Sensor and Sink Placement, Scheduling and Routing Algorithms for Connected Coverage of Wireless Sensor Networks

Banu Kabakulak*1

1Department of Industrial Engineering, Boğaziçi University, İstanbul, Turkey

A sensor is a small electronic device which has the ability to sense, compute and communicate either with other sensors or directly with a base station (sink). In a wireless sensor network (WSN), the sensors monitor a region and transmit the collected data packets through routes to the sinks. In this study, we propose a mixed–integer linear programming (MILP) model to maximize the number of time periods that a WSN carries out the desired tasks with limited energy and budget. Our sink and sensor placement, scheduling, routing with connected coverage (SPSRC) model is the first in the literature that combines the decisions for the locations of sinks and sensors, activity schedules of the deployed sensors, and data flow routes from each active sensor to its assigned sink for connected coverage of the network over a finite planning horizon. The problem is NP–hard and difficult to solve even for small instances. Assuming that the sink locations are known, we develop heuristics which construct a feasible solution of the problem by gradually satisfying the constraints. Then, we introduce search heuristics to determine the locations of the sinks to maximize the network lifetime. Computational experiments reveal that our heuristic methods can find near optimal solutions in an acceptable amount of time compared to the commercial solver CPLEX 12.7.0.

Keywords: Wireless sensor networks, coverage, α–connectivity, activity scheduling, data routing, heuristics.

*Corresponding author. Tel.: +90 2123596771; fax: +90 2122651800. E-mail addresses: banu.kabakulak@boun.edu.tr (B. Kabakulak).
1 Introduction

Wireless sensor networks (WSNs) are composed of a large number of wireless devices, called sensors, equipped with communication and computing capabilities to monitor a region. A homogeneous WSN consists of identical sensors, whereas the communication and computing capability of the sensors are different in a heterogeneous network. WSNs are applied to various fields of technology thanks to their easy and cheap deployment features. They are used to gather information about human activities in health care, battlefield surveillance in military, monitor wildlife or pollution in environmental sciences, and so on [30].

A sensor can collect data within its sensing range, process data as packets and transmit to a base station (sink) either directly or through other sensors which are within its communication range. Sensors consume energy for sensing, receiving data from other sensors and transmitting data to other sensors or a sink. Energy-aware operating is important for a sensor since it has limited battery energy. A sensor can carry out sensing and communicating tasks when it is active and consumes negligible energy in standby mode [13]. A sensor is no more a member of the WSN, when its battery energy depletes.

The number of time periods that a WSN operates as desired is its lifetime and depends highly on the limited energy of the sensors. Hence, energy-aware usage of the sensors helps to prolong the network lifetime. The key factors that affect the energy consumption can be listed as follows: locations of the sensors and sinks in the network, schedule of the active or standby periods of the sensors, sink assignments of the sensors and data transmission routes from the sensors to their assigned sinks.

Design problems related with WSNs are well studied in the literature. The Coverage Problem (CP) focuses on how well the sensors observe the region they deployed. Each node in the region has a coverage requirement, i.e., least number of sensors that should monitor the node. In order to minimize the energy consumption, CP aims to locate smallest number of sensors in the region to satisfy the coverage requirements of the nodes [28, 33]. The reviews [7, 8] summarize the works on CP. A genetic algorithm is proposed in [21] to cover each node with at least one sensor. The greedy algorithm in [4] provides coverage when the sensing ranges of the sensors are adjustable. In [15], the column generation algorithm locates the sensors while minimizing the sensing energy consumption of the sensors. In [23], mobile robots are used in cooperation with a WSN to monitor a disaster area. In this study, we consider differentiated CP, where each node in the network has different coverage requirement.

The active sensors should be connected to a sink to transmit the sensed data. Coverage and connectivity are important for a WSN, since they are quality of service (QoS) measures. There are different
definitions in the literature for network connectivity. A network can be considered to be connected if the active sensors can communicate at least one of the other active sensors [25]. The authors restore the connectivity among the sensors with a heuristic algorithm in the case of a failure of sensors in [14]. Another definition is the existance of a path from each active sensor to a sink in order to transmit the data packets [2]. Hence, the locations of the sensors are determined to satisfy the coverage requirements of the nodes and provide the connectivity in the Connected Coverage Problem (CCP). In [9], the authors give energy efficient algorithms for CCP. There are numerous works in the literature on CCP such as [1, 31, 34]. In our study, we make a broader definition of connectivity and we assume a network is $\alpha$–connected in a period if each active sensor can communicate with $\alpha$–many active sensors and there is a data flow path to its assigned sink. Hence, each active sensor can communicate with its assigned sink either directly or via other active sensors by multi-hop transmission. Moreover, one can find alternative paths from the sensors to the sinks in the case of unexpected failure of a sensor in the network, since there are at least $(\alpha - 1)$ active sensors in the communication range of each sensor.

One can use the limited energy of the sensors more economically by keeping unnecessary sensors in standby mode in a period. That is, the Activity Scheduling Problem (ASP) determines the active and standby periods of the sensors in order to extend the network lifetime [17, 20]. The authors schedule the sensors in homogeneous and heterogeneous WSNs in [27]. Heuristic algorithms are developed in [16] to deploy sensors for CP and schedule their activities to maximize the lifetime. In some WSNs, such as cognitive WSNs, the sensed data packets can be processed by only some of the sink nodes. The Sink Assignment Problem (SAP) assigns the active sensors to an appropriate sink. SAP allows to control the data traffic load on the sinks. This helps to avoid congestion, i.e., traffic load exceeds the available capacity, on the sinks as noted in [10]. The authors formulate the SAP for cognitive WSNs as MILP in [26] and develop a linear relaxation technique for its solution.

Sensors also consume energy while transmitting the sensed data to the sinks. Hence, the Data Routing Problem (DRP) finds the minimum energy consuming paths from the active sensors to the sinks [32]. A column generation algorithm is developed in [22] for CCP, ASP, and DRP. An integer programming formulation for DRP, when the sensor locations are known, is given in [20]. The authors introduce centralized heuristics to minimize the energy consumption. In the literature, it is sufficient to find a data flow path from a sensor to “a sink.” On the other hand, we extend DRP in this study to design more secure networks. That is, we make sink assignment for each active sensor and construct a data flow path from a sensor to “its assigned sink.” Such a network design securely collects data, since only the desired sink can receive the data packets. The locations of the sinks also affect the energy consumption
in data routing. The Sink Location Problem (SLP) investigates the optimum locations of the sinks that require the least transmission energy. In [18, 24], mobile sinks are considered to collect data from the sensors. The paths of the mobile sensors are also addressed in [12]. The authors develop heuristics to find the locations of the sinks at each period. In our study, we assume that sink nodes are immobile.

In this study, we consider heterogeneous WSNs. We consider to deploy sensor in the network to satisfy differentiated coverage requirements of the nodes and α–connectivity of the sensors in CCP. We schedule the active and standby periods of the sensors for energy–aware design of the network in ASP. We assign a sink for each active sensor in a period in SAP. We detect minimum energy consuming paths from the active sensors to their assigned sinks in DRP. We find the best locations of the sinks to maximize the network lifetime in SLP. We name the network design problem that combines CCP, ASP, SAP, and DRP as the Sensor Placement, Scheduling and Routing with Connectivity (PSRC) problem. Additionally, we have the SPSRC problem, which deals with SLP and PSRC simultaneously. SPSRC is difficult to solve exactly for large networks since it includes the set covering problem, which is known to be NP–complete [5], as a subproblem.

We can list our contribution to the literature as follows:

- We extend the definitions of CCP and DRP by introducing α–connectivity and SAP.
- This paper makes original contributions to the literature by dealing with SPSRC model which integrates SLP, CCP, ASP, SAP, and DRP. To the best of our knowledge, the design issues mentioned are not undertaken within a single model in the literature before.
- We reduce SPSRC model to PSRC by assuming that the sink locations are known. We propose Constructive Heuristic (CH) and Disjunctive Heuristic (DH) methods for the solution of PSRC.
- We generate feasible solutions of SPSRC with our Local Search (LS) and Tabu Search (TS) algorithms, which determine the sink locations.
- Our solution methods can provide near optimal feasible solutions of SPSRC in an acceptable amount of time. Our algorithms outperform CPLEX 12.7.0 in terms of solution quality and computation time.

In the remainder of the paper, we formally define our problem in Section 2 and introduce our SPSRC formulation in Section 3. We give the details of our heuristic methods for the solution of SPSRC in Section 4. We test the efficacy of our methods via computational experiments in Section 5. Some concluding remarks and comments on future work appear in Section 6.

4
2 Problem Definition

In this paper, we consider a heterogeneous WSN with \( K \) sensor types given with set \( K = \{1, ..., K\} \). We assume sink is type–0 and set \( \bar{K} = K \cup \{0\} \) represents the sensor and sink types. WSN monitors a region defined by a set of nodes, i.e., \( N = \{1, ..., N\} \). We deploy and activate sensors at each time period \( t \in T = \{1, ..., T\} \) to cover each node \( i \in N \) with at least \( f_i \)–many sensors. The cost of placing a sensor \((j, k)\), i.e., type-\( k \) sensor at location \( j \in N \), is given as \( c_{jk} \). We have a total budget of \( B \) monetary units.

An active sensor \((j, k)\) can cover nodes within its sensing \((r_s^k)\) range and generates \( h_{jk} \)–many data packets in a period. The coverage coefficient \( a_{jki} \) is one if sensor \((j, k)\) can cover node \( i \), and zero otherwise. A type-\( k \) sensor can receive and transmit data packets within its communication \((r_c^k)\) range. The communication coefficient \( b_{jki} \) is one if sensor \((j, k)\) can communicate with a sensor or a sink at node \( i \), and zero otherwise. In order to have a robust communication, we require that an active sensor communicates with at least \( \alpha \)–many active sensors in a period. A type-\( k \) sensor has \( E_k \) units of energy initially. It consumes \( e_s^k \) units for sensing the network, \( e_r^k \) units for receiving and \( e_c^k \) units for transmitting data packets in its active periods.

At each period \( t \) within network lifetime \( L \), we aim to cover each node \( i \) and route the data packets collected by the active sensors to their assigned sinks. We consider \( T \) as the planning horizon, which is an upper bound on the network lifetime \( L \). We assume that a sink receives data packets but cannot transmit them to other sensors or sinks. That is when there is a sink \((j, 0)\), we have \( b_{j0i} \) equal to zero for all \( i \in N \). Furthermore, sinks cannot contribute to the coverage of nodes. Hence, for a sink \((j, 0)\), we have \( a_{j0i} \) equal to zero for all \( i \in N \).

We illustrate a feasible solution of the \( SPSRC \) problem with planning horizon \( T = 2 \) in Figure 1. In this example, we have a 4 \times 4 grid network, i.e., \( N = 16 \) candidate nodes to deploy sinks and sensors. We have two different types of sensors (\( K = 2 \)) and two sinks to collect the data. Coverage requirement is one \((f_i = 1)\) for each node \( i \in N \). The sensing range of type–1 sensors is one, and half of the sensing range of type–2 sensors, i.e., \( 2r_1^s = r_2^s = 2 \). The communication range of type–1 sensors is two, and half of the communication range of type–2 sensors, i.e., \( 2r_1^c = r_2^c = 4 \). Each active sensor communicates with one other active sensor \((\alpha = 1)\) and generates \( h_{jk} = 24 \) data packets in a period.

In the feasible solution given in Figure 1 the decisions for the network are as follows:

- (SLP) Two sinks are located at nodes 8 and 14 (rectangles), i.e., \((8, 0)\) and \((14, 0)\). They are active throughout the planning horizon \( T \).
Figure 1: A sample sensor network operating for $T = 2$

- (CCP) We deploy four type–1 sensors (circles), i.e., (1, 1), (2, 1), (10, 1) and (16, 1). There are four type–2 sensors (diamonds), which are (3, 2), (4, 2), (11, 2) and (15, 2).

- (ASP) In period $t = 1$, the sensors (1, 1), (10, 1), (3, 2) and (15, 2) are active (black). The sensors (2, 1), (16, 1), (4, 2) and (11, 2) are in standby mode (white) at $t = 1$. In period $t = 2$, the sensors (2, 1), (10, 1), (16, 1), (4, 2), and (11, 2) are active, whereas (1, 1), (3, 2) and (15, 2) are in standby mode.

One can observe that each node is within the sensing range of at least one active sensor ($f_i = 1$). Furthermore, there is at least one active sensor ($\alpha = 1$) in the communication range of each active sensor at each period.

- (SAP) The active sensors (1, 1) and (3, 2) are assigned to the sink (8, 0) at $t = 1$. Besides, (10, 1) and (15, 2) are assigned to the sink (14, 0) in the same period. At $t = 2$, the active sensors (2, 1) and (4, 2) transmit to the sink (8, 0), and the sink (14, 0) collects data from the sensors (10, 1), (16, 1) and (11, 2).

- (DRP) In period $t = 1$, the sensor (1, 1) transmits 24 data packets to its sink (8, 0) at 2–hops. The other active sensors can send data to their sinks directly. In period $t = 2$, the sensor (2, 1) connects with its sink (8, 0) via sensor (4, 2), whereas the other sensors can communicate with
their sinks at 1-hop.

The solution of DRP provides the paths that connect the sensors to their assigned sinks.

3 Mathematical Formulations

In this section we introduce a MILP model, i.e, \( SPSRC \), that locates sinks and sensors, determine activity schedules of the sensors, assigns a sink for each active sensor and determines the sensor-to-sink data flow routes while maximizing the network lifetime. In the case of known sink locations, \( SPSRC \) reduces to the sensor placement, scheduling and routing with connected coverage, i.e., \( PSRC \), model.

We have eight decision variables in our mathematical model. The continuous variable \( L \) is the network lifetime that we aim to maximize. The binary decision variable \( n_t \) is one if a period \( t \) is within the lifetime \( L \), and zero otherwise. The variables \( x_{jk} \), \( z_{jkt} \) and \( u_{ijkt} \) are binary. The variable \( x_{jk} \) is one if there is a type-\( k \) sensor at location \( j \), \( z_{jkt} \) is one if the deployed sensor \((j,k)\) is active in period \( t \), and \( u_{ijkt} \) is one if the active sensor \((j,k)\) is assigned to the sink at location \( i \) in period \( t \). The amount of flow from sensor \((i,l)\) to sensor \((j,k)\) in period \( t \) is given with the continuous variable \( y_{iljkt} \). The continuous variable \( g_{iljt} \) represents the incoming flow to a sink \((j,0)\) from sensor \((i,l)\) in period \( t \). The binary variable \( w_{iljkt} \) is one if there is a positive flow from sensor \((i,l)\) to sensor/sink \((j,k)\) in period \( t \), and zero otherwise.

We assume that a sink has sufficiently large battery energy. Hence, a sink is active during the network lifetime. We summarize the terminology used in this paper in Table 1.

One can observe that setting all decision variables to zero gives a trivial solution with lifetime \( L = 0 \). Constraint [1] eliminates this solution from the solution space. We have \( n_t = 0 \) when \( t > L \), which implies \( L \leq t - 1 \) in constraints [2].

\[
1 \leq L \leq T \tag{1}
\]

\[
Tn_t \geq L + 1 - t, \quad t \in T \tag{2}
\]

At each period \( t \) within lifetime \( L \), we should cover a point \( i \in N \) with at least \( f_i \)-many active sensors as given in constraints [3].

\[
\sum_{(j,k) \in N \times K} a_{ijk}z_{jkt} \geq f_in_t, \quad i \in N, \ t \in T \tag{3}
\]

Constraints [4] force to deploy a sensor before activating it and constraints [5] provide not to activate a sensor for the periods out of the lifetime \( L \).
Table 1: List of symbols

| Index sets                          | Parameters                      | Decision Variables      |
|-------------------------------------|---------------------------------|-------------------------|
| \(N\) set of sensor and sink locations | \(a_{ijk}\) 1 if node \(i\) is within the coverage range of sensor \((j, k)\), 0 otherwise | \(L\) lifetime of the WSN |
| \(K\) set of sensor types           | \(b_{ilj}\) 1 if a sensor located at node \(j\) is within the communication range of sensor \((i, l)\), 0 otherwise | \(n_t\) 1 if period \(t\) is within the lifetime \(L\), 0 otherwise |
| \(\tilde{K}\) set of sensor and sink types | \(c_{jk}\) cost of placing a type-\(k\) sensor at node \(j\) | \(x_{jk}\) 1 if a type-\(k\) sensor is placed at node \(j\), 0 otherwise |
| \(T\) set of periods                | \(\epsilon_{jk}^c\) energy consumption of a type-\(k\) sensor for transmitting one unit of flow | \(z_{jkt}\) 1 if a sensor \((j, k)\) is active in period \(t\), 0 otherwise |
|                                     | \(\epsilon_{jk}^e\) energy consumption of a type-\(k\) sensor for receiving one unit of flow | \(u_{ijkl}\) 1 if a sensor \((j, k)\) is assigned to a sink located at node \(i\) in period \(t\), 0 otherwise |
|                                     | \(\epsilon_{jk}^s\) energy consumption of a type-\(k\) sensor for sensing and processing during a period | \(y_{ijkl}\) amount of data flow from sensor \((i, l)\) to sensor \((j, k)\) in period \(t\) |
|                                     | \(E_k\) initial battery energy of a type-\(k\) sensor | \(g_{iljt}\) amount of data flow from sensor \((i, l)\) to sink \((j, 0)\) in period \(t\) |
|                                     | \(\alpha\) minimum number of sensors that a sensor can communicate | \(w_{ijkl}\) 1 if data flows from sensor \((i, l)\) to sensor \((j, k)\) in period \(t\), 0 otherwise |

In order to have a \(\alpha\)–connected network, as given in constraints (6), there should be at least \(\alpha\)–many active sensors that an active sensor \((i, l)\) can communicate at a period \(t\).

\[
\sum_{(j,k) \in \mathbb{N} \times \mathbb{K}, (j,k) \neq (i,l)} b_{ilj} z_{jkt} \geq \alpha z_{ilt}, \quad (i, l) \in \mathbb{N} \times \mathbb{K}, t \in \mathbb{T}
\]  

We make the unique sink assignments of the active sensors in a period with constraints (7)–(9). Constraints (7) require a sink \((i, 0)\) is deployed and constraints (8) guarantee that the sensor \((j, k)\) is
active for assigning the sensor \((j,k)\) to the sink \((i,0)\). Moreover, constraints \((9)\) assign one and only one sink for each active sensor \((j,k)\) in a period \(t\).

\[
u_{ijkt} \leq x_{i0}, \quad i \in \mathbb{N}, (j,k) \in \mathbb{N} \times \mathbb{K}, \quad t \in \mathbb{T} \quad (7)
\]

\[
u_{ijkt} \leq z_{jkt}, \quad i \in \mathbb{N}, (j,k) \in \mathbb{N} \times \mathbb{K}, \quad t \in \mathbb{T} \quad (8)
\]

\[
\sum_{i \in \mathbb{N}} u_{ijkt} = z_{jkt}, \quad (j,k) \in \mathbb{N} \times \mathbb{K}, \quad t \in \mathbb{T} \quad (9)
\]

We avoid flows from a sensor \((i,l)\) to itself with constraints \((10)\). We assume that there can be only one active sensor of type-\(k\) at a node in a period, since most of the time an energy efficient optimal solution has such an activity schedule. Hence, the maximum number of sensors that can be active in a period is given by \(NK\). Then, an active sensor \((j,k)\) can send the total flow of \((NK-1)\) sensors to a sensor \((i,l)\). From here, we can bound the total outflow from an active sensor \((j,k)\) by \(M_1 = \max_{(j,k) \in \mathbb{N} \times \mathbb{K}} \{h_{jk}\}(NK-1)\). A sensor \((i,l)\) can send flow to the sensors that are within its communication range as given in constraints \((11)\). Constraints \((12)\) guarantee that there is no outflow from a sensor \((j,k)\) in standby mode. Similarly, total inflow to an active sensor \((j,k)\) can be at most \(M_1\) as given in constraints \((13)\).

\[
y_{ililt} = 0, \quad (i,l) \in \mathbb{N} \times \mathbb{K}, \quad t \in \mathbb{T} \quad (10)
\]

\[
y_{iljkt} \leq M_1 b_{ilj}, \quad (i,l), (j,k) \in \mathbb{N} \times \mathbb{K}, \quad t \in \mathbb{T} \quad (11)
\]

\[
\sum_{(i,l) \in \mathbb{N} \times \mathbb{K}} y_{jkilt} \leq M_1 z_{jkt}, \quad (j,k) \in \mathbb{N} \times \mathbb{K}, \quad t \in \mathbb{T} \quad (12)
\]

\[
\sum_{(i,l) \in \mathbb{N} \times \mathbb{K}} y_{ijklt} \leq M_1 z_{jkt}, \quad (j,k) \in \mathbb{N} \times \mathbb{K}, \quad t \in \mathbb{T} \quad (13)
\]

A sink \((j,0)\) can collect the data packets of all sensors. A sink \((j,0)\) can receive flows from the sensors that can communicate with the sink as enforced by constraints \((14)\). Constraints \((15)\) bound the total inflow to a sink \((j,0)\) by \(M_2 = \max_{(j,k) \in \mathbb{N} \times \mathbb{K}} \{h_{jk}\}NK\).

\[
g_{iltjt} \leq M_2 b_{iltj}, \quad (i,l) \in \mathbb{N} \times \mathbb{K}, \quad j \in \mathbb{N}, \quad t \in \mathbb{T} \quad (14)
\]

\[
\sum_{(i,l) \in \mathbb{N} \times \mathbb{K}} g_{iltjt} \leq M_2 x_{j0}, \quad j \in \mathbb{N}, \quad t \in \mathbb{T} \quad (15)
\]

If a sensor \((j,k)\) is active in period \(t\), then it generates \(h_{jk}\) units of flow, i.e., data packets, to the incoming data flow from the active sensors and sends the total flow to other active sensors or a sink as outflow. Constraints \((16)\) ensure the data flow balance for each active sensor \((j,k)\) in each period \(t\).
\[
\sum_{(i,l) \in N \times K} y_{iljkt} + h_{jk} z_{jk} = \sum_{(i,l) \in N \times K} y_{ijkl} + \sum_{i \in N} g_{ijkl}, \quad (j,k) \in N \times K, \quad t \in T
\]  

(16)

We assign a sink for each active sensor \((j,k)\), which initiates \(h_{jk}\) units of data flow. We expect that the data flow of each active sensor reaches to its assigned sink. Constraints (17) determine the total inflow to a sink \((i,0)\) as the sum of data flows of the assigned sensors at period \(t\).

\[
\sum_{(j,k) \in N \times K} g_{ijkl} = \sum_{(j,k) \in N \times K} h_{jk} u_{ijkl}, \quad i \in N, \quad t \in T
\]  

(17)

An active sensor can send its flow to its assigned sink either directly or through the other active sensors. Then, the sink assignment of a sensor \((i,l)\) can be one of the sink assignments of the sensors that it sends flow. In constraints (18), the sink assignment variables \(u_{ijkl}\) of the sensor \((i,l)\) is bounded with the ones of the sensor \((j,k)\), i.e., \(u_{ijkl}\), if there is a positive flow from the sensor \((i,l)\) to the sensor \((j,k)\). We assume that a sink \((j,0)\) is assigned to itself at all periods. In constraints (19), \(1_j(v)\) is the indicator function, which takes value one if \(v = j\), and zero otherwise. Constraints (19) guarantee that the sink assignment of a sensor \((i,l)\) is the sink \((j,0)\), if there is a positive flow among them.

\[
\begin{align*}
\sum_{t \in T} \left( e_{k} z_{jk} + e_{k} \sum_{(i,l) \in N \times K} y_{iljkt} + e_{k} \sum_{(i,l) \in N \times K} y_{ijkl} + e_{k} \sum_{i \in N} g_{ijkl} \right) & \leq E_k, \quad (j,k) \in N \times K \\
\end{align*}
\]  

(20)

Constraints (21) force that the total deployment cost of sensors and sinks does not exceed the total available budget.

\[
\sum_{(j,k) \in N \times K} c_{jk} x_{jk} \leq B
\]  

(21)

Constraints (22) limit the number of sinks in the network with \(S\).
\[ \sum_{i \in \mathbb{N}} x_{i0} = S \]  \hspace{1cm} (22)

Finally, constraints \((23) - (28)\) are the binary and nonnegativity restrictions on the decision variables.

\[ n_t \in \{0,1\}, \quad t \in T \]  \hspace{1cm} (23)

\[ x_{jk} \in \{0,1\}, \quad (j,k) \in \mathbb{N} \times \bar{K} \]  \hspace{1cm} (24)

\[ z_{jkt} \in \{0,1\}, \quad (j,k) \in \mathbb{N} \times K, t \in T \]  \hspace{1cm} (25)

\[ u_{ijkt} \in \{0,1\}, \quad i \in \mathbb{N}, (j,k) \in \mathbb{N} \times K, t \in T \]  \hspace{1cm} (26)

\[ y_{iljkt} \geq 0, \quad (i,l), (j,k) \in \mathbb{N} \times K, t \in T \]  \hspace{1cm} (27)

\[ g_{iljt} \geq 0, \quad (i,l) \in \mathbb{N} \times K, j \in \mathbb{N}, t \in T \]  \hspace{1cm} (28)

We formulate the sink and sensor placement, scheduling, routing while providing connected coverage \((SPSRC)\) as mixed–integer linear programming problem subject to the constraints \((1) – (28)\). We aim to maximize the network lifetime \((29)\) under limited energy and budget resources.

\[ \max L \]  \hspace{1cm} (29)

One can observe that constraints \((18) \text{ and } (19)\) are not linear. We introduce binary variables \(w_{iljkt}\), which is one if there is a positive flow from a sensor \((i,l)\) to a sensor/sink \((j,k)\) in period \(t\), and zero otherwise. Then, we linearize constraints \((18) \text{ and } (19)\) as follows:

\[ y_{iljkt} \leq M_1 w_{iljkt}, \quad (i,l), (j,k) \in \mathbb{N} \times K, t \in T \]  \hspace{1cm} (30)

\[ g_{iljt} \leq M_2 w_{iljot}, \quad (i,l) \in \mathbb{N} \times K, j \in \mathbb{N}, t \in T \]  \hspace{1cm} (31)

\[ u_{vilt} - u_{vikt} \leq 1 - w_{iljkt}, \quad v \in \mathbb{N}, (i,l), (j,k) \in \mathbb{N} \times K, t \in T \]  \hspace{1cm} (32)

\[ u_{vilt} - 1 - v_{j(0)x_{j0}} \leq 1 - w_{iljot}, \quad v,j \in \mathbb{N}, (i,l) \in \mathbb{N} \times K, t \in T \]  \hspace{1cm} (33)

We have battery energy limitations of the sensors as constraints \((20)\). This means, there can be alternative optimum solutions of \(SPSRC\) model which consume less energy if we can avoid unnecessary flow loops. In constraints \((6)\), we know that an active sensor can communicate with at least \(\alpha\)–many active sensors. From here, we can save energy in the network if we limit the number of outflow paths from a sensor \((i,l)\) by \(\alpha\) as in constraints \((34)\).
\[
\sum_{(j,k) \in N \times K} w_{ilkt} \leq \alpha, (i, l) \in N \times K, t \in T
\] (34)

SPSRC combines multiple decisions related with a network in one model. SPSRC is an NP–complete problem [5, 19]. Hence, we propose heuristic approaches to generate a near optimal solution of SPSRC in acceptable amount of time. We first consider that we are given the locations of \( S \) sinks, i.e., \( x_{i0} \) values are known. We name this problem as PSRC. We generate feasible solutions via our constructive heuristic (CH) and disjunctive heuristic (DH) approaches for PSRC. Then, we find feasible solutions of SPSRC with our local search (LS) and tabu search (TS) methods, which determine the locations of the sinks. We explain the details of these algorithms in the next section.

4 Solution Methods

We introduce CH (see Section 4.1.1) and DH (see Section 4.1.2) algorithms to generate feasible solutions of PSRC. We can construct a feasible solution of PSRC with CH or DH either from scratch (all-0 solution), or improve a given feasible solution, or repair an infeasible solution. These heuristics are embedded into our sink location search algorithms, i.e., LS (see Section 4.2.1) and TS (see Section 4.2.2), explained in Section 4.2 to maximize the network lifetime \( L \).

4.1 Feasible Solution Generation Algorithms for PSRC

4.1.1 Constructive Heuristic

In Algorithm 1, we explain the main steps of CH. In the first step of CH, we obtain a feasible solution of PSRC with respect to the coverage (3) and budget (21) constraints with Algorithm 2. Then, we satisfy the connectivity (6) and sink assignment (7) - (9) constraints by assigning sinks to the active sensors using Algorithm 3. We determine the amount of flows \( y \) and \( g \) from the active sensors to their assigned sinks by solving Routing Problem \( (RP(t)) \) at each period \( t \).
Algorithm 1. (Constructive Heuristic)

**Input:** An instance of PSRC, a vector \((L, n, x, u, y, g, w)\)

1. Set \(L = T\).
2. Apply Algorithm 2 to obtain a feasible solution subject to coverage (3) and budget (21) constraints.
3. If \(L = 0\), then Stop.
4. For Each \(t \leq L\), apply Algorithm 3 to generate a feasible solution subject to connectivity (6) and sink assignment (7) - (9) constraints in period \(t\).
5. End For Each
6. If \(L = 0\), then Stop.
7. Else For Each \(t \leq L\), solve \(RP(t)\) to determine the data flows \(y\) and \(g\).
8. End If

**Output:** A feasible solution of PSRC with lifetime \(L\).

In order to satisfy the constraints (3) and (21), we implement Algorithm 2. We estimate the maximum possible outflow from an active sensor \((j, k)\) in period \(t\) as

\[
\xi_t = \left( \sum_{(j, k) \in N \times K} z_{jkt} \right) \left( \max_{(j, k) \in N \times K} h_{jk} \right), \quad t \in T.
\] (35)

At the beginning of each period \(t\), we check whether an active sensor \((j, k)\) \((z_{jkt} = 1)\) has sufficient battery energy for sensing and communicating tasks or not. At period \(t = 1\), we have \(E_{jak}^{rem} = E_k\). A sensor \((j, k)\) can be active in period \(t\) if the remaining battery energy satisfies \(E_{jak}^{rem} \geq (e_s^k + \xi_t(e_r^k + e_c^k))\), we set \(z_{jkt} = 0\) otherwise (Step 3). We update \(E_{jak}^{rem}\) for each period \(t\) using Equation (36).

\[
E_{jak,t+1}^{rem} = E_{jak}^{rem} - z_{jkt}(e_s^k + \xi_t(e_r^k + e_c^k)), \quad (j, k) \in N \times K.
\] (36)

We check whether each node \(i\) in the region is covered by \(f_i\)–many active sensors or not in each period \(t\). If each node is covered within the budget in a period \(t\), then we move to the next period. If budget constraint is not held, we consider to remove some of the deployed sensors without harming the coverage constraints at Step 7. For this purpose, we delete sensors, i.e., set \(x_{jk} = 0\), that are in standby mode until the current period since they do not contribute to the coverage of the nodes in any of the periods. Deleting sensor \((j, k)\) improves remaining budget by \(c_{jk}\) monetary units. We continue with the process while the remaining budget is negative or we cannot find a sensor to delete.

After this procedure, it is possible that we could not provide budget feasibility. In this case, one can consider to delete active sensors whose removal will not harm the coverage of the nodes. This strategy is used only when we are in the first period, i.e., \(t = 1\), for the sake of simplicity of the heuristic. If budget still cannot be provided then we set \(L = t - 1\) and stop the algorithm (Step 8). On the other
hand, if there is an undercovered node in the region, then we first try to ensure coverage in the network by activating the existing sensors. Observe that activating a standby sensor does not demand budget usage.

Algorithm 2. (Satisfying Coverage (3) and Budget (21) Constraints)

**Input:** An instance of PSRC, a partial (in)feasible solution \((L, n, x, z)\)

0. Set \(t = 1\).
1. For Each \(t \leq L\)
2. For Each active sensor \((j, k)\) in period \(t\) \((z_{jkt} = 1)\), calculate \(E_{jkt}^{rem}\)
3. If \(E_{jkt}^{rem}\) is not sufficient, Then \(z_{jkt} = 0\).
4. End For Each
5. If every node is covered, Then
6. If budget is satisfied, Then \(t \leftarrow t + 1\).
7. Else order standby sensors in \([0, t]\) in nonincreasing cost and remove them until budget is satisfied.
8. If budget is violated, Then \(nt = 0, L = t - 1\) and Stop.
9. Else While there is \(CEP_{jk} > 0\) with sufficient \(E_{jkt}^{rem}\) for some sensor \((j, k)\) activate \((z_{jkt} = 1)\) the standby sensor \((j, k)\) with the highest \(CEP_{jk}\).
10. End While
11. While there is a sensor \((j, k)\) with the highest \(CCR_{jk} > 0\)
12. If budget is not enough for the sensor \((j, k)\), Then order standby sensors in \([0, t]\) in nonincreasing cost and remove them until enough budget is obtained.
13. If budget is enough,
14. Then deploy \((x_{jk} = 1)\) and activate \((z_{jkt} = 1)\) the sensor \((j, k)\) with the highest \(CCR_{jk}\).
15. Else \(nt = 0, L = t - 1\) and Stop.
16. End If
17. End While
18. End If
19. End For Each

**Output:** A solution satisfying the constraints (3) and (21) with lifetime \(L\).

We choose the standby sensor to activate in a greedy way by calculating the coverage energy product, i.e., \(CEP_{jk}\), for each sensor \((j, k)\) in a period \(t\). Let \((n, x, z)\) be the possibly infeasible partial initial solution given as input to Algorithm 2. In a period \(t\), we calculate the amount of violation in constraints (3) for each node \(i\) with the undercoverage \(U_i\) values as given in Equation (37).

\[
U_i = \max \left\{ \left( f_{ni} - \sum_{(j, k) \in N \times K} a_{ijk} z_{jkt} \right) , 0 \right\} , \ i \in N
\]  
(37)

For a standby sensor, it will be a reason of choice if it can cover as many nodes as possible that have positive \(U_i\). In addition, as the sensor has more remaining energy in its battery, the need for the deployment of new sensors will be less in the future, since we can activate the sensor in these periods also. Therefore, we can define \(CEP_{jk}\) for a standby sensor \((j, k)\) in a period \(t\) as
\[
CEP_{jk} = \left( \sum_{i \in N : U_i > 0} a_{ijk} \right) \frac{x_{jk} \operatorname{rem}_{jkt}}{c_{jk}}, \quad (j,k) \in N \times K
\]

(38)

where \( \operatorname{rem}_{jkt} \) represents the remaining energy of a sensor \((j,k)\) in period \(t\).

Then, we activate the standby sensors starting from the one with the highest positive \(CEP_{jk}\) value and continue until we provide coverage constraints or we do not have standby sensors with positive \(CEP_{jk}\) value (Step 9). If we could not satisfy constraints (3), then we calculate a coverage cost ratio, i.e., \(\text{CCR}_{jk}\), for each sensor \((j,k)\) as

\[
\text{CCR}_{jk} = \frac{\left( \sum_{i \in N : U_i > 0} a_{ijk} \right) (1 - x_{jk}) E_k}{c_{jk}}, \quad (j,k) \in N \times K
\]

(39)

where \(E_k\) represents the initial battery energy of a type-\(k\) sensor. We deploy \((x_{jk} = 1)\) and activate \((z_{jkt} = 1)\) a new sensor \((j,k)\) in period \(t\) with the highest \(\text{CCR}_{jk}\) value (Step 14). In the case remaining budget is not enough to deploy the selected sensor, we try to generate sufficient budget by deleting standby sensors that are not used until the current period (Step 12). If coverage is not provided within the budget, then we set \(L = t - 1\) and stop the algorithm. The computational complexity of Algorithm 2 is given as \(O(\tau_1 N^3 K T + \tau_1 N^2 K T^2)\) where \(\tau_1 = \max_{i \in N} \left\{ \sum_{(j,k) \in N \times K} a_{ijk} \right\} \) is the maximum number of sensors that can cover a node.
Algorithm 3. (Satisfying Connectivity (6) and Sink Assignment (7) - (9) Constraints)

Input: An instance of PSRC, period $t$, a partial (in)feasible solution $(L, n_t, x_t, z_t, u_t)$

0. Set $u_t = 0$

Let $L$ be a list of sensors and sinks. Add $S$—many sinks to $L$ and label them.

1. While $L$ is not empty

2. If the first element of $L$ is a sink (say $(j,0)$), Then

   add unlabeled active sensors $(i,l)$ that communicate with the sink $(j,0)$, i.e., $b_{ij} = 1$, to list $L$. Set $u_{jilt} = 1$.

   Remove sink $(j,0)$ from list $L$.

3. Else /* the first element of $L$ is a sensor (say $(j,k)$) */

   add unlabeled active sensors $(i,l)$ that communicate with the active sensor $(j,k)$, i.e., $b_{ilj} = 1$, to list $L$. Set $u_{vilt} \rightarrow u_{vjlkt}$ for all $v \in N$.

   Remove sensor $(j,k)$ from list $L$.

4. End If

5. End While

6. If connectivity is not satisfied, Then

7. While there is $CoEP_{jk} > 0$ with sufficient $E_{rem}^{jk}$ for some sensor $(j,k)$

   activate ($z_{jkt} = 1$) the standby sensor $(j,k)$ with the highest $CoEP_{jk}$.

8. End While

9. While there is a sensor $(j,k)$ with the highest $CoCR_{jk} > 0$

10. If budget is not enough for the sensor $(j,k)$, Then order standby sensors in $[0,t]$ in nonincreasing cost and remove them until enough budget is obtained.

11. If budget is enough,

12. Then deploy ($x_{jk} = 1$) and activate ($z_{jkt} = 1$) the sensor $(j,k)$ with the highest $CoCR_{jk}$.

   Pick a sensor $(i,l)$ with $b_{jki} = 1$, and set $u_{vjlkt} \leftarrow u_{vilt}$ for all $v \in N$.

13. Else $n_t = 0$, $L = t - 1$ and Stop.

14. End If

15. End While

16. End If

Output: A solution satisfying the constraints (6) and (7) - (9) for period $t$.

We satisfy the connectivity (6) and the sink assignment (7) - (9) constraints in a period $t$ with Algorithm 3. We carry out a breadth–first–search starting from $S$—many sinks to detect the active sensors that can communicate with one of the sinks (Step 2) or another active sensor that has a sink assignment (Step 3). If each active sensor cannot communicate with at least $\alpha$—many active sensors as constraints (6) require, we first consider to activate some of the standby sensors $(j,k)$, since this is cost free. Among the standby sensors $(j,k)$ having $E_{rem}^{jk} \geq (e_k^s + \zeta_t (e_k^r + e_k^c))$ remaining battery energy, we activate the one which has the highest positive communication energy product, i.e., $CoEP_{jk}$ (Step 7).

Let $(x_t, z_t)$ be the partial initial solution obtained with Algorithm 2 in period $t$. Then, we calculate the underconnectivity $UC_{il}$ of the active sensors $(i,l)$ with Equation (40).

$$UC_{il} = \max \left\{ \alpha z_{ilt} - \sum_{(j,k) \in N \times K, (j,k) \neq (i,l)} b_{ij} z_{jkt}, 0 \right\}, (i,l) \in N \times K \quad (40)$$

In order to provide connectivity, we prefer to activate the cheapest standby sensor that has the highest
remaining battery energy and can communicate with most of the underconnected sensors. Hence, we define \( \text{CoEP}_{jk} \) for a standby sensor \((j, k)\) in period \(t\) as

\[
\text{CoEP}_{jk} = \frac{\left( \sum_{(i,l) \in \mathbb{N} \times \mathbb{K}, UC_{il} > 0} b_{jki} \right) \sum_{(i,l) \in \mathbb{N} \times \mathbb{K}, UC_{il} > 0} b_{jki} x_{jkt} E_{rem}^{jk}}{c_{jk}}, \quad (j, k) \in \mathbb{N} \times \mathbb{K}.
\] (41)

If connectivity is not held in Step 12, we consider to deploy \((x_{jk} = 1)\) and activate \((z_{jkt} = 1)\) the sensor \((j, k)\) with the highest positive communication cost ratio, i.e., \(\text{CoCR}_{jk}\) given as

\[
\text{CoCR}_{jk} = \frac{\left( \sum_{(i,l) \in \mathbb{N} \times \mathbb{K}, UC_{il} > 0} b_{jki} \right) (1 - x_{jk}) E_k}{c_{jk}}, \quad (j, k) \in \mathbb{N} \times \mathbb{K}.
\] (42)

where \(E_k\) represents the initial battery energy of a type-\(k\) sensor.

We continue with the sensor deployment as we can find a sensor that has positive \(\text{CoCR}_{jk}\) value. In the insufficient budget case, we remove the sensors that are in standby mode upto period \(t\). If we cannot generate the required budget, we update the network lifetime as \(L = t - 1\) and stop the algorithm.

In the last step of CH (Step 8 of Algorithm 1), we solve \(RP(t)\) for each period \(t\) within the lifetime \(L\) to find the minimum energy consuming sensor–to–sink data flow paths. In \(RP(t)\), \(y_{jkil}\) and \(g_{jki}\) are continuous variables representing the data flows in period \(t\) from sensor \((j,k)\) to sensor \((i,l)\) and from sensor \((j,k)\) to sink \((i,0)\), respectively. We have a partial solution \((L, \bar{x}, \bar{z}, \bar{u})\) obtained by applying Algorithms 2 and 3. We initialize the remaining energy of a sensor \((j,k)\) in period \(t = 1\) as \(E_{rem}^{jk} = E_{jkt}\) at the beginning of the period \(t + 1\) as

\[
E_{rem}^{jk,t+1} = E_{jkt} - \varepsilon_{jkt}, \quad (j, k) \in \mathbb{N} \times \mathbb{K}.
\] (43)

Observe that we can find a feasible solution of \(RP(t)\) for each period \(t\), since we activate the sensors that have sufficient energy for transmitting maximum possible flow \(\zeta_t\) in Algorithms 2 and 3. We rewrite the constraints \(\{20\}, \{16\}, \{17\}, \{11\} - \{13\}, \{14\}, \{15\}, \{10\}\) as the constraints \(\{44b\}, \{44d\}, \{44e\}, \{44i\} - \{44l\}, \{44m\}\) respectively, with \(g_{jkil}\) and \(g_{jki}\) variables and given \((\bar{x}, \bar{z}, \bar{u})\) values in \(RP(t)\). Constraints \(\{44c\}\) bound the energy consumption with \(E_{rem}^{jk,t}\) which is calculated with the Equations \(\{43\}\). Observe that \(w\) variables are not required in \(RP(t)\), since we can represent the constraints \(\{30\} - \{33\}\) with the constraints \(\{44i\}\) and \(\{44l\}\). Furthermore, we do not need constraints \(\{34\}\), since \(RP(t)\) avoids unnecessary flows by minimizing the energy consumption. \(RP(t)\) can be solved in polynomial.
time, since it is a linear programming formulation [11].

\[ \text{RP}(t): \]
\[
\min \sum_{(j,k) \in N \times K} \varepsilon_{jk} \\
\text{s.t.:} \\
\varepsilon_{jk} = e_k^r z_{jkt} + e_k^l \sum_{(i,l) \in N \times K} w_{iljk} + e_k^u \sum_{i \in N} g_{jki}, \ (j,k) \in N \times K \\
\varepsilon_{jk} \leq E_{\text{rem}}^{jkt}, \ (j,k) \in N \times K \\
\sum_{(i,l) \in N \times K} w_{iljk} + h_{jkt} \tilde{z}_{jkt} = \sum_{(i,l) \in N \times K} y_{jkl} + \sum_{i \in N} g_{jki}, \ (j,k) \in N \times K \\
\sum_{(j,k) \in N \times K} g_{jki} = \sum_{(j,k) \in N \times K} h_{jkt} \tilde{u}_{ijk t}, \ i \in N \\
y_{iljk} \leq M_1 b_{il}, \ (i,l),(j,k) \in N \times K \\
\sum_{(i,l) \in N \times K} y_{jkl} \leq M_1 \tilde{z}_{jkt}, \ (j,k) \in N \times K \\
\sum_{(i,l) \in N \times K} y_{iljk} \leq M_1 \tilde{z}_{jkt}, \ (j,k) \in N \times K \\
y_{iljk} \leq M_1 \left(1 - \sum_{v \in N} |\tilde{u}_{vilt} - \tilde{u}_{v jkt}| \right), \ (i,l),(j,k) \in N \times K \\
g_{ilj} \leq M_2 b_{ilj}, \ (i,l) \in N \times K, j \in N \\
\sum_{(i,l) \in N \times K} g_{ilj} \leq M_2 \tilde{x}_{j0}, \ j \in N \\
g_{ilj} \leq M_2 \left(1 - \sum_{v \in N} |\tilde{u}_{vilt} - 1_j(v)\tilde{x}_{j0}| \right), \ (i,l) \in N \times K, j \in N \\
y_{ilid} = 0, \ (i,l) \in N \times K \\
y_{iljk} \geq 0, \ (i,l),(j,k) \in N \times K \\
g_{ij} \geq 0, \ (i,l) \in N \times K, j \in N \\
\varepsilon_{jk} \geq 0, \ (j,k) \in N \times K.
\]

CH method provides feasibility with respect to coverage (3) and budget (21) constraints for all periods in the planning horizon \( T \) (Step 1 of Algorithm 1). Then, we assign a sink to each of the active sensors while satisfying connectivity (6) constraints for all periods (Step 4 of Algorithm 1). Lastly, we find the minimum energy consuming paths from sensors to their assigned sinks for all periods (Step 8 of Algorithm 1). CH consumes most of the budget to cover the network as more periods as possible in Step 1. This strategy decreases the chance of providing connectivity by deploying new sensors in Step 4, which adversely affects the lifetime \( L \). From this observation, we propose DH method (see Section 4.1.2), which considers feasibility with respect to coverage (3), budget (21), connectivity (6), sink assignment (7) - (9) constraints and determination of data routes for each period independently.

4.1.2 Disjunctive Heuristic

We summarize our DH method in Algorithm 4. We first set the active sensors, which do not have sufficient battery energy \( E_{\text{rem}}^{jkt} \) in a period \( t \), to standby mode (Steps 1 - 4). We treat each period
individually, and satisfy constraints in the order of coverage (3), budget (21), connectivity (6), sink assignment (7) - (9). We remove standby sensors up to the current period when the budget is not enough to deploy new sensors for coverage and connectivity (Steps 11 and 18).

**Algorithm 4. (Disjunctive Heuristic)**

**Input:** An instance of PSRC, a vector \((L, n, x, z, u, y, g, w)\)

0. Set \(t = 1, L = T\).
1. For Each \(t \leq L\)
2. For Each active sensor \((j, k)\) in period \(t\) \((z_{jkt} = 1)\), calculate \(E_{jkt}^{rem}\)
3. If \(E_{jkt}^{rem}\) is not sufficient, Then \(z_{jkt} = 0\).
4. End For Each
5. For Each \(t \leq L\)
6. If every node is covered, Then
7. If budget is satisfied, Then
8. If Algorithm 3 generates a feasible solution subject to constraints (6) and (7) - (9) in period \(t\), Then
   Apply Algorithm 3 to deactivate \((z_{jkt} = 0)\) the sensors without violating constraints (3), (6) and (7) - (9).
   Solve RP\((t)\) to determine the data flows \(y\) and \(g\).
   \(t \leftarrow t + 1\)
9. Else Stop.
10. End If
11. Else order standby sensors in \([0, t]\) in nonincreasing cost and remove them until budget is satisfied.
12. If budget is violated, Then \(n_t = 0, L = t - 1\) and Stop.
13. Else apply Steps 8 - 10.
14. End If
15. Else While there is \(CEP_{j,k} > 0\) with sufficient \(E_{jkt}^{rem}\) for some sensor \((j, k)\) activate \((z_{jkt} = 1)\) the standby sensor \((j, k)\) with the highest \(CEP_{j,k}\).
16. End While
17. While there is a sensor \((j, k)\) with the highest \(CCR_{j,k} > 0\)
18. If budget is not enough for the sensor \((j, k)\), Then order standby sensors in \([0, t]\) in nonincreasing cost and remove them until enough budget is obtained.
19. If budget is enough,
20. Then deploy \((x_{j,k} = 1)\) and activate \((z_{jkt} = 1)\) the sensor \((j, k)\) with the highest \(CCR_{j,k}\).
21. Else \(n_t = 0, L = t - 1\) and Stop.
22. End If
23. End While
24. Apply Steps 8 - 10.
25. End If
26. End For Each

**Output:** A feasible solution of PSRC with lifetime \(L\).

Furthermore, we save energy with Algorithm 5 by setting the active sensors, which will not harm coverage and connectivity restrictions, to standby mode (Step 8). We start deactivation from the most expensive active sensor, since we can generate more budget if we consider to remove standby sensors for coverage or connectivity in some period. \(RP(t)\) finds the minimum energy consuming paths for the active sensors to their assigned sinks in period \(t\) (Step 8). We move to the next period if we satisfy the
constraints for the current period. This budget and battery energy utilization strategy improves the 
lifetime $L$ compared to CH as we report in Section 5.

Algorithm 5. (Deactivating Unnecessary Sensors)

Input: An instance of PSRC, period $t$, a partial feasible solution $(L, n_t, x_t, z_t, u_t)$

0. Let $isActive = false$.
   Let $L$ be the nonincreasing cost order of the active sensors $(j, k)$ in period $t$ ($z_{jkt} = 1$).
1. For Each active sensor $(j, k) \in L$
   2. $isActive = false$.
   3. For Each node $i$ with $a_{ijk} = 1$
      4. If constraints (3) are violated for node $i$, Then $isActive = true$.
   5. End For Each
   6. If $isActive = false$, Then
      7. For Each active sensor $(i, l)$ with $b_{jkl} = 1$
         8. If constraints (6) are violated for sensor $(i, l)$, Then $isActive = true$.
     9. End For Each
    10. End If
   11. If $isActive = false$, Then
     12. Deactivate sensor $(j, k)$ in period $t$ ($z_{jkt} = 0$).
    13. For Each active sensor $(i, l)$ with $b_{jkl} = 1$, set $u_{vilt} = null$ for all $v \in N$.
    14. For Each active sensor $(i, l)$ without sink assignment
       15. Pick one of the active sensors $(i', l')$ with $b_{i'kl'} = 1$ and set $u_{vilt} \leftarrow u_{v'i'l't}$ for all $v \in N$.
    16. End For Each
    17. End If
   18. End For Each

Output: Unnecessary active sensors are in standby mode in period $t$.

The computational complexity of Algorithm 5 is given as $O(N^4K^3)$.

4.2 Sink Search Algorithms for SPSRC

In this section, we describe our search heuristics, i.e., LS and TS, to satisfy constraint (22) while maximizing the network lifetime $L$. These algorithms estimate the lifetime $L$ with a heuristic for PSRC (see Section 4.1) while searching the locations of $S$ many sinks.

4.2.1 Local Search Heuristic

In LS heuristic, given in Algorithm 6, we explore $N$ candidate locations to deploy $S$ sinks. We assume that $x_0 = (x_{10}, x_{20}, ..., x_{N0})$ with $x_{j0} \in \{0, 1\}$ is the binary vector of sink locations. We sort $N$ candidate locations in nondecreasing sink deploy cost $c_{j0}$ order. Initially, we locate $S$ sinks to the cheapest nodes. We implement a PSRC heuristic (CH or DH) to obtain an initial network lifetime $L$ with the remaining budget $(B - \sum_{j \in N} c_{j0}x_{j0})$ (Step 0).

At each iteration $iter$, we randomly change the locations of $s \leq S$ sinks. We pick a location $j$ with probability $p(j)$, which is inversely proportional to the sink cost $c_{j0}$ as given in Equation (45). We
can have more remaining budget for the PSRC problem by economically locating the sinks with this strategy.

\[
p(j) = \sum_{i \neq j \in N} \frac{c_{i0}}{(N-1) \sum_{i \in N} c_{i0}}, \quad j \in N.
\]

Given the locations \(x_0\) of \(S\)-sinks, there are \(NS_s = \binom{N-S}{S}\) different alternatives to relocate \(s\)-sinks. In LS, we scan \(P_s\) percentage of the neighborhood \(NS_s\) to determine an improving solution (Step 3). Once we relocate the sinks, we compute the network lifetime \(\bar{L}\) with the remaining budget through a PSRC heuristic (Step 4). We continue with the search until we complete \(iterLim\)–many iterations or we cannot update the current lifetime \(L\) for \(nImpr\)–many consecutive iterations.

Algorithm 6. (Local Search)

\begin{itemize}
  \item \textbf{Input:} \# of candidate locations \(N\), \# of sinks \(S\), \(iterLim\), \(nImpr\)
  \item 0. Set \(iter = 0\) and \(noImpr = 0\). Let \(x_0\) be the vector of sink locations.
  \item Locate \(S\) sinks to the cheapest nodes.
  \item Let \(L\) be the lifetime found by a PSRC heuristic (CH or DH).
  \item \textbf{While} \(iter < iterLim\) and \(noImpr < nImpr\)
  \item \textbf{For Each} \(s \leq S\)
  \item \textbf{For} \(NS_s\) \(P_s\)-many trials
  \item \textbf{Randomly change locations of} \(s\)-sinks.
  \item Let \(L\) be the lifetime found by a PSRC heuristic.
  \item \textbf{If} \(\bar{L} > L\), \textbf{Then} \(L \leftarrow \bar{L}\), update \(x_0\).
  \item \textbf{End For}
  \item \textbf{End For Each}
  \item \textbf{If} \(L\) is improved, \textbf{Then} \(noImpr = 0\), \textbf{Else} \(noImpr \leftarrow noImpr + 1\).
  \item \(iter \leftarrow iter + 1\)
  \item \textbf{End While}
\end{itemize}

\textbf{Output:} A feasible solution of SPSRC with lifetime \(L\).

4.2.2 Tabu Search Heuristic

TS, given in Algorithm 7, aims to visit as distinct parts of the solution space as possible by forbidding to revisit the recent \(tabuTenure\)-many solutions from the solution space of sink locations [6]. Similar to LS, we locate \(S\) sinks to the cheapest nodes initially. We swap \(s\)-sinks \((s \leq S)\) randomly using the probability density function in Equation (45) to move another solution (Steps 2 - 6).

We implement a PSRC heuristic (CH or DH) to determine the network lifetime \(\bar{L}\) with the current sink locations (Step 4). In Step 5, we add improving solutions to \(tabuList\), which stores recent \(tabuTenure\)-many solutions. That is as we add a new solution to \(tabuList\), we remove the oldest solution from \(tabuList\). The algorithm stops after \(iterLim\)–many iterations or \(nImpr\)–many consecutive nonimproving iterations.
Algorithm 7. (Tabu Search)

**Input:** \# of candidate locations \(N\), \# of sinks \(S\), iterLim, nImpr, tabuTenure

0. Set \(t = 0\), noImpr = 0 and tabuList = \(\emptyset\). Let \(x_0\) be the vector of sink locations.

1. Locate \(S\) sinks to the cheapest nodes. Add \(x_0\) to tabuList.

2. Let \(L\) be the lifetime found by a PSRC heuristic (CH or DH).

3. While \(t < \text{iterLim}\) and noImpr < nImpr

4. For Each \(s \leq S\)

5. For \(NS_sP_s - \text{many trials}\)

6. Randomly change locations of \(s\) sinks to obtain a nontabu vector.

7. Let \(\bar{L}\) be the lifetime found by a PSRC heuristic.

8. If \(\bar{L} > L\), Then \(L \leftarrow \bar{L}\), update \(x_0\) and add to tabuList.

9. End For

10. End While

**Output:** A feasible solution of SPSRC with lifetime \(L\).

5 Computational Results

The computations have been carried out on a computer with 2.0 GHz Intel Xeon E5-2620 processor and 46 GB of RAM working under Windows Server 2012 R2 operating system. In computational experiments, we used CPLEX 12.7.0 to solve \(RP(t)\) model for period \(t\) in CH and DH algorithms. We implement all algorithms in the C++ programming language.

We summarize the computational parameters in Table 2. In our experiments, we consider \(n \times n\) square grid region to be monitored. That is, we try eight different network sizes from \(N = 16\) to \(N = 225\). Each node \(i\) in the region has a coverage requirement \(f_i = 2\). We maximize the network lifetime \(L\) within a planning horizon \(T = 400\). We provide connectivity if each active sensor communicates with at least \(\alpha = 1\) active sensor in a period. We conduct experiments with three budget \(B\) levels, i.e., low, medium and high, which are calculated as in Equations (46). As an example with the high budget, we can deploy type–1 sensors to 25% of the nodes, type–2 sensors to 75% of the nodes and we can locate \(S\) sinks of average cost.

\[
B_{\text{low}} = \frac{S}{N} \sum_{j \in N} c_{j0} + 0.75 \sum_{j \in N} c_{j1} + 0.25 \sum_{j \in N} c_{j2}, \tag{46a}
\]

\[
B_{\text{medium}} = \frac{S}{N} \sum_{j \in N} c_{j0} + 0.50 \sum_{j \in N} c_{j1} + 0.50 \sum_{j \in N} c_{j2}, \tag{46b}
\]

\[
B_{\text{high}} = \frac{S}{N} \sum_{j \in N} c_{j0} + 0.25 \sum_{j \in N} c_{j1} + 0.75 \sum_{j \in N} c_{j2}. \tag{46c}
\]
We assume a sink is of type–0 and we have $K = 2$ different sensor types for sensing the region and transmitting the data packets. There is a cost $c_{jk}$ to deploy a sensor or a sink at node $j$, which we randomly determine within the ranges given in Table 2. We take a period length as 12 hours and we assume that every half an hour a data packet is generated by a sensor. Hence, a type–$k$ sensor can generate $h_{jk} = 24$ data packets in a period. Sensing $r^s_k$ and communicating $r^c_k$ ranges of a type–2 sensor are twice as large as the ones of a type–1 sensor. A sink (type–0) cannot sense or communicate, and we assume it does not consume energy. We determine the values of the sensor parameters $e^s_k, e^c_k$ and $E_k$ based on the experimental results for a Mica2 mote studied by Calle and Kabara [3]. We experiment for three initial battery energy $E_k$ levels, i.e., low, medium and high. At high energy level, the battery of a type–$k$ sensor is full. The medium energy level is 2/3 of the full battery energy and 1/3 of the full battery energy refers to the low energy level.

Table 2: List of computational parameters

| Parameters | Sensor Specifications |
|------------|-----------------------|
| $N$        | 16, 25, 36, 49, 64,   |
|            | 81, 100, 225          |
| $K$        | 2                     |
| $T$        | 400                   |
| $\alpha$  | 1                     |
| $f_i$      | 2 for all $i \in N$   |
| $B$        | low, medium, high     |
| $c_{jk}$   | $(10, 15)$            |
| $h_{jk}$   | 0                     |
| $r^s_k$    | 0                     |
| $r^c_k$    | 0                     |
| $e^s_k$    | 0                     |
| $e^c_k$    | 0                     |
| $E_k$      | low $\infty$         |
|            | 19200                 |
|            | 28800                 |
| $E_k$      | medium $\infty$      |
|            | 38400                 |
|            | 57600                 |
| $E_k$      | high $\infty$        |
|            | 57600                 |
|            | 86400                 |
| $s$        | 1                     |
| $S$        | 2, 3                  |
| $iterLim$  | 100                   |
| $nImpr$    | 20                    |
| $tabuTenure$ | 10                |
| $P_s(\%)$  | LS 20, 40, 40         |
|            | TS 100, 20, 10        |
| Time Limit | 3600 secs             |
5.1 Computational Results for PSRC

In this section, we give the computational results for our feasible solution generation algorithms CH (see Section 4.1.1) and DH (see Section 4.1.2). We randomly locate S sinks in the network and solve for the lifetime \( L \) of the corresponding PSRC problem.

We investigate the sensitivity of our CH and DH methods to the number of sinks (\( S = 2 \) or 3), budget level (low, medium, high), energy level (low, medium, high), and network size (from \( N = 16 \) to 225). For a given set of parameters (\( S, \text{Budget}, \text{Energy}, N \)), we randomly generate 10 instances and report the average values. In the following tables, the column "L" is the network lifetime and "CPU (secs)" is the computational time in seconds.

| \( S \) | \( N \) | Low | Medium | High | Low | Medium | High | CPU (secs) |
|------|------|-----|--------|------|-----|--------|------|-----------|
| 2    | 16   | 79.8 | 1.05   | 79.8 | 1.04 | 79.8   | 1.06 | 79.8      |
|      | 25   | 83.6 | 2.29   | 83.6 | 2.06 | 83.6   | 2.05 | 83.6      |
|      | 36   | 64.8 | 3.87   | 64.8 | 4.08 | 64.8   | 3.88 | 64.8      |
|      | 49   | 74.0 | 9.08   | 74.0 | 8.82 | 74.0   | 9.03 | 74.0      |
|      | 64   | 72.0 | 15.77  | 72.0 | 15.78 | 72.0 | 15.81 | 72.0 |
|      | 81   | 75.0 | 29.63  | 75.0 | 29.55 | 75.0 | 29.82 | 75.0 |
|      | 100  | 75.6 | 52.11  | 75.6 | 52.01 | 75.6 | 52.14 | 75.6 |
|      | 225  | 62.3 | 476.95 | 62.8 | 481.37 | 62.8 | 479.64 | 62.8 |

As we summarize in Table 3, lifetime \( L \) increases as the energy level gets higher with CH method. Lifetime \( L \) improves when we increase the budget level from low to medium for \( S = 2, N = 225 \) both in medium (from 138.2 to 139.3) and high (from 213.8 to 215.4) energy levels. Besides, at low budget level and \( N = 225 \), we have larger lifetime \( L \) when we have \( S = 3 \) instead of \( S = 2 \) for both medium (from 138.2 to 139.3) and high (from 213.8 to 215.4) energy levels. It seems that, for a given budget and energy level, lifetime \( L \) gets smaller as the network size \( N \) increases, since satisfying coverage and connectivity restrictions requires more budget and energy resources.

We give the performance of DH method in Table 4. We observe that, DH gives better \( L \) values than CH for all set of parameters (\( S, \text{Budget}, \text{Energy}, N \)). This difference occurs since DH uses energy and budget resources economically by considering coverage and connectivity restrictions of each period independently. On the other hand, CH consumes most of the budget and energy to satisfy coverage constraints as more periods as possible. Hence, we may not have sufficient remaining budget to deploy...
Table 4: Computational results for DH

| Budget | N = 2 | N = 3 |
|--------|-------|-------|
| L CPU (secs) | L CPU (secs) | L CPU (secs) |
| L CPU (secs) | L CPU (secs) | L CPU (secs) |
| Low | Medium | High |
| Low | Medium | High |

The results in Table 4 show that lifetime \( L \) is not improved as we deploy more sinks in the network.

We expect to see the effect of the number of sinks \( S \) on the lifetime \( L \) for larger networks. In large networks energy consumption in routing becomes dominant, since there will be more sensors on the path from a sensor to its assigned sink. One can observe for \( N = 49 \) that increasing budget from low to medium and then to high improves the lifetime \( L \) for all energy levels. For example at medium energy level, we have \( L = 182.4 \) at low budget level, which becomes 190 for medium budget level and 197.6 for high budget level.

We also experiment for the performance of CPLEX 12.7.0 within 3600 seconds time limit. We implement \( SPSRC \) formulation with known sink locations (\( SPSRC \) formulation reduces to \( PSRC \)).

As we report in Table 5, the best known solution that CPLEX 12.7.0 can find has lifetime \( L = 1 \) within the time limit even for the instances of \((S = 2, \text{High Budget, High Energy}, N = 16)\) when \( T = 400 \).

Besides, CPLEX cannot improve the trivial upper bound \( UB = 400 \).

Table 5: Computational results for CPLEX for \( N = 16 \)

| Energy | Budget | \( S = 2 \) | \( S = 3 \) |
|--------|-------|-------|-------|
| L UB CPU (secs) | L UB CPU (secs) |
| Low | Medium | High | Low | Medium | High |

new sensors if the connectivity restrictions are not satisfied in a period.

The results in Table 4 show that lifetime \( L \) is not improved as we deploy more sinks in the network.

We expect to see the effect of the number of sinks \( S \) on the lifetime \( L \) for larger networks. In large networks energy consumption in routing becomes dominant, since there will be more sensors on the path from a sensor to its assigned sink. One can observe for \( N = 49 \) that increasing budget from low to medium and then to high improves the lifetime \( L \) for all energy levels. For example at medium energy level, we have \( L = 182.4 \) at low budget level, which becomes 190 for medium budget level and 197.6 for high budget level.

We also experiment for the performance of CPLEX 12.7.0 within 3600 seconds time limit. We implement \( SPSRC \) formulation with known sink locations (\( SPSRC \) formulation reduces to \( PSRC \)).

As we report in Table 5, the best known solution that CPLEX 12.7.0 can find has lifetime \( L = 1 \) within the time limit even for the instances of \((S = 2, \text{High Budget, High Energy}, N = 16)\) when \( T = 400 \).

Besides, CPLEX cannot improve the trivial upper bound \( UB = 400 \).
5.2 Computational Results for SPSRC

We summarize the computational results for our sink location search algorithms LS (see Section 4.2.1) and TS (see Section 4.2.2) in this section. As we note at Step 0 of Algorithms 6 and 7, LS and TS require a heuristic for PSRC. We utilize our DH method in LS and TS algorithms to generate a feasible solution of PSRC, since DH provides higher $L$ values than CH.

We generate 10 random instances for each parameter set $(S,\text{Budget, Energy, }N)$ and report the average results. We impose 3600 seconds of time limit to our search algorithms. We bound the number of iterations in LS and TS with $\text{iterLim} = 100$. Besides, we stop the search if we cannot update the lifetime $L$ for $n\text{Impr} = 20$ consecutive iterations. The length of the tabu list, i.e., $\text{tabuTenure}$, in TS is 10. As given in Table 2, we scan $P_s$ percentage of the $s$–swap neighborhood in LS and TS. For example, we scan $P_s = 20\%$ of the 1–swap neighborhood in LS, whereas $P_s = 100\%$ in TS.

We give the results for LS method in Table 6. Comparison of Tables 4 and 6 shows that the lifetime $L$ improves when we carry out search for the locations of the sinks in the network. For example, LS prolongs the average lifetime $L$ by 0.6 compared to DH for the instance $(S = 2, N = 100)$ with medium energy and low budget ($L = 185.0$ in Table 4 and $L = 185.6$ in Table 6).

| Energy | Budget | $N$ | $S = 2$ | $S = 3$ | $S = 4$ | $S = 5$ |
|--------|--------|-----|---------|---------|---------|---------|
|        |        | Low | Medium  | High    | Low     | Medium  | High    |
|        |        | $L$ | $CPU$ (secs) | $L$ | $CPU$ (secs) | $L$ | $CPU$ (secs) |
|        |        |     |          |     |          |     |          |
|        |        | 16  | 84.0     | 4.15 | 84.1     | 4.22 | 84.0    |
|        |        | 25  | 84.0     | 4.15 | 84.1     | 4.22 | 84.0    |
|        |        | 36  | 85.4     | 4.52 | 85.4     | 4.52 | 85.4    |
|        |        | 49  | 85.5     | 4.92 | 85.5     | 5.00 | 85.5    |
|        |        | 64  | 85.7     | 5.22 | 85.7     | 5.30 | 85.7    |
|        |        | 81  | 85.7     | 5.52 | 85.7     | 5.60 | 85.7    |
|        |        | 100 | 85.8    | 6.15 | 85.8     | 6.23 | 85.8    |
|        |        | 225 | 85.9    | 6.65 | 85.9     | 6.77 | 85.9    |

Table 6: Computational results for LS with DH

It is crucially important to locate the sinks closer to their assigned sensors to improve the network lifetime $L$. In such a case, one activates fewer sensors, i.e., consumes less energy, to provide communication among the sensors and their assigned sinks. As we can see from Table 6, increasing the budget level extends the lifetime $L$ at a certain energy level. For example, at low energy level and $(S = 2, N = 81)$, the lifetime $L$ gradually takes values 83.0, 83.9 and 84.7 as the budget level gets higher. Moreover, increasing the number of sinks in the network from $S = 2$ to 3 helps to the network lifetime $L$. 26
SPSRC finds the optimal locations of the sensors and sinks, active/standby periods of the sensors and the data transmission routes from each active sensor to its assigned sink. At each period, we need to cover each node in the network and the active sensors should communicate with each other to reach their assigned sinks. The aim is to maximize the number of such periods within limited budget and battery energy resources.

**SPSRC** is NP-complete since it includes the set covering problem as a subproblem. Hence, exact solution of the problem cannot be found even for small networks in acceptable amount of time. For the solution of the problem, we first assume that the sink locations are given. For the reduced problem, i.e., **PSRC**, we propose constructive heuristic (CH) and disjunctive heuristic (DH). Then, we develop
local search (LS) and tabu search (TS) methods to determine the best locations of the sinks for the network lifetime. In our computational experiments, we observe that DH is better than CH in terms of lifetime. Hence, we proceed with DH while experimenting for LS and TS. TS scans larger proportion of the solution space compared to LS. That is TS provides higher lifetime values within the time limit. Our solution techniques are heuristic approaches. That is, development of optimization algorithms for SPSRC problem can be a future research. Furthermore, we assume that the sinks are at fixed locations. The problem can be further investigated for mobile sinks.

References

[1] Aneja, Y., Chandrasekaran, R., Li, X., and Nair, K. (2010). A branch-and-cut algorithm for the strong minimum energy topology in wireless sensor networks. *European Journal of Operational Research*, 204(3):604 – 612.

[2] Arai, S., Iwatani, Y., and Hashimoto, K. (2010). Fast sensor scheduling with communication costs for sensor networks. In *Proceedings of the 2010 American Control Conference*, pages 295–300.

[3] Calle, M. and Kabara, J. (2006). Measuring energy consumption in wireless sensor networks using gsp. In *2006 IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 1–5.

[4] Cerulli, R., Donato, R. D., and Raiconi, A. (2012). Exact and heuristic methods to maximize network lifetime in wireless sensor networks with adjustable sensing ranges. *European Journal of Operational Research*, 220(1):58 – 66.

[5] Garey, M. R. and Johnson, D. S. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

[6] Gendreau, M. and Potvin, J.-Y. (2005). Metaheuristics in combinatorial optimization. *Annals of Operations Research*, 140(1):189–213.

[7] Gupta, N., Kumar, N., and Jain, S. (2016). Coverage problem in wireless sensor networks: A survey. In *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, pages 1742–1749.

[8] Guvensan, M. A. and Yavuz, A. G. (2011). On coverage issues in directional sensor networks: A survey. *Ad Hoc Networks*, 9(7):1238 – 1255.

[9] Han, G., Liu, L., Jiang, J., Shu, L., and Hancke, G. (2017). Analysis of energy-efficient connected target coverage algorithms for industrial wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 13(1):135–143.
[10] Kafi, M. A., Djenouri, D., Ben-Othman, J., and Badache, N. (2014). Congestion control protocols in wireless sensor networks: A survey. *IEEE Communications Surveys Tutorials*, 16(3):1369–1390.

[11] Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395.

[12] Keskin, M. E. (2017). A column generation heuristic for optimal wireless sensor network design with mobile sinks. *European Journal of Operational Research*, 260(1):291 – 304.

[13] Lajara, R., Pelegri-Sebastiá, J., and Solano, J. J. P. (2010). Power consumption analysis of operating systems for wireless sensor networks. *Sensors*, 10(6):5809 – 5826.

[14] Lee, S., Younis, M., and Lee, M. (2015). Connectivity restoration in a partitioned wireless sensor network with assured fault tolerance. *Ad Hoc Networks*, 24:1 – 19.

[15] Lersteau, C., Rossi, A., and Sevaux, M. (2018). Minimum energy target tracking with coverage guarantee in wireless sensor networks. *European Journal of Operational Research*, 265(3):882 – 894.

[16] Mini, S., Udgata, S. K., and Sabat, S. L. (2014). Sensor deployment and scheduling for target coverage problem in wireless sensor networks. *IEEE Sensors Journal*, 14(3):636–644.

[17] Rault, T., Bouabdallah, A., and Challal, Y. (2014). Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, 67:104 – 122.

[18] Salarian, H., Chin, K. W., and Naghdy, F. (2014). An energy-efficient mobile-sink path selection strategy for wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 63(5):2407–2419.

[19] Slavík, P. (1996). A tight analysis of the greedy algorithm for set cover. In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC ’96, pages 435–441, New York, NY, USA. ACM.

[20] Soua, R. and Minet, P. (2011). A survey on energy efficient techniques in wireless sensor networks. In 2011 4th Joint IFIP Wireless and Mobile Networking Conference (WMNC 2011), pages 1–9.

[21] Tian, J., Gao, M., and Ge, G. (2016). Wireless sensor network node optimal coverage based on improved genetic algorithm and binary ant colony algorithm. *EURASIP Journal on Wireless Communications and Networking*, 2016(1):104.

[22] Türkoğulları, Y. B., Aras, N., Altınel, K., and Ersoy, C. (2010). A column generation based heuristic for sensor placement, activity scheduling and data routing in wireless sensor networks. *European Journal of Operational Research*, 207(2):1014 – 1026.

[23] Tuna, G., Gungor, V. C., and Gulez, K. (2014). An autonomous wireless sensor network deployment system using mobile robots for human existence detection in case of disasters. *Ad Hoc Networks*, 13:54
68. (1)Special Issue: Wireless Technologies for Humanitarian Relief & (2)Special Issue: Models And Algorithms For Wireless Mesh Networks.

[24] Tunca, C., Isik, S., Donmez, M. Y., and Ersoy, C. (2015). Ring routing: An energy-efficient routing protocol for wireless sensor networks with a mobile sink. *IEEE Transactions on Mobile Computing*, 14(9):1947–1960.

[25] Wang, L. and Xiao, Y. (2006). A survey of energy-efficient scheduling mechanisms in sensor networks. *Mob. Netw. Appl.*, 11(5):723–740.

[26] Wang, X., Huang, L., Leng, B., Xu, H., and Yang, C. (2017). Joint channel and sink assignment for data collection in cognitive wireless sensor networks. *International Journal of Communication Systems*, 30(5).

[27] Wu, Y., Li, X. Y., Li, Y., and Lou, W. (2010). Energy-efficient wake-up scheduling for data collection and aggregation. *IEEE Transactions on Parallel and Distributed Systems*, 21(2):275–287.

[28] Yang, Q., He, S., Li, J., Chen, J., and Sun, Y. (2015). Energy-efficient probabilistic area coverage in wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 64(1):367–377.

[29] Yao, Y., Cao, Q., and Vasilakos, A. V. (2015). Edal: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for heterogeneous wireless sensor networks. *IEEE/ACM Trans. Netw.*, 23(3):810–823.

[30] Younis, M., Senturk, I. F., Akkaya, K., Lee, S., and Senel, F. (2014). Topology management techniques for tolerating node failures in wireless sensor networks: A survey. *Computer Networks*, 58(Supplement C):254 – 283.

[31] Yu, Z., Teng, J., Bai, X., Xuan, D., and Jia, W. (2014). Connected coverage in wireless networks with directional antennas. *ACM Trans. Sen. Netw.*, 10(3):51:1–51:28.

[32] Zhang, D., Li, G., Zheng, K., Ming, X., and Pan, Z. H. (2014). An energy-balanced routing method based on forward-aware factor for wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 10(1):766–773.

[33] Zhu, C., Zheng, C., Shu, L., and Han, G. (2012). A survey on coverage and connectivity issues in wireless sensor networks. *Journal of Network and Computer Applications*, 35(2):619 – 632. Simulation and Testbeds.

[34] Zorbas, D. and Douligeris, C. (2011). Connected coverage in wsns based on critical targets. *Computer Networks*, 55(6):1412 – 1425.