fraction than the legacy and Always Active policies. The lower active burden preserves residual energy and delays the decline in alive nodes. The trajectories also clarify why mean coverage alone is insufficient: methods that initially provide excessive coverage can deplete the network faster and produce a shorter coverage lifetime.

Table 5. Main numerical results.

Nodes	Method	Coverage	Connectivity	Active ratio	PDR	Saving (%)	FND	Coverage life
50	DGNSS	0.9059	1.0000	0.2937	0.9698	50.87	122.67	140.00
50	Energy-Aware Greedy	0.9089	1.0000	0.3044	0.9690	49.22	118.33	140.00
50	LEACH-like	0.9111	1.0000	0.3175	0.9683	47.50	102.33	138.67
50	Greedy Oracle	0.8974	0.9978	0.4296	0.9589	31.69	88.00	123.33
50	Legacy Threshold	0.8759	0.9837	0.4773	0.9411	25.04	86.33	95.67
50	Always Active	0.6676	0.7619	0.6513	0.7088	0.00	82.67	89.67
100	DGNSS	0.9120	1.0000	0.4287	0.9623	24.47	88.00	123.00
100	Energy-Aware Greedy	0.9132	1.0000	0.4467	0.9609	21.45	88.00	122.00
100	LEACH-like	0.9140	1.0000	0.4524	0.9602	20.69	88.00	121.50
100	Greedy Oracle	0.8860	0.9974	0.4853	0.9514	15.61	83.50	107.00
100	Legacy Threshold	0.8426	0.9661	0.4979	0.9194	14.12	83.00	91.00
100	Always Active	0.6105	0.8312	0.5940	0.7663	0.00	82.50	90.50

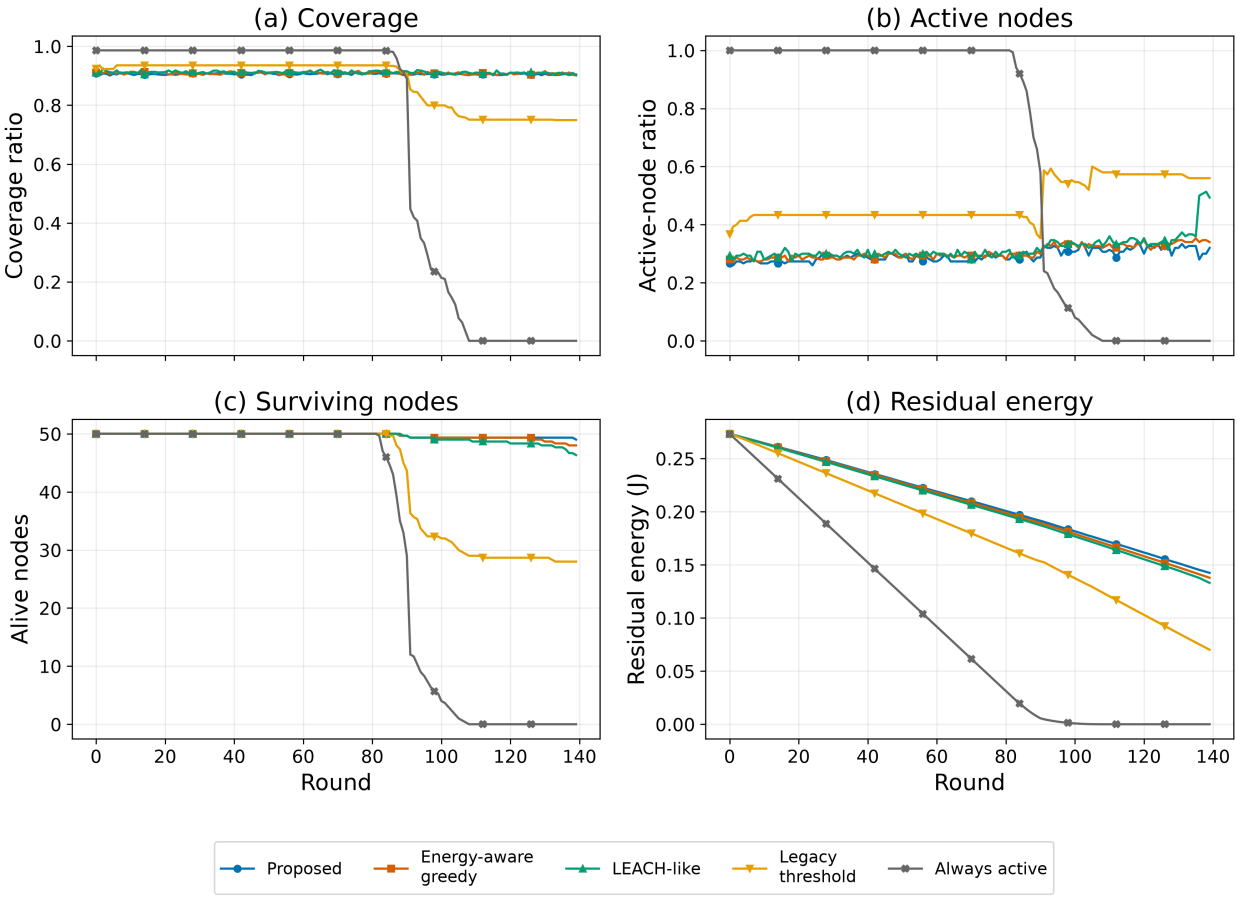


Figure 6. Round-level trajectories.

6.4. Scalability from 50 to 100 nodes

The 100-node scenario is more demanding because the deployment area grows and the simulation horizon is 160 rounds. DGNSS achieves 24.47% energy saving, compared with 21.45% for Energy-Aware Greedy and 20.69%

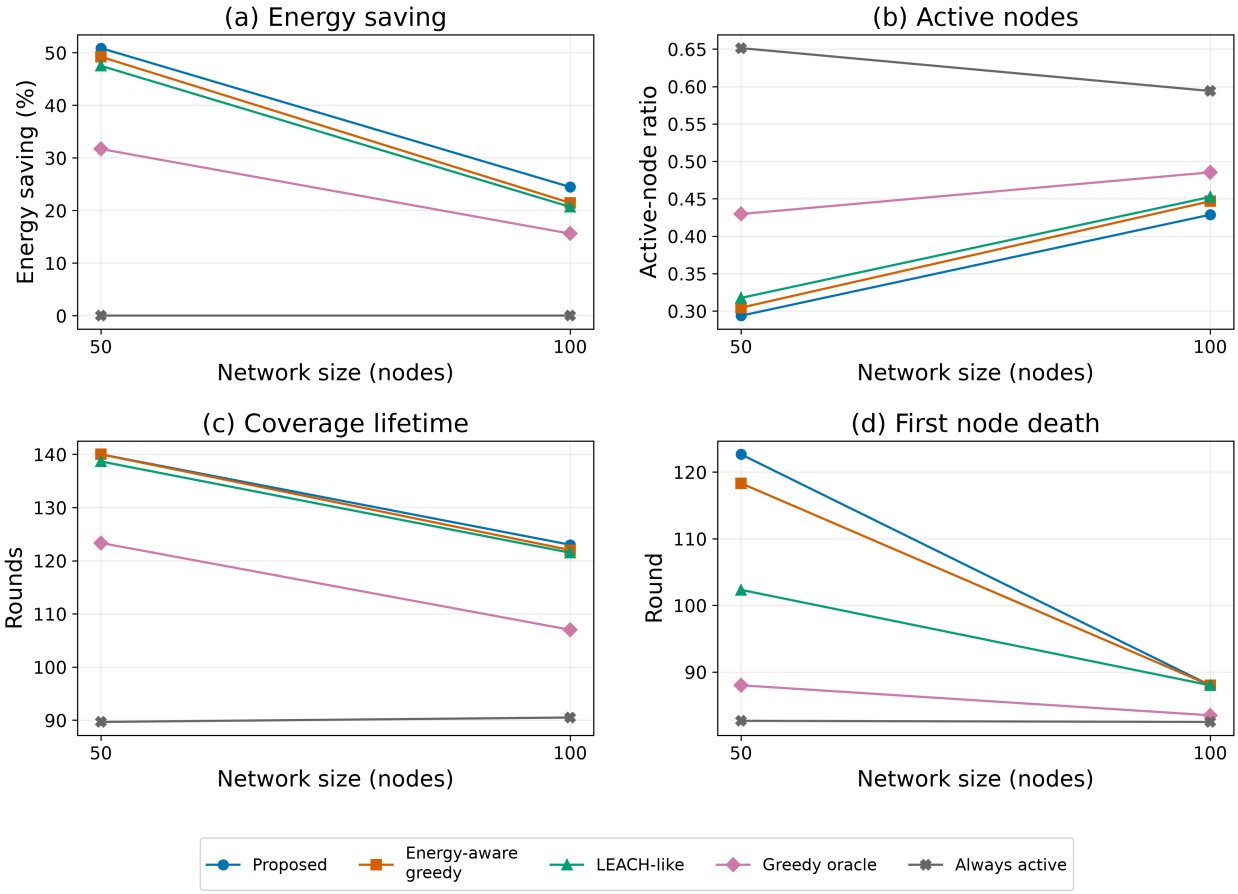


Figure 7. 50- and 100-node comparison.

for LEACH-like. The proposed method reduces the active ratio by 1.80 percentage points relative to Energy-Aware Greedy and extends coverage lifetime by one round. Its mean connectivity remains one.

Figure 7 shows the scale transition. Energy-saving percentages decline for all sparse schedulers because the larger deployment requires a greater relay and sensing burden. Nevertheless, DGNSS retains the best energy saving among the common methods at both scales. The absolute FND advantage over Energy-Aware Greedy is scale-dependent: it is 4.33 rounds at 50 nodes and zero at 100 nodes. Thus, the evidence supports an improved energy–coverage trade-off, not uniform superiority in every metric.

6.5. Paired improvement analysis

Table 6 reports paired differences computed on identical topology seeds. DGNSS improves energy saving over Energy-Aware Greedy on all five test topologies across both scales. At 50 nodes, it also improves FND by 4.33 rounds without reducing coverage lifetime. At 100 nodes, it gains 3.02 percentage points in energy saving and one coverage-lifetime round, while FND is unchanged.

6.6. Ten-topology robustness and statistical tests

Table 7 reports the extended ten-topology evaluation against Energy-Aware Greedy. The 50-node result remains positive: DGNSS saves 49.04% energy on average, Energy-Aware Greedy saves 45.54%, and the paired energy-saving gain is 3.51 percentage points. The Wilcoxon signed-rank test gives $p = 0.0488$ for the energy-saving gain

Table 6. Paired gains over Energy-Aware Greedy.

Metric	50 nodes	100 nodes
Energy-saving gain (percentage points)	+1.65	+3.02
Energy-saving win rate	3/3	2/2
Active-ratio reduction	0.0107	0.0180
Coverage-lifetime gain (rounds)	0.00	+1.00
FND gain (rounds)	+4.33	0.00

Table 7. Ten-topology robustness.

Nodes	Topologies	DGNSS saving (%)	Greedy saving (%)	Gain (pp)	p_E	Active reduction	Coverage-life gain	FND gain
50	10	49.04 ± 3.99	45.54 ± 4.80	3.51	0.0488	0.0239	0.70	1.20
100	10	29.68 ± 9.11	28.41 ± 8.55	1.27	0.1602	0.0071	-2.10	-3.10

Table 8. Ranker ablation.

Nodes	Ranker	Coverage	Active ratio	Saving (%)	FND	Coverage life
50	Dual-graph GraphSAGE	0.9055	0.3076	49.04	108.70	135.90
50	Feature-only MLP	0.9053	0.3069	49.17	106.70	136.10
100	Dual-graph GraphSAGE	0.9162	0.3952	29.68	96.20	132.40
100	Feature-only MLP	0.9165	0.3907	30.35	98.90	134.80

and $p = 0.0488$ for the active-ratio reduction. Coverage lifetime is not significantly different because both methods frequently remain feasible until the end of the 140-round horizon.

At 100 nodes, the extended result is more cautious. DGNSS saves 29.68% energy, compared with 28.41% for Energy-Aware Greedy, but the paired energy-saving gain of 1.27 percentage points is not statistically significant ($p = 0.1602$ by Wilcoxon test). The coverage-lifetime and FND differences also do not support a uniform lifetime advantage. Therefore, the 100-node evidence should be interpreted as an energy-saving trend with preserved feasibility, not as a statistically established lifetime improvement. Table 7 reports method means with 95% confidence intervals; the p-values are paired Wilcoxon signed-rank tests on topology-wise differences.

6.7. Ranker and structural ablations

Table 8 compares the dual-graph GraphSAGE ranker with a feature-only MLP ranker under the same decoder and topology seeds. The MLP is competitive at both scales. At 50 nodes, the two rankers are nearly indistinguishable in energy saving and coverage lifetime. At 100 nodes, the MLP obtains slightly higher mean energy saving and coverage lifetime. This result narrows the architectural claim: the dual-graph encoder is a principled relational representation, but the present evidence does not prove that GraphSAGE alone is superior to a strong feature-only ranker. The coverage-aware features and deterministic decoder are major contributors to the final performance.

The single-graph ablations show the same pattern. At 50 nodes, removing the communication graph slightly increases energy saving but reduces coverage lifetime, while removing the coverage graph produces almost the same mean energy saving as the full model. At 100 nodes, the no-coverage-graph and no-communication-graph variants remain close to the full model. These results indicate that the dual-graph design is useful as a physically interpretable representation, but the current experimental scale is insufficient to isolate a large independent contribution from either message-passing branch.

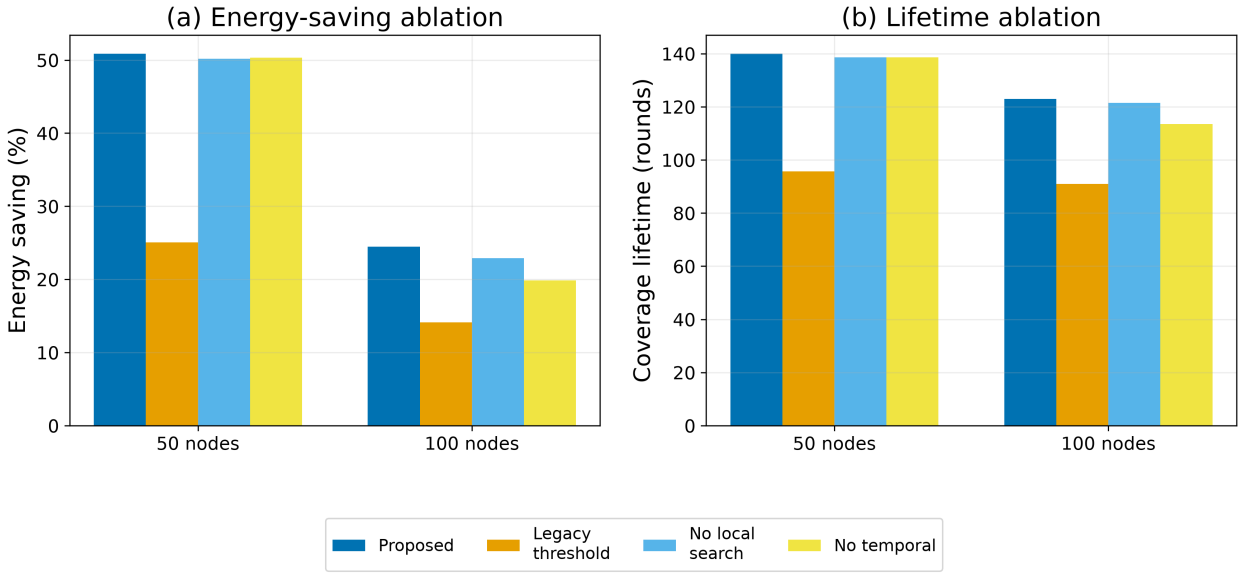


Figure 8. Ablation results.

Table 9. Loss-weight sensitivity.

Variant	λ_{cov}	λ_{rank}	Coverage	Active ratio	Saving (%)	Coverage life
Coverage low	0.60	0.30	0.9065	0.2897	51.58	140.00
Coverage high	1.80	0.30	0.9063	0.2888	51.74	140.00
Rank low	1.20	0.15	0.9064	0.2899	51.56	140.00
Rank high	1.20	0.45	0.9059	0.2882	51.82	140.00

6.8. Ablation study

Figure 8 compares the complete method with the legacy threshold decoder, removal of local search, and removal of temporal mechanisms. The threshold decoder is consistently weakest, confirming that independent node decisions followed by repair are unsuitable for this constrained subset problem. At 100 nodes, removing local search lowers energy saving from 24.47% to 22.90%, and removing temporal features and rotation lowers it to 19.88% while reducing coverage lifetime from 123 to 113.5 rounds.

The 50-node no-witness variant saves 51.03%, slightly more than the complete method, but its FND is 115.67 rather than 122.67 rounds. Witness protection therefore trades a very small amount of immediate energy saving for a more robust distribution of critical sensing responsibility. This is an example of why energy saving should not be optimized in isolation.

6.9. Loss-weight sensitivity

Table 9 reports a limited sensitivity check for the two loss weights most directly tied to feasibility and ranking. The variants use five 50-node test topologies and a shorter eight-epoch training budget, so the table is a robustness check rather than a full hyperparameter search. Across the tested values, energy saving remains between 51.56% and 51.82%, mean coverage remains above 0.9059, connectivity remains one, and coverage lifetime remains the full 140 rounds. The exact Wilcoxon p-value for the energy-saving gain over Energy-Aware Greedy is 0.0625 in all four variants, which is the smallest attainable two-sided value for five nonzero matched differences. These results do not identify an optimal loss setting, but they show no brittle collapse under the tested perturbations.

Table 10. Controlled 200-node check.

Method	Coverage	Connectivity	Active ratio	Saving (%)	FND	Coverage life
DGNSS	0.9237	1.0000	0.1382	71.29	138.67	160.00
No local search	0.9238	1.0000	0.1384	71.26	138.67	160.00
No temporal	0.9254	1.0000	0.1388	71.17	124.00	160.00
Energy-Aware Greedy	0.9255	1.0000	0.1408	70.86	150.33	160.00
Always Active	0.6212	0.8083	0.5852	0.00	82.33	92.00

Table 11. Per-round runtime.

Scenario	Feature extraction	Model inference	Decoder	Total decision
50-node GraphSAGE	0.0054	0.0019	0.0203	0.0276
100-node GraphSAGE	0.0178	0.0026	0.0836	0.1040
100-node MLP	0.0184	0.0004	0.0827	0.1016
200-node GraphSAGE	0.0699	0.0045	0.0737	0.1481

6.10. Controlled 200-node scale check

Table 10 gives a controlled 200-node check using the 100-node area, sensing radius, communication radius, energy, threshold, and horizon. This denser setting is easier for coverage and relay redundancy than the field-expansion setting from 50 to 100 nodes, so it should not be read as a general 200-node benchmark. It does show that the centralized decoder remains computationally feasible and can preserve coverage for all 160 rounds. DGNSS saves 71.29% energy with mean active ratio 0.1382, while Energy-Aware Greedy saves 70.86% with mean active ratio 0.1408. Energy saving is slightly higher for DGNSS, but Energy-Aware Greedy has a later FND in this dense case. Values are means over three test topologies.

6.11. Runtime profile

Table 11 reports wall-clock timing from the Python implementation. The total DGNSS decision time is 0.0276 s per round at 50 nodes, 0.1040 s at 100 nodes, and 0.1481 s at 200 nodes. The decoder and feature construction dominate runtime; model inference is much smaller, ranging from 0.0019 s to 0.0045 s for the GraphSAGE ranker. The MLP inference time is lower, but total time changes little because the same exact decoder is used. These timings support offline or centralized planning at the tested scales, while large-scale distributed deployment would require incremental graph maintenance, spatial indexing, or decentralized approximations.

Five observations emerge from the full set of experiments. First, the ranker and decoder serve different roles. The ranker provides topology- and state-dependent priorities, while exact decoding ensures that those priorities become an executable connected cover. The MLP ablation shows that the current evidence supports the hybrid ranking–decoding design more strongly than it supports a claim of GraphSAGE superiority. Second, minimum cardinality is not sufficient for long-term performance. Temporal rotation and witness protection determine whether repeatedly selected critical nodes die early. Third, a guardrail is valuable in learned combinatorial optimization. Including a strong constructive candidate makes the learned method a safe-improvement system: the model must improve or match an already competitive feasible solution. Fourth, the 50-node energy-saving gain over Energy-Aware Greedy is supported by the ten-topology paired tests, while the 100-node extension shows a smaller non-significant trend. Fifth, centralized exact decoding remains practical at the tested scales but is the dominant runtime component.

The results also explain the weak performance of Always Active. Activating every available node maximizes instantaneous redundancy but accelerates energy consumption. Once many nodes die, both coverage and connectivity decline. A sparse but carefully constructed active set preserves the ability to monitor and relay for longer.

7. Limitations and threats to validity

The evaluation has several limitations. First, the main comparison table uses three independent 50-node test topologies and two independent 100-node test topologies. The added robustness runs increase this to ten topologies at each scale for selected comparisons, but ten deployments are still not enough to characterize all random WSN geometries. The statistical tests therefore support the specific configured regimes rather than a universal performance claim.

Second, the packet-delivery model is an analytical proxy based on sink reachability and residual energy. It does not model interference, contention, retransmissions, fading, queueing, or protocol-specific medium access control (MAC) behavior. The OMNeT++ component currently validates offline schedule replay with a simplified packet path. A full radio, MAC, routing, and channel model is required before making claims about deployment-level throughput and latency.

Third, coverage is evaluated on a finite regular grid under a binary disk-sensing model. Probabilistic sensing, obstacles, directional sensors, target coverage, and heterogeneous sensing radii are outside the present scope. Fourth, all nodes are static and the sink is fixed. Mobility and topology uncertainty could require online graph updates and uncertainty-aware decoding.

Fifth, the 50-node and 100-node energy-stress configurations use different areas, radii, initial energies, training-set sizes, and horizons. They are intended as two reproducible operating regimes rather than a controlled single-factor scaling experiment. The controlled 200-node check isolates node-count growth only in a denser setting; it does not replace large-scale tests over sparse 500-node or 1000-node deployments. The MLP ablation also shows that feature design and deterministic decoding explain much of the measured performance, so larger experiments are needed before assigning the gains specifically to graph message passing. Finally, the decoder is centralized and performs repeated exact checks. Distributed approximation and latency profiling are important future work.

8. Conclusion

DGNSS was developed as a dual-graph neural sleep scheduler that preserves sensing coverage and sink connectivity through deterministic constrained decoding. Communication and coverage-overlap graphs are encoded separately, fused through learned gates, and combined with structural, energy, witness, and temporal features. The neural network supplies node keep scores, while multi-start pruning, constructive candidates, local exchanges, energy rotation, and a safe-improvement guardrail produce an executable active set.

The method corrected the central failure of independent thresholding, which tended to keep most nodes active. In the reproduced 50-node scenario, DGNSS achieved 50.87% energy saving, maintained coverage for all 140 rounds, and delayed first-node death by 4.33 rounds relative to Energy-Aware Greedy. In the 100-node scenario, it achieved 24.47% energy saving and extended coverage lifetime by one round relative to the same baseline. Ablations confirmed the importance of constrained decoding, local exchanges, and temporal energy mechanisms. Additional ten-topology runs support the 50-node energy-saving gain statistically, but show that the 100-node advantage over Energy-Aware Greedy is smaller and not statistically significant. The feature-only MLP ranker is competitive with the dual-graph GraphSAGE ranker, so the strongest conclusion is that coverage-aware ranking plus deterministic connected-cover decoding is effective; the independent value of graph message passing needs larger and more diverse experiments.

Future work will investigate larger ensembles of random deployments, probabilistic sensing and channel models, distributed decoding, heterogeneous nodes, mobile sinks, online adaptation, and detailed OMNeT++ or hardware-in-the-loop validation. An additional direction is to train the ranking model directly against decoder-level lifetime outcomes while retaining the deterministic feasibility guardrail.

Declarations

Funding This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of interest The authors declare that they have no conflict of interest.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Data availability The generated topology files, round-level metrics, node-state time series, figure data, decision schedules, and aggregate tables are included with the experimental artifacts associated with this study.

Code availability The implementation includes topology generation, dual-graph construction, model training, constrained decoding, baseline evaluation, artifact export, and OMNeT++ schedule replay. The source package is prepared for repository release.

Author contributions Ahmida Djediai contributed to conceptualization, methodology, software, experimentation, analysis, visualization, and article preparation. Amine Khaldi contributed to supervision, validation, interpretation, and article review.

REFERENCES

1. Tossa, F., Faga, Y., Abdou, W., Ezin, E.C., Gouton, P.: Wireless sensor network deployment: Architecture, objectives, and methodologies. Sensors **25**(11), 3442 (2025)
2. El Khediri, S.: Wireless sensor networks: A survey, categorization, main issues, and future orientations for clustering protocols. Computing **104**(8), 1775–1837 (2022) https://doi.org/10.1007/s00607-022-01071-8
3. Rana, A., Kaur, K., Kaur, P., Bhatti, E.: Energy-efficient protocols for environmental monitoring in wireless sensor networks: A review. Journal of Ambient Intelligence and Smart Environments **17**(2), 139–163 (2025) https://doi.org/10.1177/18761364241297221
4. Chakravarti, B., Chatterjee, S.: A HEED-inspired hybrid-score clustering protocol for energy-efficient border surveillance in wireless sensor networks. Adamas Technical Review 54 (2025)
5.
6. Sharma, D., Tomar, G.S.: Enhance PEGASIS algorithm for increasing the life time of wireless sensor network. Materials Today: Proceedings **29**, 372–380 (2020) https://doi.org/10.1016/j.matpr.2020.07.291
7.
8. Khan, M.N., Rahman, H.U., Khan, M.Z., Mehmood, G., Sulaiman, A., Shaikh, A., Alqhatani, A.: Energy-efficient dynamic and adaptive state-based scheduling (EDASS) scheme for wireless sensor networks. IEEE Sensors Journal **22**(12), 12386–12403 (2022) https://doi.org/10.1109/JSEN.2022.3174050
9. Aslanyan, L., Aslanyan, H., Khosravi, H.: Optimal node scheduling for integrated connected-coverage in wireless sensor networks. In: Ninth International Conference on Computer Science and Information Technologies Revised Selected Papers, pp. 1–13. IEEE, Piscataway, NJ (2013). https://doi.org/10.1109/csitechnol.2013.6710363 . http://dx.doi.org/10.1109/CSITechnol.2013.6710363
10. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016) https://doi.org/10.48550/ARXIV.1609.02907
11. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. arXiv preprint arXiv:1706.02216 (2017) https://doi.org/10.48550/ARXIV.1706.02216
12. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017) https://doi.org/10.48550/ARXIV.1710.10903
13. Eisen, M., Ribeiro, A.: Optimal wireless resource allocation with random edge graph neural networks. IEEE Transactions on Signal Processing **68**, 2977–2991 (2020) https://doi.org/10.1109/tsp.2020.2988255
14. Shen, Y., Shi, Y., Zhang, J., Letaief, K.B.: Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis. IEEE Journal on Selected Areas in Communications **39**(1), 101–115 (2021) https://doi.org/10.1109/jsac.2020.3036965
15. Neji, W., Othman, S.B., Sakli, H.: T-leach: Threshold sensitive low energy adaptive clustering hierarchy for wireless sensor networks. In: 2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), pp. 336–342. IEEE, Piscataway, NJ (2020). https://doi.org/10.1109/sta50679.2020.9329354 . http://dx.doi.org/10.1109/STA50679.2020.9329354
16. S.S., S., Ansar A., S., Pillai, M.J.: Leach based clustering protocols in wireless sensor networks. In: 2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT), pp. 1–6. IEEE, Piscataway, NJ (2019). https://doi.org/10.1109/iciict1.2019.8741502 . http://dx.doi.org/10.1109/ICIICT1.2019.8741502
17. Yufu, J., Hongjun, L.: A node scheduling scheme based on coverage-preserving for wireless sensor network. In: 2007 Second International Conference on Communications and Networking in China, pp. 856–860. IEEE, Piscataway, NJ (2007). https://doi.org/10.1109/chinacom.2007.4469519 . http://dx.doi.org/10.1109/CHINACOM.2007.4469519

18. Qu, W., Wang, J., Liu, Z.: An energy-efficiency coverage-preserving node scheduling scheme in wireless sensor networks. In: 2009 International Symposium on Computer Network and Multimedia Technology, pp. 1–4. IEEE, Piscataway, NJ (2009). https://doi.org/10.1109/cnmt.2009.5374731 . http://dx.doi.org/10.1109/CNMT.2009.5374731
19. Bulut, E., Korpeoglu, I.: Sleep scheduling with expected common coverage in wireless sensor networks. Wireless Networks **17**(1), 19–40 (2010) https://doi.org/10.1007/s11276-010-0262-2
20. Dineshkumar, P., Geetha, K., Rajan, C.: An energy-aware sleep scheduling coverage protocol for wireless sensor network. Journal of Circuits, Systems and Computers **34**(14) (2025) https://doi.org/10.1142/s0218126625503037
21. Wu, J., Tian, S., Qi, X., Cai, Z., Wang, P.: Optimizing sleep scheduling in wireless sensor networks via node utility and critical target prioritization. Scientific Reports **16**(1) (2026) https://doi.org/10.1038/s41598-026-40548-w
22. Kumar, V., Kumar, S.: Coverage preserving scheduling for life span maximization in wireless sensor network based internet of things. In: Communication and Computing Systems, pp. 206–213. CRC Press, Boca Raton, FL (2019). https://doi.org/10.1201/9780429444272-33 . http://dx.doi.org/10.1201/9780429444272-33
23. Biradar, S., Mallikarjuna Shastry, P.M.: Redundancy elimination with coverage preserving algorithm in wireless sensor network. International Journal of Communication Networks and Information Security (IJCNIS) **10**(3) (2022) https://doi.org/10.17762/ijcnis.v10i3.3576
24. Tian, D., Georganas, N.D.: Connectivity maintenance and coverage preservation in wireless sensor networks. Ad Hoc Networks **3**(6), 744–761 (2005) https://doi.org/10.1016/j.adhoc.2004.03.001
25. Cardei, M., Thai, M.T., Li, Y., Wu, W.: Energy-efficient target coverage in wireless sensor networks. In: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies., vol. 3, pp. 1976–1984. IEEE, Piscataway, NJ (2005). https://doi.org/10.1109/infcom.2005.1498475 . http://dx.doi.org/10.1109/INFCOM.2005.1498475
26. He, S., Chen, J., Sun, Y.: Coverage and connectivity in duty-cycled wireless sensor networks for event monitoring. IEEE Transactions on Parallel and Distributed Systems **23**(3), 475–482 (2012) https://doi.org/10.1109/tpds.2011.191
27. Chandra, A., Tarasia, N., Swain, A.R., Kumar, M.: A distributed prime node-id connected dominating set for target coverage using adjustable sensing range. In: Intelligent Computing, Communication and Devices, pp. 99–104. Springer, New Delhi (2014). https://doi.org/10.1007/978-81-322-2009-1_12 . http://dx.doi.org/10.1007/978-81-322-2009-1_12
28. Ingelrest, F., Simplot-Ryl, D., Stojmenovic, I.: Smaller connected dominating sets in ad hoc and sensor networks based on coverage by two-hop neighbors. In: 2007 2nd International Conference on Communication Systems Software and Middleware, pp. 1–8. IEEE, Piscataway, NJ (2007). https://doi.org/10.1109/comswa.2007.382576 . http://dx.doi.org/10.1109/COMSWA.2007.382576
29. Chenait, M., Zebbane, B., Badache, N.: A new k-coverage model to determine redundant sensors in wireless sensor networks. In: 2018 International Conference on Smart Communications in Network Technologies (SaCoNeT), pp. 149–154. IEEE, Piscataway, NJ (2018). https://doi.org/10.1109/saconet.2018.8585545 . http://dx.doi.org/10.1109/SACONET.2018.8585545
30. Chenait, M., Benzaid, C., Kheddache, T., Sadouni, A., Zebbane, B.: Energy-aware k-coverage protocol for heterogeneous sensor iot environment. In: 2025 International Conference on Communication, Computing, Networking, and Control in Cyber-Physical Systems (CCNCPS), pp. 99–104. IEEE, Piscataway, NJ (2025). https://doi.org/10.1109/ccncps66785.2025.11135554 . http://dx.doi.org/10.1109/CCNCPS66785.2025.11135554
31. Sinde, R., Begum, F., Njau, K., Kaijage, S.: Refining network lifetime of wireless sensor network using energy-efficient clustering and drl-based sleep scheduling. Sensors **20**(5), 1540 (2020) https://doi.org/10.3390/s20051540
32. Verma, P., Dumka, A., Vyas, D., Bhardwaj, A.: Reinforcement learning based node sleep or wake-up time scheduling algorithm for wireless sensor network. International Journal of Mathematical, Engineering and Management Sciences **5**(4), 707–731 (2020) https://doi.org/10.33889/ijmems.2020.5.4.057
33. R, H.: Intelligent scheduling in wireless sensor networks using reinforcement learning based deep q-networks. Journal of Information Systems Engineering and Management **10**(24s), 530–546 (2025) https://doi.org/10.52783/jisem.v10i24s.3935
34. Xing, Y., Zeng, Z., Liu, C., Verma, A., Lee, T., Hou, D., Pan, H., Edwards, S.: Optimizing sleep schedules for energy-efficient agricultural wireless sensor networks using deep reinforcement learning. In: 2024 IEEE 15th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), pp. 66–72. IEEE, Piscataway, NJ (2024). https://doi.org/10.1109/uemcon62879.2024.10754747
35. Jurado-Lasso, F.F., Orfanidis, C., Jurado, J.F., Fafoutis, X.: Hrl-tsch: A hierarchical reinforcement learning-based tsch scheduler for iiot. IEEE Transactions on Cognitive Communications and Networking **10**(6), 2102–2118 (2024) https://doi.org/10.1109/tccn.2024.3408459
36. Fey, M., Lenssen, J.E.: Fast graph representation learning with pytorch geometric. arXiv preprint arXiv:1903.02428 (2019) https://doi.org/10.48550/ARXIV.1903.02428
37. Vo, V.-V., Raza, S.M., Le, D.-T., Kim, M., Choo, H.: Graph neural networks for iot data aggregation scheduling. In: NOMS 2024-2024 IEEE Network Operations and Management Symposium, pp. 1–7. IEEE, Piscataway, NJ (2024). https://doi.org/10.1109/noms59830.2024.10575067 . http://dx.doi.org/10.1109/NOMS59830.2024.10575067
38. Xu, X., Lu, Y., Fu, Q.: Applying graph neural network in deep reinforcement learning to optimize wireless network routing. In: 2021 Ninth International Conference on Advanced Cloud and Big Data (CBD), pp. 218–223. IEEE, Piscataway, NJ (2022). https://doi.org/10.1109/cbd54617.2021.00045 . http://dx.doi.org/10.1109/CBD54617.2021.00045
39. Lu, P., Jing, C., Zhu, X.: Graphsage-based multi-path reliable routing algorithm for wireless mesh networks. Processes **11**(4), 1255 (2023) https://doi.org/10.3390/pr11041255
40. Dai, H., Khalil, E.B., Zhang, Y., Dilkina, B., Song, L.: Learning combinatorial optimization algorithms over graphs. arXiv preprint arXiv:1704.01665 (2017) https://doi.org/10.48550/ARXIV.1704.01665
41. Darvariu, V.-A., Hailes, S., Musolesi, M.: Graph reinforcement learning for combinatorial optimization: A survey and unifying perspective. arXiv preprint arXiv:2404.06492 (2024) https://doi.org/10.48550/ARXIV.2404.06492
42. Bouarourou, S., Zannou, A., Nfaoui, E.H., Kanzouai, C., Boulaalam, A.: An energy-efficient pathfinding model for wireless sensor networks in IoT using whale optimization algorithm. Statistics, Optimization & Information Computing **14**(5), 2379–2395 (2025) https://doi.org/10.19139/soic-2310-5070-2433

43. Ouaarouss, A., Chabbout, H., Zannou, A., Isaad, J.: Optimized data offloading in IoT-based wireless sensor networks. Statistics, Optimization & Information Computing **15**(1), 921–935 (2025) https://doi.org/10.19139/soic-2310-5070-3041
44. Abdulghani, B.W., Shukur, O.B.S.: Missing values imputation for spatio-temporal climate variable using graph convolutional networks (GCN). Statistics, Optimization & Information Computing (2026) https://doi.org/10.19139/soic-2310-5070-3928