Computers & Operations Research 39 (2012) 1530–1539

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Efficient solution techniques for the integrated coverage, sink location and routing problem in wireless sensor networks

Evren Güney [a], Necati Aras [b], İ. Kuban Altınel [b,*], Cem Ersoy [c]

[a] Industrial Engineering Department, Istanbul Arel University, Istanbul, Turkey
[b] Industrial Engineering Department, Boğaziçi University, İstanbul, Turkey
[c] Computer Engineering Department, Boğaziçi University, İstanbul, Turkey

ARTICLE INFO

Available online 16 September 2011

Keywords:
Wireless sensor networks
Integer programming
Tabu search

ABSTRACT

Sensors are tiny electronic devices having limited battery energy and capability for sensing, data processing and communicating. They can collectively behave to provide an effective wireless network that monitors a region and transmits the collected information to gateway nodes called sinks. Most of the applications require the operation of the network for long periods of times, which makes the efficient management of the available energy resources an important concern. There are three major issues in the design of sensor networks: sensor deployment or the coverage of the sensing area, sink location, and data routing. In this work, we consider these three design problems within a unified framework and develop two mixed-integer linear programming formulations. They are difficult to solve exactly. However, it is possible to compute good feasible solutions of the sink location and routing problems easily, when the sensors are deployed and their locations in the sensor field become known. Therefore, we propose a tabu search heuristic that tries to identify the best sensor locations satisfying the coverage requirements. The objective value corresponding to each set of sensor locations is calculated by solving the sink location and routing problem. Computational tests carried out on randomly generated test instances indicate that the proposed hybrid approach is both accurate and efficient.

1. Introduction

Advances in micro-electromechanical systems, digital electronics and wireless communications have enabled the development of low-cost, low-power multi-functional tiny devices called sensors. They use battery power as the energy source and can communicate through wireless channels over relatively small distances. Each sensor is usually equipped with one or more sensing units, one or more transceivers, actuators, processors and storage units. As a result of significant resource limitations such as limited memory, battery power, signal processing, computation and communication capability, an individual sensor can only sense a small portion of its environment. However, when a large number of these devices work in a collaborative fashion to carry out a certain task, they form a wireless sensor network (WSN). WSNs give rise to a wide range of real-life applications in areas such as military, homeland security, health care, environment, agriculture, logistics, smart home or office design and other areas [26,5].

WSNs can be homogeneous consisting of identical sensors, or heterogeneous consisting of sensors with possibly different technical characteristics and costs. Sensors that collectively form a WSN are deployed over an area of interest called the sensor field. The energy source provided for a sensor is usually the battery power which has not yet reached the stage to operate for a very long time without being recharged. Moreover, in a large number of applications sensors are intended to work in remote or hostile environments, and it is undesirable or impossible to recharge or replace their batteries. The lifetime of a WSN is measured by the time until it is disconnected; namely there exist holes that do not collect or transmit any information. In other words, the WSN cannot provide an acceptable level of operating quality beyond its lifetime. Therefore, conserving energy and prolonging the network lifetime is an important issue in the design of WSNs [27].

The transmission range of sensors is restricted as a consequence of their energy and size limitations, hence they cannot communicate through large distances. Therefore, each sensor needs to transmit its data to a central unit called "base station" or "sink", which is a larger device with a comparatively large energy supply and long-range transmission capabilities such as internet or satellite communication.

There are various design issues in the construction of a WSN. One of the most important problems addressed in the literature is the sensor coverage problem (CP). This problem is centered

* Corresponding author. Tel.: +90 212 3597076; fax: +90 212 2651800.
E-mail address: altinel@boun.edu.tr (İ.K. Altınel).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.09.002

around a fundamental question: "How well do the sensors observe the physical space?" As pointed out by Meguerdichian et al. [23], coverage can be regarded as a measure of the quality of service of the sensing function in WSNs. Therefore, the CP is studied thoroughly and there are many exact or approximate methods to solve it (e.g., [6,14]). Also an interesting line of research has emerged, where the CP is modeled as a mixed-integer linear programming (MILP) formulation [7,2].

In a typical WSN, each sensor collects and processes data, and tries to send this information to a sink. Since sensors' communication ranges are limited, the data packets carrying the sensed information usually have to follow multi-hop paths. The routing problem (RP), which involves finding the most energy efficient sensor-to-sink routes for a given set of sensor and sink locations, is an essential problem in WSNs because data transmission is an energy consuming task [1].

Determining the optimal sink locations is also an important design issue to extend the lifetime of a WSN. The average number of hops to reach a sink becomes a critical factor that is affected by the sink locations. In a significant number of research studies, the sink locations are assumed to be determined a priori [18,20]. However, the energy consumption and thus the lifetime of the WSN can be improved by finding the best location(s) of the sink(s) since these locations have an impact upon the sensor-to-sink data transmission routes [3]. The joint optimization of sink locations and data routing is addressed in the sink location and routing problem (SLRP) that has received considerable attention in the literature as it results in a more efficient network design [13,25,17].

In this work, we jointly consider the coverage, sink location, and data routing problems in heterogeneous WSNs. We refer to this integrated problem as the coverage, sink location and routing problem (CSLRP), and develop two mathematical programming formulations by unifying these three design issues within a single model. Unfortunately, the resulting MILP formulations can only be solved for small-sized instances using commercial solvers. For medium and large-sized instances, we propose a hybrid solution procedure. In the outer loop, tabu search is employed to find near-optimal sensor locations satisfying the coverage requirements. In the inner loop, the remaining sink location and routing problems are solved using various methods for the sensor locations fixed in the outer loop. This integrated problem is also addressed in a very recent work [16]. However, the proposed benchmark model has a major flaw, which makes the results given in that work unreliable. The flaw stems from the fact that the data flow balance equations in the MILP formulation are erroneous.

The rest of the paper is organized as follows. In the next section, we introduce the CSLRP and its mathematical programming formulations. The hybrid solution approach is explained in Section 3, while experimental results are reported in Section 4. We conclude the paper in Section 5 with some remarks.

2. Problem definition and mathematical programming formulations

The objective of the CSLRP is to design a WSN such that the total available battery energy of the sensors in the WSN is spent in the most efficient way for transmitting (routing) data packets from sensors to sinks. This also helps to prolong the lifetime of the WSN as much as possible. Before we present the mathematical programming formulations for the CSLRP, we describe the problem while defining the decision variables common for both formulations.

The CSLRP is defined in a two-dimensional sensor field $\mathcal{N}$ with $|\mathcal{N}| = N$ points. We assume that every point of sensor field is a candidate location for a sensor and/or sink, and there are $|\mathcal{K}| = K$ sensor types, where $\mathcal{K}$ denotes the set of sensor types. Each sensor type has a different cost, sensing and transmission range. The cost of deploying a type-k sensor is equal to h_k, and the available budget for sensor deployment is H monetary units. Given that there may be obstacles in the terrain where the sensors are deployed, it is expected that there is inherent uncertainty associated with sensor readings and thus sensor detections must be formulated probabilistically as discussed in Brooks and Iyengar [4]. Furthermore, it is also expected that sensors closer to a point in the sensor field generally provide better coverage. To account for both issues, we adopt *probabilistic* detection and let parameter q_{ijk} denote the probability that a type-k sensor deployed at point i generates sensing data for point j. These probabilities can be calculated for each (i,j,k) triplet by adopting an appropriate sensor terrain model. A common approach is to assume that a target at Euclidean distance d_{ij} from a type-k sensor is detected with probability $q_{ijk} = e^{-\alpha_k d_{ij}}$ [9,10], where α_k is a decay parameter that determines the rate at which the detection probability of a type-k sensor decreases with the distance. When binary decision variables are defined as $s_{ik} = 1$ if a type-k sensor is deployed at point $i \in \mathcal{N}$, and $s_{ik} = 0$ otherwise, $(1-q_{ijk}s_{ik})$ gives the probability that a target located at point j is missed by a type-k sensor deployed at point i. Hence, under the assumption that sensing probabilities q_{ijk} are independent, the restriction which ensures that the overall miss probability at point j does not exceed a given maximum allowable level $0 < t_j < 1$ can be written as $\prod_{i \in \mathcal{N}} \prod_{k \in \mathcal{K}} (1-q_{ijk}s_{ik}) \le t_j$. Although this inequality is non-linear in binary decision variables s_{ik}, it can be linearized as follows. When we take the natural logarithm of both sides of the inequality, we obtain $\sum_{i \in \mathcal{N}} \sum_{k \in \mathcal{K}} \ln(1-q_{ijk}s_{ik}) \le \ln t_j$, which can be rewritten as $\sum_{i \in \mathcal{N}} \sum_{k \in \mathcal{K}} \ln(1-q_{ijk})s_{ik} \le \ln t_j$. This follows from the fact that $\ln(1-q_{ijk}s_{ik}) = 0$ when $s_{ik} = 0$ and $\ln(1-q_{ijk}s_{ik}) = \ln(1-q_{ijk})$ when $s_{ik} = 1$. By multiplying both sides of the last inequality with -1 and defining coefficients $a_{ijk} = -\ln(1-q_{ijk})$ and $b_j = -\ln t_j$, we obtain $\sum_{i \in \mathcal{N}} \sum_{k \in \mathcal{K}} a_{ijk}s_{ik} \ge b_j$, which can be regarded as a coverage constraint with b_j representing a minimum coverage threshold. We would like to note that a_{ijk} and b_j are always positive.

There are two other modeling alternatives for the sensor coverage models in the literature. In *perfect* detection (see, for example, [2,8,22]), where a sensor always detects a target remaining in its range, $a_{ijk} = 1$ if a type-k sensor deployed at point i covers (senses) point j, and $a_{ijk} = 0$ otherwise. The coverage threshold parameter b_j represents the number of sensors that need to cover point j. In *imperfect* detection (see, for example, [22]), a_{ijk} is called the coverage intensity for point j provided by a type-k sensor at point i and b_j coverage intensity threshold at point j. The coverage constraint in imperfect sensing ensures that the total coverage intensity at all points of the sensor field has to exceed the threshold level b_j.

Given that the number of sinks to be installed is equal to p, we propose two different MILP formulations for the CSLRP. Two of the decision variables are shared by both of the formulations. The first one is the binary variable s_{ik} which has been introduced before. The second one is the binary variable y_j which is equal to one if a sink is installed at point j, and zero otherwise. It is assumed that a point in the sensor field can accommodate a sensor and a sink together. Furthermore, different types of sensors can be placed at the same point. The parameters and common decision variables used in the models are shown in Table 1 for a quick reference. Besides these two sets of decision variables, there are others that have to be defined depending on the formulation. The first formulation, CSLRP-1 is an arc-flow based network design model, where the total routing energy is computed by summing up the energy consumptions on the arcs

Table 1
Parameters and common decision variables.

Parameter	Definition
N	The number of points in the sensor field
K	The number of sensor types
a_{ijk}	The coverage intensity at point j by a type-k sensor deployed at point i
b_j	The coverage threshold at point j
d_{ij}	The Euclidean distance between points i and j
α_k	The rate at which the detection intensity of a type-k sensor decreases
h_k	The cost of deploying a type-k sensor
H	The available budget for sensor deployment
p	The number of sinks to be installed
Decision variable	**Definition**
s_{ik}	One if a type-k sensor is deployed at point i, zero otherwise
y_j	One if a sink is installed at point j, zero otherwise

between all pairs of points in the sensor field. The energy consumption on an arc from point i to point j is a function of both the distance and amount of data flow between the two points. CSLRP-1 can be used to efficiently solve small instances of the CSLRP using a commercial MILP solver. The second formulation, CSLRP-2, is a facility location model, where the total routing energy is calculated in terms of paths connecting sensors to sinks. Although CSLRP-2 has more decision variables and constraints than CSLRP-1, it lends itself to an efficient heuristic algorithm to solve large instances of the CSLRP, as will be shown later.

2.1. A single-commodity network flow formulation

In this formulation, two sets of decision variables are defined for the data flows. u_{ijkl} represents the amount of sensor-to-sensor flow from a type-k sensor at point i to a type-l sensor at point j. Since it is possible to deploy different types of sensors at the same point, variables u_{iikl} do exist in the formulation. v_{ijk} represents the amount of sensor-to-sink flow from a type-k sensor at point i to a sink at point $j \neq i$. Parameters c_{ijk} represent the energy expenditure for unit data flow from sensors to sensors and from sensors to sinks. They are the objective function coefficients computed using the formula $c_{ijk} = \gamma_k d_{ij}^{\theta}$. Here, γ_k is the amount of energy spent by a type-k sensor for sending one unit of data flow along a unit distance, and θ is the path loss factor whose value is usually in the interval $[2,4]$ (see [5]). Variable w_i is used to absorb the total data inflow to a point i when a sink is installed there. The single-commodity network flow formulation CSLRP-1 is given below:

$$\min \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{N}}\sum_{\substack{l\in\mathcal{K}\\ l\neq k\\ \text{for } i=j}} c_{ijk}u_{ijkl} + \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{N}} c_{ijk}v_{ijk} \tag{1}$$

$$\text{subject to} \quad \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} a_{ijk}s_{ik} \geq b_j, \quad j\in\mathcal{N} \tag{2}$$

$$\sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} h_k s_{ik} \leq H \tag{3}$$

$$\sum_{j\in\mathcal{N}}\sum_{l\in\mathcal{K}}\sum_{\substack{k\in\mathcal{K}\\ k\neq l\\ \text{for } i=j}} u_{jilk} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{l\in\mathcal{K}} v_{jil} + \sum_{k\in\mathcal{K}} s_{ik} = \sum_{j\in\mathcal{N}}\sum_{l\in\mathcal{K}}\sum_{\substack{k\in\mathcal{K}\\ k\neq l\\ \text{for } i=j}} u_{ijkl} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{k\in\mathcal{K}} v_{ijk} + w_i, \quad i\in\mathcal{N} \tag{4}$$

$$\sum_{j\in\mathcal{N}}\sum_{\substack{l\in\mathcal{K}\\ l\neq k\\ \text{for } i=j}} u_{ijkl} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} v_{ijk} \leq NKs_{ik}, \quad i\in\mathcal{N}, k\in\mathcal{K} \tag{5}$$

$$\sum_{\substack{i\in\mathcal{N}\\ j\neq i}}\sum_{k\in\mathcal{K}} v_{ijk} \leq NKy_j, \quad j\in\mathcal{N} \tag{6}$$

$$w_i \leq NKy_i, \quad i\in\mathcal{N} \tag{7}$$

$$\sum_{j\in\mathcal{N}} y_j = p \tag{8}$$

$$u_{ijkl} \geq 0, \quad i,j\in\mathcal{N}; k,l\in\mathcal{K} : k\neq l \text{ for } i=j \tag{9}$$

$$v_{ijk} \geq 0, \quad i\neq j\in\mathcal{N}; k\in\mathcal{K} \tag{10}$$

$$w_i \geq 0, \quad i\in\mathcal{N} \tag{11}$$

$$y_i, s_{ik} \in \{0,1\}, \quad i\in\mathcal{N}; k\in\mathcal{K} \tag{12}$$

The objective function (1) minimizes the total routing energy which consists of the sensor-to-sensor and sensor-to-sink routing energies. Coverage constraints (2) ensure that a sufficient number of sensors are deployed so that the detection probability at all points in the sensor field is greater than or equal to the coverage threshold. Constraint (3) guarantees that the total cost of the sensors does not exceed the budget. Constraints (4) are the flow balance equations for each point i. They guarantee that the total inflow from other sensors to point i and the data generated by the sensor(s) at this point must either be equal to the total outflow to other points with sensors and/or a sink or be absorbed at the sink installed at point i. Constraints (5) state that sensor-to-sensor or sensor-to-sink flow from a type-k sensor at point i can only occur if such a sensor is deployed there. The upper limit is obtained by considering that the total data generated in the sensor field cannot be more than NK, where the limit is obtained when all points are deployed with all types of sensors. Constraints (6) ensure that there is no sensor-to-sink flow to point j if there is no sink at this point. Constraints (7) link variables y_i and w_i by guaranteeing that no data packets can be absorbed at a point without a sink. Constraint (8) sets the number of sinks to be installed equal to p. Finally, constraints (9)–(11) are the non-negativity restrictions on continuous variables, while constraints (12) are binary restrictions on discrete variables y_i and s_{ik}. Observe that there are $O(N^2K^2)$ variables and $O(NK)$ constraints in the formulation.

At this point it may be worthwhile to list two observations about this formulation

1. When variable v_{ijk}, $i,j\in\mathcal{N}$, $k\in\mathcal{K}$ is positive on the right-hand side of balance equation (4), there is an outflow from type-k sensor deployed at point i to a sink at another point. This implies that there is no sink at point i and $w_i = 0$, because otherwise (i.e., there exists a sink at point i), there would not be a sensor-to-sink flow to a sink at another point in an optimal solution due to the fact that $c_{iik} = 0$ when $d_{ii} = 0$.
2. Since $c_{iik} = 0$, there will always be an alternative optimal solution with respect to the flow variables u_{ijkl} and v_{ijk} if it is optimal that a sink is installed at a point where a sensor is deployed. Note that the total inflow from other sensors to such a sink can be either direct or through sensor k placed at that point. Namely, either $\sum_{j\in\mathcal{N}, j\neq i}\sum_{l\in\mathcal{K}} u_{jilk} = 0$ and $\sum_{j\in\mathcal{N}, j\neq i}\sum_{l\in\mathcal{K}} v_{jil} > 0$ or $\sum_{j\in\mathcal{N}, j\neq i}\sum_{l\in\mathcal{K}} u_{jilk} > 0$ and $\sum_{j\in\mathcal{N}, j\neq i}\sum_{l\in\mathcal{K}} v_{jil} = 0$. In either case, w_i is equal to the total inflow to point i plus one (the data generated by the sensor at this point).

We notice that the flow balance equations in [16] are not correct, because they cannot handle the cases where two types of sensors are deployed at the same point or a sink is co-located with a sensor. We solve CSLRP-1 by means of a general-purpose MILP solver in order to compute optimal values that can be used in assessing the accuracy of the proposed heuristics. The integrated problem we deal with in this paper can also be formulated as a multi-commodity network flow model where each sensor is assumed to generate a commodity, which makes the total number of commodities $N \times K$. We do not consider this type of formulation here since it is computationally very demanding.

2.2. An assignment formulation

In this second formulation, we use binary variables x_{ijk} where $x_{ijk}=1$ if a type-k sensor at point i is assigned to a sink at point j, and $x_{ijk}=0$ otherwise. To compute the corresponding routing energy we need to know the arcs that are used in this assignment, i.e., the arcs of the path connecting the sensor at point i to the sink at point j. For this purpose, we introduce binary variable z_{lm}^{ijk}, which is equal to one when arc (l,m) is used in the assignment of a type-k sensor at point i to a sink at point j. Similar to the first formulation, c_{lmk} represents the flow energy between points l and m when there is a type-k sensor at point l. The resulting mathematical model CSLRP-2 is given below:

$$\min \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}\sum_{l\in\mathcal{N}}\sum_{m\in\mathcal{N}}\sum_{k\in\mathcal{K}} c_{lmk} z_{lm}^{ijk} \quad (13)$$

$$\text{subject to} \quad \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} a_{ijk} s_{ik} \ge b_j, \quad j\in\mathcal{N} \quad (14)$$

$$\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{N}} h_k s_{ik} \le H \quad (15)$$

$$\sum_{j\in\mathcal{N}} x_{ijk} = s_{ik}, \quad i\in\mathcal{N}, k\in\mathcal{K} \quad (16)$$

$$x_{ijk} \le y_j, \quad i,j\in\mathcal{N}, k\in\mathcal{K} \quad (17)$$

$$\sum_{j\in\mathcal{N}} y_j = p \quad (18)$$

$$\sum_{m\in\mathcal{N}} z_{lm}^{ijk} - \sum_{m\in\mathcal{N}} z_{ml}^{ijk} = \begin{cases} x_{ijk}, & l=i \\ 0, & l\in\mathcal{N}\setminus\{i,j\}, \\ -x_{ijk}, & l=j \end{cases} \quad i,j,l\in\mathcal{N}, i\ne j, k\in\mathcal{K} \quad (19)$$

$$\sum_{m\in\mathcal{N}} z_{lm}^{ijk} \le e_l, \quad i,j,l\in\mathcal{N}, i\ne j, k\in\mathcal{K} \quad (20)$$

$$e_l \le \sum_{k\in\mathcal{K}} s_{lk}, \quad l\in\mathcal{N} \quad (21)$$

$$Ke_l \ge \sum_{k\in\mathcal{K}} s_{lk}, \quad l\in\mathcal{N} \quad (22)$$

$$z_{lm}^{ijk} \ge 0, \quad i,j,l,m\in\mathcal{N}, k\in\mathcal{K} \quad (23)$$

$$s_{ik}, y_i, x_{ijk}, e_l \in \{0,1\}, \quad i,j,l\in\mathcal{N}, k\in\mathcal{K} \quad (24)$$

The objective function (13) minimizes the total routing energy over all arcs. Coverage constraints (14) and the budget constraint (15) are the same as before. Assignment constraints (16) ensure that each sensor is assigned to one and only one sink. Constraints (17) prohibit any assignment to point j if there is no sink at that point. Here we prefer the strong version of this constraint set, which can be replaced by $\sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} x_{ijk} \le NKy_j$ to obtain an equivalent formulation. This version has fewer constraints, but it yields a weaker LP relaxation. Constraint (18) sets the number of sinks to be installed in the sensor field to p. Constraints (19) are the flow balance constraints. If there is an assignment x_{ijk}, then there must be a unit outflow from the source (type-k sensor at point i), a unit inflow to the destination (sink at point j), and for all other intermediate points the total inflow must be equal to the total outflow. Constraints (20)–(22) together ensure that an arc (l,m) can only be used if there is a sensor at the tail l of arc (l,m). Note that if there is no sensor at point l (i.e., $\sum_{k\in\mathcal{K}} s_{lk}=0$), then constraint (21) forces e_l to be zero. On the other hand, if there is at least one sensor at point l, then constraint (22) makes sure that $e_l=1$. Recall that the only possibility of having more than one sensor at some point is that the deployed sensors are of different types, and there can be at most K sensors at a point. Constraints (23) and (24) are, respectively, non-negativity and binary restrictions on the variables. Note that although z_{lm}^{ijk} are non-negative variables, at the optimum solution they only take binary values. This is due to the fact that constraints (19) are the flow conservation equations for the shortest path problem when $x_{ijk}=1$.

In this formulation, the number of variables is $O(N^4K)$ and the number of constraints is $O(N^3K)$. This means that it is a larger formulation than CSLRP-1. Computational experiments also revealed that it is less efficient than CSLRP-1 when both formulations are solved by a commercial state-of-the-art MILP solver within a time limit of 4 h. On the other hand, CSLRP-2 has an important property which makes it very useful for computing approximate solutions of CSLRP as will be explained in the next section: when the sensor locations are given, CSLRP-2 reduces to the classical *p-median* problem.

3. A hybrid solution procedure

CSLRP is computationally very difficult and solving any of the proposed formulations by a general-purpose MILP solver usually generates an optimal solution only for small instances. For medium and large instances there is a need for efficient heuristic methods. In this paper, we propose a hybrid solution procedure for the solution of the CSLRP which utilizes the CSLRP-2 formulation. The outer loop of this procedure uses tabu search to identify the best sensor locations satisfying the coverage and budget requirements, while in the inner loop sink locations and data flow routes from sensors to sinks are determined in the best way for the sensor locations fixed by the outer loop.

Given a set of sensors that satisfies the coverage and budget constraints (i.e., s_{ik} are fixed), both CSLRP-1 and CSLRP-2 can be re-written in a simpler form by dropping the s_{ik} variables and some of the constraints. For CSLRP-2 in particular, constraints (14), (15), (19), (20), (21), and (22) can be removed. The resulting formulation SLRP, which is given below, deals with the problem of locating a prespecified number of sinks and assigning the given set of sensors to these sinks. In this formulation, $\mathcal{S}$ denotes the set of points in the sensor field where sensors are deployed, and $\mathcal{K}_i$ is the set of sensor types deployed at point i:

$$\text{SLRP}: \min \sum_{i\in\mathcal{S}}\sum_{k\in\mathcal{K}_i}\sum_{j\in\mathcal{N}} g_{ijk} x_{ijk} \quad (25)$$

$$\text{subject to} \quad \sum_{j\in\mathcal{N}} x_{ijk} = 1, \quad i\in\mathcal{S}, k\in\mathcal{K}_i \quad (26)$$

$$x_{ijk} \le y_j, \quad i\in\mathcal{S}, k\in\mathcal{K}_i, j\in\mathcal{N} \quad (27)$$

$$\sum_{j\in\mathcal{N}} y_j = p \quad (28)$$

$$x_{ijk}, y_j \in \{0,1\}, \quad i\in\mathcal{S}, k\in\mathcal{K}_i, j\in\mathcal{N} \quad (29)$$

As can be noticed, SLRP is the formulation of the well-known p-median problem where g_{ijk}'s, which are the objective function coefficients, constitute the energy consumption corresponding to the minimum energy path between a type-k sensor at point i and a sink at point j. In other words, they are simply the sum of the energy consumptions on the arcs that forms the shortest path in terms of energy. Note that when the locations and the types of the deployed sensors are given, we can perform a preprocessing step to determine the minimum energy path from each sensor to every candidate sink location. These paths can be determined by solving a shortest path problem between the given sensor locations and all the points in the sensor field since all points are candidate sink locations. This task can effectively be carried out using a many-to-many shortest path algorithm such as the one by Floyd [12].

Tabu search (TS) is a metaheuristic algorithm that guides the local search to prevent it from being trapped in premature local optima or cycling [15]. This is achieved by prohibiting the moves that cause to return to previously visited solutions throughout a certain number of iterations. We use TS to make a search in the solution space of sensor locations that are feasible with respect to coverage and budget constraints. The objective values corresponding to these feasible sensor locations are computed by solving the SLRP using three different methods which provide the routing of the data flows from sensors to sinks in the best possible way. Hence, we propose actually three heuristics that differ in the inner loop of the hybrid solution procedure.

We apply an easy and quick strategy to find the initial sensor deployment. It is based on the greedy heuristic introduced by Altınel et al. [2], and deploys sensors at minimum cost without violating the budget constraint by reducing the maximum level of undercoverage. At each iteration of the heuristics, we generate candidate neighborhood sets by using the following *Add*, *Drop*, and *Swap* moves: 1-Add, 2-Add, 1-Drop, 2-Drop, 1-Swap, and 2-Swap. While one and two sensors are added randomly to the existing set of sensors in the 1-Add and 2-Add moves, respectively, 1-Drop and 2-Drop moves involve the removal of one and two sensors, respectively. 1-Swap and 2-Swap moves, on the other hand, perform a random exchange of the locations of one and two sensors, respectively. Effectively, this corresponds to randomly dropping an existing sensor (two existing sensors), and then adding a new sensor (two new sensors) leaving the total number of sensors unchanged. If the new sensor set obtained by any move is feasible, the objective value associated with it is computed by solving the corresponding SLRP model. In a heterogeneous WSN, where there exist sensors with different sensing ranges and costs, the above-mentioned moves may cause an infeasibility by violating the coverage and/or budget constraints. When such an infeasibility is encountered, that move is not accepted. The total number of neighbors generated by all the moves is equal to r. In our experiments we use different r values to work with different neighborhood sizes that may affect the quality of the solutions obtained by the proposed heuristic. While generating these neighbors, those moves that are in the tabu list are not allowed for a certain number of iterations unless the corresponding objective value is better than the objective value of the incumbent.

Since the SLRP belongs to the class of the p-median problem, it can be solved by one of the available solution methods that are known to be efficient for this problem type. To this end, we choose three methods. Two of them are the Lagrangean heuristic (LH) and the nested-dual heuristic (NDH). The remaining method, greedy heuristic (GH), is a fast heuristic which is also easy to implement. As a result, we have three TS heuristics, which differ in the inner loop of the hybrid solution procedure when solving the SLRP. We refer to them as TS-LH, TS-NDH, and TS-GH. Below, we briefly overview LH, NDH, and GH that are used to solve the SLRP.

In our application of LH, we relax the assignment constraints (26) in the SLRP defined by expressions (25)–(29) with unrestricted Lagrange multipliers. The resulting Lagrangean subproblem decomposes with respect to each grid point $j \in \mathcal{N}$. Each of these subproblems can be solved by inspection, and their objective values can be ranked. The total of the sum of the p smallest objective values and the sum of the Lagrange multipliers constitutes a lower bound for the SLRP. Subgradient optimization is used to solve the Lagrangean dual problem, whose objective is to find the largest lower bound. At each iteration of the subgradient optimization procedure, an upper bound is computed on the optimal objective value of the SLRP. It is used to determine the value of the step length which is necessary to update the Lagrange multipliers. The upper bound is obtained by constructing a feasible solution to the SLRP by making use of the binary location variables given by the Lagrangean subproblem that provides the lower bound.

The NDH heuristic was proposed by Mirchandani et al. [24] to solve the p-median problem. This heuristic uses the approach developed by Erlenkotter [11] for the solution of the uncapacitated facility location problem. Erlenkotter's DUALOC method is a simple ascent adjustment procedure that solves the LP relaxation of the Lagrangean dual problem in which the median constraint (28) is dualized. This enables the computation of a lower bound on the optimal value. Then, a heuristic is used to obtain a feasible solution to the p-median problem. The details of the application of LH and NDH heuristics to the SLRP can be found in Güney et al. [17].

The greedy heuristic (GH) for the p-median problem, suggested by Kuehn and Hamburger [19], is initialized with an empty set of facility locations with an infinite objective value. Then facilities are added to this set one at a time by choosing the facility whose addition results in the largest reduction in the objective value.

In TS-LH, TS-NDH, and TS-GH, we use two termination criteria: the maximum number of iterations and the maximum number of iterations without any improvement in the incumbent's objective value. In the TS-LH, TS-NDH, and TS-GH heuristics, *num_iter* and *num_nonimp_iter* are counters that are used to keep the total number of iterations and the number of iterations without an improvement in the incumbent (the best objective value obtained so far). The parameters *max_iter* and *max_nonimp_iter* are, respectively, the limits for *num_iter* and *num_nonimp_iter*. *Obj* is the objective value of the current sensor set obtained by solving the SLRP. *Obj_Best_Neigh* is the best solution among all the neighboring solutions at a given iteration and *Obj** is the best solution obtained throughout the algorithm. Finally, *max_tabu_tenure* is a parameter that sets the number of iterations during which a solution is kept in the tabu list. The basic steps of the TS heuristics are given in Appendix using the aforementioned definitions.

4. Computational experiments

In this section, we report the accuracy and efficiency of the TS-LH, TS-NDH, and TS-GH heuristics. As the basis of our comparisons, we take the objective value of the best feasible solution provided by CPLEX 11.2 when solving the CSLRP-1 within the allowed time limit of 4 h (14,400 s). By letting z_{IP} to denote the best objective value CPLEX computes for an instance and $X \in \{z_{LH}, z_{NDH}, z_{GH}\}$ representing the best objective value obtained by the proposed methods for the same instance, we can measure the accuracy of the methods by computing the percent deviations from z_{IP} according to the formula:

$$100 \times \frac{X - z_{IP}}{z_{IP}} \tag{30}$$

We also report the CPU times to assess the efficiency of the different solution methods. All experiments are carried out on a Dell PowerEdge 2400 computer with two 64-bit, 2.66-GHz Xeon 5355 Quad Core processors and 28 GB memory.

4.1. Instance generation

We consider an open area without many obstacles as the sensor field and choose a path loss factor of $\theta = 2$. The sensor field is assumed to consist of $N = n \times n$ grid points in a lattice structure where each point is a candidate site for the deployment of sensors and locating the sinks. We generate 14 data sets where n takes values from the set {3–15,20}. Later on, we also assess the performance of the solution methods on a set of test instances where the points are distributed uniformly within the sensor field rather than being grid points. The coverage threshold b_i is set to 0.99 for all points, while the value of the available budget H for sensor deployment is determined as follows. First, the minimum-weight coverage problem is solved to optimality where the weights correspond to the unit costs of the sensors. Then, the optimal objective value gives the minimum budget level $H_{\min}$ that is required to obtain a sensor network where all the grid points are covered. In other words, to prevent the infeasibility of the CSLRP we have to set the budget H at least equal to $H_{\min}$. For the experiments, where we assess the accuracy and efficiency of the solution methods, we set $H = 1.5H_{\min}$. In a subsequent section, we carry out sensitivity analysis with respect to the parameters b_i and H. We do this by changing the value of H to various multiples of $H_{\min}$ and varying the coverage threshold b_i.

In our experiments, we consider two types of sensors with different characteristics. First, their costs are different as follows: $h_1 = 10$ and $h_2 = 20$. Consistent with their costs, the following values are assigned to the coverage parameter α_k that determines the rate at which the detection intensity of a type-k sensor decreases with the distance: $\alpha_1 = 0.5$ and $\alpha_2 = 0.4$. Note that the more expensive sensor has a lower α-value. The parameter γ_k is the amount of energy spent by a type-k sensor for sending one unit of data for unit distance. It is chosen as $\gamma_1 = 10$ nA h and $\gamma_2 = 20$ nA h for the two types of sensors, respectively [21]. The ampere-hour (A h) (its sub-units are milliampere-hour (mA h) and nanoampere-hour (nA h)) is a unit of electric charge, and is frequently used in measurements of electrochemical systems such as electroplating and electrical batteries. For example, a typical AA type of battery contains 2200 mA h of electric charge.

The value of the path loss factor θ that is used in the formula $c_{ijk} = \gamma_k d_{ij}^{\theta}$ (which computes the energy consumption of the sensors) is set to two. In a real-life application, the number of sinks to be installed is quite small. Therefore, we set $p=1$, $p=2$, and $p=3$ in the experiments.

4.2. Performance of the hybrid solution procedure

As mentioned before, the outer loop of our hybrid solution procedure calls for tabu search to identify the best sensor locations satisfying the coverage and budget requirements. The inner loop involves solving the sink location and routing problem SLRP formulated as a p-median model using three methods (LH, NDH, and GH) for each sensor location set fixed in the outer loop by tabu search. Among these three methods, LH is computationally the most intensive one, whereas GH is the fastest consuming the smallest amount of CPU time. Therefore, when determining the size of the neighborhood at each tabu search iteration we consider the difference in the efficiency of the p-median heuristics. As a result, TS-LH has the smallest neighborhood size ($r=50$) where 10 solutions are generated randomly in the neighborhood of the current solution using each of the 1-Add, 2-Add, 1-Swap, and 2-Swap moves, while five solutions are generated using the 1-Drop and 2-Drop moves. For TS-NDH, we double the number of neighboring solutions in each category which gives rise to a neighborhood size $r=100$. TS-GH, due to its CPU time efficiency, has the largest neighborhood with $r=150$ where the number of solutions generated by each move is three times as large as that of TS-LH. The other parameters of the TS heuristics are set to the following values: $max\_iter = 500$ and $max\_nonimp\_iter = 30$ iterations.

First, we consider the case with one sink ($p=1$), and solve each of the 14 test instances five times. Since the neighboring solutions are generated randomly at each tabu search iteration, our heuristics may yield a different solution for each run. Therefore, we report in Table 2 both the best and average results generated by five runs for each test instance. The first column includes the number of points N in the sensor field. The second column shows the amount of energy spent corresponding to the best feasible solution obtained within a time limit of 14,400 s when the CSLRP-1 is solved using CPLEX 11.2. CPLEX can provide optimal solutions only for problems with $N \leq 49$ points. The values marked with "*" are the corresponding optimal values. The remaining entries of this column are the objective values of the best feasible solutions obtained within the time limit. The third to fifth columns display the percent deviations of the best and average objective values obtained by TS-LH, TS-NDH, and TS-GH heuristics from the objective value found by solving CSLRP-1 using CPLEX. The last four columns give the CPU times spent by the methods. As can be seen by the best percent deviations, 6.9%, 6.3%, and 4.1% more energy is spent on the average when the WSN is setup by making

Table 2
Comparison of the results when $p=1$.

N	Energy	Best % dev. (avg. % dev.)			CPU time (s)			
	CSLRP-1	TS-LH	TS-NDH	TS-GH	CSLRP-1	TS-LH	TS-NDH	TS-GH
9	5*	0.0 (12.0)	0.0 (4.0)	0.0 (8.0)	0.1	9.8	0.9	0.1
16	11*	9.1 (16.4)	9.1 (14.5)	18.2 (21.8)	0.2	61.4	4.2	0.6
25	16*	0.0 (2.5)	0.0 (1.3)	0.0 (1.3)	1.2	399.3	77.8	24.2
36	30*	16.7 (24.0)	10.0 (13.3)	3.3 (7.3)	24.1	726.1	171.3	48.2
49	44*	22.7 (33.2)	15.9 (18.6)	22.7 (24.5)	901.4	886.4	382.5	89.5
64	75	6.7 (10.1)	6.7 (8.8)	2.7 (6.4)	14,400.0	1565.5	667.2	205.3
81	96	10.4 (12.9)	15.6 (16.7)	4.2 (5.6)	14,400.0	3945.9	720.5	130.9
100	132	6.8 (9.2)	6.1 (6.7)	4.5 (5.8)	14,400.0	6272.3	1217.9	238.1
121	177	10.7 (13.4)	13.6 (16.5)	7.3 (8.6)	14,400.0	8991.4	1802.5	384.6
144	225	12.0 (13.2)	14.7 (15.6)	10.2 (11.3)	14,400.0	12,016.2	4384.7	607.3
169	309	8.7 (9.7)	8.1 (9.0)	3.6 (4.3)	14,400.0	14,400.0	5979.4	990.8
196	440	−5.0 (−4.2)	−0.9 (0.6)	−7.7 (−7.2)	14,400.0	14,400.0	14,400.0	1378.1
225	532	0.2 (0.8)	−1.5 (−0.7)	−4.5 (−2.9)	14,400.0	14,400.0	14,400.0	2062.2
400	1750	−1.8 (−0.4)	−9.2 (−7.1)	−6.6 (−2.9)	14,400.0	14,400.0	14,400.0	8194.1
Avg.	274.4	6.9 (10.9)	6.3 (8.4)	4.1 (6.6)	9323.4	6605.3	4186.4	1025.3

the decisions of sensor deployment, sink location and routing using the three heuristics. TS-GH outperforms the other heuristics in terms of accuracy, while it is also the best in terms of efficiency.

The results for $p=2$ are displayed in Table 3. As is the case with $p=1$, CPLEX cannot find any optimal solution for test instances with $N>49$ within the allowed time limit. It can be observed that the best performing heuristic in terms of accuracy is TS-NDH with an average of 4.1% deviation, while TS-GH is once more the most efficient. Note that there are some negative values for percent deviations. They indicate that the heuristic solution is better than the best feasible solution CPLEX computes in 4 h.

Results given in Table 4 for $p=3$ indicate that the best performing heuristics with respect to both criteria remain the same. With regard to the efficiency, we can conclude that solving the CSLRP-1 by CPLEX requires 11.5 times as much time as needed by the fastest heuristic TS-GH, on the average.

When we compare the objective values for the same test instances (i.e., instances with the same number of grid points N), we observe that the energy consumptions decrease as the number of the sinks increases. This is an expected result because when there are more sinks in the WSN, each sensor is, on the average, closer to the sinks, and thus less routing energy is consumed by the sensors.

4.3. Sensitivity analyses

We conduct further experiments to investigate the effect of three factors on the results: the available budget, the coverage threshold, and the configuration of the candidate points in the sensor field. Each of these factors is considered separately in the following.

4.3.1. The effect of the budget on the routing energy

The available budget H allocated for sensor deployment is a significant factor in the WSN design. Note that in our experiments we set $H=\mu H_{\min}$ with $\mu=1.5$ where $H_{\min}$ is the minimum budget level that is required to obtain a sensor network in which all grid points are covered. To investigate the effect of the budget on the routing energy spent, we set μ to various values between 1.01 and 4.0, and solve the CSLRP-1 by CPLEX for five test problems from $N=9$ to $N=49$. Note that these are the instances that are solved to optimality by CPLEX at the default budget level as given in Tables 2–4. Fig. 1 displays at each budget level H the average routing energy spent in the WSN over the test problems for $p=1$, $p=2$, and $p=3$. As can be seen from the plots in Fig. 1, the routing

Table 3
Comparison of the results for $p=2$.

N	Energy	Best % dev. (avg. % dev.)			CPU time (s)			
	CSLRP-1	TS-LH	TS-NDH	TS-GH	CSLRP-1	TS-LH	TS-NDH	TS-GH
9	3*	0.0 (13.3)	0.0 (6.7)	0.0 (6.7)	0.1	20.3	0.7	0.1
16	6*	16.7 (30.0)	16.7 (26.7)	16.7 (26.7)	0.8	52.6	2.9	0.6
25	9*	11.1 (22.2)	0.0 (8.9)	11.1 (17.8)	5.3	735.3	47.0	24.4
36	18*	5.6 (12.2)	0.0 (7.8)	5.6 (10.0)	500.7	1299.8	177.1	58.2
49	25*	20.0 (23.2)	8.0 (10.4)	72.0 (76.8)	3995.3	1638.0	229.1	123.5
64	40	12.5 (16.0)	12.5 (16.5)	32.5 (43.0)	14,400.0	2613.5	453.7	232.2
81	55	30.9 (38.9)	20.0 (23.6)	32.7 (37.5)	14,400.0	3234.6	387.1	136.8
100	85	14.1 (17.6)	14.1 (16.9)	18.8 (22.1)	14,400.0	5255.4	606.8	245.9
121	138	1.4 (6.4)	−0.7 (4.5)	0.0 (3.5)	14,400.0	7655.3	991.9	408.3
144	171	11.1 (14.7)	10.5 (12.4)	11.7 (14.3)	14,400.0	10,660.2	1609.5	689.6
169	232	0.4 (3.0)	−2.6 (0.3)	1.3 (2.5)	14,400.0	14,400.0	2411.3	919.9
196	288	2.1 (4.7)	5.2 (7.7)	−0.3 (2.4)	14,400.0	14,400.0	3136.1	1273.4
225	392	−4.1 (−1.2)	−12.8 (−9.5)	−2.6 (1.3)	14,400.0	14,400.0	4571.1	1790.5
400	1010	−11.0 (−8.7)	−14.2 (−13.4)	−14.7 (−13.7)	14,400.0	14,400.0	14,400.0	7383.5
Avg.	176.6	7.9 (13.7)	4.1 (8.5)	13.2 (17.9)	9578.7	6483.2	2073.2	949.1

Table 4
Comparison of the results for $p=3$.

N	Energy	Best % dev. (avg. % dev.)			CPU time (s)			
	CSLRP-1	TS-LH	TS-NDH	TS-GH	CSLRP-1	TS-LH	TS-NDH	TS-GH
9	1*	0.0 (20.0)	0.0 (0.0)	0.0 (20.0)	0.1	13.2	0.9	0.2
16	4*	25.0 (35.0)	0.0 (10.0)	25.0 (30.0)	5.2	54.6	4.9	1.0
25	7*	0.0 (8.6)	0.0 (5.7)	14.3 (17.1)	28.4	471.3	55.4	30.8
36	12*	16.7 (23.3)	8.3 (15.0)	16.7 (26.7)	1968.9	1284.1	89.1	49.2
49	18*	16.7 (24.4)	11.1 (16.7)	11.1 (15.6)	13,985.6	1694.4	211.4	124.3
64	27	29.6 (36.3)	18.5 (23.0)	37.0 (44.4)	14,400.0	2393.4	394.1	199.7
81	39	23.1 (29.7)	17.9 (26.7)	23.1 (29.2)	14,400.0	2784.8	308.1	133.1
100	53	28.3 (36.2)	17.0 (23.0)	24.5 (30.2)	14,400.0	5140.1	531.5	242.7
121	81	9.9 (14.1)	6.2 (9.6)	11.1 (15.1)	14,400.0	8315.2	873.7	366.4
144	114	11.4 (16.5)	3.5 (7.4)	2.6 (6.8)	14,400.0	10,155.1	1374.5	581.2
169	161	−3.1 (−0.5)	−5.0 (−1.5)	−5.6 (−3.1)	14,400.0	14,400.0	1951.8	863.1
196	236	−6.8 (−4.2)	−11.0 (−7.5)	−10.6 (−7.7)	14,400.0	14,400.0	2952.3	1210.6
225	246	1.6 (6.7)	0.4 (3.8)	0.8 (5.4)	14,400.0	14,400.0	4382.6	1602.5
400	853	−18.1 (−14.2)	−26.1 (−19.3)	−24.6 (−20.4)	14,400.0	14,400.0	14,400.0	7356.8
Avg.	132.3	9.6 (16.6)	2.9 (8.0)	9.0 (15.0)	10,428.8	6421.9	1966.5	911.5

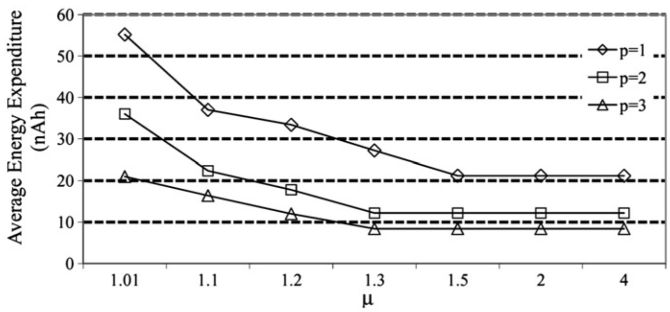

Fig. 1. The effect of the budget on the routing energy.

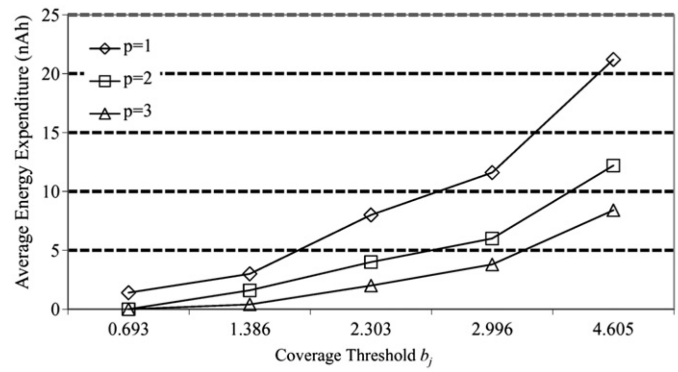

Fig. 2. The effect of the coverage threshold on the routing energy.

energy consumption first decreases as the budget level increases. This initial reduction in the energy is due to the increased number of deployed sensors. Note that a larger number of sensors can be deployed due to the higher budget levels. This, in turn, decreases the average distance between sensors and sinks, and a smaller average distance reduces the routing energy spent. Being able to deploy more sensors due to a higher budget availability does not contribute to a reduction in the routing energy after a certain point, however, because the number of data packets transmitted from sensors to sinks also increases. This latter factor leads to an increase in the routing energy. Hence, the routing energy attains its optimal value at a certain budget level, and remains at this value for larger budget levels.

4.3.2. The effect of the coverage threshold on the routing energy

The coverage threshold is commonly used as a measure that shows the quality of service (QoS) in a WSN. Recall that coverage threshold parameter b_j at point j and the maximum allowable miss probability t_j at point j are related to each other, which is given as $b_j = -\ln t_j$. For example, a value of $t_j = 0.01$ corresponds to $b_j = -\ln 0.01 = 4.605$. A smaller maximum allowable miss probability or equivalently a higher coverage threshold corresponds to a better QoS since each grid point in the sensor field is then covered at a larger intensity level. This implies, however, that a large number of sensors is required to guarantee a high quality of service in a WSN.

We perform experiments with various values of t_j ranging from 0.01 to 0.5, which corresponds to coverage threshold values b_j varying from 4.605 to 0.693. We solve the CSLRP-1 by CPLEX for the same five test problems used in the investigation of the effect of different budget levels. Fig. 2 shows at each coverage threshold level the average routing energy spent in the WSN for $p=1$, $p=2$, and $p=3$. We can see that the average routing energy increases as b_j increases. The reason lies in the fact that the level of data transmission and thus the routing energy increases as the number of deployed sensors in the WSN increases, which stems from the requirement of a higher coverage threshold. This graph can help a WSN designer in making a decision about the trade-off between the QoS required in the application and the consumed energy in the WSN and thus the lifetime of the WSN.

4.3.3. Randomly generated candidate points for sensors

The assumption of having grid points in a lattice structure as candidate points within the sensor field may be restrictive in some WSN applications. Therefore, it may be of interest to assess the quality of the solutions generated by the heuristics for the case where the candidate points of the sensors are distributed randomly within the sensor field. Since TS-NDH is the best performing heuristic in terms of accuracy, we solve the test problems only by this heuristic and compare the results with those obtained by solving CSLRP-1 using CPLEX.

Table 5
Accuracy of TS-NDH with random points.

N	p=1		p=2		p=3	
	CSLRP-1	TS-NDH (%)	CSLRP-1	TS-NDH (%)	CSLRP-1	TS-NDH (%)
9	143.5	0.0	17.6	14.2	10.1	22.8
16	747.9	0.1	93.1	1.1	51.3	5.5
25	3075.4	0.0	229.7	0.0	147.5	0.5
36	8925.3	0.0	363.3	0.1	223.2	0.0
49	19,063.9	12.1	641.9	1.6	405.9	4.1
Avg.		2.4		3.4		6.6

Table 6
Efficiency TS-NDH with random points.

N	p=1		p=2		p=3	
	CSLRP-1	TS-NDH	CSLRP-1	TS-NDH	CSLRP-1	TS-NDH
9	0.4	2.2	0.2	2.1	0.3	2.8
16	30.2	9.3	45.4	6.8	149.8	12.6
25	285.7	111.8	369.8	91.7	2087.7	151.2
36	755.5	317.2	287.1	255.4	6287.9	312.1
49	14,400	754.8	14,400	543.6	14,400	749.9
Avg.	3094.4	239.1	3020.5	179.9	4585.1	245.7

Table 5 contains the energy consumptions obtained by CPLEX and the percent deviations of the objective values found by TS-NDH from those of CPLEX for five test problems where the number of candidate points N varies between 9 and 49. Regarding the solution quality of our hybrid solution method TS-NDH, we can say that it performs quite satisfactory. The best percent deviations are 2.4%, 3.4%, and 6.6% for $p=1$, $p=2$, and $p=3$, respectively. The comparison of the CPU performances of CPLEX and TS-NDH is displayed in Table 6. As can be seen, TS-NDH is approximately 13, 17, and 19 times faster than CPLEX on the average for $p=1$, $p=2$, and $p=3$, respectively.

Notice that when the points in the sensor field are randomly distributed, the energy spent in the WSN increases considerably for all three values of p. This increase can be observed in particular for larger network sizes. For example, the ratio of the energy spent when candidate points are distributed randomly to the energy consumed in the case of grid structured candidate points is given in Table 7. A final analysis can be made regarding the average solution times when candidate sensor points are grid points versus randomly generated points. When we compute the average solution times of the five test instances with a number points $9 \le N \le 49$ and $p = 1,2,3$ obtained by CPLEX and TS-NDH using the values in

Table 7
Increase in the consumed energy for random points.

N	E_{random}/E_{grid}		
	$p=1$	$p=2$	$p=3$
9	29	6	10
16	68	16	13
25	192	26	21
36	298	20	19
49	433	26	23

Table 8
CPU time comparison between grid and random points.

Method	$p=1$		$p=2$		$p=3$	
	Grid	Random	Grid	Random	Grid	Random
CSLRP-1	185.4	3094.4	900.4	3020.5	3197.6	4585.1
TS-NDH	127.3	239.1	91.4	179.9	72.3	245.7

Tables 2–4 and compare them with those in Table 6, we conclude that the CPU times attained by CPLEX and TS-NDH increase in the case of randomly generated points as shown in Table 8.

5. Conclusion

In this work, we study the joint optimization of point coverage, sink location, and data routing problems in wireless sensor networks. Two new MILP formulations are developed for the integrated model. The first one involves data routing variables defined on the arcs of the network, while the second one uses data routing variables based on sensor-to-sink assignments. Since the solution of these formulations by means of commercial MILP solvers is inefficient for medium and large-sized problems, we develop a hybrid solution procedure based on TS metaheuristic using the assignment formulation. The outer loop of the hybrid solution procedure involves determining the best sensor configuration by TS, and the inner loop calls for solving the remaining sink location and routing problem for each sensor configuration fixed in the outer loop.

We propose three heuristic strategies, TS-LH, TS-NDH, and TS-GH, which differ in the inner loop that involves the solution of the sink location and routing problem as a p-median problem. Experimental results show that this approach can provide close-to-optimal solutions within much less time than solving the more efficient one of the two formulations by CPLEX. We also conduct a sensitivity analysis with respect to three issues and determine their effects on the total routing energy and computational performance. These are random distribution of the candidate points of the sensors, the available budget for sensor deployment, and the coverage threshold. The results can be utilized by a wireless sensor network designer while assessing the trade-off between the total energy expenditure (or network lifetime) and the number of deployed sensors.

Acknowledgments

This research has been supported by TÜBİTAK (The Scientific and Technological Research Council of Turkey) under Grant no. 107M250 and by Boğaziçi University Research Fund under Grant no. 05HA303.

Appendix A. Pseudocode of the tabu search heuristics

```
1.  Initialize the parameters and find a starting solution;
2.  While(num_iter < max_iter)
    and(num_nonimp_iter < max_nonimp_iter) do
3.    Obj_Best_Neigh := 0.
4.    Foreach move type i do
5.      Calculate size_neigh_i and set num_neigh := 0.
6.      While num_neigh < size_neigh_i do
7.        Generate a new neighbor by move i.
8.        Calculate Obj for this new solution by solving the
          SLRP using LH, NDH, or GH.
9.        Record this new solution not to regenerate it.
10.         If Obj > Obj_Best_Neigh, do
11.           Check if the new solution is tabu-active.
12.           Ifit is not tabu-active or if Obj > Obj*, then
13.             Set Obj_Best_Neigh := Obj
14.             If Obj > Obj*, then
15.               Set this neighbor as the incumbent
                  Update Obj* := Obj
                  Set num_nonimp_iter := 0.
16.             End If
17.           End If
18.         End If
19.       num_neigh := num_neigh + 1.
20.     End While
21.   End For
22. Add the best neighboring solution to the tabu list and set
    it as the current solution. Generate a random integer
    number in the interval [1, max_tabu_tenure] and set it as
    the current solution's tabu tenure. Record num_iter as the
    start time of the current solution's tabu status.
23. num_nonimp_iter := num_nonimp_iter + 1,
    num_iter := num_iter + 1.
24. End While
```

References

[1] Akyıldız IF, Melodia T, Chowdhury KR. A survey on wireless multimedia sensor networks. Computer Networks 2007;51:921–60.
[2] Altınel İK, Aras N, Güney E, Ersoy C. Binary integer programming formulations and heuristics for differentiated coverage in heterogeneous sensor networks. Computer Networks 2008;52:2419–31.
[3] Bogdanov A, Maneva E, Riesenfeld S. Power-aware base station positioning for sensor networks. in: Proceedings of the 23rd annual joint conference of the IEEE computer and communications conference (INFOCOM 2004); 2004. p. 575–85.
[4] Brooks RR, Iyengar SS. Multi-sensor fusion: fundamentals and applications with software. Upper Saddle River, NJ: Prentice Hall PTR; 1998.
[5] Callaway Jr EH. Wireless sensor networks: architectures and protocols. Boca Raton: Auerbach Publications; 2004.
[6] Cardei M, Wu J. Handbook of sensor networks, coverage in wireless sensor networks. CRC Press; 2004.
[7] Cardei M, Thai MT, Li Y, Wu W. Energy-efficient target coverage in wireless sensor networks. in: Proceedings of the 24th annual joint conference of the IEEE computer and communication societies (INFOCOM), vol. 3; 2005. p. 1976–84.
[8] Chakrabarty K, Iyengar SS, Qi H, Cho E. Grid coverage for surveillance and target location in distributed sensor networks. IEEE Transactions on Computers 2002;51:1448–53.
[9] Dhillon SS, Chakrabarty K, Iyengar SS. Sensor placement for grid coverage under imprecise detections. in: Proceedings of the international conference on information fusion; 2002. p. 1581–7.
[10] Dhillon SS, Chakrabarty K. Sensor placement for effective coverage and surveillance in distributed sensor networks. in: Proceedings of the IEEE wireless communications and networking conference; 2003. p. 1609–14.
[11] Erlenkotter D. A dual-based procedure for uncapacitated facility location. Operations Research 1978;26:992–1009.
[12] Floyd RW. Algorithm 97 shortest path. Communication of the ACM 1962;6:345.
[13] Gandam SR, Dawande M, Prakash R, Venkatesan S. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. in: Proceedings of the IEEE global telecommunications conference (GLOBECOM'03); 2003. p. 377–81.

[14] Ghosh A, Das SK. Coverage and connectivity issues in wireless sensor networks: a survey. Pervasive and Mobile Computing 2008;4:303–34.
[15] Glover F, Laguna M. Tabu search. Dordrecht: Kluwer Academic Publishers; 1997.
[16] Güney E, Altınel İK, Aras N, Ersoy C. A tabu search heuristic for point coverage, sink location, and data routing in wireless sensor networks. in: EvoCOP 2010, Lecture notes in computer science, vol. 6022; 2010. p. 83–94.
[17] Güney E, Aras N, Altınel İK, Ersoy C. Efficient integer programming formulations for optimum sink location and routing in heterogeneous wireless sensor networks. Computer Networks 2010;54(6):1805–22.
[18] Kim H, Seok Y, Choi N, Choi Y, Kwon T. Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks. in: ICOIN 2005. Lecture notes in computer science, vol. 3391; 2005. p. 264–75.
[19] Kuehn AA, Hamburger MJ. A heuristic program for locating warehouses. Management Science 1963;9:643–66.
[20] Lin L, Shroff NB, Srikant R. Energy-aware routing in sensor networks: a large system approach. Ad Hoc Networks 2007;5:818–31.
[21] Mainwaring A, Polastre J, Szewczyk R, Culler S, Anderson J. Wireless sensor networks for habitat monitoring. In: WSNA'02, Atlanta, Georgia; 2002.
[22] Meguerdichian S, Potkonjak M. Low power 0/1 coverage and scheduling techniques in sensor networks. UCLA Technical Reports 030001. Los Angeles, USA: UCLA; 2003.
[23] Meguerdichian S, Koushanfar F, Potkonjak M, Srivastava MB. Coverage problems in wireless ad-hoc sensor networks. in: Proceedings of INFOCOM 2001, twentieth annual joint conference of the IEEE computer and communications societies, vol. 3; 2001. p. 1380–7.
[24] Mirchandani PB, Oudjit A, Wong RT. 'Multidimensional' extensions and a nested dual approach for the *m*-median problem. European Journal of Operational Research 1985;21:121–37.
[25] Shi Y, Hou YT, Efrat A. Algorithm design for a class of base station location problems in sensor networks. Wireless Networks 2009;15:21–8.
[26] Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. International Journal of Computer and Telecommunications Networking 2008;52:2292–330.
[27] Younis M, Akkaya K. Strategies and techniques for node placement in wireless sensor networks: a survey. Ad Hoc Networks 2008;6:621–55.