

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

R. Cerulli<sup>a</sup>, R. De Denato<sup>a</sup>, A. Raiconi<sup>a</sup>

<sup>a</sup>*Department of Mathematics and Computer Science, 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, [6],[9],[11])

---

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

A common scenario in these applications consider the deployment of a large quantity of low-cost, 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 sensor has a sensing range, which can be fixed or adjustable, and therefore is 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 by the single targets can be shared either between themselves or communicating with a central station and, therefore, they can be coordinated to perform together 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 [1] 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 have 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 [4], we consider a finite number of alternative *power levels* and associate a measure of battery consumption to each of them.

Indeed, a clever use of the sensors can effectively increase the *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 individuate 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 *Maximum 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 possible covers for the whole set of targets are  $\{s_1, s_2\}$  or  $\{s_2, s_3\}$ .

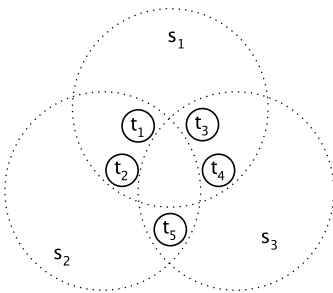


Figure 1: Example network with 5 targets and 3 sensors.

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 units of time. Further extensions of the network lifetime are not possible since only  $s_1$  has residual lifetime and it doesn't 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, network lifetime is equal to 1.5 units of time, and therefore this results to be a better strategy. MLP was proved to be NP-complete by reduction from the 3-SAT problem in [3]. Different solution approaches were proposed to solve it either exactly (see [12]) or approximately ([1],[2],[3],[14]). Variants of the problem consider covers which may neglect some of the targets (Minimum Coverage Breach ([5],[15]), Maximum Network  $\alpha$ -Lifetime [10]).

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 batteries be able to keep sensors active for 1 unit of time at power level 1 and 0.5 units at power level 2.

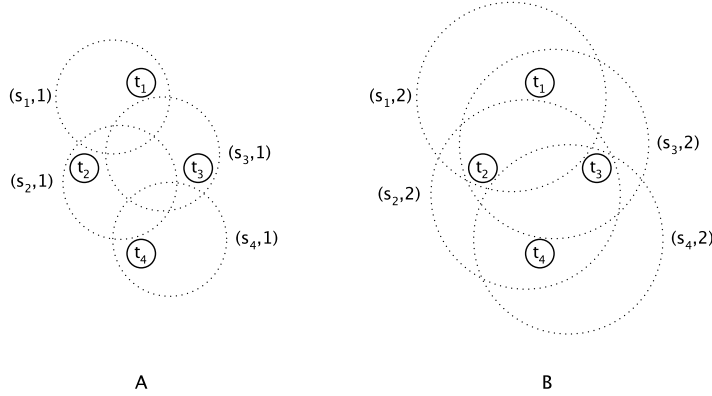


Figure 2: Example network with 4 targets, 4 sensors and 2 power levels.

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 *Maximum Network Lifetime with Adjustable Ranges Problem (MLARP)*. In [4] the authors address the problem of maximizing the number of covers, called Adjustable Range Set Covers (AR-SC). They present some heuristic resolution approaches, based both on greedy and LP relaxation methods. In [7] 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 [8]. In [13], [16] and [17] 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. Section 4 presents the mathematical formulation we developed to describe the problem, which was then embedded in a Column Generation procedure as described in Section 5. Section 6 presents our heuristic procedures. Section 7 describes a procedure to evaluate upper bounds on the solution value. The results of our extensive experimental tests are presented in Section 8. Finally, Section 9 contains some final remarks.

## 2. Notation

Let  $N = (T, S)$  be a wireless sensor network, where  $T = \{t_1, \dots, t_n\}$  is the set of the target nodes and  $S = \{s_1, \dots, s_m\}$  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's set at level  $a$ . The positions of targets and nodes don't 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  $T_{(s_i, b)} \subseteq T_{(s_i, a)} \forall b \in \{1, \dots, a - 1\}$ . 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  $t_j \notin T_{(s_i, b)} \forall b \in \{1, \dots, a - 1\}$ . 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  $C_l = \{(s_i, a) | s_i \in S, a \in 1, \dots, k\}$ , 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 require an increasing 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  $\Delta^a$  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 least expensive level). For example,  $\Delta^a = 2$  means that level  $a$  consumes twice the energy of level 1. It's 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's always set at level 1.

### 3. Problem Definition and Complexity

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  $(C_l, w_l)$ ,  $l = 1, \dots, p$ , where  $C_l$  is a cover and  $w_l$  is its corresponding activation time, such that the sum of all the activation times (that is the network life time)  $\sum_{l=1}^p w_l$  is maximized, and the power consumption of each sensor does not exceed its battery, i.e.  $\sum_{l \in \{1, \dots, p\} | (s_i, a) \in C_l} \Delta^a w_l \leq 1$  for each  $s_i \in S$ .

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

### 4. Mathematical Formulation

Let  $C_1, \dots, C_\ell$  be the family of all the feasible covers. We model with variables  $w_1, \dots, w_\ell$  the activation times we want to determine for each of them (note that feasible covers that we do not want to be part of the solution will have an activation time equal to 0). For each sensor  $s_i$ , level  $a$  and cover  $C_l$ , let  $\Phi_{il}^a$  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.

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

s.t.

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

$$w_l \geq 0 \quad \forall l = 1, \dots, \ell \tag{3}$$

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

The total number of feasible covers  $\ell$  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.

## 5. Column Generation Approach

Our Column Generation approach is a variant of the method proposed in [12] for the classic MLP problem. Another variant of this method was presented in [10] to solve the Maximum Network  $\alpha$ -Lifetime and the Maximum Network Regular  $\alpha$ -Lifetime Problems.

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't 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 to generate 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 Primal*) 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  $\pi_i, i = 1, \dots, m$ , 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  $\sum_{i:(s_i, \cdot) \in C_l} \pi_i - c_l \geq 0$ , 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:

$$\mathbf{[SEP]} \quad \min \sum_{i=1}^m \sum_{a=1}^k \pi_i x_i^a \quad (4)$$

s.t.

$$\sum_{i=1}^m \sum_{a=1}^k \phi_{ji}^a x_i^a \geq 1 \quad \forall j = 1, \dots, n \quad (5)$$

$$\sum_{a=1}^k x_i^a \leq 1 \quad \forall i = 1, \dots, m \quad (6)$$

$$x_i^a \in \{0, 1\} \quad \forall i = 1, \dots, m; a = 1, \dots, k \quad (7)$$

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;
- $\phi_{ji}^a$  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 adjusted sensor for each sensor.

If the optimal objective function of **[SEP]** is  $\geq 1$ , 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.

As already carried out in [12], to avoid the generation of previously generated covers we add the following constraints to [SEP]. Let  $C_1, \dots, C_g$  be the covers generated by the algorithm so far:

$$\sum_{i=1}^m \sum_{a=1}^k \Phi_{il}^a x_i^a \leq \sum_{i=1}^m \sum_{a=1}^k \Phi_{il}^a - 1 \quad \forall l = 1, \dots, g \quad (8)$$

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.

The CG procedure is initialized with a single cover composed of each sensor  $s_i \in S$  activated on the highest power level  $k$ , which is valid unless there are no feasible covers in the network.

## 6. Heuristic Approaches

### 6.1. Adjustable Ranges Greedy (AR-Greedy)

In this section we present a greedy heuristic which shares some ideas of Centralized Greedy Algorithm presented in [4], bringing further refinements.

The main idea of AR-Greedy is to iteratively build new covers by selecting specific targets (*critical targets*), and then, selecting the *best* adjusted sensors to cover them. Appropriate coverage times within a given upper bound are also selected in order to keep each cover feasible.

The pseudocode of the procedure is presented in Algorithm 1.

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 during the algorithm execution. The  $\Gamma$  vector is used while determining which sensors will be included in the generated covers, as explained in Section 6.1.2. 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 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. We will describe these choices in Sections 6.1.1 and 6.1.2. 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. Further details on how the activation time is evaluated are reported in section 6.1.3. 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

---

```

1: input: wireless network  $N = (T, S)$ , number of power levels  $k$ , granularity factor  $gf \in (0, 1]$ , criteria
   weighting parameter  $\Gamma = (\gamma_1, \gamma_2, \gamma_3)$ ,  $\gamma_i \geq 0$ ,  $\gamma_1 + \gamma_2 + \gamma_3 = 1$ 
2:  $S_R \leftarrow S$ 
3: for each  $s_i \in S_R$  do
4:    $r_{s_i} \leftarrow 1$ 
5: end for
6:  $SOL \leftarrow \emptyset$ 
7:  $lt \leftarrow 0$ 
8: while  $\bigcup_{s_i \in S_R} T_{(s_i, k)} \equiv T$  do
9:   Create a new empty cover  $C_l$ 
10:   $T_U \leftarrow T$ 
11:   $S_I \leftarrow \emptyset$ 
12:  while  $T_U \neq \emptyset$  do
13:    Find a critical target  $t_c \in T_U$ 
14:    Select  $s_c \in S_R \setminus S_I$  and  $a \in \{1, \dots, k\}$  s.t.  $t_c \in T_{(s_c, a)}$  and  $(s_c, a)$  has the maximum contribution
    according to  $\Gamma$ 
15:     $S_I \leftarrow S_I \cup \{s_c\}$ 
16:    for each  $t_j \in T_U$  s.t.  $t_j \in T_{(s_c, a)}$  do
17:       $T_U \leftarrow T_U \setminus \{t_j\}$ 
18:    end for
19:     $C_l \leftarrow C_l \cup \{(s_c, a)\}$ 
20:  end while
21:   $w_l = \text{max feasible activation time} \leq gf$  for  $C_l$ 
22:  for each  $(s_i, a) \in C_l$  do
23:     $r_{s_i} \leftarrow r_{s_i} - (\Delta^a w_l)$ 
24:    if  $r_{s_i} = 0$  then
25:       $S_R \leftarrow S_R \setminus \{s_i\}$ 
26:    end if
27:  end for
28:   $SOL \leftarrow SOL \cup \{(C_l, w_l)\}$ 
29:   $lt \leftarrow lt + w_l$ 
30: end while
31: return  $(SOL, lt)$ 

```

---

### 6.1.1. 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.

More in 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:

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

where

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

### 6.1.2. 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  $s_i \in S_R \setminus S_I$  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  $\Delta^a$ ; that is,

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

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 don't bring significant improvements.

*Covering Waste.* During the generation of a new cover  $C_l$ , 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 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,

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

The greatest contribution according to the criterion is determined by the minimum CW score; it penalizes the selection of sensors and power levels that use inefficiently significant amounts of energy.

*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 by definition). We define the contribution of  $(s_i, a)$  according to  $\Gamma$  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 such value.

### 6.1.3. Maximum Feasible Activation Time

Given a newly generated cover  $C_l$ , it 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  $r_{s_i} - \Delta^a$ ; 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  $(s_i, a) \in C_l$ .

### 6.2. 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  $\Gamma$  weighting parameter. The best solution identified during this phase is used as starting point for the local search phase and the related  $\Gamma^*$  value is used for every other execution of the heuristic throughout the algorithm. The chosen tested values for  $\Gamma$  are discussed in Section 8.

Solution neighborhoods are built by executing a variant of AR-Greedy that avoids the selection of certain adjusted sensors (*banned* adjusted sensors). More in 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 by 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 significance of the improvement is evaluated using a parameter  $\epsilon$  (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.

## 7. Upper bound computation

As we will show in Section 8, on some instances we did not execute the CG algorithm to completion 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 6.1.1, given a target  $t_j$  and a sensor  $s_i$  such that  $t_j \in T_{(s_i,k)}$ , let  $a_{ij}$  be the

---

**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:  $AS_B \leftarrow \emptyset$ 
3: find the best weighting parameter  $\Gamma^*$ 
4:  $(SOL, lt, AS) = AR - Greedy'(N, k, gf, \Gamma^*, AS_B)$ 
5:  $stop \leftarrow \text{false}$ 
6: while  $stop = \text{false}$  do
7:   let  $AS = \{(s_1, a_1), \dots, (s_z, a_z)\}$ 
8:    $i \leftarrow 0$ 
9:    $improvement \leftarrow \text{false}$ 
10:  while  $improvement = \text{false}$  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:       $improvement \leftarrow \text{true}$ 
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  $improvement = \text{false}$  then
20:     $stop \leftarrow \text{true}$ 
21:  end if
22: end while
23: return  $(SOL, lt)$ 
```

---

power level such that  $(s_i, a_{ij})$  is minimal for  $t_j$ . We have

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

where

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

## 8. 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 internal procedure for AR-Iterative. The upper bound described in Section 7 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 Subsection 8.1 we describe our test instances; subsection 8.2 contains the values we used for the parameters used by our algorithm and describes our testing environment; finally Subsection 8.3 contains our results divided in tables as well as

some comments.

### 8.1. Instances Description

The instances are generated by randomly disposing targets and sensors on a  $200n \times 200n$  area. We considered test instances composed of  $n = 50, 100, 200, 400, 800, 1200$  target nodes. We consider a parameter *depth* which represents a lower bound on the minimum number of sensors that cover each target when set on 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 ( $A^a = 1 + \frac{2}{3} \frac{a-1}{k-1}$ ). If the area coverage for a given level  $a$  is  $\alpha A^1$ , its consumption ratio  $\Delta^a$  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  $\Delta^{1H} = \frac{5}{3}\Delta^1 = \frac{5}{3}$ ).

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 since developing a good heuristic for this case is outside the focus of this paper, while we compared CG results for the various cases in order to get an estimate of the advantage given by the multi-level approach.

### 8.2. Parameters and Testing Environment

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  $\Gamma$  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. IBM ILOG CPLEX 12 with Concert Technology was used to solve the mathematical formulations within the Column Generation algorithm.

### 8.3. Results

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

As already introduced, 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 in order to evaluate their quality. More in detail, for each instance  $inst$  let  $U'(inst)$  be the value returned by the Column Generation algorithm if it terminated its execution before the time limit, the upper bound value otherwise. Tables 1 contain average percentage ratios between Column Generation solution values and U' values (computed as  $\frac{CG}{U'} \times 100$ ). Therefore, regarding the entries with a 100 value the Column



Generation produced a certified optimum for each of the related instances. It can be seen that the ratio gets worse as we increase both the depth parameter and the power mode parameter. This is justified by the additional complexity of the problem when a larger set of sensors has to be considered. The performances of the CG algorithm do not significantly deteriorate if we consider larger sets of targets, showing a good scalability to high dimensional problems. Overall, the worst average ratios are obtained for  $n = 100$ ,  $pm = 5$  and  $depth = 6$  and  $9$ , where they are equal to 89% and 86.96%, respectively. For all the other 52 scenarios, the average ratio between the CG algorithm and  $U'$  is never worse than 90% and in 31 of them it is higher than 95%.

In Tables 2, average percentage ratios between AR-Iterative and CG are reported. Those values can be higher than 100% in case AR-Iterative finds on average better solutions than the CG algorithm within the time limit. This actually happens in 8 scenarios; more in detail, in 7 out of those 8 scenarios we had  $pm = 5$  and in the remaining  $pm = 3$  and  $depth = 9$ , confirming that the Column Generation is affected by high values for these parameters, while the heuristic criteria keep being effective. Overall, the average ratio on all scenarios varies from 90.05% and 105.87%. Most of the smallest percentage ratios can be found when there is a small number of targets: 10 scenarios fall in the range between 90% and 95% when the number of targets is between 50 and 200, while only for one scenario it is smaller than 95% in the other cases (94.63% for  $n = 1200$ ,  $depth = 3$ ,  $pm = 3$ ). This can be explained by the nature of our local search scheme, which might not be able to recover from a "wrong" greedy choice on smaller instances where the algorithm has to perform less choices during its execution.

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  $pm = 1$ . 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 28.25% to a maximum of 73.28% with respect to  $pm = 1$  and from a minimum

of 34.04% to a maximum of 43.20% with respect to  $pm = 1H$ . Adding new power levels brings obviously better or equal optimal solutions; we could expect such improvements to be incrementally smaller up to a stabilization point where further levels are redundant. We can actually verify this behavior in our instances for  $pm = 3$ , which brings an average improvement of 3.39% with respect to  $pm = 2$ . However, it can also be noticed from our tables that in one scenario with  $pm = 3$  and  $depth = 9$  and in 14 scenarios with  $pm = 5$  increasing the number of power levels brings slightly worse solutions; this is explained by the high complexity of these instances (recall that the accuracy of the procedure gets worse when  $pm$  and  $depth$  increase).

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. More in 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 we 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 was easy predictable, many instances can be solved for small values of  $pm$  and *density*, 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 scenarios can solve one of their instances to completion 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 we previously noticed the quality of the returned solutions keeps being good with respect to the upper bound. Finally, two particular cases can be noticed for scenarios  $n = 200$ ,  $pm = 5$ ,  $depth = 3$  and  $n = 1200$ ,  $pm = 5$ ,  $depth = 3$ ; in both cases a single instance proved to be easy for the CG approach and was solved in a few seconds.

Now consider time averages for our heuristic in Tables 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

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	2	100	98.79	97.23	98.67	100	96.18	100	99.11	97.14
	3	100	95.72	94.15	98.69	97.76	93.15	97.64	93.58	94.45
	5	98.02	92.41	90.93	96.95	89	86.96	95.59	90.03	90.73
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	2	98.55	97.33	98.59	99.53	96.86	94.82	100	98.66	96.14
	3	94.48	95.05	94.84	96.68	94.63	93.97	99.47	96.59	96.43
	5	91.59	92.51	91.6	94.9	94.47	90.77	96.11	93.18	94.86

Table 1: CG/U' solution values percentage ratio

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	2	97.65	93.38	96.27	96.26	95.63	96.52	97.58	96.22	93.52
	3	94.29	90.05	98.24	93.93	94.86	99.52	97.33	98.45	94.69
	5	93.88	94.81	103.13	92.75	101.49	105.87	95.43	102.74	96.67
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	2	96.91	96.73	98.1	98.65	97.43	95	99.02	96.3	98.34
	3	98.46	97.62	102.69	98.63	97.41	95.89	94.63	95.97	97.04
	5	101.49	101.11	104.99	96.29	95.29	95.46	95.75	96.75	98.45

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

times are very reasonable, varying from an average of 0.12 seconds for  $n = 50$ ,  $depth = 3$ ,  $pm = 2$  to 346.99 for  $n = 1200$ ,  $depth = 9$ ,  $pm = 5$ .

## 9. Conclusions

In this work we addressed the Maximum Network Lifetime with Adjustable Ranges Problem, that is a generalization of the classical Maximum Network Lifetime Problem defined on wireless sensor networks. We developed an exact approach based on a Delayed Column Generation technique and a greedy heuristic

n		50			100			200		
depth		3	6	9	3	6	9	3	6	9
pm	1H	104	104	100.00	92	92	93.33	104.00	122.00	110.67
	2	141.18	139.31	139.10	129.42	129.99	128.25	146.00	158.27	148.97
	3	146.40	146.26	139.27	135.35	135.24	130.16	150.59	159.64	150.79
	5	148.47	144.14	137.73	135.14	128.08	124.01	150.57	157.42	147.85
n		400			800			1200		
depth		3	6	9	3	6	9	3	6	9
pm	1H	116.00	124.00	112.00	136.00	124.00	125.33	100.00	96.00	109.33
	2	159.20	161.28	151.87	173.28	162.01	159.37	137.33	136.18	146.86
	3	160.37	164.14	151.47	180.26	163.32	162.74	143.33	137.94	152.10
	5	157.70	163.67	149.50	183.50	167.82	162.38	142.24	135.63	152.78

Table 3: CG/OPT<sup>1</sup> 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	10.44(5)	142.99(4)	140.55(1)	26.48(5)	438.62(5)	1380.64(1)	72.17(5)	1495.4(1)	2070.31(1)
	3	127.25(5)	410.48(1)	dnf	113.07(4)	2425.62(1)	dnf	947.62(2)	dnf	dnf
	5	1931.05(3)	dnf	dnf	740.5(3)	dnf	dnf	1.6(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	160.45(4)	dnf	dnf	776.76(4)	dnf	dnf	399.73(5)	2203.21(3)	dnf
	3	dnf	dnf	dnf	234.62(2)	dnf	dnf	1212.43(3)	dnf	dnf
	5	dnf	dnf	dnf	dnf	dnf	dnf	2.63(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

which was embedded in a local search scheme. An extensive experimental phase was carried on in order to validate the proposed methods. The exact approach proved to be able to obtain optimal solutions in reasonable time on many instances. Even on high dimensional instances, where we did not find a certified optimum within the considered time limit, the returned solutions proved to be accurate when evaluated with respect to an upper bound. The heuristic approach provides high quality solutions in fast computational times, outperforming the Column Generation on some of the complex ones.

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 might bring to a direct comparison among resolution methods which consider discrete and continuous adjustable ranges models. New classes of instances, possibly related to real-world applications, will be also investigated.

## References

- [1] P. Berman, G. Calinescu, C. Shah, and A. Zelikovsky, Power Efficient Monitoring Management in Sensor Networks, Proceedings of the Wireless Communications and Networking Conference '04, pp. 2329 - 2334 (2004).
- [2] 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).
- [3] 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).
- [4] 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).
- [5] 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).
- [6] 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).
- [7] 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).
- [8] 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).

- [9] 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).
- [10] 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).
- [11] 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).
- [12] 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).
- [13] 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).
- [14] S. Slijepcevic and M. Potkonjak, Power Efficient Organization of Wireless Sensor Networks, IEEE International Conference on Communications, Volume 2, pp. 472-476 (2001).
- [15] 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).
- [16] 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).
- [17] 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).