# Design and Evaluation of a Hybrid Wireless Sensor Network Combining Cluster-Tree and Mesh Routing

Nicholas Palmer
*Dept. of Electrical and Computer Engineering*
*San Diego State University*
San Diego, CA, USA
npalmer2267@sdsu.edu

***Abstract*—Wireless sensor networks (WSNs) are self-organizing networks of sensor nodes deployed to monitor the environment. These networks are typically battery-powered with limited processing and storage capabilities, requiring energy-efficient communication protocols and resilience to node and link failures. To address these challenges, we propose a hybrid WSN that combines cluster-tree and mesh routing to enhance network performance. We also implement router-bridge nodes to improve energy efficiency between cluster heads. We evaluate our hybrid WSN using a Python simulation tool and compare it with existing WSNs.**

***Index Terms*—Wireless Sensor Network, AODV, Cluster tree**

## I. Introduction

Wireless sensor networks (WSNs) are used in a variety of applications, such as environmental monitoring, industrial automation, healthcare, and increasingly, in consumer devices. The necessity for ease of deployment and lower cost hardware have made WSNs popular in recent years, but they also pose significant challenges in terms of self-organization, energy efficiency, and robustness.

### *A. Self-Organization*

Self-organization is a crucial advantage of WSNs, allowing deployments in remote or inaccessible locations, making it impractical to pre-configure infrastructure. Self-organization allows nodes to form a network topology and select routing paths dynamically based on the network conditions and the environment. Building an optimal topology presents many challenges given the dynamic nature of the environment, limited computational resources, and energy resources of most low-cost hardware. Most networks use one of two topologies: a hierarchical cluster-tree or a mesh topology.

- **Cluster-Tree**
  - Hierarchical network with a single root node and multiple cluster-heads
  - Cluster-heads are responsible for routing data to and from their cluster, only they must maintain routing information for their child members and child networks below them
  - Centralized routing allows for greater energy efficiency as the routing can be optimized for the entire network
- **Mesh**
  - Peer-to-peer network with no central nodes or defined hierarchy
  - Each node is responsible for routing data to and from its neighbors, maintaining routing information for all its n-hop neighbors
  - Decentralized routing allows for greater robustness to node and link failures as the routing can adapt on-demand to changes in the network topology

Despite these advantages, both topologies have their limitations. Cluster-tree topologies can be fragile, as the failure of a cluster-head can isolate an entire cluster from the network causing a cascading failure. While mesh topologies can incur significant overhead due to the need for each node to maintain routing information for all its neighbors, leading to increased overhead and energy use. These shortcomings motivate the need for a hybrid approach that combines the strengths of both topologies.

### *B. Hybrid WSNs*

To address these limitations, hybrid WSNs that incorporate both cluster-tree and mesh routing have been proposed. In these approaches, nodes are organized into a cluster-tree topology, but they can also use mesh links to route data between clusters. This allows for greater robustness to node failures, as nodes can use mesh links to route bypass failed cluster-heads. Additionally, the cluster-tree structure allows for more efficient routing within clusters, reducing the need for each node to maintain routing information for all its neighbors.

### *C. Contributions*

In this paper, we present the design and evaluation of a hybrid WSN that combines cluster-tree and mesh routing. We first describe the design of the hybrid topology formation and routing protocol, including the use of router-bridge nodes to improve energy efficiency between cluster heads. We then implement the protocol via a Python simulation and evaluate its performance in terms of average join time, connectivity, packet delivery, and energy use, under node and link losses. Finally, we compare the performance of our hybrid WSN with existing WSNs.

## II. Related Work

We briefly review prior works on cluster-tree and mesh routing protocols in WSNs, as well as existing hybrid approaches.

### *A. Cluster-Tree Protocols*

### *B. Mesh Routing Protocols*

### *C. Hybrid Approaches*

### *D. Summary*

## III. System Model and Assumptions

We describe the system model and assumptions used in our design and evaluation.

### *A. Network Model*

We consider a network consisting of $N$ sensor nodes deployed uniformly at random in a defined deployment area. The network designates one root node that initiates network formation and acts as the gateway, and the remaining nodes self-organize into a hybrid cluster-tree mesh topology. Nodes communicate via a single radio channel with a dynamically selectable transmission range. We assume static node placement throughout the simulation and an ideal MAC layer with IEEE 802.15.4 characteristics operating at 2.4 GHz (i.e., collisions are not modeled). To capture link-layer unreliability, we apply a uniform packet loss probability $p_{\text{loss}} \in [0, 0.1]$ to all transmissions, varied across experiments to evaluate protocol performance under various conditions.

### *B. Traffic Model*

To evaluate protocol performance under, we consider a many-to-many traffic pattern where each node generates periodic data packets destined for a randomly selected node in the network. This pattern stress-tests the routing protocol more rigorously than the traditional many-to-one (sensor-to-sink) model typical of data-collection WSNs, as it exercises a wider variety of routing paths. Once a node has joined the network and is in the `REGISTERED`, `CLUSTER_HEAD`, or `ROUTER` state, it begins generating traffic at regular intervals. The packet generation interval is dynamically configurable for each simulation run, resulting in a variable data rate per node.

### *C. Energy Model*

Each node is initialized with a finite energy (battery capacity) and consumes energy due to radio activity in the transmit (TX) and receive (RX) states. We model a TI/Chipcon CC2420 IEEE 802.15.4 radio operating at 2.4 GHz with data rate $R = 250\,\text{kbps}$ and supply voltage $V = 3.0\,\text{V}$.

*1) Initial Energy:* Battery capacity is configured in mAh and converted to energy as

$$E_{\text{init,mJ}} = C_{\text{mAh}} \cdot V \cdot 3600.$$

In our experiments, we test $C_{\text{mAh}}$ across the range 0.25–1.0 mAh at $V = 3.0\,\text{V}$, corresponding to initial energies from $E_{\text{init}} = 2700\,\text{mJ}$ (2.7 J) to $E_{\text{init}} = 10{,}800\,\text{mJ}$ (10.8 J). These values are lower than typical sensor node batteries to accelerate simulation convergence while preserving relative energy consumption behavior. Unless otherwise stated, results use $C_{\text{mAh}} = 0.25\,\text{mAh}$.

*2) Packet Size:* Let $B(p)$ denote the over-the-air size in bytes of packet $p$. This includes PHY overhead (6 B: preamble, SFD, PHR), MAC overhead (8 B), a fixed 13 B network header, and a byte count corresponding to the payload type:

$$B(p) = 6\ (\text{PHY}) + 8\ (\text{MAC}) + 13\ (\text{header}) + \text{payload\_bytes}(p).$$

*3) TX and RX Energy:* For a given TX power level $\ell$, the CC2420 draws a current $I_{\text{TX}}(\ell)$, which we precompute into per-byte TX energies

$$e_{\text{TX}}(\ell) \in \{0.82,\ 0.95,\ 1.34,\ 1.67\}\ \mu\text{J/byte},$$

corresponding to output power settings with $I_{\text{TX}} \in \{8.5,\ 9.9,\ 14.0,\ 17.4\}\,\text{mA}$. We include a fixed per-packet overhead $E_{\text{start}} = 10\,\mu\text{J}$ to model fixed radio energy consumption.

The RX current $I_{\text{RX}} \approx 18.8\,\text{mA}$ yields a per-byte receive cost of $e_{\text{RX}} \approx 1.8\,\mu\text{J/byte}$. Thus, the energy to transmit and receive a packet $p$ is

$$E_{\text{TX}}(p, \ell) = E_{\text{start}} + B(p)\, e_{\text{TX}}(\ell), \qquad E_{\text{RX}}(p) = B(p)\, e_{\text{RX}}.$$

The simulator tracks energy per node in milliampere-hours. Nodes whose energy reaches zero are disabled and cease all TX and RX activity.

### *D. Scope and Simplifications*

Since our focus is on evaluating protocol-level behavior rather than hardware-accurate energy profiling, we make the following simplifications:

- We omit MCU processing, idle listening, and sleep-state power consumption, only TX and RX radio activity is modeled.
- We assume an ideal MAC layer without collisions and link unreliability is captured via a uniform packet loss probability.
- We do not model physical characteristics beyond a simple connectivity model based on transmission range and TX power level.
- All nodes are static throughout each simulation run.

These simplifications allow us to isolate and evaluate the energy use and routing efficiency of the protocol itself, independent of other effects that would vary across hardware and deployment.

## IV. Hybrid Cluster-Tree + Mesh Network Design

The proposed protocol features a lightweight, self-organizing **cluster-tree** that guarantees reachability in the network, and an opportunistic **mesh** that can bypass the tree when a route exists in the N-hop neighbor table. The backbone provides a structured hierarchy that ensures reachability; while the mesh opportunistically reduces hop count, latency, and energy when local connectivity permits.

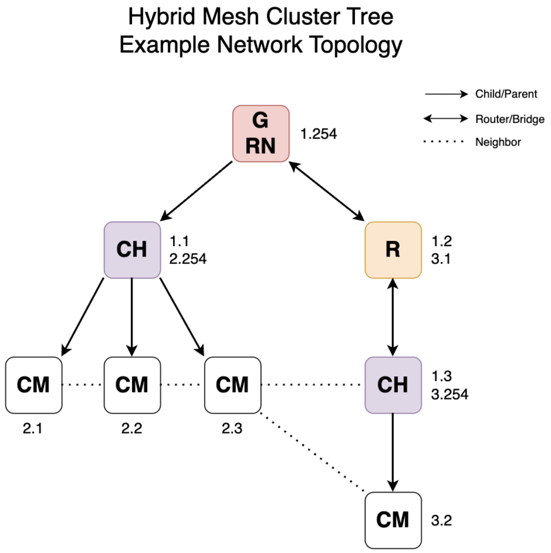

Fig. 1: A high-level architecture of the Hybrid Cluster Tree Mesh Network

### A. Design Overview

**Cluster-Tree:** Nodes form clusters, each managed by a Cluster Head (CH) with a unique NET_ID (Network ID). Each cluster member (CM) receives a unique NODE_ID within that NET_ID. CHs connect upward toward the Root Node (RN), which allocates NET_IDs to CHs. This forms a tree of clusters that can route traffic between any two clusters using hierarchical parent/child relationships.

**Mesh:** Every node maintains an N-hop neighbor table populated by HEARTBEAT overhearing and neighbor table sharing. If a destination (or its parent CH) is reachable via the neighbor table, packets are forwarded directly (peer-to-peer) instead of traversing the tree (except for some packet types which must traverse the tree to build the hierarchy).

**Dynamic Addressing:** Every dynamic address is the combination of a NET_ID and NODE_ID. A dynamic address (NET_ID.NODE_ID) resolves to a single node at any moment, but a single physical node may be addressed by multiple dynamic addresses (e.g., acting as a CM in one cluster while simultaneously acting as a CH for another). A globally unique 8 byte UID (EUI-64) is used to validate identity across role/address changes.

A high-level figure illustrating the architecture (tree backbone with cross-layer mesh links) is given in Figure 1.

### B. Roles, Capabilities, and Role Transitions

The proposed design separates what a node is doing now (attributes) from what it can do (capabilities). Broadcasting these fields enables distributed, local decisions without centralized orchestration.

**Roles:**

- **CM:** Ordinary member of a cluster; may also act as CH/R/RN/GW if capable.
- **CH:** Manages one cluster address space; assigns NODE_IDs and maintains a Member Table.
- **R:** Bridges clusters when topology suggests a dedicated bridge is preferable to redundant CH chains.
- **RN:** Allocates new NET_IDs and acts as a CH for its local cluster (1.254 reserved).
- **GW:** Provides external network connectivity; may coincide with RN or be separate.

**Promotion triggers:**

1) **CH promotion (CM → CH):** If an unregistered node attempts to join through a CM and no reachable CH can serve it, the CM requests a NET_ID allocation from the RN (NETID_REQ/NETID_RESP) and becomes a CH upon success.
2) **Router formation (CM/CH → R):** If a node primarily relays between clusters (e.g., it sits between two CHs and has no local members), it may assume an R role to reduce unnecessary cluster depth.
3) **CH demotion (CH → R/CM):** If a CH becomes redundant (no members; only bridging), it may demote to R or CM to simplify the backbone and reduce maintenance overhead.

### C. Cluster Formation and Join Workflow

The protocol supports cold-start: nodes can discover a network, join it, and receive an address lease without pre-configuration.

**Discovery:** An unregistered node broadcasts PROBE messages. Responders send HEARTBEAT messages describing role flags, capabilities, and a path_cost (to root). The joining node populates its neighbor table with the received HEARTBEAT messages and selects a candidate CH (or CM that can promote) using lowest local (distance cost) cost and network cost (path cost).

**Join (CH available):**

1) Node sends JOIN_REQ (unicast) to selected CH, including UID and capabilities.
2) CH assigns a NODE_ID, creates a Member Table entry, and responds with JOIN_ACK (broadcast) containing targetUid, assigned clusterAddr, and address validity time.
3) New CM confirms via ACK (matching seqNo); CH clears renew state.

**Join (promotion required):** If no CH is reachable directly, the CM triggers promotion: it requests a NET_ID from the RN, becomes CH, and then completes the join.

### D. Routing Rules

Every forwarded packet carries a DST and a NH (Next Hop). Forwarders *rewrite NH* based on local tables, while preserving the DST. This keeps the end-to-end destination stable while allowing routing rules to be applied hop-by-hop.

- **Mesh:** For data packets, if DST or its CH (NET_ID.254) exists in the Neighbor Table, forward to the recorded next hop. Control packets (NETID_REQ, NETID_RESP, JOIN_ACK, ACK) bypass mesh routing to preserve the tree hierarchy.

- **Router:** A Router (R) bridges clusters by forwarding packets destined for its downstream network, or packets arriving from its parent, to its downstream CH.
- **Tree:** CHs deliver locally if target_net == my_net, or forward via the Child Net Table for child networks. If no route is found, packets are forwarded to the parent.

**Rationale:** Mesh routing reduces hop count and energy cost in dense deployments by exploiting local connectivity. The tree provides guaranteed reachability along a known hierarchy: ordinary nodes need only a parent entry and Neighbor Table, while only CHs need to maintain Member and Child Net Tables.

### E. Self-Healing Protocol

The protocol treats topology as dynamic, continuously expiring stale state to adapt to node failures and mobility.

**Neighbor Table:** Updated on any receive (including overheard packets). Entries age out after a configurable window to prevent routing through dead nodes.

**Member Table (CH):** Uses address leases. If a CM fails to send a HEARTBEAT before expiry, the CH removes the entry and frees the NODE_ID.

**Child Net Table (CH):** Populated by observing NETID_REQ/NETID_RESP traversals. Entries transition PENDING → VALID on promotion, becoming STALE if unheard. Stale entries are purged, forcing packets to route upward until a valid path is re-learned.

**Failure handling examples:**

- **Lost parent CH:** CM times out neighbor/parent; re-enters discovery (PROBE) and rejoins.
- **Lost CH with active members:** Members simultaneously lose parent; reattach to alternate CHs or trigger local promotions, rebuilding clusters outward from surviving nodes.
- **Broken inter-cluster bridge:** Tree routing naturally falls back upward;

### F. Energy and Resource Usage

**Memory:** Ordinary nodes store only the Neighbor Table, bounded by $N$ entries at $N$ hops. CHs additionally maintain the Member Table (bounded by cluster size) and Child Net Table (bounded by child networks).

**Energy:** Neighbor learning via passive overhearing of HEARTBEAT messages requires no TX energy. Nodes dynamically adjust transmission power based on distance to the next hop, reducing energy consumption and extending network lifetime.

**Hop Count:** Mesh forwarding reduces hop count in dense deployments by enabling horizontal shortcuts. In sparse deployments, tree routing guarantees reachability via hierarchical forwarding. Without mesh links, packets must traverse upward before descending to the destination cluster.

TABLE I: Default Simulation Parameters

| Parameter | Value |
|---|---|
| Network size ($N$) | 100 nodes |
| Deployment area | 100 × 100 m |
| Initial energy ($E_0$) | 0.25 mAh |
| Traffic interval ($T_s$) | 1.0 s |
| Packet loss probability ($p_{\text{loss}}$) | 0.01 |

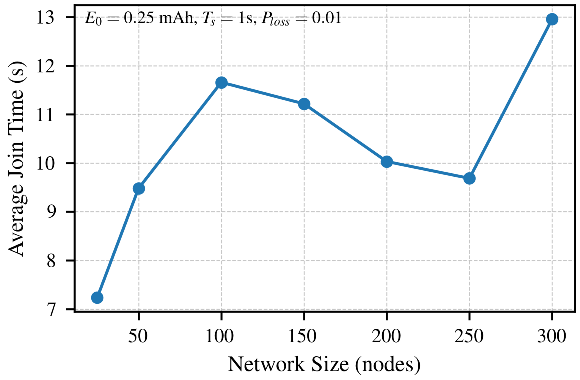

Fig. 2: Average join time versus network size. Parameters: $E_0 = 0.25\,\text{mAh}$, $T_s = 1\,\text{s}$, $p_{\text{loss}} = 0.01$.

## V. Simulation and Experiments

We implement the hybrid protocol in WSNLAB, a discrete-event Python simulator. The network, traffic, and energy models follow Section III. Unless otherwise noted, experiments use the default parameters in Table I.

### A. Metrics

We evaluate the protocol using the following metrics:

- **Join time:** Time from node power-on until successful registration with a CH.
- **Connectivity:** Fraction of nodes with a valid route to the sink over time.
- **Packet delivery ratio (PDR):** Fraction of generated packets successfully delivered.
- **Node lifetime:** Time until a node depletes its energy budget. Nodes with $E_i(t) \leq 0$ are permanently disabled.
- **Network lifetime:** Time until connectivity drops below 80%, capturing when the network is no longer operationally useful.

### B. Experiments

We organize experiments into three categories: (1) scalability, varying network size from 25–300 nodes; (2) reliability, varying packet loss probability $p_{\text{loss}}$; and (3) energy, varying initial energy $E_0$ and traffic interval $T_s$. We also compare tree-only routing against the hybrid tree+mesh configuration.

*1) Network Lifetime vs. Initial Energy:* We vary $E_0$ while holding network size, placement, and traffic constant. Network lifetime is defined as the time until connectivity drops below 80%.

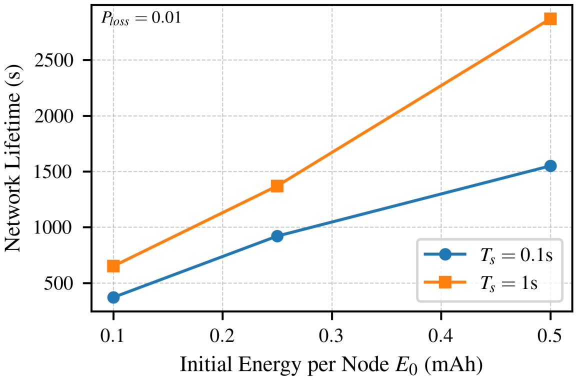

Fig. 3: Network lifetime as a function of the initial energy budget $E_0$ per node, for different traffic intervals ($T_s = 0.1$ s and $T_s = 1.0$ s). Network lifetime is defined as the time until fewer than 80% of the nodes remain connected to the sink. Parameters: $p_{\text{loss}} = 0.01$.

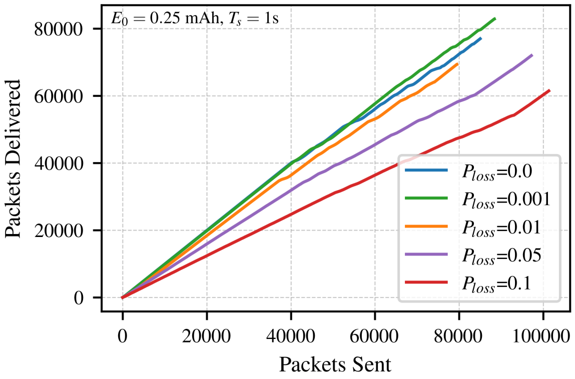

Fig. 4: Cumulative packets delivered versus packets sent under different packet loss probabilities ($p_{\text{loss}} = 0.0$, 0.001, 0.01, 0.1). Higher loss probabilities result in greater deviation between sent and delivered packets. Parameters: $E_0 = 0.25$ mAh, $T_s = 1$ s.

*2) Role-Based Energy Depletion:* In the second experiment set, we analyze the energy consumption of different node roles in the hybrid cluster-tree and mesh topology. Cluster heads and mesh forwarders are expected to relay a larger fraction of the traffic and to transmit more control packets than leaf nodes. To quantify this effect, we log for each node its role, time of death (if any), and total time spent in each radio state. We then compare:

- the average lifetime of cluster heads and mesh forwarders against that of leaf nodes, and
- the average energy breakdown (TX/RX/IDLE/SLEEP) per role.

This experiment highlights whether critical nodes (e.g., cluster heads and mesh forwarders) tend to deplete their batteries prematurely and how their failure impacts the remaining topology.

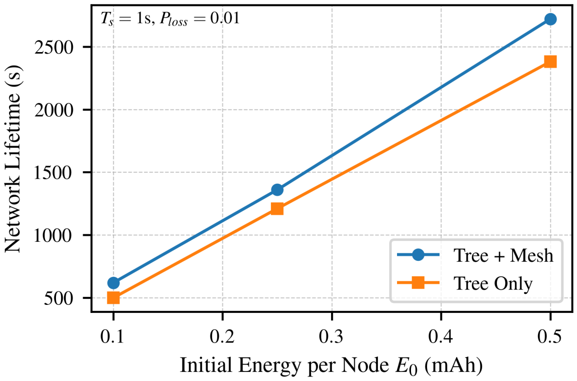

Fig. 5: Comparison of network lifetime between tree-only and tree+mesh (hybrid) routing as a function of initial energy $E_0$. Network lifetime is defined as the time until fewer than 80% of the nodes remain connected to the sink. Parameters: $T_s = 1$ s, $p_{\text{loss}} = 0.01$.

*3) Impact of Traffic Load on Energy Consumption:* In the third experiment set, we study the trade-off between traffic load and network lifetime. The topology and initial energy $E_0$ are kept fixed, while the application data rate is varied (e.g., low, medium, and high packet generation rates per node). For each traffic level, we monitor:

- the average remaining energy over time,
- the number and fraction of alive nodes,
- the number and fraction of nodes with a route to the sink, and
- the packet delivery ratio (PDR) computed over sliding time windows.

By comparing these metrics across traffic configurations, we evaluate how increased load accelerates energy depletion and reduces the time for which the network can sustain a target connectivity or reliability level.

### C. Results

*1) Scalability: Join Time:* Fig. 2 shows the average join time as a function of network size. Join time increases from approximately 7.2 s at 25 nodes to 13 s at 300 nodes—a sub-linear increase relative to the 12× growth in network size. Interestingly, a plateau occurs between 150–250 nodes, where denser deployments provide more CH candidates nearby, reducing discovery latency. The increase at 300 nodes suggests control packet congestion during formation in very dense networks.

*2) Network Lifetime vs. Initial Energy:* Fig. 3 confirms a linear relationship between initial energy $E_0$ and network lifetime. At $T_s = 1$ s, lifetime increases from 650 s to 2850 s as $E_0$ grows from 0.1 to 0.5 mAh. High traffic ($T_s = 0.1$ s) reduces lifetime by approximately 45% compared to low traffic. The widening gap at higher energy levels indicates that traffic load becomes the dominant factor when energy budgets are less constrained.

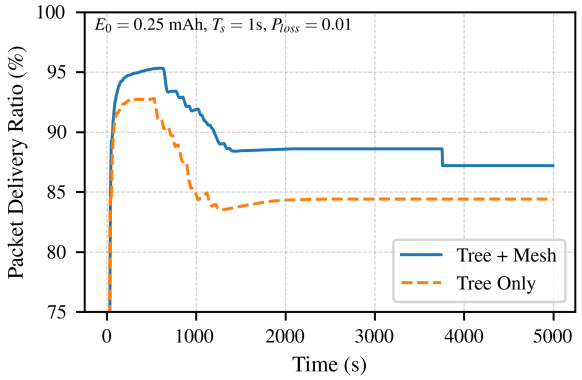

Fig. 6: Packet delivery ratio (PDR) over time comparing tree-only and tree+mesh routing. Parameters: $E_0 = 0.25\,\text{mAh}$, $T_s = 1\,\text{s}$, $p_{\text{loss}} = 0.01$.

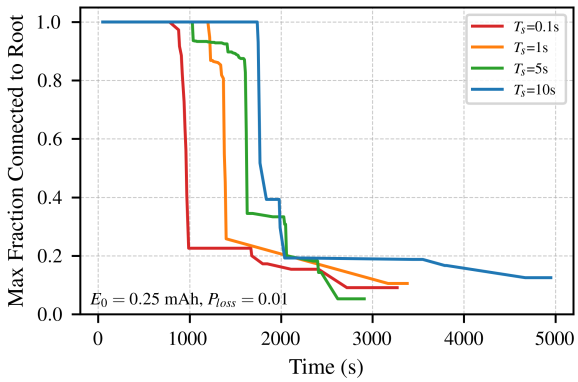

Fig. 7: Fraction of nodes connected to the sink over time for different traffic intervals ($T_s = 0.1,\ 1.0,\ 5.0,\ 10.0\,\text{s}$). Higher traffic loads reduce the effective network lifetime. Parameters: $E_0 = 0.25\,\text{mAh}$, $p_{\text{loss}} = 0.01$.

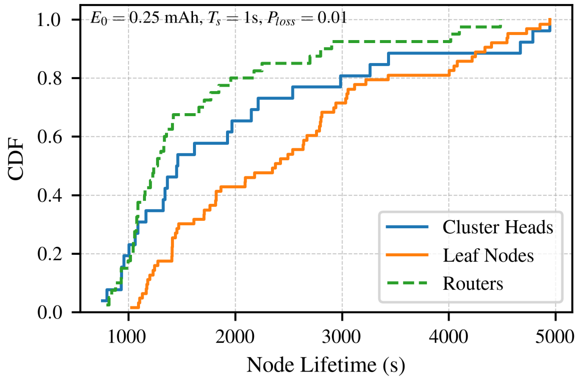

Fig. 8: Cumulative distribution function (CDF) of node lifetimes by role: cluster heads, leaf nodes, and routers. Parameters: $E_0 = 0.25\,\text{mAh}$, $T_s = 1\,\text{s}$, $p_{\text{loss}} = 0.01$.

*3) Reliability Under Packet Loss:* Fig. 4 shows cumulative packet delivery under varying loss probabilities. The protocol maintains good delivery up to $p_{\text{loss}} = 0.01$, achieving approximately 75% cumulative PDR. At $p_{\text{loss}} = 0.1$, cumulative loss reaches 38%, as multi-hop paths compound per-hop losses. The near-linear curves indicate steady-state behavior without catastrophic failure, demonstrating robustness to link unreliability.

*4) Tree vs. Mesh: Network Lifetime:* Fig. 5 compares network lifetime between tree-only and hybrid tree+mesh routing. The hybrid approach consistently outperforms tree-only across all energy levels: 24% improvement at $E_0 = 0.1\,\text{mAh}$ (620 s vs. 500 s), 14% at 0.25 mAh (1370 s vs. 1200 s), and 12.5% at 0.5 mAh (2700 s vs. 2400 s). The benefit is most pronounced at lower energy budgets, confirming that mesh shortcuts reduce hop count and energy per packet.

*5) Tree vs. Mesh: Packet Delivery Ratio:* Fig. 6 shows PDR over time for both routing modes. The hybrid approach achieves a peak PDR of 95% compared to 93% for tree-only. More importantly, tree-only PDR drops sharply to 84% at approximately 1000 s as CH failures disrupt routes, while tree+mesh maintains 88–89% PDR until 3700 s. The sustained 4% PDR advantage demonstrates that mesh links provide alternative paths when tree links fail, improving resilience.

*6) Impact of Traffic Load:* Fig. 7 shows connectivity over time for different traffic intervals. Network lifetime (80% connectivity threshold) varies dramatically: approximately 800 s for $T_s = 0.1\,\text{s}$, 1000 s for $T_s = 1\,\text{s}$, 1500 s for $T_s = 5\,\text{s}$, and 1800 s for $T_s = 10\,\text{s}$. Notably, all curves exhibit steep "cliff" drops when critical CHs fail—connectivity collapses rapidly rather than degrading gradually. A 10× reduction in traffic rate roughly doubles network lifetime.

*7) Role-Based Energy Depletion:* Fig. 8 presents the CDF of node lifetimes by role. Routers die earliest with a median lifetime of approximately 1500 s, followed by cluster heads at 2000 s, while leaf nodes survive longest with a median of 2500 s. The 1000 s spread between roles confirms significant energy imbalance—critical infrastructure nodes deplete faster due to forwarding and control overhead. This explains the connectivity cliff: the network disconnects while many leaf nodes still have substantial remaining energy. The results suggest that CH rotation or load-balancing mechanisms could significantly extend network lifetime.

## VI. Conclusion and Future Work

## References

[1] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, 2000, pp. 10 pp. vol.2–.

[2] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*, 1999, pp. 90–100.

[3] D. B. Johnson, D. A. Maltz, J. Broch *et al.*, "Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks," *Ad hoc networking*, vol. 5, no. 1, pp. 139–172, 2001.

[4] S. Lindsey and C. Raghavendra, "Pegasis: Power-efficient gathering in sensor information systems," in *Proceedings, IEEE Aerospace Conference*, vol. 3, 2002, pp. 3–3.

[5] S. C. Ergen, "Zigbee/ieee 802.15. 4 summary," *UC Berkeley, September*, vol. 10, no. 17, p. 11, 2004.

[6] O. Younis, M. Krunz, and S. Ramasubramanian, "Node clustering in wireless sensor networks: recent developments and deployment challenges," *IEEE Network*, vol. 20, no. 3, pp. 20–25, May 2006.

[7] T. M. Behera, S. K. Mohapatra, U. C. Samal, M. S. Khan, M. Daneshmand, and A. H. Gandomi, "Residual energy-based cluster-head selection in wsns for iot application," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5132–5139, June 2019.

[8] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Ad Hoc Networks*, vol. 2, no. 1, pp. 1–22, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S157087050300043X