Agent Deployment Algorithms for Heterogeneous Wireless Sensor Networks in a Remote Monitoring Mission

Banu Kabakulak*, *Member, IEEE*, E. Alper Yıldırım

***Abstract*—This study addresses the Network Agent Deployment (NAD) problem in heterogeneous wireless sensor networks (WSNs), which aims to determine (i) the locations of sensor nodes, actuators, routers, and gateways, and (ii) the configuration of sensor nodes, including sensor types as well as battery, memory, and transceiver models, so as to minimize total deployment cost. The design ensures network lifetime, connectivity, coverage, and reliability requirements while achieving balanced gateway loads through proper clustering and agent-to-gateway assignments.**

We propose three solution approaches: CNAD for global optimality, DeNAD as a relaxed formulation yielding near-optimal solutions with lower computational effort, and a Greedy heuristic for lightweight, scalable deployment suitable for embedded systems. Computational experiments demonstrate trade-offs between solution quality and computation time, and suitability of the proposed methods for applications in agricultural monitoring, environmental surveillance, and battlefield scenarios.

***Index Terms*—Agent deployment, heuristics, internet of things, narrow-band wireless sensor networks, network layer, mixed-integer programming.**

I. Introduction

WIRELESS Sensor Networks (WSNs) are composed of spatially distributed, autonomous, low-power devices—referred to as *agents*—that wirelessly collect and transmit data from a target region [1]. They are particularly effective in remote or harsh environments where human intervention is infeasible, enabling applications across environmental monitoring, agriculture, smart cities, and military surveillance [2]–[5].

This research aims to generate an optimal *network topology* for a *heterogeneous* WSN to continuously collect data from a remote agricultural land by deploying *immobile* multi-type agents, namely *sensor nodes*, *actuators*, *routers*, and *gateways*. A sensor node includes multiple sensors, and uses a battery, an onboard memory, and a radio transceiver in common as depicted in Fig. 1. That is, a node perceives the neighborhood with the sensing unit, processes the data in the computing unit, and shares the data via the communication unit relying on the limited energy of the power unit [6], [7].

This research has been supported by The Scientific and Technological Research Council of Türkiye (TÜBİTAK).

B. Kabakulak is with the Department of Industrial Engineering, İstanbul Bilgi University, İstanbul, Türkiye. *Corresponding author, e-mail: banu.kabakulak@boun.edu.tr, url: https://www.banukabakulak.com.

E.A. Yıldırım is with the Maxwell Institute of Mathematical Sciences, School of Mathematics, University of Edinburgh, Edinburgh, United Kingdom. E-mail: E.A.Yildirim@ed.ac.uk

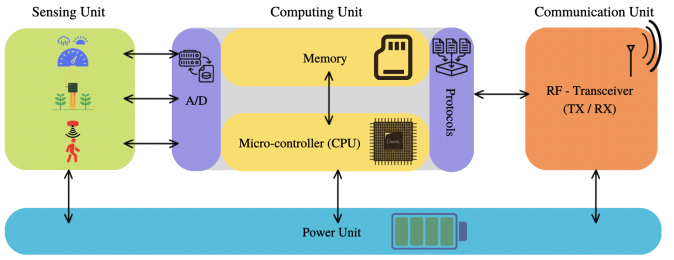

Fig. 1. Generic sensor node infrastructure.

An *actuator* can be employed for various control tasks, such as initiating or terminating irrigation, or regulating light and heating in a greenhouse. A *router* functions as a relay device, collecting sensed data from neighboring sensor nodes and forwarding it toward the *gateways*, which ultimately transmit the data to a cloud database for storage and analysis.

The design of a WSN relies on five protocol layers—*physical*, *data link*, *network*, *transport*, and *application*—which define its communication architecture [8]. The *network layer* is especially critical, managing routing from sensor nodes to gateways while supporting mobility, dynamic topology changes, and fault tolerance. Since sensor nodes are typically battery-powered without energy harvesting, this layer must be highly energy-efficient, as the radio transceiver consumes a significant portion of total energy [9], [10]. Key *Quality of Service* (QoS) metrics for real-time agricultural monitoring include *latency*, *coverage*, *energy efficiency*, *network lifetime*, *reliability*, *scalability*, and *fault tolerance* [11], [12]. Prioritizing these metrics ensures timely and reliable delivery of actionable data for precision agriculture.

Various strategies have been developed to address agent deployment, connected coverage, and energy efficiency in WSNs. For agent deployment, mobile agents allow dynamic adaptation to environmental changes, ensuring reliable coverage in applications such as crisis management and object tracking [13], [14]. Optimization-based methods, including Orthogonal Teaching-Learning-Based Optimization (OTLBO), determine optimal node placements to enhance coverage, connectivity, and cost efficiency [15], while hybrid approaches integrate intelligent algorithms with robotic deployment to handle challenging environments [16]. Connected coverage techniques aim to monitor all areas of interest while maintaining network connectivity, ranging from energy-efficient

protocols that balance coverage and network lifetime [17], to location-unaware connected dominating sets [18], hybrid optimization for router placement [19], and $k-$coverage or multi-point coverage strategies [20], [21]. Energy efficiency is further promoted through routing protocols that minimize transmission energy [22]–[24], data aggregation to reduce redundant transmissions [25], [26], selective node activation or sleep scheduling [27], [28], and AI- or cross-layer-based optimization for adaptive energy management [29]. Together, these strategies improve WSN performance in deployment, coverage, and energy management.

Building on these strategies, the major contributions of this research can be summarized as follows:

1) We formally define the *Network Agent Deployment* (NAD) problem to determine the optimal locations and configurations of sensor nodes, actuators, routers, and gateways, aiming to minimize deployment cost while meeting coverage, connectivity, network lifetime, reliability, and balanced gateway load requirements.
2) We introduce centralized (CNAD) and decomposed (DeNAD) mixed-integer linear formulations. To the best of our knowledge, these are the first cross-layer optimization models that enable custom sensor node configurations while guaranteeing bidirectional agent-to-gateway communication.
3) We develop three solution methods: CNAD for global optimality, DeNAD for near-optimal solutions with lower computational effort, and a Greedy heuristic that is lightweight, scalable, and suitable for embedded systems.
4) The cross-layer deployment framework ensures energy-efficient, fault-tolerant, and low-latency operation by clustering agents, assigning them to gateways, and enforcing hop and coverage constraints.

The rest of the paper is organized as follows: we describe the NAD problem in a remote monitoring mission formally in Sec. II, introduce the centralized and decomposed mathematical formulations for the NAD problem in Sec. III, develop exact and heuristic solution approaches in Sec. IV, evaluate performance of the proposed algorithms via computational experiments in Sec. V, and provide some concluding remarks on the future research directions in Sec. VI.

II. System Description

A WSN monitors a discrete set $\mathcal{N}$ of target points remotely with multi-type sensors located in a sensor node as in Fig. 2. The data packet of a sensor node follows a multi-hop transmission route, which may possibly go through routers, to reach a gateway. The gateways then direct the instantaneous network data to a cloud database via GSM. The data is processed in the cloud to determine the abnormal cases in the region such as drought, infestation and intruders. The cloud automatically updates the dashboard on the farmer's cellular phone about the current status of the region. Then, either the farmer or the cloud intelligence can command the actuators to react to the unexpected cases immediately. This research ultimately focuses on the optimal configuration of the sensor nodes when multi-type sensors and alternative hardware models are given, and on the construction of an energy-efficient network topology with connected coverage and budget restrictions in the ground network.

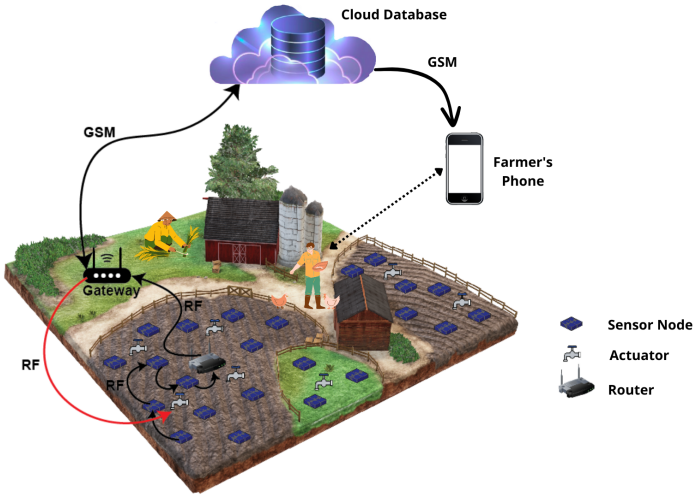

Fig. 2. A WSN in an agricultural remote monitoring mission.

The sensor nodes, actuators, routers and gateways are represented with the agent type indices n, a, r and g, respectively. The set $\mathcal{N}$ of target points is also the set of candidate deployment positions for the network agents. In the network, the set $\mathcal{A} = \{n, a, r, g\}$ denotes the agent types, and the set $\mathcal{S}$ collects the sensor types. A pair $(j, k) \in \mathcal{N} \times \mathcal{A}$ indicates a type k agent located at point j, while a type $s \in \mathcal{S}$ sensor at point $j \in \mathcal{N}$ is represented by the pair (j, s).

Each agent is equipped with a battery, memory and radio transceiver characterized by model indices b, m and t, respectively. For agents of type $k \in \{a, r, g\}$, the hardware models b, m, and t are standardized. In contrast, sensor nodes can be customized by selecting from a range of hardware models with varying technical capabilities (see Fig. 1). A key objective of this research is to optimally select battery models $b \in \mathcal{B}$, memory models $m \in \mathcal{M}$, and a transceiver model $t \in \mathcal{T}$ for each sensor node.

Each network agent is stored in a standard protective box to avoid hardware failures under harsh environmental conditions such as heat, cold, rain, and so on. A protective box costs c_p monetary units, and has a volume of V cm^3 with P pin connections, which bound the number of sensors in a sensor node. An agent of type $k \in \{a, r, g\}$ costs c_k monetary units, whereas the cost of a type $s \in \mathcal{S}$ sensor is c_s monetary units with a volume of v_s cm^3. The cost of a battery model b, memory model m, and transceiver model t is denoted by c_b, c_m, and c_t, respectively, where $b \in \mathcal{B}$, $m \in \mathcal{M}$, and $t \in \mathcal{T}$.

A point $i \in \mathcal{N}$ has a *differentiated coverage* requirement by at least f_{is} sensors of type s per τ_s minutes. The length of a period is $\tau^* = \max_{s \in \mathcal{S}} \tau_s$ mins, and the planning horizon is L periods. A sensor $(j, s) \in \mathcal{N} \times \mathcal{S}$ generates data of size h_s Kbits per period by detecting points located within a circle of radius ρ_s meters centered at point j. The *sensing status* parameter p_{ji} takes value one if a sensor (j, s) can sense point

TABLE I
LIST OF SENSOR NODE SPECIFICATIONS DECIDED BY NAD

Sensor Node Specifications	
c_j	cost of a sensor node located at point $j \in \mathcal{N}$
q_{ji}	1 if a sensor node at point $j \in \mathcal{N}$ can communicate with point i, 0 otherwise
h_j	data packet size of a sensor node at point $j \in \mathcal{N}$ generated per period
w_j	protocol overhead of a sensor node at point $j \in \mathcal{N}$
d_j	bandwidth of a sensor node at point $j \in \mathcal{N}$
e_j^a	energy consumed by a sensor node at point $j \in \mathcal{N}$ (with asleep sensors) per period
e_j^{tx}	energy consumed by a sensor node at point $j \in \mathcal{N}$ per unit data transmission
e_j^{rx}	energy consumed by a sensor node at point $j \in \mathcal{N}$ per unit data reception from other agents
E_j	available battery energy of a sensor node at point $j \in \mathcal{N}$
M_j	memory capacity of a sensor node at point $j \in \mathcal{N}$

i, and zero otherwise. A sensor of type s consumes e_s mAh to sense and process data per period.

An agent of type $k \in \{a, r, g\}$ generates a data packet of size h_k Kbits per period and transmits if the target agent at point i is within a circle of radius ρ_k^c meters centered at point j. The *communication status* parameter q_{jki} is set to one if an agent $(j, k) \in \mathcal{N} \times \{a, r, g\}$ can communicate with a point i, and zero otherwise.

The energy capacity of a nonrenewable battery model $b \in \mathcal{B}$ is E_b mAh, while the storage capacity of a memory model $m \in \mathcal{M}$ is Ω_m Kbits. Each protective box accommodates up to D battery docks and C memory compartments, limiting the number of battery and memory units that can be installed in a sensor node. A transceiver model $t \in \mathcal{T}$ has a communication range of ρ_t^c meters, and q_{jti} is the communication status of a transceiver model $t \in \mathcal{T}$ located at point j with point i. The total data traffic on a transceiver model $t \in \mathcal{T}$ is limited with the bandwidth d_t Kbits per period. A transceiver model $t \in \mathcal{T}$ consumes e_t^a mAh per period in active mode, e_t^{tx} mAh/Kbits for data transmission, and e_t^{rx} mAh/Kbits for data reception with a negligible deep sleep mode energy consumption.

The NAD problem customizes sensor nodes by optimally assigning sensors of type $s \in \mathcal{S}$ and selecting hardware models $b \in \mathcal{B}$, $m \in \mathcal{M}$, and $t \in \mathcal{T}$ (see TABLE I). That is, NAD designs the composition of a sensor node by optimizing its cost c_j, which accounts for the deployed sensors, shared hardware, and protective box. The configuration of a sensor node is defined as follows: (i) the data packet size h_j is determined by the total data generated by the assigned sensors, (ii) the communication status $\mathsf{q}_{ji} = q_{jti}$, bandwidth $\mathsf{d}_j = d_t$, and energy consumption parameters (active $\mathsf{e}_j^a = e_t^a$, transmission $\mathsf{e}_j^{tx} = e_t^{tx}$, and reception $\mathsf{e}_j^{rx} = e_t^{rx}$) are dictated by the selected transceiver model $t \in \mathcal{T}$, (iii) the memory capacity M_j is determined by the total capacity of the allocated memory units, (iv) the total available energy E_j is based on the combined capacity of the allocated batteries. We assume that the standard hardware models b, m, and t for agents of type $k \in \{a, r, g\}$ are sufficient to sustain operation throughout the lifetime of the network.

A gateway can collect data in a multi–hop fashion from type $k \in \{n, a, r\}$ agents. A gateway can command the agents within its communication range *directly*, such as triggering an actuator or switching an agent or a sensor between active and sleep modes. A type $k \in \{n, a, r\}$ agent located at point i must be covered by at least f_{ig} gateways to have a reliable network control. To have a fault tolerant and reliable agent–to–gateway routing, we assume that a type $k \in \{n, a, r\}$ agent must communicate with at least λ neighbors.

Due to bidirectional communication between agents and gateways, an agent of type $k \in \{n, a, r\}$ incurs a *protocol overhead*, which comprises the sum of the agent's transmitted data and the control commands received from the gateway. Specifically, the protocol overhead for agents of type $k \in \{a, r\}$ is given by $w_k = \mu_k + h_k$ Kbits per period, where μ_k represents the gateway control command in Kbits. For a sensor node at point j, the protocol overhead w_j additionally includes control commands for the allocated sensors, i.e., $\mathsf{w}_j = \mu_n + \sum_s \mu_s + \mathsf{h}_j$ Kbits per period. Here, μ_n and μ_s (in Kbits) correspond to the control commands for the sensor node and a sensor of type s, respectively.

The objective of NAD is to choose

(i) the locations of sensor nodes, actuators, routers, and gateways, and
(ii) the configuration of sensor nodes, i.e., determining the types of sensors and the battery, memory, and transceiver models

so as to minimize the total agent deployment cost while ensuring

(i) network lifetime, connectivity, coverage, and reliability requirements,
(ii) a balanced overhead allocation of gateways by properly clustering actuators, routers, and sensor nodes and assigning them to gateways.

III. MATHEMATICAL FORMULATIONS

A. Centralized Network Agent Deployment Model

The *centralized* NAD (CNAD) problem configures the sensor nodes and deploys the network agents while minimizing the budget required for operation over a specified planning horizon. In the CNAD model, the binary decision variable x_{jk} is equal to one if an agent $(j, k) \in \mathcal{N} \times \mathcal{A}$ is present in the network (see Table II).

The variables s_{js}, b_{jb}, m_{jm} and t_{jt} determine the configuration of a sensor node located at point $j \in \mathcal{N}$. In particular, $\mathsf{s}_{js} = 1$ indicates the deployment of a sensor (j, s), while $\mathsf{t}_{jt} = 1$ signifies that transceiver model $t \in \mathcal{T}$ is selected for the sensor node at point j. The variables b_{jb} and m_{jm} represent the number of allocated hardware units of model $b \in \mathcal{B}$ and $m \in \mathcal{M}$ at point j, respectively. The flow from an agent $(j, k) \in \mathcal{N} \times \mathcal{A}$ to another agent $(i, l) \in \mathcal{N} \times \mathcal{A}$ is captured by the continuous variable y_{jkil}, ensuring the establishment of an agent-to-gateway data route within the network. Additionally, a deployed agent $(j, k) \in \mathcal{N} \times \{n, a, r\}$ is assigned to an agent (i, g) using the binary variable u_{jki}, which is responsible for forming clusters in the network.

The CNAD objective (1) aims to minimize the total deployment cost of type $k \in \{n, r, g\}$ agents. The deployment

TABLE II
LIST OF PARAMETERS AND VARIABLES

Model Parameters	
ε	maximum ratio of protocol overhead deviation
δ	maximum number of hops in a data route
Decision Variables	
x_{jk}	1 if agent $(j,k) \in \mathcal{N} \times \mathcal{A}$ is used in the network, 0 otherwise
s_{js}	1 if sensor $(j,s) \in \mathcal{N} \times \mathcal{S}$ is used in the network, 0 otherwise
b_{jb}	number of battery model $b \in \mathcal{B}$ at point $j \in \mathcal{N}$
m_{jm}	number of memory model $m \in \mathcal{M}$ at point $j \in \mathcal{N}$
t_{jt}	1 if transceiver model $t \in \mathcal{T}$ used at point $j \in \mathcal{N}$, 0 otherwise
y_{jkil}	amount of flow from agent $(j,k) \in \mathcal{N} \times \mathcal{A}$ to agent $(i,l) \in \mathcal{N} \times \mathcal{A}$
u_{jki}	1 if agent $(j,k) \in \mathcal{N} \times \{n,a,r\}$ is assigned to agent (i,g), 0 otherwise

decision x_{ja} for actuators is assumed to be predetermined, as they are not flexible to relocate in practice.

$$\min \quad \sum_{j \in \mathcal{N}} \mathsf{c}_j + \sum_{(j,k) \in \mathcal{N} \times \{r,g\}} c_k \, \mathsf{x}_{jk} \tag{1}$$

Constraints (2) define the cost variable c_j of a sensor node based on the deployment decision x_{jn}, the selected sensors s_{js}, and the hardware models b_{jb}, m_{jm} and t_{jt}.

$$\begin{aligned} \mathsf{c}_j = c_p \, \mathsf{x}_{jn} + \sum_{s \in \mathcal{S}} c_s \, \mathsf{s}_{js} + \sum_{t \in \mathcal{T}} c_t \, \mathsf{t}_{jt} \\ + \sum_{m \in \mathcal{M}} c_m \, \mathsf{m}_{jm} + \sum_{b \in \mathcal{B}} c_b \, \mathsf{b}_{jb} \quad \forall j \in \mathcal{N} \end{aligned} \tag{2}$$

Constraints (3) ensure that each agent located at point j is covered by at least f_{jg} gateways. In order to deploy a sensor (j,s), an agent (j,n) must be present, as specified by constraints (4). Furthermore, constraints (5) prevent the deployment of sensor nodes without any assigned functionality. Constraints (6) and (7) impose upper bounds on the number of sensors at point j, limited by the available pins P and node volume V, respectively.

$$\sum_{i \in \mathcal{N}} q_{igj} \, \mathsf{x}_{ig} \;\geq\; f_{jg} \, \mathsf{x}_{jk} \qquad \forall \, (j,k) \in \mathcal{N} \times \{n,a,r\} \tag{3}$$

$$\mathsf{s}_{js} \;\leq\; \mathsf{x}_{jn} \qquad \forall \, (j,s) \in \mathcal{N} \times \mathcal{S} \tag{4}$$

$$\mathsf{x}_{jn} \;\leq\; \sum_{s \in \mathcal{S}} \mathsf{s}_{js} \qquad \forall \, j \in \mathcal{N} \tag{5}$$

$$\sum_{s \in \mathcal{S}} \mathsf{s}_{js} \;\leq\; P \qquad \forall \, j \in \mathcal{N} \tag{6}$$

$$\sum_{s \in \mathcal{S}} v_s \, \mathsf{s}_{js} \;\leq\; V \, \mathsf{x}_{jn} \qquad \forall \, j \in \mathcal{N} \tag{7}$$

Constraints (8) define the differentiated coverage requirement f_{is} of a point i by a sensor of type s. To extend the lifetime of the network, the parameter f_{is} can be increased, enabling the deployment of redundant sensor nodes. Finally, the data packet size h_j of an agent (j,n) is calculated as specified in constraints (9).

$$\sum_{j \in \mathcal{N}} p_{ji} \, \mathsf{s}_{js} \;\geq\; f_{is} \qquad \forall \, i \in \mathcal{N}, \; s \in \mathcal{S} \tag{8}$$

$$\mathsf{h}_j \;=\; \sum_{s \in \mathcal{S}} h_s \, \mathsf{s}_{js} \qquad \forall \, j \in \mathcal{N} \tag{9}$$

Constraints (10) allocate a transceiver of model $t \in \mathcal{T}$ to each agent (j,n). Subsequently, the communication status q_{ji} of an agent (j,n) with a point i is established by constraints (11). To ensure reliable communication, nonlinear constraints (12) and linear constraints (13) mandate the deployment of at least λ neighbors for each agent of type $k \in \{n,a,r\}$. This strategy ensures that there are at least λ alternative routes from an agent, meaning the agent remains connected as long as no more than λ neighbors fail.

$$\sum_{t \in \mathcal{T}} \mathsf{t}_{jt} \;=\; \mathsf{x}_{jn} \quad \forall \, j \in \mathcal{N} \tag{10}$$

$$\mathsf{q}_{ji} = \sum_{t \in \mathcal{T}} q_{jti} \; \mathsf{t}_{jt} \quad \forall \, i,j \in \mathcal{N} \tag{11}$$

$$\sum_{\substack{(i,l) \in \mathcal{N} \times \{n,r,g\} \\ (i,l) \neq (j,n)}} \mathsf{q}_{ji} \, \mathsf{x}_{il} \;\geq\; \lambda \, \mathsf{x}_{jn} \quad \forall \, j \in \mathcal{N} \tag{12}$$

$$\sum_{\substack{(i,l) \in \mathcal{N} \times \{n,r,g\} \\ (i,l) \neq (j,k)}} q_{jki} \, \mathsf{x}_{il} \;\geq\; \lambda \, \mathsf{x}_{jk} \quad \forall \, j \in \mathcal{N}, \; k \in \{a,r\} \tag{13}$$

Constraints (12) can be linearized as shown in constraints (15) using the auxiliary variable $\chi_{ilj} = \mathsf{q}_{ji} \, \mathsf{x}_{il}$, along with additional constraints (14), defined for $(i,l) \in \mathcal{N} \times \{n,r,g\}$ and $j \in \mathcal{N}$.

$$0 \leq \chi_{ilj} \leq \mathsf{x}_{il}, \;\; \chi_{ilj} \leq \mathsf{q}_{ji}, \;\; \chi_{ilj} \geq \mathsf{x}_{il} + \mathsf{q}_{ji} - 1 \tag{14}$$

$$\sum_{\substack{(i,l) \in \mathcal{N} \times \{n,r,g\} \\ (i,l) \neq (j,n)}} \chi_{ilj} \;\geq\; \lambda \, \mathsf{x}_{jn} \quad \forall \, j \in \mathcal{N} \tag{15}$$

Furthermore, constraints (16) specify the bandwidth d_j of an agent (j,n). Constraints (17) select a transceiver for an agent (j,n) with adequate bandwidth to handle both its own node data h_j and the data from its neighboring agents.

$$\mathsf{d}_j = \sum_{t \in \mathcal{T}} d_t \; \mathsf{t}_{jt} \quad \forall \, j \in \mathcal{N} \tag{16}$$

$$\begin{aligned} \mathsf{d}_j \;\geq\; \mathsf{h}_j + \sum_{i \in \mathcal{N}} \mathsf{q}_{ij} \, \mathsf{h}_i \\ + \sum_{\substack{i \in \mathcal{N} \\ l \in \{a,r\}}} q_{ilj} \, h_l \, \mathsf{x}_{il} - H \, |\mathcal{N}| \, (1 - \mathsf{x}_{jn}) \quad \forall \, j \in \mathcal{N} \end{aligned} \tag{17}$$

It is important to note that constraints (17) are nonlinear because h_i and q_{ij} depend on variables s_{is} and t_{jt}, respectively. To address this, we introduce the auxiliary variable $\phi_{ij} = \mathsf{q}_{ij} \, \mathsf{h}_i$, and the parameter $H = P \cdot \max_{s \in \mathcal{S}} \{h_s\}$. Using these, we derive the linear constraints (18) and (19). Consequently, the nonlinear constraints (17) are reformulated into their linear equivalents as shown in constraints (20).

$$0 \;\le\; \phi_{ij} \;\le\; H\,|\mathcal{N}|\,\mathsf{q}_{ij} \quad \forall\, i,j \in \mathcal{N} \tag{18}$$

$$\phi_{ij} \;\ge\; \mathsf{h}_i - H|\mathcal{N}|\,(1-\mathsf{q}_{ij}) \quad \forall\, i,j \in \mathcal{N} \tag{19}$$

$$\mathsf{d}_j \;\ge\; \mathsf{h}_j + \sum_{i\in\mathcal{N}} \phi_{ij} + \sum_{\substack{i\in\mathcal{N}\\ l\in\{a,r\}}} q_{ilj}\, h_l\, \mathsf{x}_{il} - H\,|\mathcal{N}|\,(1-\mathsf{x}_{jn}) \quad \forall\, j \in \mathcal{N} \tag{20}$$

Constraints (21) guarantee that each agent (j,n) is assigned at least one memory unit of model $m \in \mathcal{M}$, while the total number of memory units is limited by constraints (22). The memory capacity M_j of an agent (j,n) is given by constraints (23). Additionally, constraints (24) require that the total storage capacity of an agent (j,n) is sufficient to accommodate both its own data h_j and the data from its neighboring agents.

$$\sum_{m\in\mathcal{M}} \mathsf{m}_{jm} \;\ge\; \mathsf{x}_{jn} \quad \forall\, j \in \mathcal{N} \tag{21}$$

$$\sum_{m\in\mathcal{M}} \mathsf{m}_{jm} \;\le\; C \quad \forall\, j \in \mathcal{N} \tag{22}$$

$$\mathsf{M}_j = \sum_{m\in\mathcal{M}} \Omega_m\, \mathsf{m}_{jm} \quad \forall\, j \in \mathcal{N} \tag{23}$$

$$\mathsf{M}_j \;\ge\; \mathsf{h}_j + \sum_{i\in\mathcal{N}} \phi_{ij} + \sum_{\substack{i\in\mathcal{N}\\ l\in\{a,r\}}} q_{ilj}\, h_l\, \mathsf{x}_{il} - H\,|\mathcal{N}|\,(1-\mathsf{x}_{jn}) \quad \forall\, j \in \mathcal{N} \tag{24}$$

Network connectivity ensures that each agent has at least one data route to a gateway within the network. This requirement is modeled as a modified shortest-path problem, where the number of hops to a gateway is constrained by an upper bound δ. Specifically, constraints (25) limit the total inflow to each agent (j,k) to δ, while constraints (26) ensure that the path length does not exceed δ hops. This approach facilitates an even distribution of data routes across the network, balancing energy consumption during transmission, improving network latency, and enhancing reliability by minimizing the risk of data loss.

$$\sum_{(i,l)\in\mathcal{N}\times\mathcal{A}} \mathsf{y}_{iljk} \;\le\; \delta\,\mathsf{x}_{jk} \quad \forall\,(j,k) \in \mathcal{N}\times\{n,r\} \tag{25}$$

$$\mathsf{y}_{iljg} \;\le\; \delta\,\mathsf{x}_{jg} \quad \forall\, j \in \mathcal{N},\,(i,l) \in \mathcal{N}\times\{n,a,r\} \tag{26}$$

Constraints (27) ensure that there is at least one unit of outflow from an agent (j,k), under the assumption that *all* deployed agents are active. The inflow–outflow balance for an agent $(j,k) \in \mathcal{N}\times\{n,a,r\}$ is maintained by constraints (28). For gateway agents (i,g), constraints (29) ensure that the total inflow matches the total number of deployed agents.

$$\sum_{(i,l)\in\mathcal{N}\times\mathcal{A}} \mathsf{y}_{jkil} \;\ge\; \mathsf{x}_{jk} \quad \forall\,(j,k) \in \mathcal{N}\times\{n,a,r\} \tag{27}$$

$$\sum_{(i,l)\in\mathcal{N}\times\mathcal{A}} \mathsf{y}_{jkil} - \sum_{(i,l)\in\mathcal{N}\times\mathcal{A}} \mathsf{y}_{iljk} \;=\; \mathsf{x}_{jk} \quad \forall\,(j,k) \in \mathcal{N}\times\{n,a,r\} \tag{28}$$

$$\sum_{\substack{(j,k)\in\mathcal{N}\times\mathcal{A}\\ i\in\mathcal{N}}} \mathsf{y}_{jkig} \;=\; \sum_{(j,k)\in\mathcal{N}\times\{n,a,r\}} \mathsf{x}_{jk} \tag{29}$$

Constraints (30) and (31) guarantee that a recipient agent (i,l) is within the communication range of the transmitting agent (j,k). Type a agents are designed to transmit only their own data. Therefore, constraints (32) ensure that no inflow is allowed to agents of this type. Additionally, constraints (33) eliminate any outflow from gateways, and constraints (34) prohibit self–loops within the network.

$$\mathsf{y}_{jkil} \;\le\; \delta\, q_{jki} \quad \forall\,(j,k) \in \mathcal{N}\times\{a,r\}, (i,l) \in \mathcal{N}\times\mathcal{A} \tag{30}$$

$$\mathsf{y}_{jnil} \;\le\; \delta\,\mathsf{q}_{ji} \quad \forall\, j \in \mathcal{N}, (i,l) \in \mathcal{N}\times\mathcal{A} \tag{31}$$

$$\mathsf{y}_{ilja} = 0 \quad \forall\, j \in \mathcal{N},\; (i,l) \in \mathcal{N}\times\mathcal{A} \tag{32}$$

$$\mathsf{y}_{igjk} = 0 \quad \forall\, i \in \mathcal{N},\; (j,k) \in \mathcal{N}\times\mathcal{A} \tag{33}$$

$$\mathsf{y}_{jkjk} = 0 \quad \forall\,(j,k) \in \mathcal{N}\times\mathcal{A} \tag{34}$$

The formulation assumes all agents are active, with redundancy improving reliability. As CNAD is an offline topology design problem, its goal is to ensure a data route from each agent to a gateway. Data flow is modeled by hop counts without bandwidth or energy constraints, which are instead handled later by an online scheduling model.

The decision variable t_{jt}, representing the transceiver model $t \in \mathcal{T}$, determines the energy specifications e_j^a, e_j^{rx}, and e_j^{tx} for an agent (j,n). These specifications are governed by constraints (35), (36), and (37), respectively. Constraints (38) define the estimated energy consumption $\hat{\mathsf{E}}_j$ per period for an agent (j,n) as the total energy required to remain active, sense and process the neighborhood, receive data from neighboring agents and transmit data packets to the other agents.

$$\mathsf{e}_j^a = \sum_{t\in\mathcal{T}} e_t^a\, \mathsf{t}_{jt} \quad \forall\, j \in \mathcal{N} \tag{35}$$

$$\mathsf{e}_j^{rx} = \sum_{t\in\mathcal{T}} e_t^{rx}\, \mathsf{t}_{jt} \quad \forall\, j \in \mathcal{N} \tag{36}$$

$$\mathsf{e}_j^{tx} = \sum_{t\in\mathcal{T}} e_t^{tx}\, \mathsf{t}_{jt} \quad \forall\, j \in \mathcal{N} \tag{37}$$

$$\hat{\mathsf{E}}_j = \mathsf{e}_j^a + \sum_{s\in\mathcal{S}} e_s\, \mathsf{s}_{js} + \lambda H\, \mathsf{e}_j^{rx} + (1+\lambda)\, H\, \mathsf{e}_j^{tx} \quad \forall\, j \in \mathcal{N} \tag{38}$$

Constraints (39) ensure that each agent (j,n) is assigned at least one battery, while constraints (40) impose an upper limit on the number of battery units. The initial energy E_j specification of an agent (j,n) is then determined by constraints (41). This energy must be sufficient to sustain the agent's operation throughout the network planning horizon L, as required by constraints (42).

$$\sum_{b\in\mathcal{B}} \mathsf{b}_{jb} \;\geq\; \mathsf{x}_{jn} \quad \forall\, j \in \mathcal{N} \tag{39}$$

$$\sum_{b\in\mathcal{B}} \mathsf{b}_{jb} \;\leq\; D \quad \forall\, j \in \mathcal{N} \tag{40}$$

$$\mathsf{E}_j = \sum_{b\in\mathcal{B}} E_b\, \mathsf{b}_{jb} \quad \forall\, j \in \mathcal{N} \tag{41}$$

$$\mathsf{E}_j \;\geq\; \hat{\mathsf{E}}_j\, L \quad \forall\, j \in \mathcal{N} \tag{42}$$

An agent (j,k) with $k \in \{n,a,r\}$ is assigned to a unique gateway (i,g) through constraints (43). For this assignment to be feasible, constraints (44) ensure that the gateway (i,g) is deployed, while constraints (45) confirm that the gateway can communicate with the deployed agent (j,k).

$$\sum_{i\in\mathcal{N}} \mathsf{u}_{jki} = \mathsf{x}_{jk} \quad \forall\, (j,k) \in \mathcal{N}\times\{n,a,r\} \tag{43}$$

$$\sum_{(j,k)\in\mathcal{N}\times\{n,a,r\}} \mathsf{u}_{jki} \;\leq\; 3N\, \mathsf{x}_{ig} \quad \forall\, i \in \mathcal{N} \tag{44}$$

$$\mathsf{u}_{jki} \;\leq\; q_{igj}\, \mathsf{x}_{jk} \quad \forall\, i \in \mathcal{N}, (j,k) \in \mathcal{N}\times\{n,a,r\} \tag{45}$$

The protocol overhead *parameter* w_k of a type $k \in \{a,r\}$ agent is defined as the total bidirectional data flow with a gateway, as described in constraints (46). In contrast, the overhead *variable* w_j of an agent (j,n) also accounts for the total gateway command size μ_s of the deployed sensors as specified in constraints (47). Consequently, the total protocol overhead O_i of a gateway (i,g) is calculated as the sum of the protocol overheads of assigned agents, as expressed by nonlinear constraints (48).

$$w_k = \mu_k + h_k \quad \forall\, k \in \{a,r\} \tag{46}$$

$$\mathsf{w}_j = \mu_n\, \mathsf{x}_{jn} + \sum_{s\in\mathcal{S}} \mu_s\, \mathsf{s}_{js} + \mathsf{h}_j \quad \forall\, j \in \mathcal{N} \tag{47}$$

$$\mathsf{O}_i \;=\; \sum_{j\in\mathcal{N}} \mathsf{w}_j\, \mathsf{u}_{jni} + \sum_{(j,k)\in\mathcal{N}\times\{a,r\}} w_k\, \mathsf{u}_{jki} \quad \forall\, i \in \mathcal{N} \tag{48}$$

We introduce auxiliary variable $\psi_{ji} = \mathsf{w}_j\, \mathsf{u}_{jni}$ and define constraints (49) and (50) to obtain the linear equivalent, as presented in constraints (51).

$$0 \leq \psi_{ji} \leq H\, |\mathcal{N}|\, \mathsf{u}_{jni} \quad \forall\, i,j \in \mathcal{N} \tag{49}$$

$$\psi_{ji} \geq \mathsf{w}_j - H\, |\mathcal{N}|\, (1 - \mathsf{u}_{jni}) \quad \forall\, i,j \in \mathcal{N} \tag{50}$$

$$\mathsf{O}_i \;=\; \sum_{j\in\mathcal{N}} \psi_{ji} + \sum_{(j,k)\in\mathcal{N}\times\{a,r\}} w_k\, \mathsf{u}_{jki} \quad \forall\, i \in \mathcal{N} \tag{51}$$

To ensure a fair distribution of protocol overhead among the gateways, nonlinear constraints (52) and (53) enforce that the protocol overhead O_i of a gateway (i,g) deviates by no more than the specified ratio ε.

$$\sum_{j\in\mathcal{N}} \mathsf{x}_{jg}\, \mathsf{O}_i \;\geq\; (1-\varepsilon) \sum_{j\in\mathcal{N}} \mathsf{O}_j - 3H\, |\mathcal{N}|\, (1 - \mathsf{x}_{ig}) \tag{52}$$

$$\sum_{j\in\mathcal{N}} \mathsf{x}_{jg}\, \mathsf{O}_i \;\leq\; (1+\varepsilon) \sum_{j\in\mathcal{N}} \mathsf{O}_j + 3H\, |\mathcal{N}|\, (1 - \mathsf{x}_{ig}) \tag{53}$$

The linear versions can be derived as shown in constraints (56) and (57) using the auxiliary variable $\theta_{ij} = \mathsf{x}_{jg}\, \mathsf{O}_i$ and the associated constraints (54) and (55).

$$0 \;\leq\; \theta_{ij} \;\leq\; 3H\, |\mathcal{N}|\, \mathsf{x}_{jg} \quad \forall\, i,j \in \mathcal{N} \tag{54}$$

$$\theta_{ij} \;\geq\; \mathsf{O}_i - 3H\, |\mathcal{N}|\, (1 - \mathsf{x}_{jg}) \quad \forall\, i,j \in \mathcal{N} \tag{55}$$

$$\sum_{j\in\mathcal{N}} \theta_{ij} \;\geq\; (1-\varepsilon) \sum_{j\in\mathcal{N}} \mathsf{O}_j - 3H\, |\mathcal{N}|\, (1 - \mathsf{x}_{ig}) \tag{56}$$

$$\sum_{j\in\mathcal{N}} \theta_{ij} \;\leq\; (1+\varepsilon) \sum_{j\in\mathcal{N}} \mathsf{O}_j + 3H\, |\mathcal{N}|\, (1 - \mathsf{x}_{ig}) \tag{57}$$

Finally, constraints (58)–(60) and (63) impose the binary restrictions, constraints (61) and (62) establish the integrality conditions, and constraints (64) enforce non-negativity.

$$\mathsf{x}_{jk} \in \{0,1\} \quad \forall\, (j,k) \in \mathcal{N}\times\mathcal{A} \tag{58}$$

$$\mathsf{s}_{js} \in \{0,1\} \quad \forall\, (j,s) \in \mathcal{N}\times\mathcal{S} \tag{59}$$

$$\mathsf{t}_{jt} \in \{0,1\} \quad \forall\, j \in \mathcal{N},\; t \in \mathcal{T} \tag{60}$$

$$\mathsf{b}_{jb} \in \mathbb{Z}^+ \cup \{0\} \quad \forall\, j \in \mathcal{N},\; b \in \mathcal{B} \tag{61}$$

$$\mathsf{m}_{jm} \in \mathbb{Z}^+ \cup \{0\} \quad \forall\, j \in \mathcal{N},\; m \in \mathcal{M} \tag{62}$$

$$\mathsf{u}_{jki} \in \{0,1\} \quad \forall\, i \in \mathcal{N},\; (j,k) \in \mathcal{N}\times\{n,a,r\} \tag{63}$$

$$\mathsf{y}_{jkil} \geq 0 \quad \forall\, (j,k),(i,l) \in \mathcal{N}\times\mathcal{A} \tag{64}$$

The CNAD model is a Mixed–Integer Linear Programming (MILP) problem, which can be solved by state-of-the-art solvers. However, due to the exponential worst-case complexity of the underlying solution algorithms, CNAD can be solved to global optimality only for very small network sizes (see Section V). We therefore propose another solution method based on decomposing the problem into smaller subproblems in the next section.

B. Decomposed Network Agent Deployment Models

The *decomposed* NAD (DeNAD) approach addresses the computational complexity of the CNAD model by breaking it down into relaxed subproblems. Specifically, DeNAD determines decisions in the following sequence:

1) Gateway Deployment (GD) Model: The GD model deploys agents (j,g) to establish direct communication with all candidate points $i \in \mathcal{N}$ at minimal cost.

$$\min \quad \sum_{j\in\mathcal{N}} c_g\, \mathsf{x}_{jg} \tag{65}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{N}} q_{jgi}\, \mathsf{x}_{jg} \;\geq\; f_{ig} \quad \forall\, i \in \mathcal{N} \tag{66}$$

$$\mathsf{x}_{jg} \in \{0,1\} \quad \forall\, j \in \mathcal{N} \tag{67}$$

2) Sensor Deployment (SD) Model: The SD model determines the locations $j \in \mathcal{N}$ for deploying sensor nodes and assigns sensors of type $s \in \mathcal{S}$ to these nodes while minimizing costs. The sensor assignments s_{js} obtained from SD specify the data packet size h_j of an agent (j,n), as defined by Equation (9).

$$\min \quad \sum_{j\in\mathcal{N}} c_p\,\mathsf{x}_{jn} + \sum_{(j,s)\in\mathcal{N}\times\mathcal{S}} c_s\,\mathsf{s}_{js} \tag{68}$$
$$\text{s.t.} \quad (4)-(8),\ (59)$$
$$\sum_{(i,l)\in\mathcal{N}\times\{n,g\}} q_{jai}\,\mathsf{x}_{il} \ \geq\ \lambda\,\mathsf{x}_{ja} \quad \forall\, j\in\mathcal{N} \tag{69}$$
$$\mathsf{x}_{jn}\in\{0,1\} \quad \forall\, j\in\mathcal{N} \tag{70}$$

3) Node Transceiver Deployment (NTD) Model: Using the decisions x_{jg} from GD, and x_{jn} and h_j from SD as input parameters, the *linear* NTD model assigns a transceiver model $t\in\mathcal{T}$ to each agent (j,n) while minimizing costs. For an agent (j,n), the relaxed NTD model assigns the transmitter t_{jt}, determines the communication status q_{ji} with point i, and allocates the bandwidth d_j, assuming that *all* agents are active. These decisions define the energy consumption parameters e^a_j, e^{rx}_j, and e^{tx}_j through Equations (35), (36), and (37), respectively. Consequently, the estimated energy consumption $\hat{\mathsf{E}}_j$ is computed according to Equation (38).

$$\min \quad \sum_{j\in\mathcal{N},t\in\mathcal{T}} c_t\,\mathsf{t}_{jt} \tag{71}$$
$$\text{s.t.} \quad (10),(11),(16),(60)$$
$$\sum_{\substack{(i,l)\in\mathcal{N}\times\{n,g\}\\(i,l)\neq(j,n)}} \mathsf{q}_{ji}\,\mathsf{x}_{il} \ \geq\ \lambda\,\mathsf{x}_{jn} \quad \forall\, j\in\mathcal{N} \tag{72}$$
$$\mathsf{d}_j \ \geq\ \mathsf{h}_j + \sum_{i\in\mathcal{N}} \mathsf{q}_{ij}\,\mathsf{h}_i + \sum_{i\in\mathcal{N}} q_{iaj}\,h_a\,\mathsf{x}_{ia} + \lambda\,h_r - H\,|\mathcal{N}|\,(1-\mathsf{x}_{jn}) \quad \forall\, j\in\mathcal{N} \tag{73}$$
$$\mathsf{q}_{ij}\in\{0,1\},\ \mathsf{d}_j\geq 0 \quad \forall\, i,j\in\mathcal{N} \tag{74}$$

4) Node Memory Deployment (NMD) Model: The *linear* NMD model, which is decomposable for each agent (j,n), takes the decisions x_{jn} and h_j from SD, and q_{ij} from NTD as input parameters. In the relaxed NMD model, the number of memory units m_{jm} of model m is determined under the assumption that *all* agents are active, which in turn defines the memory capacity M_j as per Equation (23).

$$\min \quad \sum_{j\in\mathcal{N},m\in\mathcal{M}} c_m\,\mathsf{m}_{jm} \tag{75}$$
$$\text{s.t.} \quad (21)-(23),(62)$$
$$\mathsf{M}_j \ \geq\ \mathsf{h}_j + \sum_{i\in\mathcal{N}} \mathsf{q}_{ij}\,\mathsf{h}_i + \sum_{i\in\mathcal{N}} q_{iaj}\,h_a\,\mathsf{x}_{ia} + \lambda\,h_r - H\,|\mathcal{N}|\,(1-\mathsf{x}_{jn}) \quad \forall\, j\in\mathcal{N} \tag{76}$$
$$\mathsf{M}_j\geq 0 \quad \forall\, j\in\mathcal{N} \tag{77}$$

5) Router Deployment (RD) Model: The RD model utilizes the decisions x_{jg} from GD, x_{jn} from SD, and q_{ij} from NTD as input parameters. The deployment guarantees that each agent has a data flow route to a gateway within at most δ hops while minimizing costs.

$$\min \quad \sum_{j\in\mathcal{N}} c_r\,\mathsf{x}_{jr} \tag{78}$$
$$\text{s.t.} \quad (25)-(34),(64)$$
$$\sum_{\substack{(i,l)\in\mathcal{N}\times\{n,r,g\}\\(i,l)\neq(j,r)}} q_{jri}\,\mathsf{x}_{il} \ \geq\ \lambda\,\mathsf{x}_{jr} \quad \forall\, j\in\mathcal{N} \tag{79}$$
$$\mathsf{x}_{jr}\in\{0,1\} \quad \forall\, j\in\mathcal{N} \tag{80}$$

6) Node Battery Deployment (NBD) Model: The NBD model, which is decomposable for each agent (j,n), takes the decisions x_{jn} from SD, and $\hat{\mathsf{E}}_j$ from NTD as input parameters. The initial energy E_j of an agent (j,n) is determined by Equation (41).

$$\min \quad \sum_{j\in\mathcal{N},b\in\mathcal{B}} c_b\,\mathsf{b}_{jb} \tag{81}$$
$$\text{s.t.} \quad (39)-(42),(61)$$
$$\mathsf{E}_j\geq 0 \quad \forall\, j\in\mathcal{N} \tag{82}$$

7) Cluster Formation (CF) Model: The *linear* CF model calculates the protocol overhead parameters w_k and w_j using the decisions s_{js} and h_j from SD as defined in Equations (46) and (47), respectively. CF assigns each agent to a gateway based on the decisions x_{jg} from GD, x_{jn} from SD, and x_{jr} from RD, ensuring that the overhead deviation does not exceed a specified ratio ε. CF is formulated as a constraint programming problem and has a solution if and only if $\mathcal{F}_{\text{CF}}\neq\emptyset$, as defined in Equation (83).

$$\mathcal{F}_{\text{CF}} = \big\{\mathsf{O}_i\geq 0,(63)\mid (43)-(45),(48),(52),(53)\big\} \tag{83}$$

Consequently, DeNAD addresses the NAD problem by sequentially solving the relaxed models—GD, SD, NTD, NMD, RD, NBD, and CF—using an optimization solver. The deployment cost is then evaluated through the objective function (1) based on these decisions. In the next section, we present solver-independent Greedy heuristics designed for each subproblem.

IV. Greedy Heuristic Algorithms for DeNAD

For embedded system implementations, we develop solver-independent Greedy heuristics to address each subproblem of DeNAD.

The greedy heuristic G-GD solves the GD problem by iteratively placing gateways so every agent is covered by at least one gateway, selecting at each step the location j^* that covers the most uncovered agents. It runs in $\mathcal{O}(|\mathcal{N}|^2)$ time.

The greedy heuristic G-SD addresses the SD problem by selecting sensor types and locations to maximize coverage. For each candidate node $j\in\mathcal{N}$, V_j and P_j represent the remaining volume and available pins, defining the deployable set $\mathcal{D}$. At each step, the algorithm selects j^* and sensor type $s\in\mathcal{S}$ that cover the most undersensed points, using the indicator function $\mathbb{I}_{\{\text{condition}\}}$, which equals one if the *condition* is true and zero otherwise.

Algorithm 1 Greedy GD (G-GD) Algorithm

Input: An instance of GD

Output: Gateway deployment x_{jg} for $j \in \mathcal{N}$

1: Let $\mathsf{x}_{jg} = 0$ for $j \in \mathcal{N}$,
$\mathsf{U}_i = \max\{0, f_{ig} - \sum_{j\in\mathcal{N}} q_{jgi}\,\mathsf{x}_{jg}\}$ for $i \in \mathcal{N}$, and
$\mathcal{U} = \{i \in \mathcal{N} \mid \mathsf{U}_i > 0\}$ be the set of undercovered points.
2: **while** $\mathcal{U} \neq \emptyset$ **do**
3: $j^* = 0$
4: **if** $j^* = \arg\max_{j\in\mathcal{N}} \Big\{ \sum_{i\in\mathcal{U}} q_{jgi}\,(1 - \mathsf{x}_{jg}) \Big\} > 0$ **then**
5: Set $\mathsf{x}_{j^*g} = 1$, update $\mathsf{U}_i = \max\{0, \mathsf{U}_i - q_{j^*gi}\}$ values for $i \in \mathcal{N}$, and the set $\mathcal{U}$.
6: **else**
7: **break** *(infeasibility)*
8: **end if**
9: **end while**

Algorithm 2 Greedy SD (G-SD) Algorithm

Input: An instance of SD

Output: Sensor node x_{jn} and sensor s_{js} deployment for $j \in \mathcal{N}$, $s \in \mathcal{S}$

1: Let $\mathsf{x}_{jn} = 0$ and $\mathsf{s}_{js} = 0$ for $j \in \mathcal{N}$, $s \in \mathcal{S}$
$\mathsf{U}_{is} = \max\{0, f_{is} - \sum_{j\in\mathcal{N}} p_{ji}\,\mathsf{s}_{js}\}$ for $i \in \mathcal{N}$, $s \in \mathcal{S}$ and
$\mathcal{U}_s = \{i \in \mathcal{N} \mid \mathsf{U}_{is} > 0\}$ be the set of undersensed points.
2: Let $\mathsf{V}_j = \max\{0, V - \sum_{s\in\mathcal{S}} v_s\,\mathsf{s}_{js}\}$ and
$\mathsf{P}_j = \max\{0, P - \sum_{s\in\mathcal{S}} \mathsf{s}_{js}\}$ be the remaining volume and pins at sensor node $j \in \mathcal{N}$.
3: Let $\mathcal{D} = \{j \in \mathcal{N} \mid \min\{\mathsf{V}_j, \mathsf{P}_j\} > 0\}$ be the set of deployable points.
4: **for** each sensor $s \in \mathcal{S}$ **do**
5: **while** $\mathcal{U}_s \neq \emptyset$ **do**
6: $\mathsf{r}_j = \sum_{i\in\mathcal{U}_s} p_{ji}\,(1 - \mathsf{s}_{js})$ for $j \in \mathcal{D}$, and set $j^* = 0$
7: **if** $\max_{j\in\mathcal{D}} \Big\{ \mathsf{r}_j\,\mathsf{x}_{jn}\,\mathbb{I}_{\{v_s \le \mathsf{V}_j\}} \Big\} > 0$ **then**
8: $j^* = \arg\max_{j\in\mathcal{D}} \Big\{ \mathsf{r}_j\,\mathsf{x}_{jn}\,\mathbb{I}_{\{v_s \le \mathsf{V}_j\}} \Big\}$ and $\mathsf{s}_{j^*s} = 1$
9: **else**
10: $j^* = \arg\max_{j\in\mathcal{D}} \Big\{ \mathsf{r}_j\,(1 - \mathsf{x}_{jn}) \Big\}$
and $\mathsf{x}_{j^*n} = 1$, $\mathsf{s}_{j^*s} = 1$
11: **end if**
12: **if** $j^* > 0$ **then**
13: Update $\mathsf{U}_{is} = \max\{0, \mathsf{U}_{is} - p_{j^*i}\}$ for $i \in \mathcal{N}$,
$\mathsf{V}_{j^*} \leftarrow \mathsf{V}_{j^*} - v_s$, $\mathsf{P}_{j^*} \leftarrow \mathsf{P}_{j^*} - 1$,
and the sets $\mathcal{U}_s$ and $\mathcal{D}$.
14: **else**
15: **break** *(infeasibility)*
16: **end if**
17: **end while**
18: **end for**

Algorithm 3 Greedy NTD (G-NTD) Algorithm

Input: An instance of NTD

Output: Transceiver assignment t_{jt} to sensor node (j, n) for $j \in \mathcal{N}$, $t \in \mathcal{T}$

1: Let $\mathsf{t}_{jt} = 0$ and $\mathsf{C}_{jt} = \sum_{\substack{(i,l)\in\mathcal{N}\times\{n,g\} \\ (i,l)\neq(j,n)}} q_{jti}\,\mathsf{x}_{il} - \lambda\,\mathsf{x}_{jn}$.
2: **if** $\exists\, j \in \mathcal{N}$ with $(\mathsf{C}_{jt} < 0 \quad \forall\, t \in \mathcal{T})$ **then**
3: **break** *(infeasibility)*
4: **else**
5: **for** $|\mathcal{T}|$ rounds **do**
6: **for** each sensor node (j, n) **do**
7: $\mathsf{B}_{jt} = d_t - \left(\mathsf{h}_j + \sum_{\substack{i\in\mathcal{N} \\ t'\in\mathcal{T}}} \mathsf{q}_{it'j}\,\mathsf{t}_{it'}\,\mathsf{h}_i + \sum_{i\in\mathcal{N}} q_{iaj}\,h_a\,\mathsf{x}_{ia} + \lambda\,h_r \right)$
8: **if** $(\mathsf{B}_{jt} < 0 \quad \forall\, t \in \mathcal{T})$ **then**
9: **break** *(infeasibility)*
10: **else**
11: Let $t^* = \arg\max_{t\in\mathcal{T}} \Big\{ \frac{\mathsf{C}_{jt}\,\mathsf{B}_{jt}}{c_t} \Big\}$, and set $\mathsf{t}_{jt^*} = 1$
as the unique nonzero entry for $j \in \mathcal{N}$.
Break ties by selecting the minimum c_t.
12: **end if**
13: **end for**
14: **if** no update in t_{jt} for $j \in \mathcal{N}$, $t \in \mathcal{T}$ **then**
15: **break** *(feasibility)*
16: **end if**
17: **end for**
18: **end if**

If no existing node can host a new sensor, G-SD deploys a new node by setting $\mathsf{x}_{j^*n} = 1$ (Step 10). It runs in polynomial time with complexity $\mathcal{O}(|\mathcal{S}|\,|\mathcal{N}|^2)$.

The heuristic G-NTD solves the NTD problem by assigning transceivers to sensor nodes (j, n). It evaluates connectivity C_{jt} (Step 1) and available bandwidth B_{jt} (Step 7), then selects t^* maximizing connectivity and bandwidth while minimizing cost (Step 11). The process repeats up to $|\mathcal{T}|$ rounds and runs in $\mathcal{O}(|\mathcal{T}|\,|\mathcal{N}|^2)$ time.

The G-NMD algorithm solves the NMD problem by assigning memory units to each sensor node (j, n). It computes the *memory requirement* M_j (Step 3) and allocates up to $(C - 1)$ compartments to the cost-optimal model m^* (Step 5), assigning one more unit if needed (Steps 7–14). It runs in $\mathcal{O}(|\mathcal{N}|^2)$ time.

The G-RD algorithm solves the RD problem by deploying the minimum number of routers to ensure agent-to-gateway connectivity. It models the *communication network* $\mathcal{G}$ as a graph with agents as vertices and links as arcs. Using Breadth–First Search (BFS) up to depth δ, it identifies agents unable to reach a gateway and adds them to the disconnected set $\mathcal{D}$. In each iteration, it computes the *connectivity level* F_j of each node j, selecting j^* that maximizes communication with disconnected agents while satisfying the λ-connectivity constraint. The process repeats until all agents connect to a gateway. G-RD runs in $\mathcal{O}(|\mathcal{N}|^3)$ time.

The G-NBD heuristic provides an efficient solution to the

Algorithm 4 Greedy NMD (G−NMD) Algorithm

Input: An instance of NMD
Output: Memory assignment m_{jm} to sensor node (j,n) for $j \in \mathcal{N}$, $m \in \mathcal{M}$

Let $\mathsf{m}_{jm} = 0$ for $j \in \mathcal{N}$, $m \in \mathcal{M}$
for each sensor node (j,n) **do**
  $\mathsf{M}_j = \mathsf{h}_j + \sum_{i \in \mathcal{N}} \mathsf{q}_{ij}\, \mathsf{h}_i + \sum_{i \in \mathcal{N}} q_{iaj}\, h_a\, \mathsf{x}_{ia} + \lambda\, h_r$
  $m^* = \min_{m \in \mathcal{M}} \left\{ c_m \left\lfloor \frac{\mathsf{M}_j}{\Omega_m} \right\rfloor \right\}$
  Set $\mathsf{m}_{jm^*} = \min \left\{ C-1, \left\lfloor \frac{\mathsf{M}_j}{\Omega_{m^*}} \right\rfloor \right\}$
  $\mathsf{M}_j \leftarrow \mathsf{M}_j - \Omega_{m^*}\, \mathsf{m}_{jm^*}$ and $m^* = 0$
  if $\mathsf{M}_j > 0$ **then**
    $m^* = \min_{m \in \mathcal{M}} \left\{ c_m\, \mathbb{I}_{\{\Omega_m \geq \mathsf{M}_j\}} \right\}$
    if $m^* > 0$ **then**
      $\mathsf{m}_{jm^*} \leftarrow \mathsf{m}_{jm^*} + 1$
    else
      break *(infeasibility)*
    end if
  end if
end for

Algorithm 5 Greedy RD (G−RD) Algorithm

Input: An instance of RD
Output: Router deployment x_{jr} to for $j \in \mathcal{N}$

Let $\mathsf{x}_{jr} = 0$ for $j \in \mathcal{N}$
Apply BFS up to depth δ for each agent $(j,k) \in \mathcal{N} \times \mathcal{A}$ in the communication network $\mathcal{G}$ to reach a gateway. Let $\mathcal{D}$ be the set of disconnected agents in $\mathcal{G}$.
while $\mathcal{D} \neq \emptyset$ **do**
  Set $j^* = 0$
  Calculate $\mathsf{F}_j = \sum_{(i,l) \in \mathcal{N} \times \{n,r,g\}} q_{jri}\, \mathsf{x}_{il}$ for $j \in \mathcal{N}$
  $j^* = \arg\max_{j \in \mathcal{N}} \left\{ \left[\sum_{(i,l) \in \mathcal{D}} q_{ilj} + \sum_{(i,n) \in \mathcal{D}} \mathsf{q}_{ij} \right] \mathsf{F}_j\, \mathbb{I}_{\{\mathsf{F}_j \geq \lambda\}} \right\}$
  if $j^* > 0$ **then**
    Set $\mathsf{x}_{j^*r} = 1$
    Update the network $\mathcal{G}$ and the set $\mathcal{D}$.
  else
    break *(infeasibility)*
  end if
end while

NBD problem by assigning at least one battery unit to each sensor node (j,n). In Step 3, the algorithm estimates the energy consumption $\hat{\mathsf{E}}_j$ per period. The cost-optimal battery model b^*, determined in Step 4, is allocated up to $(D-1)$ units as described in Step 5. If additional capacity is required, one more unit is assigned to the final dock. The G−NBD algorithm operates in polynomial time, with a computational complexity of $\mathcal{O}\big(|\mathcal{B}|\,|\mathcal{N}|\big)$.

The G−CF algorithm efficiently addresses the CF problem by uniquely assigning agents to gateways, forming clusters with balanced communication overhead. In Step 3, the protocol overheads w_k and w_j for different agent types are computed, followed by the calculation of the average overhead $\bar{\mathsf{O}}$ in

Algorithm 6 Greedy NBD (G−NBD) Algorithm

Input: An instance of NBD
Output: Battery assignment b_{jb} to sensor node (j,n) for $j \in \mathcal{N}$, $b \in \mathcal{B}$

Let $\mathsf{b}_{jb} = 0$ for $j \in \mathcal{N}$, $b \in \mathcal{B}$
for each sensor node (j,n) **do**
  Calculate $\hat{\mathsf{E}}_j$ using Equation (38).
  $b^* = \min_{b \in \mathcal{B}} \left\{ c_b \left\lfloor \frac{\hat{\mathsf{E}}_j\, L}{E_b} \right\rfloor \right\}$
  Set $\mathsf{b}_{jb^*} = \min \left\{ D-1, \left\lfloor \frac{\hat{\mathsf{E}}_j\, L}{E_{b^*}} \right\rfloor \right\}$
  $\hat{\mathsf{E}}_j \leftarrow \hat{\mathsf{E}}_j - \frac{E_{b^*}\, \mathsf{b}_{jb^*}}{L}$ and $b^* = 0$
  if $\hat{\mathsf{E}}_j > 0$ **then**
    $b^* = \min_{b \in \mathcal{B}} \left\{ c_b\, \mathbb{I}_{\{E_b \geq \hat{\mathsf{E}}_j\, L\}} \right\}$
    if $b^* > 0$ **then**
      $\mathsf{b}_{jb^*} \leftarrow \mathsf{b}_{jb^*} + 1$
    else
      break *(infeasibility)*
    end if
  end if
end for

Algorithm 7 Greedy CF (G−CF) Algorithm

Input: An instance of CF
Output: Clustering assignment u_{jki} to agent (j,k) and gateway (i,g) for $i,j \in \mathcal{N}$, $k \in \{n,a,r\}$

Let $\mathsf{u}_{jki} = 0$ and $\mathsf{O}_i = 0$ for $i,j \in \mathcal{N}$, $k \in \{n,a,r\}$
$\mathcal{U} = \{(j,k) \in \mathcal{N} \times \{n,a,r\} \,|\, \mathsf{u}_{jki} = 0 \quad \forall i \in \mathcal{N}\}$ is the set of unclustered agents.
Calculate w_k and w_j using Equations (46) and (47).
Let $\mathsf{W}_{jk} = \mathsf{w}_j\, \mathbb{I}_{\{k=n\}} + w_k\, \mathbb{I}_{\{k \in \{a,r\}\}}$ be the agent overhead.
$\bar{\mathsf{O}} = \frac{\sum_{(j,k) \in \mathcal{N} \times \{n,a,r\}} \mathsf{W}_{jk}\, \mathsf{x}_{jk}}{\sum_{j \in \mathcal{N}} \mathsf{x}_{jg}}$ is the average overhead.
while $\mathcal{U} \neq \emptyset$ **do**
  Randomly pick an agent $(j,k) \in \mathcal{U}$.
  for (i,g) with $q_{igj} = 1$ and $\mathsf{O}_i < (1-\varepsilon)\, \bar{\mathsf{O}}$ **do**
    Update $\mathsf{O}_i \leftarrow \mathsf{O}_i + \mathsf{W}_{jk}$
    Set $\mathsf{u}_{jki} = 1$ and $\mathcal{U} \leftarrow \mathcal{U} \setminus \{(j,k)\}$
    break
  end for
  if the agent $(j,k) \in \mathcal{U}$ **then**
    for (i,g) with $q_{igj} = 1$ and $\mathsf{O}_i + \mathsf{W}_{jk} < (1+\varepsilon)\, \bar{\mathsf{O}}$ **do**
      Update $\mathsf{O}_i \leftarrow \mathsf{O}_i + \mathsf{W}_{jk}$
      Set $\mathsf{u}_{jki} = 1$ and $\mathcal{U} \leftarrow \mathcal{U} \setminus \{(j,k)\}$
      break
    end for
  end if
  if the agent $(j,k) \in \mathcal{U}$ **then**
    break *(infeasibility)*
  end if
end while

Step 5. An agent (j,k) may be assigned to a cluster (i,g) if the gateway can communicate with the agent ($q_{igj} = 1$) and the protocol overhead lies within the balanced range $\big[(1-\varepsilon)\,\bar{\mathsf{O}},\ (1+\varepsilon)\,\bar{\mathsf{O}}\big]$. The assignment process continues until all agents are clustered. The G−CF algorithm operates in polynomial time, with a computational complexity of $\mathcal{O}(|\mathcal{N}|^2)$.

As a result, the subproblems in DeNAD can be efficiently solved using the greedy heuristics G−GD, G−SD, G−NTD, G−MD, G−RD, G−NBD, and G−CF. This approach ensures a scalable and computationally efficient solution for large-scale sensor networks.

V. Computational Results

The simulation parameters are summarized in Table III. We first demonstrate the solutions of CNAD, DeNAD, and Greedy on a demo instance (D) with $|\mathcal{N}| = 15$ candidate points in a 5×4 km^2 region. We then assess solution quality and runtime on three larger regions: small (S, 10×8 km^2), medium (M, 15×14 km^2), and large (L, 20×18 km^2). For each region size, we consider three different values of $|\mathcal{N}|$, as detailed in Table III.

TABLE III
Summary of simulation parameters.

Parameter		Value
$\lvert\mathcal{N}\rvert$	D	15
	S	40, 60, 80
	M	100, 150, 200
	L	200, 250, 300
$\lvert\mathcal{S}\rvert$, $\lvert\mathcal{B}\rvert$, $\lvert\mathcal{M}\rvert$, $\lvert\mathcal{T}\rvert$		3, 2, 2, 3
τ^*, L		30 mins, 2 years (34,560 periods)
λ, δ, ε		1, 5, 10%

We consider $|\mathcal{S}| = 3$ sensor types, $|\mathcal{B}| = 2$ battery models, $|\mathcal{M}| = 2$ memory models, and $|\mathcal{T}| = 3$ transceiver models. The period length is set to $\tau^* = 30$ minutes, and the planning horizon is $L = 2$ years, corresponding to 34,560 periods. In our experiments, each agent is required to communicate with at least $\lambda = 1$ other agent and can reach a gateway within at most $\delta = 5$ hops. Additionally, the overhead deviation among gateways is constrained to a maximum of $\varepsilon = 10\%$. We randomly deploy $|\mathcal{N}|/4$ actuators on the candidate positions. The coverage requirement f_{is} for a point i with respect to a sensor of type s is modeled as Bernoulli random variable with probability 0.5.

Zigbee is a low-power wireless communication protocol tailored for mesh networking, operating primarily in the 2.4 GHz band and conforming to the IEEE 802.15.4 standard. For the experimental setup, the simulation parameters are derived from the Zigbee-compatible network components summarized in Table IV. Based on these components, the simulation parameters are organized as follows: network agents and the protective box in Table V, sensor parameters in Table VI, battery and memory parameters in Table VII, and transceiver specifications in Table VIII. Note that in Table V, the sensor node cost, data packet size and communication range are marked as to be determined (tbd), since they will be optimally specified by the NAD problem.

TABLE IV
Reference network components employed in the experiments.

Component	Model
Actuator	Rain Bird CP075 Irrigation Valve
Router	XBee 3 Zigbee Module
Gateway	Digi XBee Gateway
Protective Box	Hammond 1554F2GYCL IP67
Sensor 1	DFRobot Gravity Analog Soil Moisture Sensor
Sensor 2	SHT31-D Temperature & Humidity Sensor
Sensor 3	Zio Qwiic TSL2561 Light Intensity Sensor
Battery 1	Tadiran TL-5903 3.6V Primary Battery
Battery 2	Panasonic NCR18650B 3.6V Rechargeable Battery
Memory 1	Adesto AT45DB641E Flash, 8 MB
Memory 2	Microchip 25AA1024 EEPROM, 128 KB
Transceiver 1	EFR32MG21 SoC Transceiver
Transceiver 2	nRF52840 SoC Transceiver
Transceiver 3	Silicon Labs MGM210PA

TABLE V
Specifications of network agents and protective box.

k	c_k(\$)	h_k (Kbits/period)	μ_k (Kbits)	ρ_k^c (km)
n	tbd	tbd	0.28	tbd
a	45	0.24	0.28	0.8
r	30	0.24	0.28	1.5
g	150	–	–	2
	c_p(\$)	V (cm^3)	P, D, C	
Box	25	5	3, 3, 2	

All algorithms were implemented in Python and executed on a computer running macOS Sequoia 15.5 with 8 GB of RAM and an Apple M2 processor. The CNAD and DeNAD mathematical models were solved using IBM CPLEX 22.11 with default settings and a 5-minute time limit. The Python source codes are available at our GitHub repository [30].

A. Results for Instance D

We illustrate the solutions to the NAD problem obtained by the CNAD, DeNAD, and Greedy methods for a demonstration instance D with $|\mathcal{N}| = 15$ candidate poses, as shown in Figs. 3, 4, and 5, respectively. Candidate poses are represented by blue dots, actuators by yellow triangles, routers by pink diamonds, and gateways by red squares. Sensor nodes are depicted as green rectangles, which may contain type-1 (blue), type-2 (orange), or type-3 (green) sensors.

The red discs centered at the gateways indicate their communication ranges. In all methods, three gateways are deployed, and their combined coverage spans all candidate poses. Sensor types are distributed across sensor nodes to meet the *coverage* requirements of each pose, with at most $|\mathcal{S}| = 3$ sensors per node. Increasing the coverage requirement f_{is} improves the network's *fault tolerance*. Furthermore, each agent has at least $\lambda = 1$ neighboring agent within its communication range, thereby improving *connectivity* and ensuring *reliable* field monitoring. Larger values of λ further support the design of a *fault-tolerant* wireless sensor network.

Each gateway defines a cluster, with agent-to-gateway communication paths illustrated by blue, green, and orange arcs.

TABLE VI
SPECIFICATIONS OF SENSOR MODELS.

s	c_s ($)	ρ_s (km)	v_s (cm³)	τ_s (sec)	h_s (Kbits/period)	μ_s (Kbits)	e_s $\times 10^{-3}$ (mAh/period)
1	20	0.1	0.10	1800	0.36	0.30	0.280
2	12	0.1	0.10	1800	0.32	0.28	0.078
3	6	0.1	0.92	1800	0.32	0.28	0.060

TABLE VII
SPECIFICATIONS OF BATTERIES AND MEMORY MODELS.

b	c_b ($)	E_b (mAh)	m	c_m ($)	Ω_m (Kbits)
1	6	2400	1	2.5	65,536
2	8	3400	2	1.5	1,024

Every agent is assigned to a unique cluster, ensuring *connectivity*, and communicates with its designated gateway for data transmission and control commands. All agents lie within the communication range of their gateway, enabling direct command reception. Agent-to-gateway paths are constrained to at most $\delta = 5$ hops, thereby reducing the probability of *data loss*, and controlling both *latency* and *energy consumption* in data collection. Additionally, gateway data load is balanced by restricting overhead deviation to no more than $\varepsilon = 10\%$, which promotes *fairness* in network utilization.

Battery and memory allocations are determined under the assumption that all agents are active. The optimal allocation of batteries ensures a minimum *network lifetime* of $L = 2$ years, while memory allocation guarantees sufficient storage capacity, thereby enhancing *reliability*. The lifetime can be further extended by optimizing the active/sleep cycles of sensor nodes in networks designed by the CNAD, DeNAD, and Greedy methods using an additional optimization algorithm.

Analyzing the computational results, CNAD achieves the global optimal solution (Fig.3) with an objective value of $1392 in 0.17 seconds. DeNAD solves each subproblem to optimality and produces a relaxed solution (Fig.4) with an objective value of $1464 in 0.04 seconds. The Greedy method yields locally optimal subproblem solutions and produces the solution in Fig. 5 with an objective value of $1480 in 0.02 seconds. These results highlight a clear trade-off: solution quality decreases as the problem is relaxed and local optima are constructed, whereas computation time improves. This trade-off is further explored in the next section for larger regions, i.e., the S, M, and L instances with varying numbers of poses $|\mathcal{N}|$.

B. Results for Instances S, M, and L

TABLE IX presents a comparison of the average performance of the CNAD, DeNAD, and Greedy methods. The evaluation covers nine instance settings (from S40 to L300), each repeated over 10 random cases, and the reported values represent averages. Out of the 10 trials, "#Solved" indicates the number of cases for which a feasible solution was found within the 5-minute time limit. "#Opt" denotes the number of cases where CNAD obtained the global optimum, while "#Impr" reports the number of cases in which CNAD's solution

TABLE VIII
SPECIFICATIONS OF TRANSCEIVER MODELS.

t	c_t ($)	ρ_t^c (km)	d_t (Kbits/period)	e_a^t $\times 10^{-3}$ (mAh)	e_t^{tx} $\times 10^{-3}$ (mAh/Kbit)	e_t^{rx} $\times 10^{-3}$ (mAh/Kbit)
1	4.5	1.0	250	15.84	100	11
2	2.5	0.8	150	8.82	9	5
3	60.0	2.0	250	6.00	0.111	0.0106

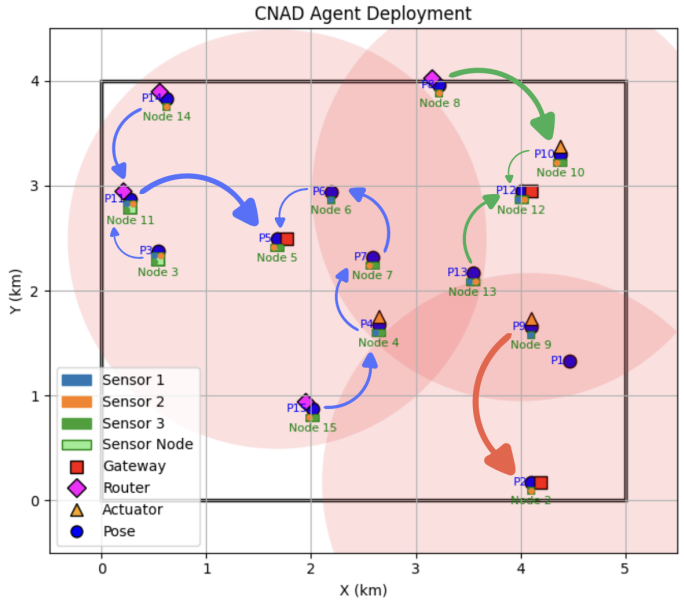

Fig. 3. CNAD deployment and clustering for instance D with $|\mathcal{N}| = 15$, $|\mathcal{S}| = 3$, $|\mathcal{B}| = 2$, $|\mathcal{M}| = 2$, $|\mathcal{T}| = 3$, and $L = 2$ years.

was improved by DeNAD or Greedy. Finally, "CPU (secs)" gives the average execution time in seconds.

As region size and the number of poses increase, CNAD solves fewer instances to optimality. For L300, it finds feasible (not necessarily optimal) solutions in only 4 out of 10 cases within the time limit. In contrast, DeNAD consistently solves all subproblems to optimality, and Greedy always yields feasible solutions, both with significantly lower computation times. For larger instances (M–L), DeNAD improves upon CNAD's best feasible solutions in more cases, while Greedy achieves fewer improvements, as expected.

The *optimality gap* of a method (CNAD, DeNAD, or Greedy) is defined as the percentage difference between the objective value obtained by the method and the best-known lower bound produced by CPLEX when solving CNAD. Fig. 6 compares the optimality gaps of the methods across different instance sizes. CNAD achieves the narrowest gap in the S region, all three methods exhibit comparable performance in the M region, and DeNAD demonstrates a marked improvement over the others in the L region.

According to our results, CNAD yields the global optimal solution but at high computational cost, limiting scalability. DeNAD, a relaxed variant, offers near-optimal local solutions with lower complexity and is thus more efficient. Both methods, being mathematically formulated, are well-suited for

TABLE IX
COMPARISON OF CNAD, DeNAD, AND GREEDY METHODS ACROSS INSTANCES.

	CNAD			DeNAD			Greedy		
Instance	#Solved	#Opt	CPU (secs)	#Solved	#Impr	CPU (secs)	#Solved	#Impr	CPU (secs)
S40	10	9	80.28	10	0	0.19	10	0	0.03
S60	10	10	121.99	10	0	0.24	10	0	0.09
S80	10	5	189.00	10	0	0.31	10	0	0.20
M100	10	2	209.12	10	1	0.37	10	0	0.34
M150	10	0	time	10	2	0.66	10	1	0.68
M200	10	0	time	10	3	1.19	10	3	1.21
L200	10	0	time	10	2	1.34	10	1	1.33
L250	10	0	time	10	7	2.00	10	6	2.22
L300	4	0	time	10	4	2.87	10	4	2.94

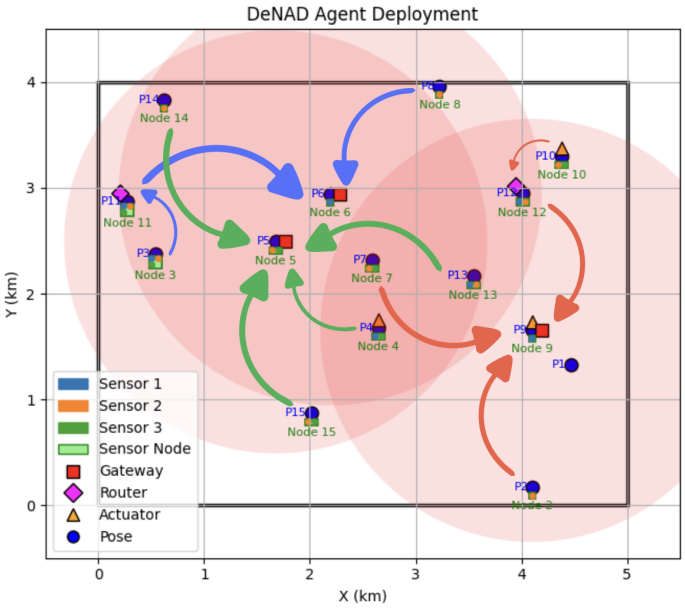

Fig. 4. DeNAD deployment and clustering for instance D with $|\mathcal{N}| = 15$, $|\mathcal{S}| = 3$, $|\mathcal{B}| = 2$, $|\mathcal{M}| = 2$, $|\mathcal{T}| = 3$, and $L = 2$ years.

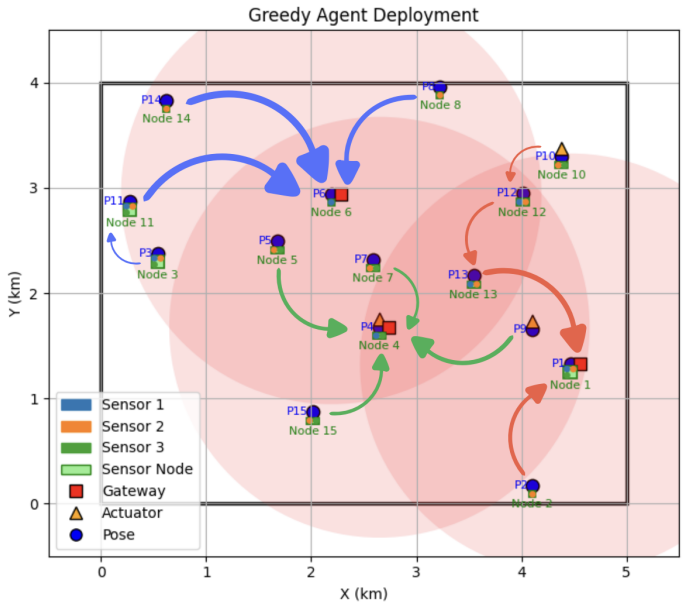

Fig. 5. Greedy deployment and clustering for instance D with $|\mathcal{N}| = 15$, $|\mathcal{S}| = 3$, $|\mathcal{B}| = 2$, $|\mathcal{M}| = 2$, $|\mathcal{T}| = 3$, and $L = 2$ years.

cloud-based quantum computing platforms (e.g., D-Wave) to obtain high-quality solutions in reasonable time. In contrast, the Greedy heuristic is lightweight, *solver-independent*, and *scalable*. It produces feasible local solutions with minimal computational effort and can be implemented on embedded systems, making it especially suitable for commercial applications in remote monitoring.

In our experiments, only actuators are assumed to be deployed. CNAD, DeNAD, and Greedy can still be applied by fixing decision variables when other agents already exist, focusing on deploying additional agents to meet coverage and connectivity requirements with minimal cost. These methods also adapt to regional changes, such as adding or removing candidate poses, by updating the corresponding variables.

VI. CONCLUSIONS

This study addressed the Network Agent Deployment (NAD) problem in heterogeneous WSNs for remote monitoring. The

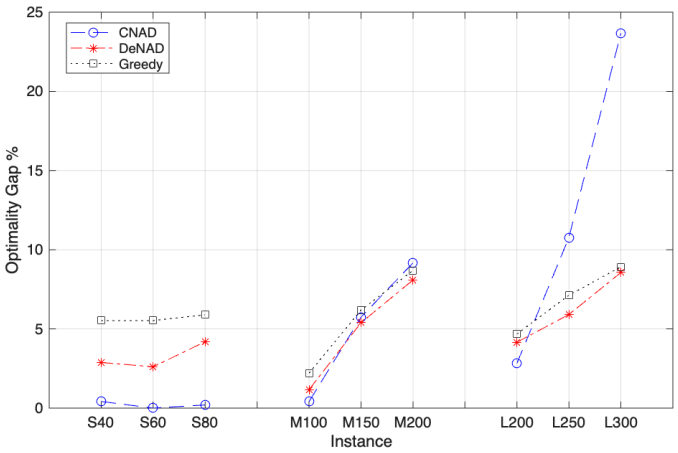

Fig. 6. Optimality gap comparison of CNAD, DeNAD and Greedy methods.

proposed approaches ensure coverage, connectivity, fault tolerance, and network lifetime by optimally assigning sensor types, maintaining agent neighborhoods, limiting agent-to-gateway hops, and balancing gateway loads. Battery and memory allocations guarantee at least L years of operation, with further extensions possible through activity scheduling.

We developed three solution approaches: `CNAD` for global optimality, `DeNAD` for near-optimal solutions with lower computational effort, and a Greedy heuristic for lightweight, scalable deployment suitable for embedded systems. The methods support reliable, energy-efficient, and fault-tolerant network operation, applicable to agricultural monitoring, environmental surveillance, and battlefield scenarios. Future work could explore mobile agents and adaptive scheduling to further reduce energy consumption and deployment costs while enhancing reliability.

REFERENCES

[1] M. A. Jamshed, K. Ali, Q. H. Abbasi, M. A. Imran, and M. Ur-Rehman, "Challenges, applications, and future of wireless sensors in Internet of Things: A review," *IEEE Sensors Journal*, vol. 22, no. 6, pp. 5482–5494, 2022.

[2] M. N. Mowla, N. Mowla, A. F. M. S. Shah, K. M. Rabie, and T. Shongwe, "Internet of Things and wireless sensor networks for smart agriculture applications: A survey," *IEEE Access*, vol. 11, pp. 145 813–145 852, 2023.

[3] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[4] Z. Nurlan, T. Zhukabayeva, M. Othman, A. Adamova, and N. Zhakiyev, "Wireless sensor network as a mesh: Vision and challenges," *IEEE Access*, vol. 10, pp. 46–67, 2022.

[5] M. Chen, S. Gonzalez, V. C. Leung, Q. Zhang, and M. Li, "A survey of research on military wireless sensor networks," *Proceedings of the IEEE*, vol. 102, no. 11, pp. 1749–1776, 2014.

[6] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[7] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[8] Q. Zhao, "Protocol stack architecture for wireless sensor networks," in *Encyclopedia of Wireless Networks*, X. Shen, X. Lin, and K. Zhang, Eds. Springer, 2020, pp. 1117–1120.

[9] A. Odey and D. Li, "Low power transceiver design parameters for wireless sensor networks," *Wireless Sensor Network*, vol. 4, no. 10, pp. 243–249, 2012.

[10] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: a top-down survey," *Computer Networks*, vol. 67, pp. 104–122, 2014.

[11] S. Raj, P. Shynu, and N. Sreeram, "Wireless sensor networks in agriculture: Applications, challenges and opportunities," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 11, pp. 4627–4648, 2020.

[12] X. Li, W. Xu, and S. Yang, "QoS-aware design and optimization of wireless sensor networks for precision agriculture," *Computers and Electronics in Agriculture*, vol. 151, pp. 220–233, 2018.

[13] C. Pralet *et al.*, "Deployment of mobile wireless sensor networks for crisis situations," in *Mobile Agents and Security*. Springer, 2014, pp. 345–358.

[14] E. de Freitas *et al.*, "Mobile agents model and performance analysis of a wireless sensor network application," *SpringerLink*, 2011.

[15] A. Prakash, "Sensor node deployment based on OTLBO in WSN," *Procedia Computer Science*, vol. 52, pp. 1024–1029, 2015.

[16] Y. Zhang, Z. Liu, and Y. Bi, "Node deployment optimization of underwater wireless sensor networks using intelligent optimization algorithm and robot collaboration," *Scientific Reports*, vol. 13, p. 43272, 2023.

[17] J. Roselin, P. Latha, and S. Benitta, "Maximizing the wireless sensor networks lifetime through energy efficient connected coverage," *Ad Hoc Networks*, vol. 62, pp. 1–10, 2017.

[18] Y. Mao, L. Chen, and D. Chen, "A location-unaware connected coverage protocol in wireless sensor networks," in *Ubiquitous Intelligence and Computing*. Springer, 2008, pp. 524–534.

[19] D. A. Amer, S. A. Soliman, A. F. Hassan, and A. A. Zamel, "Enhancing connectivity and coverage in wireless sensor networks: a hybrid comprehensive learning-fick's algorithm with particle swarm optimization for router node placement," *Neural Computing and Applications*, vol. 36, p. 21671–21702, 2024.

[20] H. M. Ammari, "A computational geometry-based approach for planar k-coverage in wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 19, no. 2, Feb. 2023. [Online]. Available: https://doi.org/10.1145/3564272

[21] P.-J. Wan and C.-W. Yi, "Coverage by randomly deployed wireless sensor networks," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2658–2669, 2006.

[22] I. Korpeoglu, "Energy efficient routing," in *Sensor Networks and Configuration*. Springer, 2007, pp. 167–188.

[23] R. Sheeja *et al.*, "Multi-objective-derived energy efficient routing in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 207, p. 103464, 2023.

[24] A. Pathan *et al.*, "A secure energy-efficient routing protocol for WSN," in *Security in Wireless Sensor Networks*. Springer, 2007, pp. 387–402.

[25] N. R. Busireddy *et al.*, "Energy efficient aggregation in wireless sensor networks using artificial bee colony algorithm," in *Swarm Intelligence*. Springer, 2013, pp. 469–476.

[26] M. Migabo *et al.*, "Energy-efficient network coding for multi-hop wireless sensor networks: A survey," *Computer Communications*, vol. 82, pp. 100–113, 2016.

[27] A. Benzerbadj, "Energy-efficient approach for surveillance applications in wireless sensor networks," *Ad Hoc Networks*, vol. 24, pp. 1–15, 2015.

[28] E. Shakshuki *et al.*, "Resource management approach to an efficient wireless sensor network," *Journal of Network and Computer Applications*, vol. 115, pp. 1–16, 2018.

[29] A. Ojha *et al.*, "Evolving landscape of wireless sensor networks: A survey on ai integration for energy efficiency," *Neural Computing and Applications*, vol. 37, pp. 10 500–10 520, 2025.

[30] B. Kabakulak. Wireless sensor network agent deployment. [Online]. Available: https://github.com/banukabakulak/WSN_deployment

Banu Kabakulak received her B.Sc. degrees in Industrial Engineering and Mathematics (double major) from Boğaziçi University, İstanbul, Türkiye, in 2007, followed by M.Sc. and Ph.D. degrees in Industrial Engineering from the same institution in 2010 and 2018, respectively. Since 2019, she has been an Assistant Professor in the Department of Industrial Engineering at İstanbul Bilgi University. She is currently a visiting researcher in the School of Mathematics at the University of Edinburgh, UK. Her research interests span IoT networks, telecommunications, large-scale optimization, mathematical programming, and intelligent multi-agent systems. In 2020, she received the *Operations Engineering & Analytics Best Application Paper Award* from *IISE Transactions.*

E.Alper Yıldırım is a Reader at the School of Mathematics at the University of Edinburgh. He graduated from Bilkent University with a B.S. degree in Industrial Engineering in 1997. He earned his M.S. and Ph.D. degrees in Operations Research at Cornell University (USA) in 2000 and 2001, respectively. Prior to joining the University of Edinburgh, he held faculty positions at Stony Brook University (USA), Bilkent University (Türkiye), and Koç University (Türkiye). His research interests are in mathematical optimization and applications of optimization in different fields. He is particularly interested in convex conic optimization and various convex relaxations of nonconvex optimization problems. He currently serves on the Editorial Boards of Optimization Methods and Software, Optimization Letters, EURO Journal on Computational Optimization, and Journal of Optimization Theory and Applications.