R. Cerulli, R. De Donato, A.Raiconi, Exact and Heuristic Methods to Maximize Network Lifetime in Wireless Sensor Networks with Adjustable Sensing Ranges, European Journal of Operational Research, Volume 220, Issue 1, pp. 58-66, 2012.

https://dx.doi.org/10.1016/j.ejor.2012.01.046

Exact and Heuristic Methods to Maximize Network Lifetime in Wireless Sensor Networks with Adjustable Sensing Ranges

R. Cerulli[a], R. De Donato[a], A. Raiconi[a]

[a]*Department of Mathematics, University of Salerno, Via Ponte Don Melillo, 84084 Fisciano (SA) Italy*

Abstract

Wireless sensor networks involve many different real-world contexts, such as monitoring and control tasks for traffic, surveillance, military and environmental applications, among others. Usually, these applications consider the use of a large number of low-cost sensing devices to monitor the activities occurring in a certain set of target locations. We want to individuate a set of *covers* (that is, subsets of sensors that can cover the whole set of targets) and appropriate activation times for each of them in order to maximize the total amount of time in which the monitoring activity can be performed (*network lifetime*), under the constraint given by the limited power of the battery contained in each sensor. A variant of this problem considers that each sensor can be activated in a certain number of alternative power levels, which determine different sensing ranges and power consumptions. We present some heuristic approaches and an exact approach based on the Column Generation technique. An extensive experimental phase proves the advantage in terms of solution quality of using adjustable sensing ranges with respect to the classical single range scheme.

Keywords: Integer programming, Heuristics, Column Generation, Wireless Sensor Networks

1. Introduction

Wireless Sensor Networks have met a growing interest in the last years due to their applications in a wide range of contexts, such as national security, traffic, military, health care and environmental monitoring, among others (see for example [7],[10],[12])

A common scenario in these applications considers the deployment of a large quantity of low-cost,

Email addresses: raffaele@unisa.it (R. Cerulli), araiconi@unisa.it (A. Raiconi)

limited sensing devices (or simply *sensors*), often randomly disposed all over the geographical region of interest, in situations where an accurate individual placement of each device is not possible. Each sensing has a range, which can be fixed or adjustable, and therefore sensors are able to collect information about certain subregions of the whole space (for example, all the points whose Euclidean distance from the sensor is equal or less than a certain threshold). The information collected about the targets can be shared either between the sensors or communicating with a central station and, therefore, they can be coordinated to collectively perform a complex sensing task. We are generally interested in covering either the whole region of interest (*area coverage* problems) or specific targets inside of it (*target coverage*). However, it was shown in [2] that every possible area coverage problem can be transformed into an equivalent target coverage problem in polynomial time. For this reason, in this paper we only take into account target coverage.

Due to both size and cost constraints, each sensing device has a limited amount of battery life. Sensors can generally be in different states (such as transmit, receive, idle or sleep), however we may focus on *active* and *sleep* states, which model whether a given sensor is performing its sensing activity (and therefore consuming its battery) or not. If the sensing ranges are adjustable, different energy consumptions are likely to be required with respect to the size of each range. In this context, as in [5], we consider a finite number of alternative *power levels* and associate a measure of battery consumption with each of them.

Indeed, a clever use of the sensors can effectively increase the *sensor network lifetime* (or simply *network lifetime*), that is, the amount of time in which the monitoring activity can be performed. Since we generally have a large number of sensors, and their sensing ranges may overlap, we can find different *covers* (that is, subsets of sensors which together cover all the targets) and keep active just one cover at a time. The problem has been extensively studied in the literature in the case in which sensors have a single power level (i.e. sensing ranges are not adjustable), and is known as the *Maximum Sensor Network Lifetime Problem (MLP)*.

Consider the example network in figure 1, where there are five targets (namely t_1, t_2, t_3, t_4 and t_5) and three sensors (s_1, s_2, s_3). For each sensor we consider a single power level and its sensing area is

shown. For example, sensor s_2 covers t_1, t_2 and t_5, and the possible covers for the whole set of targets are $\{s_1, s_2\}$, $\{s_1, s_3\}$ and $\{s_2, s_3\}$.

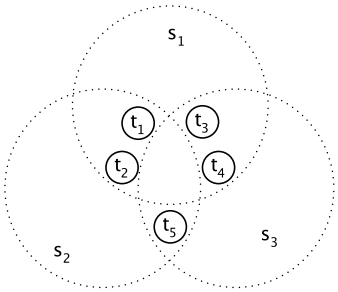

Figure 1: Example network with 5 targets and 3 sensors.

Let us consider the classical assumption that the battery of each sensor is able to keep it active for 1 unit of time. By considering one of the aforementioned covers, e.g. $\{s_2, s_3\}$, and activating it for the whole battery life of the sensor, we can monitor all the targets for 1 unit of time. Further extensions of the network lifetime are not possible, since only s_1 has residual lifetime and it does not cover all the targets alone. If we instead consider the three covers $\{s_1, s_2\}$, $\{s_1, s_3\}$, $\{s_2, s_3\}$, and activate each of them for 0.5 units of time, the network lifetime is equal to 1.5 units of time, and therefore this turns out to be a better strategy. MLP was proved to be NP-complete by reduction from the 3-SAT problem in [4]. Different solution approaches were proposed to solve it either exactly (see [13]) or approximately ([2],[3],[4],[16]). Variants of the problem consider covers which may neglect some of the targets (Minimum Coverage Breach ([6],[20]), Maximum Network $\alpha-$Lifetime [11]).

Other works consider both coverage and data routing issues. In [1], [15] and [23], by considering communication links among sensors which are close enough to communicate, the routing problem is faced by requiring connected covers. A special root node (also called sink or gateway in different works) is considered as the destination of the collected data. In [23], the authors describe an energy consumption model that takes into account the different roles of the sensors (relay, source or both) in the cover, and propose a greedy heuristic and an approximation algorithm. Both [1] and [15] describe exact approaches based on column generation, differing in the definition of the subproblem that results from a different

energy consumption model associated with each sensor. Moreover, the authors of [1] present a heuristic aiming at a distributed implementation, while in [15] a GRASP metaheuristic is proposed. The authors of [17], [18] and [19] define a problem that integrates sensor activity scheduling, data routing and sensor and sink placement. The authors propose different approaches, namely a Lagrangean and a two-stage heuristic [17], a heuristic based on the individuation of disjoint sets of sensors [18] and a column generation based one [19].

Considering different power levels has potential to further increase the network lifetime, since it increases the number of feasible covers that might be included in the solution. Depending on the specific instance, trade-offs among target coverage and battery consumption determine the optimal power level in which each used sensor should be activated, or even different power levels for the same sensor in different covers. Consider the example network in Figure 2, with four targets, four sensors and two power levels. Subfigures 2-A and 2-B show the sensing ranges of each sensor when set at level 1 and 2, respectively (by (s_i, a) we refer to sensor s_i when activated at level a). Let the batteries be able to keep sensors active for 1 unit of time at power level 1 and 0.5 units at power level 2.

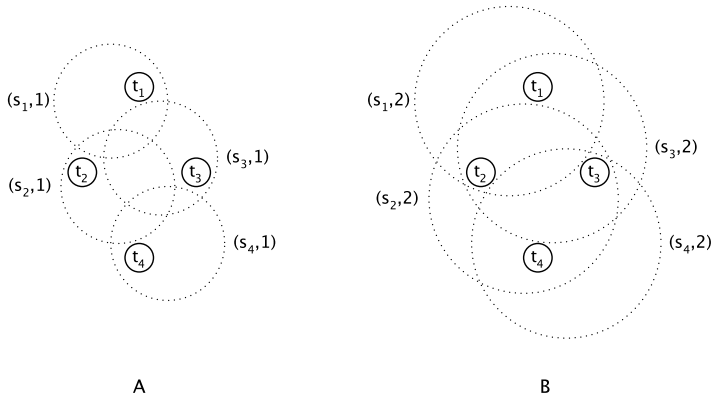

Figure 2: Example network with 4 targets, 4 sensors and 2 power levels.

Should only power level 1 be considered, there would be a single feasible cover, that is $\{(s_1, 1), (s_2, 1), (s_3, 1), (s_4, 1)\}$, with a total network lifetime of 1. By using only power level 2, we have a wider set of covers but the maximum achievable network lifetime is still equal to 1 (consider, for example, $\{(s_1, 2), (s_4, 2)\}$ and

$\{(s_2, 2), (s_3, 2)\}$ activated for 0.5 units of time each). Now let us consider covers containing sensors activated at different power levels. By using covers $\{(s_1, 1), (s_2, 2)\}$, $\{(s_4, 1), (s_3, 2)\}$ and $\{(s_1, 2), (s_4, 2)\}$ activated for 0.5, 0.5, 0.25 units of time respectively, we achieve a network lifetime of 1.25.

We define this variant of MLP as the *Maximum Network Lifetime with Adjustable Ranges Problem (MLARP)*. In [5], the authors address the problem of maximizing the number of covers, called Adjustable Range Set Covers (AR-SC). They present some heuristic solution approaches, based on both greedy and LP relaxation methods. In [8], the aim is to maximize the network lifetime while allowing smooth sensing range variations, and an approximation algorithm is proposed. In the same context, two distributed heuristics are presented in [9]. In [14], [21] and [22] the authors present models for the area coverage network lifetime problem with adjustable sensing ranges.

In our work we present an exact method based on the Delayed Column Generation Technique, a greedy heuristic and a local search procedure. The paper is organized as follows. Sections 2 and 3 formally introduce the required notation and MLARP, as well as its mathematical formulation. The formulation has been embedded in a Column Generation procedure, as described in Section 4. Section 5 presents our heuristic procedures. Section 6 describes a procedure to evaluate upper bounds on the solution value. The results of our extensive experimental tests are presented in Section 7. Finally, Section 8 contains some final remarks.

2. Notation

Let $N = (T, S)$ be a wireless sensor network, where $T=\{t_1, \ldots, t_n\}$ is the set of the target nodes and $S=\{s_1, \ldots, s_m\}$ is the set of the sensors, and let $k \geq 1$ be a positive integer value. We assume that each sensor can be activated at k alternative power levels. For each sensor s_i and for each value a between 1 and k, we will refer to sensor s_i activated at level a with (s_i, a); we will also define such a pair an *adjusted sensor*. Moreover, let $T_{(s_i,a)}$ be the subset of T containing all the targets covered by s_i when it is set at level a. The positions of targets and nodes do not change over time, therefore we can assume each $T_{(s_i,a)}$ to be known in advance. Since the power levels gradually extend the sensing ranges of the devices, for each sensor s_i and each level $a > 1$ we have $T_{(s_i,b)} \subseteq T_{(s_i,a)}$ $\forall b \in \{1, \ldots, a-1\}$. Moreover,

we define the adjusted sensor (s_i, a) *minimal* for target t_j if $t_j \in T_{(s_i,a)}$ and either $a = 1$ or $t_j \notin T_{(s_i,b)}$ $\forall b \in \{1, \ldots, a-1\}$. For example, in the network in Figure 2, $T_{(s_1,1)} = \{t_1\} \subseteq T_{(s_1,2)} = \{t_1, t_2\}$, $(s_1, 1)$ is minimal for t_1 and $(s_1, 2)$ is minimal for t_2.

Given a collection of pairs $C_l = \{(s_i, a) | s_i \in S, a \in 1, \ldots, k\}$, we define the set of targets covered by C_l as $T_{C_l} = \bigcup_{(s_i,a)\in C_l} T_{(s_i,a)}$. If C_l is such that $T_{C_l} \equiv T$ and contains at most one adjusted sensor (s_i, a) for each $s_i \in S$, we define it a *cover*; this condition is required since as already said a cover represents a subset of sensors that can be used to monitor the whole set of targets when activated at the same time. If a level switch is desired for one or more sensors belonging to the cover, it can be modeled with a different cover. Considering the example in Figure 2, we already introduced in the previous section some feasible covers, such as $\{(s_1, 2), (s_2, 2)\}$.

It is realistic to assume that higher power levels increase the consumption of energy. We assume that each device has the same hardware and, therefore, they have the same battery power and the same battery consumption for each level. In order to model the different battery consumptions, we define a positive parameter Δ^a for each power level a, which represents the ratio between battery consumption at level a and level 1 (which is the least powerful and therefore the least expensive level). For example, $\Delta^a = 2$ means that level a consumes twice the energy of level 1. It is straightforward that $\Delta^1 = 1$. We also normalize the total battery power on the energy consumption of level 1; that is, the battery of a sensor allows to keep it activated for 1 time unit if it is always set at level 1.

3. Problem Definition, Complexity and Mathematical Formulation

The Maximum Network Lifetime with Adjustable Ranges Problem is defined as follows:

Maximum Network Lifetime with Adjustable Ranges Problem (MLARP)

Find a collection of pairs (C_l, w_l), $l = 1, \ldots, \ell$, where $C_1, \ldots, C_\ell$ is the family of all the feasible covers and $w_1, \ldots, w_\ell \geq 0$ are the corresponding activation times, such that the sum of all the activation times (that is the network lifetime) $\sum_{l=1}^{\ell} w_l$ is maximized, and the power consumption of each sensor does not exceed its battery.

The problem is NP-Hard. Indeed, MLP is a special case of MLARP when $k = 1$.

For each sensor s_i, level a and cover C_l, let Φ_{il}^a be a binary parameter such that $\Phi_{il}^a = 1$ if (s_i, a) belongs to C_l, 0 otherwise. We can model the problem as follows.

$$\textbf{[MOD]} \qquad \max \sum_{l=1}^{\ell} w_l \tag{1}$$

$$\text{s.}t.$$

$$\sum_{l=1}^{\ell} \sum_{a=1}^{k} \Phi_{il}^a \Delta^a w_l \leq 1 \qquad \forall i = 1, \ldots, m \tag{2}$$

$$w_l \geq 0 \qquad \forall l = 1, \ldots, \ell \tag{3}$$

Objective function (1) maximizes the sum of the activation times of the covers, and, therefore, the network lifetime. Constraints (2) check that the total power consumption of each sensor does not exceed its battery lifetime.

The total number of feasible covers ℓ is potentially exponential; therefore, we decided to embed this model in a Column Generation approach in order to solve it optimally, as described in the next section.

4. Column Generation Approach

Our Column Generation approach is a variant of the method proposed in [13] for the classic MLP problem. Another variant of this method was presented in [15] to solve the Connected Maximum Network Lifetime Problem.

The Delayed Column Generation technique, or simply Column Generation (CG), is an efficient way to solve linear programming formulations when there is a huge set of variables and we can not therefore consider all of them explicitly. Since most of them will be nonbasic and assume a value of zero in the optimal solution, the method aims at generating only variables which have potential to improve the objective function, while the others are implicitly discarded.

The general iteration of the Column Generation considers a primal problem restricted only to a subset of variables (*Restricted Master*) and optimally solves it. In order to determine whether the returned solution is optimal for the entire problem, one should compute all the reduced costs of the nonbasic

variables and, if the optimality conditions are not satisfied, a new variable (column) should enter the basis. In the Column Generation approach, to perform these tasks an additional problem is solved (the *Separation Problem*), whose solution either returns a new column to be added to the restricted primal or verifies the optimality of the current solution.

Let us consider our previously presented [**MOD**] formulation for MLARP, restricted to a subset of p feasible covers. Let π_i, $i = 1, \ldots, m$, be the set of dual optimal multipliers associated with the primal constraints (that is, with the sensors). The current primal solution is optimal if there is no negative reduced cost associated with the nonbasic variables; that is, if for each l corresponding to a nonbasic variable w_l we have $\sum_{(i,a):(s_i,a)\in C_l} \Delta^a \pi_i - c_l \geq 0$, where c_l is the coefficient of variable w_l in the objective function (1) of the primal problem. We compute the minimum among all the reduced costs (note that $c_l = 1 \ \forall \ l$ and therefore can be excluded from the reduced costs computation). In order to do that we solve the following separation problem:

$$\textbf{[SEP]} \qquad \min \sum_{a=1}^{k} \Delta^a \sum_{i=1}^{m} \pi_i x_i^a \tag{4}$$

$$\text{s.}t.$$

$$\sum_{i=1}^{m} \sum_{a=1}^{k} \phi_{ji}^a x_i^a \geq 1 \qquad \forall j = 1, \ldots, n \tag{5}$$

$$\sum_{a=1}^{k} x_i^a \leq 1 \qquad \forall i = 1, \ldots, m \tag{6}$$

$$x_i^a \in \{0, 1\} \qquad \forall i = 1, \ldots, m; a = 1, \ldots, k \tag{7}$$

where, for each sensor s_i, power level a and target t_j:

- x_i^a is a binary variable determining whether (s_i, a) belongs to the new cover;
- ϕ_{ji}^a is a binary parameter that is equal to 1 if t_j is covered by (s_i, a).

Objective function (4) ensures that the returned cover has the minimum reduced cost. Constraints (5) make sure that each target is covered by at least one adjusted sensor. Constraints (6) impose the selection of at most one power level for each sensor.

If the optimal objective function of **[SEP]** is ≥ 1, the solution that was found by the restricted primal in the previous iteration is optimal for the whole problem, otherwise the column defined by the optimal solution values of variables x_i^a is introduced in the restricted primal and the algorithm iterates.

It is easy to check that there always exists an optimal solution such that there are not two covers C_a and C_b such that C_a is a proper subset of C_b and $w_b > 0$. Therefore, we added the following set of constraints to **[SEP]**. Let $\{C_1, C_2, \ldots, C_g\}$ be the set of connected covers generated by the algorithm so far:

$$\sum_{i=1}^{m}\sum_{a=1}^{k} \Phi_{il}^a x_i^a \leq \sum_{i=1}^{m}\sum_{a=1}^{k} \Phi_{il}^a - 1 \qquad \forall l = 1, \ldots, g \tag{8}$$

The above presented inequalities ensure that each new cover returned by the separation problem differs from the already generated ones in at least one adjusted sensor.

During our experimentation phase, whose results are presented and commented in Section 7, we initialized the CG procedure using heuristic solutions provided by the AR-Iterative algorithm described in Section 5.2.

5. Heuristic Approaches

5.1. Adjustable Ranges Greedy (AR-Greedy)

In this section we present a greedy heuristic which shares some ideas of Centralized Greedy Algorithm presented in [5], bringing many refinements related to our specific problem.

The AR-Greedy algorithm builds one cover at a time, and assigns appropriate activation times within a given upper bound to each of them in order to keep the solution feasible. Each cover, starting from an empty set, is gradually extended and finally completed by iteratively identifying specific targets that have not been covered so far (called *critical targets*) and then, adding to the cover the adjusted sensors with the *best contributions* to cover them. The algorithm ends when the residual lifetimes of the sensors do not allow the generation of a new cover.

The critical target selection phase aims to identify the target in the most unfortunate position of the network, that is, the one whose covering sensors have the least amount of residual energy. Three different

criteria are used to determine the contribution of each adjusted sensor (s_i, a), whose underlying ideas are to evaluate i) the trade-off among the number of new covered targets and consumption ratio Δ^a, ii) the percentage of sensors that would be redundantly covered more than once if (s_i, a) would be added to the current cover, and iii) the overall amount of its residual energy. These concepts are better clarified in Sections 5.1.1 and 5.1.2.

Algorithm 1 describes the procedure. In the following, we comment on this pseudocode and introduce some notations used in Sections 5.1.1 and 5.1.2. Line 1 contains the input parameters. Granularity factor $gf \in (0, 1]$ represents a maximum amount of activation time that will be assigned to each generated cover. The Γ vector is used to weight the different sensor contribution criteria, as explained in Section 5.1.2. The S_R set initialized in line 2 contains the list of sensors with a residual lifetime greater than 0. Parameters r_{s_i} initialized in lines 3-5 represent the amount of residual lifetime for each sensor s_i. The set SOL and the value lt initialized in lines 6-7 will contain the covers with related activation times composing the returned solution and the overall maximum lifetime found, respectively. Line 8 checks whether the sensors with residual positive lifetime can still cover the whole set of targets and therefore produce a new cover C_l. New covers are generated according to lines 9-28. The T_U and the S_I sets initialized in lines 10-11 keep track of the uncovered targets and of the sensors that have already been included in C_l, respectively. The next *critical* target in T_U as well as the appropriate adjusted sensor with the *greatest contribution* are iteratively selected in the loop in lines 12-20, until C_l covers all the targets. Lines 21-27 decrease the lifetime of each sensor of the cover by the maximum feasible activation time which does not exceed gf and check whether the S_R set must be updated. In more detail, C_l will be activated for $w_l = gf$ if $r_{s_i} - \Delta^a gf \geq 0$ for each $(s_i, a) \in C_l$. Otherwise, consider the adjusted sensor of C_l that minimizes $\frac{r_{s_i}}{\Delta^a}$; let us call it (s_h, b). We set $w_l = \frac{r_{s_h}}{\Delta^b}$; this guarantees a feasible activation time for each $(s_i, a) \in C_l$.

The newly generated cover and its activation time are added to the solution in line 28, and the network lifetime is updated in line 29. Finally line 31 returns the resulting set of covers and activation times.

Algorithm 1 AR-Greedy algorithm

input: wireless network $N = (T, S)$, number of power levels k, granularity factor $gf \in (0, 1]$, criteria weighting parameter $\Gamma = (\gamma_1, \gamma_2, \gamma_3)$, $\gamma_i \geq 0$, $\gamma_1 + \gamma_2 + \gamma_3 = 1$
$S_R \leftarrow S$
for each $s_i \in S_R$ **do**
  $r_{s_i} \leftarrow 1$
end for
$SOL \leftarrow \varnothing$
$lt \leftarrow 0$
while $\bigcup_{s_i \in S_R} T_{(s_i,k)} \equiv T$ **do**
  Create a new empty cover C_l
  $T_U \leftarrow T$
  $S_I \leftarrow \varnothing$
  while $T_U \not\equiv \varnothing$ **do**
    Find a **critical target** $t_c \in T_U$
    Select $s_c \in S_R \backslash S_I$ and $a \in \{1, \ldots, k\}$ s.t. $t_c \in T_{(s_c,a)}$ and (s_c, a) has the **maximum contribution** according to Γ
    $S_I \leftarrow S_I \cup \{s_c\}$
    for each $t_j \in T_U$ s.t. $t_j \in T_{(s_c,a)}$ **do**
      $T_U \leftarrow T_U \setminus \{t_j\}$
    end for
    $C_l \leftarrow C_l \cup \{(s_c, a)\}$
  end while
  $w_l =$ **max feasible activation time** $\leq gf$ for C_l
  for each $(s_i, a) \in C_l$ **do**
    $r_{s_i} \leftarrow r_{s_i} - (\Delta^a w_l)$
    if $r_{s_i} = 0$ **then**
      $S_R \leftarrow S_R \setminus \{s_i\}$
    end if
  end for
  $SOL \leftarrow SOL \cup \{(C_l, w_l)\}$
  $lt \leftarrow lt + w_l$
end while
return (SOL, lt)

5.1.1. Critical Target

At each iteration, in order to determine the critical target, we evaluate an upper bound U_{t_j} on the amount of time for which each target t_j can be covered using the residual lifetime of the sensors; the critical target will be the one with the minimal upper bound. Ties are broken randomly.

In more detail, for each target $t_j \in T_U$ and each sensor $s_i \in S_R$ such that $t_j \in T_{(s_i,k)}$, let a_{ij} be the power level such that (s_i, a_{ij}) is minimal for t_j. That is, for each covering sensor we consider the power level with the lowest possible consumption level, since it maximizes the covering time. We define t_c as follows:

$$t_c = \underset{t_j \in T_U}{\operatorname{argmin}}(U_{t_j}) \tag{9}$$

where

$$U_{t_j} = \sum_{s_i \in S_R | t_j \in T_{(s_i,k)}} \frac{r_{s_i}}{\Delta^{a_{ij}}} \tag{10}$$

Suppose that the critical target has to be selected among t_1, t_2 and t_3 in the example given in Figure 3. The minimal adjusted sensors are $\{(s_1,1),(s_2,1)\}$ for t_1, $\{(s_1,2),(s_2,1)\}$ for t_2 and $\{(s_1,2),(s_2,2)\}$ for t_3. Let us suppose that $\Delta^2 = 2$ (recall that by definition $\Delta^1 = 1$), $r_{s_1} = 1$ and $r_{s_2} = 0.25$. Then we have $U_{t_1} = \frac{r_{s_1}+r_{s_2}}{\Delta^1} = 1.25$, $U_{t_2} = \frac{r_{s_1}}{\Delta^2} + \frac{r_{s_2}}{\Delta^1} = 0.75$, $U_{t_3} = \frac{r_{s_1}+r_{s_2}}{\Delta^2} = 0.625$ and t_3 is the critical target.

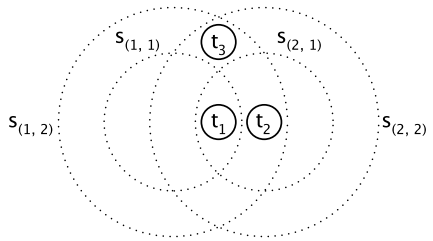

Figure 3: Example network with 3 targets, 2 sensors and 2 power levels.

5.1.2. Adjusted Sensors Contribution

The contribution of covering adjusted sensors is determined using 3 criteria: **Covering Power (CP)**, **Covering Waste (CW)** and **Residual Lifetime (RL)**. Each of these criteria returns a score for each candidate adjusted sensor, which are then combined to evaluate its overall contribution.

Covering Power. During the generation of a new cover C_l, for each adjusted sensor (s_i, a) with $s_i \in S_R \backslash S_I$ that can cover the critical target t_c, the CP score is the ratio among the total number of covered targets that still have to be covered in C_l and consumption ratio Δ^a; that is,

$$CP(s_i,a) = \frac{|T_{(s_i,a)} \bigcap T_U|}{\Delta^a} \quad \forall (s_i,a) | s_i \in S_R \setminus S_I, a \in \{1,\ldots,k\}, t_c \in T_{(s_i,a)} \tag{11}$$

The greatest contribution according to this criterion is determined by the maximum CP score; it favors sensors with relevant covering capabilities, penalizing high power levels if they do not bring significant improvements.

Let us evaluate the CP score of the four adjusted sensors in Figure 4. Suppose that the black targets have been previously covered by other sensors, that is $T_U = \{t_3, t_4\}$, and let $\Delta^2 = 2$. The CP score of the four adjusted sensors is $CP(s_1, 1) = \frac{|\emptyset|}{1} = 0$, $CP(s_2, 2) = \frac{|\{t_3\}|}{2} = 0.5$, $CP(s_2, 1) = \frac{|\{t_3, t_4\}|}{1} = 2$ and $CP(s_2, 2) = \frac{|\{t_3, t_4\}|}{2} = 1$. While $(s_2, 1)$ and $(s_2, 2)$ cover the same amount of new targets, it is easy to understand that $(s_2, 1)$ has a higher CP score.

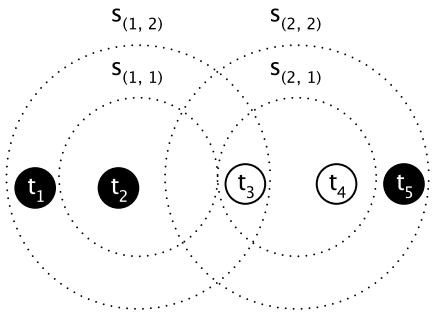

Figure 4: Example network with 5 targets, 2 sensors and 2 power levels.

Covering Waste. During the generation of a new cover C_l, for each adjusted sensor (s_i, a) with $s_i \in S_R \backslash S_I$ that can cover the critical target t_c, the CW score is the ratio among the number of covered targets that have already been covered in C_l (i.e., belonging to $T \setminus T_U$) and the total number of covered targets; that is,

$$CW(s_i, a) = \frac{|T_{(s_i,a)} \bigcap \{T \setminus T_U\}|}{|T_{(s_i,a)}|} \quad \forall (s_i, a) | s_i \in S_R \setminus S_I, a \in \{1, \ldots, k\}, t_c \in T_{(s_i,a)} \tag{12}$$

The greatest contribution according to the criterion is determined by the minimum CW score; it penalizes the selection of adjusted sensors that are unfit to the current cover since they would provide redundant coverage to many targets, wasting part of their battery lifetime. Again, let us evaluate this score for the four adjusted sensors in Figure 4, and let $\{T \setminus T_U\} = \{t_1, t_2, t_5\}$. The covering waste scores are $CW(s_1, 1) = \frac{|\{t_2\}|}{1} = 1$, $CW(s_1, 2) = \frac{|\{t_1, t_2\}|}{3} = \frac{2}{3}$, $CW(s_2, 1) = \frac{|\{\emptyset\}|}{2} = 0$ and $CW(s_2, 2) = \frac{|\{t_5\}|}{3} = \frac{1}{3}$. Note that a CW score equal to 0 means that the adjusted sensor covers only new targets.

Residual Lifetime. For each adjusted sensor (s_i, a) with $s_i \in S_R \setminus S_I$ that can cover the critical target t_c, the RL score is given by its residual lifetime ($RL(s_i, a) = r_{s_i}$). The greatest contribution is given by the maximum RL score.

Overall Sensor Contribution Evaluation. The input vector $\Gamma = (\gamma_1, \gamma_2, \gamma_3)$ is used to weight the relevance of the three criteria while determining the overall contribution of each adjusted sensor that covers the critical target. That is, for example, if $\Gamma = (1, 0, 0)$ only CP will be used, while if $\Gamma = (0, \frac{1}{2}, \frac{1}{2})$ CW and RL will be used and will be equally important. Formally, for each candidate adjusted sensor (s_i, a) let $CP'(s_i, a)$ be the associated CP score normalized in the interval $[0, 1]$ (note that the other two scores are defined in this interval). We define the contribution of (s_i, a) according to Γ as the convex combination $\gamma_1 CP'(s_i, a) + \gamma_2(1 - CW(s_i, a)) + \gamma_3 RL(s_i, a)$ and look for the adjusted sensor that maximizes this value.

5.2. Adjustable Ranges Iterative (AR-Iterative)

AR-Iterative embeds AR-Greedy in a local search scheme. The algorithm has an initialization phase, where the greedy heuristic is executed multiple times using different values of the Γ weighting parameter. The best solution identified during this phase is used as starting point for the local search phase, and the related Γ^* value is used for every other execution of the heuristic throughout the algorithm. The chosen tested values for Γ are discussed in Section 7.

Solution neighborhoods are built by executing a variant of AR-Greedy that avoids the selection of certain adjusted sensors (*banned* adjusted sensors). In more detail, we define the AR-Greedy' procedure that has a set of adjusted sensors AS_B as additional input parameter; both procedures behave the same, with the only difference that AR-Greedy' makes sure that the elements of AS_B are never selected during the procedure. In addition, we assume that AR-Greedy' also returns a set AS composed of all the adjusted sensors used in the covers (which can be easily computed in post-processing).

The AR-Iterative algorithm keeps track of the banned sensors that allow AR-Greedy' to improve the objective function value, gradually extending the AS_B set, and iteratively executes the algorithm until no significant improvements can be found in the neighborhood of the current solution.

The pseudocode is given in Algorithm 2. The AS_B set is initialized in line 2. The above described

initialization phase is performed in line 3, and the chosen starting solution is stored in line 4. The condition expressed in line 6 checks whether the main loop of the procedure (contained in lines 6-20) iterates or stops, based on the occurrence of significant objective function improvements in the last iteration. In the loop, the adjusted sensors of the current solution are added to AS_B one at a time and AR-Greedy' is executed (line 12), producing new neighbors, until a significantly better solution is found. The significancy of the improvement is evaluated using a parameter ϵ (line 12); if such a solution is found, the related neighbor is selected for the next iteration (line 15) and the adjusted sensor which led to this neighbor is permanently added to AS_B (line 16). Finally, the best solution found is returned in line 23.

Algorithm 2 AR-Iterative algorithm

1: input: wireless network $N = (T, S)$, number of power levels k, granularity factor $gf \in (0, 1]$, improvement factor $\epsilon \geq 0$
2: $AS_B \leftarrow \varnothing$
3: find the **best weighting parameter** Γ^*
4: $(SOL, lt, AS) = AR - Greedy'(N, k, gf, \Gamma^*, AS_B)$
5: $stop \leftarrow$ **false**
6: **while** $stop =$ **false do**
7: let $AS = \{(s_1, a_1), \ldots, (s_z, a_z)\}$
8: $i \leftarrow 0$
9: $improvement \leftarrow$ **false**
10: **while** $improvement =$ **false and** $i \leq z$ **do**
11: $i \leftarrow i + 1$
12: $(SOL_i, lt_i, AS_i) = AR - Greedy'(N, k, gf, \Gamma^*, AS_B \cup \{(s_i, a_i)\})$
13: **if** $lt_i > lt + \epsilon$ **then**
14: $improvement \leftarrow$ **true**
15: $(SOL, lt, AS) \leftarrow (SOL_i, lt_i, AS_i)$
16: $AS_B \leftarrow AS_B \cup \{(s_i, a_i)\}$
17: **end if**
18: **end while**
19: **if** $improvement =$ **false then**
20: $stop \leftarrow$ **true**
21: **end if**
22: **end while**
23: **return** (SOL, lt)

6. Upper bound computation

As we will show in Section 7, we did not execute the CG algorithm to completion on some instances, due to violation of the considered time limit. Therefore, a certified optimal solution is not available on these instances. In order to overcome this problem and have a measure of the quality of the solutions provided by our methods on all scenarios, we evaluated a theoretical upper bound U. The upper bound is the same as the one seen for the critical target selection, performed when $r_{s_i} = 1$ for each sensor s_i in

S. Moreover, as in 5.1.1, given a target t_j and a sensor s_i such that $t_j \in T_{(s_i,k)}$, let a_{ij} be the power level such that (s_i, a_{ij}) is minimal for t_j. We have

$$U = \min_{t_j \in T}(U_{t_j}) \tag{13}$$

where

$$U_{t_j} = \sum_{s_i \in S | t_j \in T_{(s_i,k)}} \frac{1}{\Delta^{a_{ij}}} \tag{14}$$

Given the same example sensor network in Figure 3 considered in Section 5.1.2, we have $U = 1$, which is the upper bound of target t_3 ($U_{t_3} = \frac{1+1}{\Delta^2}$).

7. Computational Results

We compared the performances of the proposed approaches (AR-Iterative and the Column Generation algorithm) on a wide set of test instances. The AR-Greedy algorithm is not explicitly reported here, since it is used as an internal procedure for AR-Iterative. The upper bound described in Section 6 is used to evaluate the quality of our solution when the CG procedure is not able to find a certified optimum within the considered time limit. The section is organized as follows: in Section 7.1 we describe our test instances; Section 7.2 contains the values of the parameters used in our algorithms and describes our testing environment; finally Section 7.3 contains our results divided in tables and some comments on them.

7.1. Instances Description

The instances are generated by randomly disposing targets and sensors on a $200n \times 200n$ area, where n is the number of target nodes. We considered test instances composed of $n = 50, 100, 200, 400, 800, 1200$ targets. We consider a parameter *depth*, which represents a lower bound on the minimum number of sensors that cover each target when set at their lowest power level. Sensors will be randomly generated until the *depth* condition is satisfied. In our experiments, we consider $depth = 3, 6, 9$. Moreover, while generating sensors, we check that each of them covers at least one target when set on its lowest power level. Regarding the number of adjustable power levels for each sensor, we generated instances according

to three different multi-level *power modes* ($pm = 2, 3, 5$). When $pm = a$ there are a different power levels to choose from. We set the sensing range of the lowest level r_1 always equal to $100n$. Let $A^1 = \pi r_1^2$ be the size of the area covered by each sensor when set to level 1. We want the areas to be covered by the other power levels to be the following:

- $A^2 = \frac{5}{3}A^1$ for $pm = 2$
- $A^2 = \frac{4}{3}A^1$, $A^3 = \frac{5}{3}A^1$ for $pm = 3$
- $A^2 = \frac{7}{6}A^1$, $A^3 = \frac{4}{3}A^1$, $A^4 = \frac{3}{2}A^1$, $A^5 = \frac{5}{3}A^1$ for $pm = 5$

That is, we always want the most expensive level to cover an area $\frac{2}{3}$ times larger than A^1, and the size of the other levels to be equally distributed in this interval ($A^a = (1 + \frac{2}{3}\frac{a-1}{k-1})A^1$). If the area coverage for a given level a is αA^1, its consumption ratio Δ^a is set to $\alpha\Delta^1 = \alpha$ accordingly (recall that $\Delta^1 = 1$ by definition). When we fix the *depth* and n parameters, increasing pm corresponds to adding new power levels to the same set of instances, making these scenarios directly comparable.

In order to validate the effectiveness of the adjustable ranges approach, we also created two single-level power modes. The first one ($pm = 1$) is obtained by considering just the smallest power level for each of our instances. The optimal solution for each instance with $pm = 1$ is known in advance by construction, and is equal to the *depth* parameter; let us denote this set of optimal solutions as OPT^1. The second single-level power mode ($pm = 1H$) is obtained by considering the most expensive power level for each instance ($A^{1H} = \frac{5}{3}A^1$ and $\Delta^{1H} = \frac{5}{3}\Delta^1 = \frac{5}{3}$), and optimal solutions for this case are not known in advance.

For each combination of the described parameters, we generated 5 instances, for a total of 54 multi-level scenarios with 270 instances and 36 single-level scenarios with 180 instances. We did not execute heuristic tests on the single-level instances with $pm = 1H$, since developing a good heuristic for this case is outside the focus of this paper, while we compared the trivial optimal solutions of $pm = 1$ with CG solutions for $pm = 1H, 2, 3, 5$, in order to get an estimate of the advantage given by the multi-level approach (see Table 3 and the related comments in Section 7.3).

7.2. Parameters, Testing Environment and Used Languages

Regarding the AR-Iterative algorithm, after a preliminary experimental phase we choose $gf = 0.2$ for the granularity factor and $\epsilon = 0.1$ for the improvement factor. As for the initialization phase, we choose the following seven values for the Γ parameter: $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$, $(\frac{1}{2}, \frac{1}{2}, 0)$, $(\frac{1}{2}, 0, \frac{1}{2})$, $(0, \frac{1}{2}, \frac{1}{2})$ and $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. For the Column Generation algorithm, we considered a time limit of 1 hour for each instance. All algorithms have been coded in C++ and executed on an Intel Xeon 2Ghz workstation with 8GB of RAM. The IBM ILOG CPLEX 12 solver with Concert Technology was used to solve the mathematical formulations within the Column Generation algorithm.

7.3. Results

The tables included in this section summarize the results of our experimental tests. We report average results for each scenario; that is, each entry in the tables contains an average value over the corresponding 5 instances. In each table, Row n represents the cardinality of the set of targets, Row *depth* the depth parameter value, and *pm* the considered power mode.

Tables 1-2 contain average performances of our Column Generation and AR-Iterative algorithms on the multi-level instances in terms of solution quality, compared with the proposed upper bound and with the CG solutions, respectively. Table 3 focuses on the significance of the proposed multi-level approach, comparing the solution values of power modes 2, 3 and 5 with the ones of the single-level power modes 1 and $1H$. Finally, Tables 4-5 contain average computational times for the two proposed algorithms. In what follows, we comment each of these tables.

As mentioned before, when we reach the considered time limit for the Column Generation procedure and therefore the provided solutions are not certified as optimal, we compare them with our upper bound described in Section 6 to evaluate their quality. In particular, for each instance, let U' be the value returned by the Column Generation algorithm if its execution is terminated before the time limit, the upper bound value U otherwise. Table 1 contains average percentage ratios between Column Generation solution values and U' values (computed as $\frac{CG}{U'} \times 100$). Therefore, regarding the entries with a value of 100 the Column Generation produced a certified optimum for each of the related instances. This happens

in 7 cases: twice for $n = 50$ with $depth = 3$ and $pm = 2, 3$, twice for $n = 100$ with $depth = 3, 6$ and $pm = 2$, and once each for $n = 200, 400, 1200$ with $depth = 3$ and $pm = 2$. Overall, considering all the scenarios, certified optimal solutions were found for 137 individual multi-level instances. It can be noticed that the performances tend to get worse for high values of the *depth* and *pm* parameters. However, the average solution value is never smaller than 95.74% of the average optimum, and in 42 out of 54 scenarios it is higher than 97%. For all the considered scenarios, CG solutions found within the time limit can therefore be considered accurate approximations of the optimal solutions.

In Table 2, the average percentage ratios between AR-Iterative and CG are reported, computed as $\frac{AR-Iterative}{CG} \times 100$. Recall that AR-Iterative solutions are used to initialize CG, therefore this ratio can be equal to 100 (if the heuristic either finds a optimal solution or the same solution of the CG within the time limit) or below. It can be seen that the average solution ratio is consistently higher than 90% for all scenarios but one, i.e. for $n = 50$, $depth = 6$ and $pm = 3$, where the average ratio is 89.94%. AR-Iterative found certified optimal solutions for 49 individual instances, and proved to be effective both as initialization method for CG and as a fast procedure to obtain good solutions.

Table 3 contains average percentage ratios between CG solution values for $pm = 1H, 2, 3, 5$ with the optimal values of the related instances for $pm = 1$, computed using the formula $\frac{OPT_1}{CG} \times 100$. The Column Generation procedure for $pm = 1H$ was initialized with covers generated by repeated executions of the subproblems with random weights associated to the sensors. Regarding the single-level power modes, it can be seen that using sensors with larger (although more expensive) ranges appears to be a generally better choice for the considered instances, as it gives on average better solutions in 14 out of 18 scenarios. However, the results for multi-level instances show that the energy of the sensors can be used in a far more effective way when different ranges are used together. Considering the case $pm = 2$, the average improvement varies from a minimum of 29.42% to a maximum of 74.4% with respect to $pm = 1$, and from a minimum of 35.31% to a maximum of 43.2% with respect to $pm = 1H$.

Additional power levels bring further extensions of the network lifetime: $pm = 3$ brings an average improvement of 6.07% with respect to $pm = 2$, and $pm = 5$ improves the results obtained with $pm = 3$

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	2	100	98.79	97.55	100	100	97.25	100	97.47	96.29
	3	100	96.72	98.23	98.69	98.27	96.83	97.65	97.82	97.28
	5	99.52	95.74	98.6	99.01	96.49	96.83	96.77	97.91	96.53
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	2	100	97.58	98.69	99.78	97.18	95.78	100	97.76	98.26
	3	97.18	98.01	98.61	97.47	97.62	96.1	98.51	98.9	98.75
	5	97.39	98.14	98.62	97.37	97.24	95.69	96.97	98.09	98.73

Table 1: CG - U' solution values percentage ratio

of 3.42% on average. We might expect the improvements brought by additional levels to keep being incrementally smaller, up to a stabilization point where they would be redundant.

Tables 4 and 5 contain average computational times (expressed in seconds) for AR-Iterative and CG, respectively. As we know, CG does not end its execution on some instances, therefore we did not consider them and evaluated average values only on the meaningful ones. In more detail, each entry in Table 4 has an associated value (reported in brackets) that expresses the number of instances that run to completion, and that were therefore used to evaluate the average. For example, when this value is 5, all the instances of the scenario run to completion. On scenarios where no instance terminated in the time limit, we just report a *dnf* (did not finish) value. As expected, many instances can be solved for small values of *pm* and *depth*, while the number of solved instances decreases and eventually drops to 0 when their values increase. For example, all instances can be solved for $pm = 1H$ and $depth = 3$, and all instances except two can be solved for $pm = 2$ and $depth = 3$; on the other hand, for $pm = 2$ and $depth = 9$, only three instances can be solved to completion when $n \leq 100$, and when $pm = 5$ and $depth = 9$ no instance can be solved. It can be also seen that the number of scenarios where some instances run to completion diminishes for high values of n, although as previously discussed the quality of the returned solutions keeps being good with respect to the upper bound.

Now, consider the time averages for our heuristic in Table 5. It can be seen that computational times increase in a very consistent way as we increment the values of n, *depth* and *pm*. Overall, computational times are reasonable, varying from an average of 0.12 seconds for $n = 50$, $depth = 3$, $pm = 2$ to 346.99 for $n = 1200$, $depth = 9$, $pm = 5$.

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	2	97.65	93.38	95.92	96.26	95.63	95.43	97.58	96.22	93.52
	3	94.29	89.94	93.86	93.93	94.35	95.34	97.32	94.15	91.89
	5	92.3	91.51	94.7	90.7	92.93	94.68	94.22	94.44	90.71
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	2	96.91	96.48	98	98.18	97.1	94.03	99.02	96.03	96.19
	3	95.52	94.67	98.41	97.84	94.35	93.77	95.57	93.66	94.74
	5	95.24	95.32	97.44	93.77	92.58	90.55	94.88	91.73	94.56

Table 2: AR-Iterative - CG solution values percentage ratio

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	1H	104	104	100	92	92	93.33	104	122	110.67
	2	141.18	139.31	139.72	129.42	129.99	129.7	146	158.27	148.97
	3	146.4	146.42	145.39	135.35	135.93	135.19	150.61	167.13	155.1
	5	150.9	149.87	148.93	138.64	137.76	137.44	152.6	171.27	157.27
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	1H	116	124	112	136	124	125.33	100	96	109.33
	2	159.2	161.7	152	174.4	162.57	160.93	137.33	136.61	150.39
	3	164.93	169.11	156.9	182.26	168.6	166.65	141.8	141.24	156.45
	5	166.92	173.76	160.93	188.81	172.77	171.12	143.58	142.14	158.86

Table 3: CG - OPT^1 solution values percentage ratio

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	1H	1.17(5)	16.21(5)	580.18(4)	1.59(5)	22.69(5)	845.06(4)	4.06(5)	762.99(4)	2518.01(2)
	2	7.69(5)	132.11(4)	961.02(2)	23.17(5)	349.99(5)	1144.25(1)	21.72(5)	2262.16(1)	dnf
	3	38.39(5)	919.92(2)	dnf	41.56(4)	1013.67(2)	dnf	1432.2(2)	dnf	dnf
	5	320.56(4)	dnf	dnf	361.8(3)	2261.12(1)	dnf	0.55(1)	dnf	dnf
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	1H	104.78(5)	287.58(2)	dnf	34.40(5)	798.69(4)	dnf	15.44(5)	248.45(4)	dnf
	2	913.99(5)	dnf	dnf	34.45(3)	dnf	dnf	96.52(5)	741.85(2)	dnf
	3	dnf	dnf	dnf	136.18(2)	dnf	dnf	209.94(2)	dnf	dnf
	5	dnf	dnf	dnf	dnf	dnf	dnf	1.05(1)	dnf	dnf

Table 4: CG time averages

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	2	0.12	0.18	0.28	0.2	0.35	0.59	0.56	1.04	1.56
	3	0.8	1.03	2.07	1.07	2.22	4.1	6.45	8.94	17.82
	5	3.22	3.37	7.45	4	7.92	15.43	14.41	21.46	40.04
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	2	1.91	3.27	5.33	5.38	9.2	15.68	5.01	6.43	12.5
	3	12.47	20.59	39	21.44	38.73	96.34	24.91	41.41	84.42
	5	35.49	71.24	107.45	67.28	150.77	250.37	119.97	221.32	346.99

Table 5: AR-Iterative time averages

8. Conclusions

In this work we addressed the Maximum Network Lifetime with Adjustable Ranges Problem (MLARP), which is a generalization of the classical Maximum Network Lifetime Problem (MLP) defined on wireless sensor networks. We developed an exact approach, based on a Delayed Column Generation technique, and a greedy heuristic which was embedded in a local search scheme. An extensive experimental phase was carried out in order to validate the proposed methods. We performed tests both on instances where the sensor ranges could be adjusted in a number of different power levels and on the same instances where sensors ranges were fixed (i.e., the classical MLP). The proposed variant allowed us to find significant improvements of the network lifetime with respect to the classical approach in each of the considered scenarios, with average improvements per scenario ranging from 24.72% to 88.81%. It has also been verified that considering a larger number of power levels brings further improvements, which get incrementally smaller.

The exact approach was able to obtain optimal solutions in reasonable time on many instances. Even on high dimensional instances with up to 1200 target points, where we did not find certified optima within the considered time limit, the returned solutions proved to be accurate when evaluated with respect to an upper bound (the average CG solution value per scenario was proved to be never smaller than 95.74% of the optimum). The local search algorithm provided high quality solutions in fast computational times (the average solution value per scenario was never smaller than 89.94% with respect to CG solutions), and was used as effective initialization method for the CG procedure.

Regarding future lines of research, we intend to bring on the study of this problem by developing appropriate metaheuristic algorithms, and to approach some variants of it (e.g., Maximum Network Lifetime Problem with adjustable ranges when a certain portion of targets can be neglected in each cover, or when connected covers are required). We think that it might also be of interest to perform a theoretical study of the possible improvement that can be obtained in terms of objective function by adding new power levels. This may lead us to a direct comparison among solution methods which consider discrete and continuous adjustable ranges models. New classes of instances, possibly related to real-world

applications, will be also investigated.

[1] A. Alfieri, A. Bianco, P. Brandimarte, and C. F. Chiasserini, Maximizing system lifetime in wireless sensor networks, European Journal of Operational Research, Volume 181, Issue 1, pp. 390-402 (2007).

[2] P. Berman, G. Calinescu, C. Shah, and A. Zelikovsky, Power Efficient Monitoring Management in Sensor Networks, Proceedings of the Wireless Communications and Networking Conference 2004, pp. 2329-2334 (2004).

[3] M. Cardei and D.-Z. Du, Improving Wireless Sensor Network Lifetime through Power-Aware Organization, ACM Wireless Networks, Volume 11, Issue 3, pp. 333-340 (2005).

[4] M. Cardei, M. T. Thai, Y. Li, and W. Wu, Energy-Efficient Target Coverage in Wireless Sensor Networks, Proceedings of the 24th conference of the IEEE Communications Society (INFOCOM), Volume 3, pp. 1976-1984 (2005).

[5] M. Cardei, J. Wu, and M. Lu, Improving Network Lifetime using Sensors with Adjustable Sensing Ranges. International Journal of Sensor Networks, Volume 1, Issue 1, pp. 41-49 (2006).

[6] M. X. Cheng, L. Ruan, and W. Wu, Achieving Minimum Coverage Breach under Bandwidth Constraints in Wireless Sensor Networks, Proceedings of the 24th conference of the IEEE Communications Society (INFOCOM), Volume 4, pp. 2638-2645 (2005).

[7] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, Habitat Monitoring: Application Driver for Wireless Communications Technology, Proceedings of ACM SIG COMM Workshop on Data Communication in Latin America and the Caribbean, Volume 31, Issue 2, pp. 20-41 (2001).

[8] A. Dhawan, C. T. Vu, A. Zelikovsky, and Y. Li, Maximum Lifetime of Sensor Networks with Adjustable Sensing Range, Proceedings of the Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, pp. 285-289 (2006).

[9] A. Dhawan, A. Aung, and S. K. Prasad, Distributed Scheduling of a Network of Adjustable Range

Sensors for Coverage Problems, Communications in Computer and Information Science, Volume 54, Chapter 3, pp. 123-132 (2010).

[10] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, Next Century Challenges: Scalable Coordination in Sensor Networks, Proceedings of the 5th annual ACM/IEEE international conference on Mobile Computing and Networking, pp. 263-270 (1999).

[11] M.Gentili and A. Raiconi, $\alpha-$Coverage to Extend Network Lifetime on Wireless Sensor Networks, Technical report n.2-2010, Department of Mathematics and Computer Science, University of Salerno (2010).

[12] J. Kahn, R. Katz, and K. Pister, Next Century Challenges: Mobile Networking for Smart Dust, Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, pp. 271-278 (1999).

[13] L. Lopes, M. Gentili, A. Efrat, and S. Ramasubramanian, Scheduling Redundant Sensors Optimally for Maximum Lifetime, Technical report n.11-2010, Department of Mathematics and Computer Science, University of Salerno (2010).

[14] N. D. Nguyen, V. Zalyubovskiy, M. T. Ha, and H. Choo, Energy-Efficient Models for Coverage Problem Using Sensors with Adjustable Sensing Ranges, Proceedings of 2010 IEEE Wireless Communications and Networking Conference, pp. 1-6 (2010).

[15] A. Raiconi and M. Gentili, Exact and Metaheuristic Approaches to Extend Lifetime and Maintain Connectivity in Wireless Sensors Networks, INOC 2011, Lecture Notes in Computer Science, Volume 6701, pp. 607-619 (2011).

[16] S. Slijepcevic and M. Potkonjak, Power Efficient Organization of Wireless Sensor Networks, IEEE International Conference on Communications, Volume 2, pp. 472-476 (2001).

[17] Y. B. Turkogullari, N. Aras, I. K. Altinel, and C. Ersoy, Optimal Placement, Scheduling, and Routing

to Maximize Lifetime in Sensor Networks, Journal of the Operational Research Society, Volume 61, Issue 6, pp. 1000-1012 (2010).

[18] Y. B. Turkogullari, N. Aras, I. K. Altinel, and C. Ersoy, An Efficient Heuristic for Placement, Scheduling and Routing in Wireless Sensor Networks, Ad Hoc Networks, Volume 8, Issue 6, pp. 654-667 (2010).

[19] Y. B. Turkogullari, N. Aras, I. K. Altinel, and C. Ersoy, A Column Generation Based Heuristic for Placement, Scheduling, and Routing in Wireless Sensor Networks, European Journal of Operational Research, Volume 207, Issue 2, pp. 1014-1026 (2010).

[20] C. Wang, M. T. Thai, Y. Li, F. Wang, and W. Wu, Minimum Coverage Breach and Maximum Network Lifetime in Wireless Sensor Networks, Proceedings of IEEE Globecom 07, pp. 1118-1123 (2007).

[21] J. Wu and S. Yang, Energy-efficient node scheduling models in sensor networks with adjustable ranges, International Journal of Foundations of Computer Science, Volume 16, Issue 1, pp. 3-17 (2005).

[22] V. Zalyubovskiy, A. Erzin, S. Astrakov, and H. Choo, Energy-efficient area coverage by sensors with adjustable ranges, Sensors, Volume 9, Issue 4, pp. 2446-2460 (2009).

[23] Q. Zhao and M. Gurusamy, Lifetime maximization for connected target coverage in wireless sensor networks, IEEE/ACM Transactions on Networking, Volume 16, Issue 6, pp. 1378-1391 (2008).