

---

## Minimum-cost sensor arrangement for achieving wanted coverage lifetime

---

Jie Wang\* and Ning Zhong

Department of Computer Science,  
University of Massachusetts,  
Lowell, MA 01854, USA  
E-mail: wang@cs.uml.edu  
E-mail: zning@cs.uml.edu  
\*Corresponding author

**Abstract:** Suppose we need to watch a set of targets continuously for a required period of time, and suppose we choose any number of sensors from a fixed set of sensor types and place them at selected sites. We want to find a sensor arrangement to achieve the required coverage lifetime such that the total (monetary) cost of the chosen sensors is minimum. This is an NP-hard problem. We approach this problem by modelling it as an Integer Linear Programming problem. We devise a polynomial-time approximation algorithm to this problem with a proven approximation guarantee. We observe that the actual approximation ratios are small in practice. We then present a time-slotted scheduling to produce a timetable for each sensor. We also consider a special case with only one type of sensor that can watch one target at a time. We present a different method for solving this problem.

**Keywords:** Wireless Sensor Networks; WSNs; minimum-cost sensor arrangement; lifetime requirement; Integer Linear Programming; ILP; approximation algorithm; time-slotted scheduling; simulation.

**Reference** to this paper should be made as follows: Wang, J. and Zhong, N. (2008) 'Minimum-cost sensor arrangement for achieving wanted coverage lifetime', *Int. J. Sensor Networks*, Vol. 3, No. 3, pp.165–174.

**Biographical notes:** Jie Wang is a Professor and Chair in the Department of Computer Science at University of Massachusetts Lowell, USA. He is also the Director of the Center for Network and Information Security at the university. He received a Doctor of Philosophy in computer science from Boston University, USA, in 1991. He received an ME in Computer Science in 1985 and his BSc in Computational Mathematics in 1982, both from Zhongshan (Sun Yat-Sen) University, Guangzhou, China.

Ning Zhong received a Doctor of Science in Computer Science from the University of Massachusetts Lowell, USA, in 2007. He received an ME in Computer Science and Engineering in 2002 and a BSc in Computer Science in 1999, both from Zhongshan (Sun Yat-Sen) University, Guangzhou, China. During his study at the University of Massachusetts Lowell, he received several scholarships including Elmer C. Matthews Memorial Scholarship and Professor Charlie Steele Memorial Scholarship, and he was named Outstanding Graduate Student. He is currently employed at Nortel, Massachusetts, USA.

---

## 1 Introduction

Recent advancement in sensing, embedded processing, and wireless communication technologies has made it feasible to construct large-scale Wireless Sensor Networks (WSNs) to solve problems that are hard to solve using traditional technologies. Surveilling a set of targets, for example, is a typical application of WSNs. A WSN is a network of sensor nodes, where a typical sensor node is a device of small size that consists of a sensor board, an embedded processor, a wireless communication device and a battery power unit. A sensor node's communication range is often much larger than its sensing range. In the sequel, we will refer to a sensor

node as a sensor. The scale of WSN deployments has reached 10,000 sensors in a single application (Culler, 2005).

Target surveillance has two basic requirements: the coverage requirement and the lifetime requirement. The coverage requirement requires that each target be watched continuously by at least one sensor at any time in a duration equal to the minimum lifetime of sensors used in the network, where the lifetime of a sensor is the period from the time when the sensor is turned on to the time when it is no longer functional because of battery power depletion. The lifetime requirement requires that each target be watched by at least one sensor at any given moment for a preset period of time, which may be several times over

the lifetime of any sensor available. The lifetime of a WSN is the duration from the time when it starts its service to the time when it is unable to provide full coverage because of sensors' power depletion.

Replacing batteries is an easy solution for achieving the required lifetime when sensors are easy to reach, but is unfeasible when sensors are in hard-to-reach locations, such as in hostile areas, in tall trees, underneath bridges over turbulent water, to name just a few. When replacing batteries is formidable, one is forced to find an alternative solution. For example, one may deploy extra sensors, synchronise their internal clocks, and schedule them to take turn watching the targets. A sensor scheduling is a plan that sets certain sensors to the sleep mode when their service is not needed, and awakes them to the active mode otherwise. We assume that a sensor in the sleep mode consumes negligible energy.

### 1.1 Previous research

WSN coverage has been a much studied topic with a long list of literature. In particular, we note that previous investigations on WSN lifetime have mainly dealt with sensor scheduling to maximise the lifetime of an existing WSN to cover a given set of targets (see, e.g. Cardei and Du, 2005; Cardei et al., 2005; Gui and Mohapatra, 2004; Hsin and Liu, 2004; Liu et al., 2005). Such WSNs are typically random deployments of a large number of sensors, providing redundant sensors in the network to work with.

For example, Cardei and Du studied how to maximise the lifetime of a randomly deployed WSN under the assumption that a sensor can watch any number of targets in its sensing range (Cardei and Du, 2005). Their method partitions sensors into a maximum number of disjoint set covers, so that sensors in disjoint sets can be set active in succession. That is, right before sensors in the current set use up their battery power, sensors in the next set will be activated and take over the current set of sensors. Active sensors watch all the targets while non-active sensors are in the sleep mode to conserve energy. They showed that finding the maximum disjoint set covers is NP-hard and they presented a heuristic for computing disjoint set covers. Cardei et al. extended this investigation by allowing a sensor's lifetime to be sliced into several sections (Cardei et al., 2005) such that the same sensor in different sections may be placed in different set covers. Obtaining a maximum set cover in this fashion is still NP-hard, and they devised heuristics.

For another example, Liu et al. studied how to maximise the lifetime of a given WSN in which there is only one type of sensors available with unit lifetime, and a sensor can only watch one target at a time (Liu et al., 2005). They provided a polynomial-time algorithm to solve the problem. Their method uses a Linear Programming (LP) model to obtain the maximum lifetime of a given WSN as well as its workload matrix. The workload matrix is then converted to a doubly stochastic matrix, which can be decomposed using Birkhoff and von Neumann's method (Bondy and Murty, 1976, p.76) into a sequence of schedule matrices for achieving the maximum coverage duration.

### 1.2 Coverage with a given lifetime requirement

We introduce in this paper an inverse WSN lifetime problem, where the network lifetime  $T$  is given as a parameter. Our

task is to find a sensor arrangement to achieve the required coverage lifetime with minimum monetary cost on sensors. (Note: The cost of a sensor may also be defined by other measures such as sensor weight or size.) We call this problem *minimum-cost WSN lifetime problem* and denote it by MCL. In particular, suppose we are given a fixed set of sensor types with different sensing capacity, cost, or lifetime, a set of stationary targets, and a set of sites to place sensors, where a target and a site are points in a bounded 3D region. The set of sites and the set of targets may or may not intersect. Our task is to select a subset from the given set of sites and a set of sensors from the given sensor types, find a mapping from the selected sites to the selected sensors, which represents a sensor placement, such that every target is watched by at least one sensor at any given moment for the entire duration of  $T$  time, and the total cost of the selected sensors is minimum among all possible solutions. The mapping may be many-to-one, meaning that several sensors may be placed at one site. This may be done, for example, using a special rack that bundles a few sensors together. The maximum number of sensors allowed at one site is given as a parameter. With such a sensor arrangement we can build a WSN using a star topology to watch the given set of targets continuously for at least  $T$  time, where each sensor communicates directly to a base station.

For the most part of this paper, we assume that a sensor can watch any number of targets in its sensing range at the same time. It is common practice to assume that a sensor's sensing range is a finite sphere with the sensor at the center point. To guarantee existence of a full coverage, we assume that for each target  $g$  there is at least one site  $s$  sufficiently near  $g$  so that a sensor placed at  $s$  can effectively watch  $g$ .

MCL contains the minimum-cost WSN coverage problem as a special case, which has been studied by several authors (see e.g. Chakrabarty et al., 2002; Sahni and Xu, 2005; Wang and Zhong, 2006). It is known that minimum-cost coverage is NP-hard (Chakrabarty et al., 2002). Thus, MCL is also NP-hard.

A naive approach to finding an approximation solution to MCL would use an efficient approximation algorithm  $\mathcal{L}$  to the minimum-cost coverage problem as follows: For each sensor at each site in the solution produced by  $\mathcal{L}$ , place  $\lceil T/e_{\min} \rceil$  sensors of the same type at the same site and activate these sensors in succession, where  $e_{\min}$  is the minimum lifetime of sensors used in the solution. This approach, however, will fail if the number of sensors allowed at one site is less than  $\lceil T/e_{\min} \rceil$ .

### 1.3 Our solution

We take the following approach to find a strong approximation. We first formulate an Integer Linear Programming (ILP) model to obtain an optimal solution to MCL and show that any feasible solution to the ILP model provides a sensor arrangement for achieving the required lifetime. We then loosen the integer constraint to allow variables to take real values, and solve a corresponding LP problem (which has fast algorithms). We devise an efficient algorithm to convert the optimal LP solution to a feasible ILP solution with a proven approximation guarantee. The approximation ratio of our algorithm depends on the LP solutions, which may be large in the worst case. But

we observe that in practice the approximation ratios are small. This is because optimal LP solutions in practice would tend to include sufficiently large number of variables whose values close to being integers, providing a nice starting point for the approximation algorithm to produce a strong approximation.

We carry out a large number of numerical experiments on sets of points that are uniformly generated at random, including sets that are extremely dense. The approximation ratio in each experiment is all less than 1.14. We use *lpsolve* (Berkelaar et al., 2005) to solve ILP and LP. While it takes days of running time to solve ILP on some of the experiments, the running time of our approximation algorithm in each case takes only a fraction of a second.

We use a time-slotted method to schedule sensors, where a sensor may change between the sleep mode and the active mode several times. Each mode change in a sensor is called *sensor slicing*. Frequent slicing is undesirable, for it adds overhead for a sensor to wake up. Our experiments show that the number of sensors that are sliced, compared to the number of sensors deployed, is small.

Finally, we consider a special case of MCL in the same line of Liu et al. (2005), where there is only one type of sensors available and a sensor can only watch one target at a time. We provide a different method for solving this problem using perfect matching in graph theory.

The rest of this paper is structured as follows. We define our ILP model for solving MCL in Section 2. In Section 3, we present our approximation algorithm, prove its correctness, time complexity and approximation ratio. In Section 4, we present a time-slotted scheduling algorithm. In Section 5, we present numerical results from experiments, compare approximation solutions and the optimal solutions, and validate our approach. In Section 6, we present a different method for the special case where there is only one type of sensor and a sensor can only watch one target at a time. In Section 7, we present final remarks and open problems.

## 2 MCL and the ILP model

Let  $t_1, \dots, t_\ell$  be  $\ell$  fixed sensor types with sensing radius  $r_1, \dots, r_\ell$ , where  $r_1 < r_2 < \dots < r_\ell$ . Let  $c_v$  denote the (monetary) cost of a type- $t_v$  sensor. Let  $e_v$  denote the lifetime of a type- $t_v$  sensor, which is equal to the duration from the time when the sensor is turned on in the active mode with full power to the time when it is no longer functional because of power depletion. Recall that a sensor can watch any number of targets in its sensing range at the same time.

Let  $R$  denote a set of targets and  $S$  a set of sites, where  $R$  and  $S$  may or may not intersect. Let  $|R| = n_R$  and  $|S| = n_S$ . We label targets and sites as  $R = \{1, 2, \dots, n_R\}$  and  $S = \{1, 2, \dots, n_S\}$ . Let  $m \geq 1$  denote the maximum number of sensors of one type that can be placed at one site. Let  $T > 0$  be the lifetime parameter. Without loss of generality, we assume that  $T$  and  $e_v$ 's are positive integers.

Denote by  $d(i, j)$  the Euclidean distance between target  $i \in R$  and site  $j \in S$ . We say that  $S$  is *fully usable* to  $R$  if for every site  $j$  there is at least one target  $i$  and one sensor

type  $t_v$  such that  $d(i, j) \leq r_v$ . We assume that the given set of sites  $S$  is fully usable to  $R$ . Let

$$E_v = \{(i, j) \mid 0 \leq d(i, j) \leq r_v, i \in R, j \in S\},$$

$$v = 1, \dots, \ell$$

$$E_v[i] = \{j \mid (i, j) \in E_v\},$$

$$i = 1, \dots, n_R$$

$$E'_v[j] = \{i \mid (i, j) \in E_v\},$$

$$j = 1, \dots, n_S$$

Let  $x_j^v$  be integer variables such that

$$x_j^v = \begin{cases} m', & \text{if } m' \text{ type-}t_v \text{ sensors are placed at site } j \\ 0, & \text{otherwise} \end{cases}$$

where  $j \in S$ ,  $v = 1, \dots, \ell$  and  $m'$  is an integer between 1 and  $m$ .

MCL can be formulated as the following ILP problem:

$$\begin{aligned} & \text{Minimise } \sum_{j=1}^{n_S} \sum_{v=1}^{\ell} c_v x_j^v \\ & \text{s.t. } (\forall i \in R) \sum_{v=1}^{\ell} \sum_{j \in E_v[i]} e_v x_j^v \geq T \quad (1) \\ & (\forall j \in S) 0 \leq x_j^v \leq m \quad (2) \end{aligned}$$

It is easy to see that if Constraint 1 is not satisfied for some target  $i$ , then this target cannot be watched for the duration of  $T$  time. We will show in Proposition 1 that Constraint 1 ensures that each target can be watched for at least  $T$  time by a simple time-slotted scheduling.

Constraint 2 ensures that the number of sensors of each type that can be placed at one site is at most  $m$ .

A *feasible* solution to the ILP model is a set of  $x_j^v$ 's with positive integer values that satisfy all the constraints. A feasible solution may or may not satisfy the objective function.

**Proposition 1:** *Any feasible solution to the ILP model guarantees a sensor arrangement such that all targets in  $R$  can be watched by the sensors in the solution for a duration of at least  $T$  time.*

*Proof:* Let  $\{t_{u_1}, \dots, t_{u_\kappa}\}$  be the set of sensor types used in the given feasible solutions, where  $1 \leq u_\kappa \leq \ell$ . Let  $e = \gcd(T, e_{u_1}, \dots, e_{u_\kappa})$  be the length of a time slot.

Choose an arbitrary target  $i$ . From the given feasible solution we have a subset  $\{j_1, \dots, j_\eta\}$  of sites and a multiset  $\{v_1, \dots, v_\eta\}$  of sensor types, where  $1 \leq j_p \leq n_S$ ,  $1 \leq v_p \leq \ell$ ,  $1 \leq p \leq \eta$  and  $v_1 \leq \dots \leq v_\eta$ , such that  $j_p \in E_{v_p}[i]$  and  $x_{j_p}^{v_p} \geq 1$  with

$$e_{v_1} x_{j_1}^{v_1} + \dots + e_{v_\eta} x_{j_\eta}^{v_\eta} \geq T$$

If target  $i$  is the first one selected, we simply use the following sequence of sensors to watch target  $i$ :

$$\underbrace{x_{j_1}^{v_1}, \dots, x_{j_1}^{v_1}}_{\text{many}}, \dots, \underbrace{x_{j_\eta}^{v_\eta}, \dots, x_{j_\eta}^{v_\eta}}_{\text{many}}$$

where type- $t_{v_p}$  sensors are placed at site  $j_p$ . These sensors are set to the sleep mode prior to the service commencement

of the network, which will be activated in succession starting from the first type- $t_{v_1}$  sensor at location  $j_1$ . Right before the current sensor uses up its battery power, the next sensor in the sequence will be activated. The total time target  $i$  is watched by these sensors is equal to  $e_{v_1}x_{j_1}^{v_1} + \dots + e_{v_\eta}x_{j_\eta}^{v_\eta} \geq T$ .

If target  $i$  is not the first one selected, then some sensors in the feasible solution that cover target  $i$  may have been used to cover previous targets, and so they have been scheduled. If there are no gaps in the time interval of length  $T$  from the previous scheduled sensors that also cover target  $i$ , then target  $i$  is under surveillance of these sensors. Otherwise, there must be sensors in the given feasible solution that cover target  $i$  but have not been used to cover previous targets. It is easy to see that the length of these gaps are multiples of  $e$  and so we can slice the unused sensors to fill in these gaps. Repeat this procedure until no targets are left, and we will obtain a sensor arrangement to watch all the targets continuously for a duration of at least  $T$  time.  $\square$

### 3 Efficient approximation

We loosen the integer constraints in the ILP model by allowing integer variables  $x_j^v$  to take real values between 0 and  $m$ . This gives rise to an LP model. LP models can be solved using fast algorithms. We then convert the optimal LP solution to a feasible solution to the ILP problem. We will first present our approximation algorithm assuming only two types of sensors are available (i.e.  $\ell = 2$ ). The conversion in the case when  $\ell = 1$  is straightforward. We will then generalise the algorithm to the general case (i.e.  $\ell > 2$ ).

#### 3.1 With two types of sensors

Denote by  $A$  and  $B$  the two types of sensors. Let

$$\{x_j^{A,*}, x_j^{B,*} \in [0, m] \mid j = 1, \dots, n_S\}$$

be an optimal solution to the LP model with

$$\text{OPT}_{\text{LP}} = \sum_{j=1}^{n_S} (c_A x_j^{A,*} + c_B x_j^{B,*})$$

To convert this solution to an integer solution, we construct a bipartite graph  $G = (R, S; E)$  as follows: For any pair of vertices  $(i, j) \in R \times S$ , we connect them if and only if target  $i$  can be covered by a sensor placed at site  $j$ . Let  $r_A < r_B$ . There are two types of connections and we use edge labels to mark these connections. That is, for each  $(i, j) \in R \times S$ , if  $d(i, j) \leq r_A$ , connect  $i$  and  $j$  and label the edge with  $A$ . If  $r_A < d(i, j) \leq r_B$ , connect  $i$  and  $j$  and label the edge with  $B$ .

The idea of our approximation algorithm is as follows: Start from a point  $i \in R$  with the largest degree in the bipartite graph  $G$  and select the smallest number of largest possible values  $x_{j_A}^{A,*}$  and  $x_{j_B}^{B,*}$ , where  $j_A \in E_A[i]$  and  $j_B \in E_B[i]$ , such that

$$\sum \left( e_A \lceil x_{j_A}^{A,*} \rceil + e_B \lceil x_{j_B}^{B,*} \rceil \right) \geq T$$

For these variables, set  $x_{j_A}^A = \lceil x_{j_A}^{A,*} \rceil$  and  $x_{j_B}^B = \lceil x_{j_B}^{B,*} \rceil$ . The selected variable  $x_{j_A}^A$  corresponds to placing  $x_{j_A}^A$  type- $A$

sensors at site  $j_A$  and  $x_{j_B}^B$  corresponds to placing  $x_{j_B}^B$  type- $B$  sensors at site  $j_B$ . Remove point  $i$  and any other point in  $R$  that is also covered by these sensors. Continue this process until all points in  $R$  are removed.

---

#### Algorithm $\mathcal{A}$ :

1. Let  $\mathbb{S} = \emptyset$  and  $L$  be a sorted list of points in  $R$  in non-increasing order according to their degrees.
2. Select  $i \in L$  such that vertex  $i$  has the largest degree. Let

$$H_{A,i} = \{x_j^{A,*} \mid j \in E_A[i]\},$$

$$H_{B,i} = \{x_j^{B,*} \mid j \in E_B[i]\},$$

$$H_i = H_{A,i} \cup H_{B,i}.$$

Sort  $H_i$  according to the values of the variables in non-increasing order. Let  $\sigma_i$  be the smallest number of variables selected from  $H_i$  such that

$$\sum_{j \in E_{\sigma_i}} \left( \chi_A^i(j) \cdot e_A \cdot \lceil x_j^{A,*} \rceil + \chi_B^i(j) \cdot e_B \cdot \lceil x_j^{B,*} \rceil \right) \geq T,$$

where  $E_{\sigma_i} = \{j \mid j \text{ is the subscript of the selected } \sigma_i \text{ variables}\}$ .

$$\chi_A^i(j) = \begin{cases} 1, & \text{if } j \in E_A[i], \\ 0, & \text{otherwise,} \end{cases}$$

$$\chi_B^i(j) = \begin{cases} 1, & \text{if } j \in E_B[i], \\ 0, & \text{otherwise.} \end{cases}$$

Denote by  $h_{j_1}, \dots, h_{j_{\sigma_i}}$  these  $\sigma_i$  variables.

3. Include  $h_{j_1}, \dots, h_{j_{\sigma_i}}$  in  $\mathbb{S}$ . Let  $W$  be the intersection of the subsets of all points in  $R$  covered by corresponding sensors placed at sites  $j_u \in S$  for  $u = 1, \dots, \sigma_i$ . Remove  $W$  from  $L$ . That is, set

$$\mathbb{S} = \mathbb{S} \cup \{h_{j_1}, \dots, h_{j_{\sigma_i}}\},$$

$$L = L - W.$$

4. Repeat Step 2 until  $L = \emptyset$ .
  5. Set the value of each variable  $h_j$  in  $\mathbb{S}$  to  $\lceil h_j \rceil$ , and the value of each variable not in  $\mathbb{S}$  to 0.
- 

Let  $k_A$  denote the maximum number of sites that fall within the radius  $r_A$  of any target in  $R$ , and  $k_B$  the maximum number of sites within the radius  $r_B$  of any target in  $R$ . Namely,

$$k_A = \max_{i \in R} \{|E_A[i]|\}, \quad k_B = \max_{i \in R} \{|E_B[i]|\}$$

Similarly, let  $\tau_A$  denote the maximum number of targets that fall within the radius  $r_A$  of any site in  $S$  and  $\tau_B$  the maximum number of targets within the radius of  $r_B$  of any site in  $S$ . Namely,

$$\tau_A = \max_{j \in S} \{|E'_A[j]|\}, \quad \tau_B = \max_{j \in S} \{|E'_B[j]|\}$$

Let  $k = \max\{k_A, k_B\}$  and  $\tau = \max\{\tau_A, \tau_B\}$ .

**Theorem 1:** Algorithm  $\mathcal{A}$  converts an optimal solution to the LP problem to a feasible solution to the ILP problem in  $O(n_R \log n_R \log k + n_R(k \log k + T^2 + T\tau) \log(m+1))$  time.

*Proof:* It is evident from the construction of the algorithm that Algorithm  $\mathcal{A}$  provides a feasible solution to the ILP model. Note that a previously selected variable at Step 2 may be selected again at a later time. The set  $W$  constructed at Step 3 in each iteration of the algorithm contains at least one point  $i$  selected from the current list  $L$  and  $W$  consists of points that can be covered by the  $\sigma_i$  sensors in the current selection. Removing  $W$  from  $L$  in each iteration, the algorithm will stop within  $n_R$  iterations.

It is straightforward to see that graph  $G$  can be constructed in  $O(kn_R)$  time and the descending list  $L$  can be constructed in time

$$O(n_R \log n_R \log k)$$

At Step 2,  $i$  can be found in constant time,  $H_i$  can be constructed in  $O(k \log(m+1))$  time,  $H_i$  can be sorted in  $O(k \log k \log(m+1))$  time and the  $\sigma_i$  variables in  $H_i$  can be selected in  $O(\sigma_i^2 \log(m+1))$  time. Since  $\sigma_i \leq T$  for all  $i$ , the  $\sigma_i$  variables in  $H_i$  can be selected in time

$$O((k \log k + T^2) \log(m+1))$$

At Step 3, the set  $W$  can be constructed in time

$$O(\sigma_i \tau \log(m+1)) \leq O(T \tau \log(m+1))$$

Removing  $W$  from  $L$  can be carried out, using a proper data structure for implementing  $L$ , in time

$$O(\sigma_i \log(m+1)) \leq O(T \log(m+1))$$

We note that Steps 2 and 3 can only be iterated at most  $n_R$  times. Thus, the running time of Algorithm  $\mathcal{A}$  is

$$O(n_R \log n_R \log k + n_R(k \log k + T^2 + T\tau) \log(m+1)) \quad \square$$

Let  $\text{OPT}_{\text{ILP}}$  denote the optimal solution of the original ILP problem and  $\Delta$  the solution of Algorithm  $\mathcal{A}$ . Let  $x_j^{A,\Delta}$  and  $x_j^{B,\Delta}$  denote the values assigned to variables  $x_j^A$  and  $x_j^B$ , respectively, at Step 5 in Algorithm  $\mathcal{A}$ . Let

$$\beta = e_A k_A + e_B k_B$$

**Theorem 2:**  $\Delta \leq m\beta \text{OPT}_{\text{ILP}}$ .

*Proof:* We first note that  $\text{OPT}_{\text{LP}} \leq \text{OPT}_{\text{ILP}}$ . We then show that for any variable  $h$  put in  $\mathbb{S}$ , we must have

$$h \geq \frac{1}{\beta} \quad (3)$$

Assume that  $h_l$  is the variable put in  $\mathbb{S}$  with respect to point  $i$  selected at Step 2, where  $1 \leq l \leq \sigma_i$ . Assume that  $h_1 \geq \dots \geq h_{\sigma_1}$ . Let  $t_l$  be the sensor type associated with variable  $h_l$ .

It follows from Constraint 1 to the LP model that

$$\sum_{j \in E_A[i]} e_A x_j^{A,*} + \sum_{j \in E_B[i]} e_B x_j^{B,*} \geq T$$

Since  $h_1$  is the largest value, we have

$$h_1 \geq \frac{T}{e_A k_A + e_B k_B}$$

It implies that

$$\sum_{j \in E_A[i]} e_A x_j^{A,*} + \sum_{j \in E_B[i]} e_B x_j^{B,*} - e_{t_1} h_1 \geq T - e_{t_1} [h_1],$$

with  $h_2$  being the largest value in the remaining variables. Thus,

$$\begin{aligned} \sum_{j \in E_A[i]} e_A x_j^{A,*} + \sum_{j \in E_B[i]} e_B x_j^{B,*} - e_{t_1} h_1 \\ \leq h_2 (e_A k_A + e_B k_B - e_{t_1}) \end{aligned}$$

This implies that

$$h_2 \geq \frac{T - e_{t_1} [h_1]}{e_A k_A + e_B k_B - e_{t_1}}$$

A straightforward induction leads us to the following inequality:

$$h_l \geq \frac{T - \sum_{j=1}^{l-1} e_{t_j} [h_j]}{e_A k_A + e_B k_B - \sum_{j=1}^{l-1} e_{t_j}}, \quad l = 1, \dots, \sigma_i$$

By Algorithm  $\mathcal{A}$  we know that for any  $l \leq \sigma_i$ :

$$\sum_{j=1}^{l-1} e_{h_j} [h_j] \leq T - 1$$

Thus,

$$T - \sum_{j=1}^{l-1} e_{t_j} [h_j] \geq 1$$

This implies that

$$h_l \geq \frac{1}{e_A k_A + e_B k_B}, \quad l = 1, \dots, \sigma_i$$

and so inequality 3 is true. Hence,

$$\begin{aligned} \text{OPT}_{\text{LP}} &= \sum_{j=1}^{n_S} \left( c_A x_j^{A,*} + c_B x_j^{B,*} \right) \\ &\geq \sum_{x_j^{A,*} \in \mathbb{S}} c_A x_j^{A,*} + \sum_{x_j^{B,*} \in \mathbb{S}} c_B x_j^{B,*} \\ &\geq \frac{1}{m\beta} \left( \sum_j c_A x_j^{A,\Delta} + \sum_j c_B x_j^{B,\Delta} \right) \\ &= \frac{\Delta}{m\beta} \end{aligned}$$

It follows from  $\text{OPT}_{\text{LP}} \geq \text{OPT}_{\text{ILP}}$  that

$$\Delta \leq m\beta \text{OPT}_{\text{ILP}} \quad \square$$

Remark 1: By a more sophisticated analysis we can prove a slightly better approximation ratio. For instance, in the special case where  $c_A = c_B = m = 1$ , we can show that the approximation ratio is  $k_A + k_B - T + 1$ . This is because in this case  $\sigma_i = T$  and we can show, similar to the proof of Theorem 2, that

$$h_l \geq \frac{T - l + 1}{k_A + k_B - l + 1}, \quad l = 1, \dots, T$$

It follows from Constraint 1 that  $k_A + k_B \geq T$ . Thus, we have

$$(T - l)(k_A + k_B - T) \geq 0, \quad 1 \leq l \leq T$$

Hence,

$$(T - l + 1)(k_A + k_B - T) - (k_A + k_B) + T \geq 0$$

$$(T - l + 1)(k_A + k_B - T + 1) - (k_A + k_B) + T - (T - l + 1) \geq 0$$

$$(T - l + 1)(k_A + k_B - T + 1) \geq k_A + k_B - l + 1$$

This implies that

$$\frac{T - l + 1}{k_A + k_B - l + 1} \geq \frac{1}{k_A + k_B - T + 1}$$

Hence, we obtain an approximation ratio of  $k_A + k_B - T + 1$ . The general case for an improved ratio is tedious and we omit it here.  $\square$

The approximation ratio  $m\beta$  in Theorem 2 is a worst-case upper bound. The actual approximation ratio depends on the optimal solution  $x_j^{A,*}$  and  $x_j^{B,*}$ . The theoretical upper bound may be reached only if in Constraint 1 in the LP model, for each target  $i$  selected in Algorithm  $\mathcal{A}$  and for each  $j \in E_A[i]$  and  $j \in E_B[i]$ , we have  $x_j^{A,*} = x_j^{B,*} = T/(e_A|E_A[i]| + e_B|E_B[i]|)$ . This situation is rare. In practice, the actual approximation ratio tends to be very small. This is because the LP solutions often provide sufficiently good values of  $x_j^{A,*}$  and  $x_j^{B,*}$  to select from for the approximation algorithm; namely, the values of the selected variables by Algorithm  $\mathcal{A}$  are either integers or close to being integers. In particular, we can show the following result.

Proposition 2: If there is a real number  $\epsilon > 0$  such that  $x_j^{A,*} \geq \lceil x_j^{A,*} \rceil / (1 + \epsilon)$  and  $x_j^{B,*} \geq \lceil x_j^{B,*} \rceil / (1 + \epsilon)$  for all  $x_j^{A,*} \in \mathbb{S}$  and  $x_j^{B,*} \in \mathbb{S}$ , then  $\Delta \leq (1 + \epsilon)OPT_{ILP}$ .

Proof: Note that

$$\Delta = \sum_{x_j^{A,*} \in \mathbb{S}} c_A \lceil x_j^{A,*} \rceil + \sum_{x_j^{B,*} \in \mathbb{S}} c_B \lceil x_j^{B,*} \rceil$$

and so we have

$$\begin{aligned} OPT_{LP} &= \sum_{j=1}^{n_R} \left( c_A x_j^{A,*} + c_B x_j^{B,*} \right) \\ &\geq \sum_{x_j^{A,*} \in \mathbb{S}} c_A x_j^{A,*} + \sum_{x_j^{B,*} \in \mathbb{S}} c_B x_j^{B,*} \\ &\geq \sum_{x_j^{A,*} \in \mathbb{S}} c_A \frac{\lceil x_j^{A,*} \rceil}{1 + \epsilon} + \sum_{x_j^{B,*} \in \mathbb{S}} c_B \frac{\lceil x_j^{B,*} \rceil}{1 + \epsilon} = \frac{\Delta}{1 + \epsilon} \end{aligned}$$

Thus,  $\Delta \leq (1 + \epsilon)OPT_{LP} \leq (1 + \epsilon)OPT_{ILP}$ .  $\square$

### 3.2 With more than two types of sensors

We generalise Algorithm  $\mathcal{A}$  to allow sensors of more than two types; that is,  $\ell > 2$ . Generalising Algorithm  $\mathcal{A}$  to the case of  $\ell = 1$  is straightforward.

As in Section 3.1, we construct a bipartite graph  $G = (R, S; E)$  such that a pair of vertices  $(i, j) \in R \times S$  are connected with a label  $v$ ,  $v = 1, \dots, \ell$ , if and only if  $r_{v-1} < d(i, j) \leq r_v$  holds, where  $r_0 = 0$ . If  $d(i, j) = 0$ , we also connect  $i$  and  $j$  with label 1.

Let  $k_v$  denote the maximum number of sites on which a type- $t_v$  sensor can be placed to cover a target. Similar to Algorithm  $\mathcal{A}$  we have the following approximation algorithm  $\mathcal{B}$ .

**Algorithm  $\mathcal{B}$ :**

1. let  $\mathbb{S} = \emptyset$  and  $L$  be a sorted list of target points in  $R$  in non-increasing order according to their degrees.
2. Select  $i \in L$  such that vertex  $i$  has the largest degree. Let

$$H_{v,i} = \{x_j^{v,*} \mid j \in E_v[i]\},$$

$$v = 1, \dots, \ell,$$

$$H_i = \bigcup_{v=1}^{\ell} H_{v,i}.$$

Sort  $H_i$  according to the values of the variables. Let  $\{h_{j_1}, \dots, h_{j_{\sigma_i}}\}$  be the smallest set of variables in  $H_i$  with the largest possible values (not in any particular order) such that the summation of the product of  $\lceil h_{j_i} \rceil$ ,  $i = 1, \dots, \sigma_i$  and corresponding  $e_v$ ,  $v = 1, \dots, \ell$  is greater than or equal to  $T$ .

Assume that  $\sigma_{v,i}$  variables of  $h_{j_1}, \dots, h_{j_{\sigma_i}}$  are from  $H_{v,i}$ .

3. Let  $U = \{v \in [1, \ell] \mid (\exists u \in [1, \sigma_i]) [h_{j_u} \in H_v]\}$ . Let

$$\mathbb{S} = \mathbb{S} \cup \{h_{j_1}, \dots, h_{j_{\sigma_i}}\},$$

$$L = L - \bigcap_{u=1}^{\sigma} \bigcap_{v \in U} \left[ \bigcap_{h_{j_u} \in H_v} E_v[j_u] \right].$$

4. Repeat Step 3 until  $L = \emptyset$ .
5. Set the value  $h_j$  in  $\mathbb{S}$  to  $\lceil h_j \rceil$ , and the value of each variable not in  $\mathbb{S}$  to 0.

Let  $x_j^{v,*} \in [0, m]$ ,  $j = 1, \dots, n_S$ , be an optimal LP solution. Let

$$OPT_{LP} = \sum_{j=1}^{n_R} \sum_{v=1}^{\ell} c_v x_j^{v,*}$$

Let  $\Delta_B$  denote the solution of Algorithm  $\mathcal{B}$ . Let

$$k_v = \max_{i \in R} \{|E_v[i]|\}, \quad k = \max_{1 \leq v \leq \ell} \{k_v\}$$

$$\tau_v = \max_{j \in S} \{|E'_v[j]|\}, \quad \tau = \max_{1 \leq v \leq \ell} \{\tau_v\}$$

$$\gamma = \sum_{v=1}^{\ell} e_v k_v$$

Similar to Theorems 1 and 2 it is straightforward to show the following result, and we omit its proof.

**Theorem 3:** *Algorithm B converts an optimal solution to the LP problem to a feasible solution to the ILP problem in  $O(n_R \log n_R \log k + n_R(k \log k + T^2 + T\tau) \log(m+1))$  time with  $\Delta_B \leq m\gamma OPT_{ILP}$ .*

Proposition 2 is also true for  $\ell \neq 2$  with a straightforward modification of its proof; namely, if there is a real number  $\epsilon > 0$  such that for all  $x_j^{v,*} \in \mathbb{S}: x_j^{v,*} \geq \lceil x_j^{v,*} \rceil / (1 + \epsilon)$ , where  $v = 1, \dots, \ell$ , then  $\Delta \leq (1 + \epsilon)OPT_{ILP}$ .

#### 4 Scheduling algorithms

Any feasible solution to the ILP model provides for each target a set of sensors and a set of sites to place these sensors that cover this target. Yet, we still need to schedule sensors' active time and sleep time as shown in Proposition 1. We implement Proposition 1 using a best-fit greedy strategy to reduce the number of sensors that need to be sliced. Recall that  $e = \gcd(T, e_{u_1}, \dots, e_{u_k})$  is the length of a time slot. Thus, a sensor's lifetime is a multiple of  $e$ . If a type- $v$  sensor has been set active for  $t$  slots, then the number of its unused time slots equals  $(e_v - t)/e$ .

The best-fit strategy is simple: In the set of sensors in the feasible solution used to cover the current target, if some of them were already chosen and scheduled to cover previously selected targets that leave gaps of no coverage of the current target in the time interval of length  $T$ , we select sensors from the feasible solution to fill in these gaps with the best-fit unused time slots. That is, if there are sensors whose unused time slots are smaller than or equal to the number of slots contained in the gap, use such a sensor to fill the gap such that the length of the new gap, if any, is minimised. If such sensors do not exist, select a sensor with the smallest number of unused time slots, fill in the gap completely, and save the unused time slots for later use.

---

##### Algorithm C (Best-fit scheduling)

The algorithm outputs a time table for each target, specifying which sensor at which site should be active in which time slot. Each sensor is identified by its type and location.

1. Let  $L$  be a descending list of targets according to the number of sensors in the feasible solution that cover them. The size of this list will be reduced at each step until it becomes empty. When a sensor is placed in a time slot, it will be set active in that time slot; otherwise, it will be set asleep.
2. For each target  $i$ ,  $i = 1, \dots, n_R$ , let  $L_i$  be a descending list of sensors in the feasible solution that cover  $i$  according to the number of sensors' unused time slots.
3. Let  $i$  be the first element in the (current) list  $L$  and remove it from  $L$ . Then remove from  $L_i$  those sensors that have been scheduled in previous targets' time tables, and use the same scheduling of these sensors to cover  $i$ .

4. Let  $[a, b]$  be the first time interval in  $i$ 's time table in the time interval of  $[0, T]$  that is not occupied by any sensor. Let  $d = b - a$ , then  $d$  is a multiple of  $e$ . Choose from  $L_i$  a type- $j$  sensor such that  $|e_j - d| \leq |e_{j'} - d|$  for any other type- $j'$  sensor  $\in L_i$ . We have the following three cases:
  - Case (1):  $|e_j - d| = 0$ . Remove  $j$  from  $L_i$ , and place sensor  $j$  in  $[a, b]$ .
  - Case (2):  $e_j > d$ . Slice sensor  $j$  into two durations, the first duration is  $d$  and the second duration is  $e_j - d$ . Place sensor  $j$  in  $[a, b]$ , set  $e_j = e_j - d$ , and put  $e_j$  back into  $L_i$  as  $j$ 's remaining energy level.
  - Case (3):  $e_j < d$ . Place sensor  $j$  into the time interval  $[a, a + e_j]$ , whose length is also a multiple of  $e$ . Set  $a = a + e_j$ .
5. Repeat Step 4 until  $i$ 's time interval  $[0, T]$  is completely filled.
6. Repeat Step 3 until  $L = \emptyset$ .

---

Proposition 3: *Algorithm C runs in  $O(n_R^2 T \log(n_R T))$  time.*

*Proof:* Step 1 can be carried out in  $O(n_R^2 T + n_R \log n_R \log(n_R T))$  time. Step 2 can be carried out in  $O(n_R^2 T \log(n_R T))$  time. Step 3 can be done in constant time (we can do so by marking each sensor when it is placed in a time slot). Steps 4 and 5 can be done in  $O(n_R T)$  time. Step 6 can be done in  $O(n_R^2 T)$  time.  $\square$

The best-fit strategy works well, for we show that, through a large number of numerical experiments, the number of sensors needed to be sliced is small.

## 5 Implementations and experiments

We present numerical experiments to compare the optimal solutions produced by the ILP model and the approximation solutions produced by approximation Algorithm B with different target densities, from sparse to dense. We also provide numerical results to show how many sensors are sliced in Algorithm C, and provide a scheduling example.

We use *lpsolve 5.5* (Berkelaar et al., 2005) to solve ILP and LP problems. We write a C program to implement our approximation algorithm and scheduling algorithm. We performed all of our experiments on a Dell Optiplex GX 260 PC, equipped with a 2.26 GHz Pentium-4 CPU and 1 GB RAM.

### 5.1 Comparison of ILP and LP solutions

All targets are generated in a  $10 \times 10$  area uniformly and independently at random. We assume that three types of sensors are available. Their attributes are listed in Table 1, with the same distance unit, monetary cost unit and time unit.

We consider  $T = 12$  and  $T = 20$ , respectively to evaluate our scheduling Algorithm C. We present numerical results obtained from our experiments with  $m = 1$  and  $m = 3$ .

We generate sensor sites randomly around each target such that at least  $\lceil T/m \rceil$  sites are within range  $r_A$  of this target and the total number of sites do not exceed  $\lceil T/m \rceil \times n_R$ .

In particular, for each target located at point  $P$ , we generate an angle and a distance in  $(0, r_A]$  uniformly and independently at random to produce a random site. We repeat this procedure to produce  $\lceil T/m \rceil$  sites. We then run our programme for 10 times and present numerical results in the case with the median total cost. Table 2(a)–(d) compare the cost of the optimal solutions given by ILP and the approximation solutions produced by Algorithm  $\mathcal{B}$ .

**Table 1** Sensor attributes

Type	Range	Cost	Lifetime
A	0.5	100	2
B	0.75	150	3
C	1	200	4

**Table 2** Cost comparisons

Number of targets	Optimal solution	Appr. solution	Actual appr. ratio
<i>(a) ILP versus Algorithm <math>\mathcal{B}</math> with <math>m = 1</math> and <math>T = 12</math></i>			
50	13,800	13,800	1
100	16,000	16,000	1
150	17,400	17,400	1
200	23,400	23,400	1
250	22,000	22,000	1
300	21,500	24,400	1.135
<i>(b) ILP versus Algorithm <math>\mathcal{B}</math> with <math>m = 1</math> and <math>T = 20</math></i>			
50	21,050	21,050	1
100	30,000	30,300	1.01
150	35,700	36,200	1.01
200	34,800	36,750	1.06
250	37,000	38,650	1.04
300	38,150	41,950	1.1
<i>(c) ILP versus Algorithm <math>\mathcal{B}</math> with <math>m = 3</math> and <math>T = 12</math></i>			
50	15,000	15,000	1
100	16,200	16,200	1
150	18,400	18,900	1.03
200	18,800	20,800	1.11
<i>(d) ILP versus Algorithm <math>\mathcal{B}</math> with <math>m = 3</math> and <math>T = 20</math></i>			
50	23,000	23,600	1.03
100	29,000	30,200	1.04
150	35,600	37,750	1.06
200	35,650	38,150	1.07

Solving ILP takes days of running time when the number of targets are 200 and above, while solving LP and executing Algorithm  $\mathcal{B}$  in each case takes only a fraction of seconds.

Although the proven worst-case approximation ratio upper bounds are large, we can see from these numerical experiments that the actual approximation ratios are either 1 or near 1, indicating that our approach provides strong approximation solutions.

Table 3(a) and (b) show the percentages of the number of sensors that are sliced by scheduling Algorithm  $\mathcal{C}$  over the total number of sensors used in experiments shown in Table 2(a)–(d). These percentages are all small.

**Table 3** Percentage of sliced sensors in scheduling

Number of targets	$T = 12$		$T = 20$	
	ILP	LP	ILP	LP
<i>(a) <math>m = 1</math></i>				
50	0	0	0	0
100	0	0	0	0
150	1.1	1.1	1.9	0
200	0	0	0	2.9
250	4.1	4.1	0	0
300	7.8	9.8	7.2	4.0
<i>(b) <math>m = 3</math></i>				
50	0	0	0	0
100	0	0	0	0
150	1.3	1.2	2.1	2.5
200	1.1	1.1	1.7	2.8

## 5.2 A scheduling example

We present a scheduling example obtained from Algorithm  $\mathcal{C}$ . We generate 10 targets independently and uniformly at random in a  $10 \times 10$  area with three types of sensors of the same attributes listed in Table 1, where  $m = 3$ . Let  $T = 20$ . The length of a time slot is  $\gcd(20, 2, 3, 4) = 1$ . Around each target, at least  $\lceil T/m \rceil$  sites are randomly generated such that they are within the range  $r_A$  of this target. We run our programme for 10 times and present the case with the median total cost. Figure 1 shows the scheduling table of 86 sensors that work together to watch those 10 target.

## 6 A special case

In this section, we study a special case of MCL; that is, we assume that there is only one type of sensors available and each sensor can only watch one target at a time. This is the assumption used in Liu et al. (2005). This assumption applies to sensors whose sensing ranges are directional. Moreover, we assume that each site may be occupied by at most  $m$  sensors of each type ( $m \geq 1$ ). Without loss of generality, we assume that the sensor type has unit cost and unit lifetime with a sensing range  $r$ . Let  $s[i]$  be the set of sites that are within range  $r$  from target  $i$ ,  $i = 1, \dots, n_R$ . Let  $x_j$ ,  $j = 1, \dots, n_S$  be integer variables such that

$$x_j = \begin{cases} m', & \text{if } m' \text{ sensor is placed at site } j \in S \\ 0, & \text{otherwise} \end{cases}$$

where  $1 \leq m' \leq m$ .

We first construct a graph  $G = (R, E)$  on target points such that a pair of targets  $i$  and  $j$  are connected with an edge if and only if these vertices can be covered by one sensor, that is, if and only if  $d(i, j) \leq 2r$ , where  $d(i, j)$  is the Euclidean distance between  $i$  and  $j$ . We denote by  $\delta_i$  the degree of vertex  $i$  in  $G$ .

This special sensor coverage lifetime problem can be formulated as the following ILP problem:

$$\text{Minimise } \sum_{j=1}^{n_S} x_j \quad (4)$$

**Figure 1** Sensor scheduling example of MCL (see online version for colours)

Scheduling table of 86 sensors to watch 10 targets when $m=3$ and $T=20$																				
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Target 1	1	2	3	4	5	6	7	8	9	10										
Target 2		11				12				13			14			15				16
Target 3	17	18	19	20	21	22	23	24	25	26										
Target 4	27	28	29	30	31	32	33	34	35	36										
Target 5		11				12				13			14			15				16
Target 6	37	38	39	40	41	42	43	44	45	46										
Target 7	47	48	49	50	51	52	53	54	55	56										
Target 8	57	58	59	60	61	62	63	64	65	66										
Target 9	67	68	69	70	71	72	73	74	75	76										
Target 10	77	78	79	80	81	82	83	84	85	86										

**Key**

	Type-A Sensor's Battery Life
	Type-B Sensor's Battery Life
	Type-C Sensor's Battery Life

$$\text{s.t. } \forall i \in R : \sum_{j \in s[i]} x_j \geq (\delta_i + 1)T \quad (5)$$

$$\forall j \in S : x_j \leq m \quad (6)$$

where Constraint 5 says that each target  $i$  should be covered by at least  $(\delta_i + 1) \cdot T$  sensors and the Constraint 6 requires that each site should be placed at most  $m$  sensors of each type.

**Theorem 4:** Any feasible solution to the ILP defined in the objective function 4 and Constraints 5 and 6 guarantees that there exists a sensor scheduling such that each target in  $R$  is watched by sensors at least  $T$  time.

*Proof:* We follow the idea of finding a perfect matching in a bipartite graph. We define a bipartite graph  $G'(I, J; E')$ , where  $I$  is a subset of targets and  $J$  subset of sensors such that there is an edge between  $i \in I$  and  $j \in J$  if and only if target  $i$  is covered by sensor  $j$ . A feasible solution to the above ILP problem generates such a bipartite graph  $G'(I, J; E')$  such that  $\delta'_i = (\delta_i + 1)T$  for each  $i \in I$ , where  $\delta'_i$  is the vertex degree of  $i$  in  $G'$ .

Define a perfect matching in  $G'$  to be an injective mapping  $f : I \rightarrow J$  such that for every  $i \in I$  there is an edge with endpoints  $i$  and  $f(x)$ . For any subset  $A \subset I$ , define  $\partial A$  to be the set of all vertices in  $J$  that are endpoints of edges with one endpoint in  $A$ . Hall's matching theorem (see, e.g. Bondy, 1976) says that there exists a perfect matching  $f : I \rightarrow J$  in  $G'$  if and only if for every subset  $A \subset I$ ,  $|\partial A| \geq A$ . Given the fact that every sensor has a unit lifetime, in order to make the system lifetime at least  $T$ , we must find  $T$  mutually disjoint perfect matchings from  $G'$ . This can be done efficiently by removing sensor nodes from  $J$  whenever they are saturated by a perfect matching. Each time a perfect matching is found,  $n_R$  sensors in  $J$  and all the associated edges are removed from  $G'$ . These  $n_R$  sensors will be active in the same time slot in our schedule. This process can be done at least  $T$  times. To see this, we note that each time when  $n_R$  sensors in  $J$  and all the associated edges are removed from  $G'$ , for each  $i \in I$ ,  $\delta'_i$  decreases by at most  $\delta_i + 1$ . Therefore, in the first  $T$  rounds of finding perfect matchings,  $|\partial A| \geq A$  holds. It follows from Hall's theorem that a

perfect matching can be found in each round. Thus, we can schedule sensors in succession to guarantee that the network lifetime is at least  $T$ .  $\square$

Since the sensor type has a unit lifetime, the proof to Theorem 4 provides a scheduling, which is obtained by finding  $T$  mutually disjoint perfect matchings from the bipartite graph  $G'(I, J; E')$ . Slicing is not needed in this case. The running time of finding a perfect matching in  $G'$  is a polynomial  $O(n^{3/2} \sqrt{|E'|/\log n})$ , where  $n = n_R + n_S$  (see, e.g. Bondy, 1976).

Similar to Section 3 we loosen the integer constraints in the ILP model by allowing integer variables  $x_j$  to take real values between 0 and  $m$ . We first solve this LP model using a fast algorithm. We then convert the optimal LP solution to a feasible solution to the ILP problem.

Algorithm  $\mathcal{D}$  is a straightforward conversion of Algorithm  $\mathcal{A}$  to the case with  $\ell = 1$  and  $e_\ell = 1$ .

---

**Algorithm  $\mathcal{D}$ :**

1. Let  $\mathbb{S} = \emptyset$  and  $L$  be a sorted list of points in  $R$  in non-increasing order according to their degrees.
2. Select  $i \in L$  such that vertex  $i$  has the largest degree. Let  $H_i = \{x_j^* \mid j \in s[i]\}$ , sort  $H_i$  according to the values of the variables in non-increasing order. Let  $\sigma_i$  be the smallest number of variables selected from  $H_i$  such that

$$\sum_{j \in E_{\sigma_i}} [x_j^*] \geq (\delta_i + 1) \cdot T,$$

where  $E_{\sigma_i} = \{j \mid j \text{ is the subscript of the selected } \sigma_i \text{ variables}\}$ , Denote by  $h_{j_1}, \dots, h_{j_{\sigma_i}}$  these  $\sigma_i$  variables.

3. Include  $h_{j_1}, \dots, h_{j_{\sigma_i}}$  in  $\mathbb{S}$ . Let  $W$  be the intersection of the subsets of all points in  $R$  covered by corresponding sensors placed at sites  $j_u \in S$  for  $u = 1, \dots, \sigma_i$ . Remove  $W$  from  $L$ . That is, set

$$\mathbb{S} = \mathbb{S} \cup \{h_{j_1}, \dots, h_{j_{\sigma_i}}\},$$

$$L = L - W.$$

4. Repeat Step 6 until  $L = \emptyset$ .
5. Set the value of each variable  $h_j$  in  $\mathbb{S}$  to  $\lceil h_j \rceil$ , and the value of each variable not in  $\mathbb{S}$  to 0.

---

Theorem 5: Algorithm  $\mathcal{D}$  provides a feasible solution  $\Delta_D$  to the ILP problem in  $O(n_R \times n_S)$  time, and

$$\Delta_D \leq mkOPT_{ILP}$$

where  $k$  is the maximum number of sites that fall in the sensing range of any target in  $R$ .

## 7 Final remarks and open problems

We have introduced in this paper the problem of arranging sensors with minimum cost to cover a set of targets continuously for a required period of time. This is an NP-hard problem. We presented an ILP model for this problem and devised an efficient approximation algorithm with a proven approximation guarantee.

Our model assumes that

- 1 sensors do not fail as long as they have sufficient power remained
- 2 sensors can communicate directly to a base station
- 3 sensors can be placed precisely at their calculated positions.

In reality, however, sensors may be prone to failure due to component or software malfunctions, severe weather or other harsh physical conditions where they are deployed. Thus, it is interesting to study how to build a reliable WSN with minimum sensor cost for achieving a given coverage lifetime using sensors that may fail. An ILP model can be formulated for this problem based on reliability theory, and efficient approximation algorithms can be found (Wang and Yu, 2006).

In reality, sensors may not always be able to communicate directly to a base station, especially in a large-scale application. Thus, it is interesting to study how to build a multihop WSN with minimum sensor cost for achieving a given coverage lifetime. By multihop, it means that each sensor in the network can either communicate directly with a base station or is on a path of sensors to a base station, where adjacent nodes are within communication range. In this study, we may need to consider the fact that a sensor used also as a relay node may have a much shorter lifetime than the lifetime of a sensor of the same type not used as a relay node, for it will cost much more power for a sensor to transmit packets. We may also deploy sensors to act only as a relay node. How to formulate an ILP model for this problem remains open.

In reality, it may not always be feasible to place sensors precisely at their calculated positions, especially in applications where sensors are air dropped. Thus, it is interesting to study how to achieve a given coverage lifetime with minimum sensor cost if sensors are placed near their calculated positions. This problem also remains open.

Finally, it is interesting to study how to design an analytical model for constructing a WSN with minimum sensor cost that satisfies all of the four requirements of coverage, lifetime, reliability and connectivity.

## Acknowledgements

We thank Benyuan Liu for several interesting discussions and Karen Daniels for a useful comment. We also thank Adam Elbirt for interesting discussions on power consumption in MICA2 motes.

## References

- Berkelaar, M., Notebaert, P. and Eikland, K. (2005) *LP Solve: Linear Programming Code, version 5.5*, Eindhoven University of Technology, The Netherlands.
- Bondy, J.A. and Murty, U.S.R. (1976) *Graph Theory with Applications*, The Macmillan Press Ltd.
- Cardei, M. and Du, D-Z. (2005) 'Improving wireless sensor network lifetime through power aware organization', *ACM Wireless Networks*, Vol. 11, No. 3, pp.333–340.
- Cardei, M., Thai, M., Li, Y. and Wu, W. (2005) 'Energy-efficient target coverage in wireless sensor networks', *Electronic Version of Proceedings of IEEE Infocom*.
- Chakrabarty, K., Iyengar, S.S., Qi, H. and Cho, E. (2002) 'Grid coverage for surveillance and target location in distributed sensor networks', *IEEE Transactions on Computing*, Vol. 51, No. 12, pp.1448–1453.
- Clouqueur, T., Phipatanasuphorn, V., Ramanathan, P. and Saluja, K.K. (2002) 'Sensor deployment strategy for target detection', *Proceedings of 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pp.42–48.
- Culler, D. (2005) 'Sensor networks—the next tier of the Internet', *Keynote speech in the Workshop on Sensor Networks—What is Real and What Lies Ahead*, Boston University, 18 November.
- Gui, C. and Mohapatra, P. (2004) 'Power conservation and quality of surveillance in target tracking sensor networks', *Proceedings of the 10th ACM Mobicom*, pp.129–143.
- Hsin, C.F. and Liu, M. (2004) 'Network coverage using low duty-cycled sensors: random and coordinated sleep algorithms', *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, pp.433–442.
- Liu, H., Wan, P., Yi, C-W., Jia, X., Makki, S. and Niki, P. (2005) 'Maximal lifetime scheduling in sensor surveillance networks', *Electronic Version of Proceedings of IEEE Infocom*.
- Sahni, S. and Xu, X. (2005) 'Algorithms for wireless sensor networks', *International Journal on Distributed Sensor Networks*, Vol. 1, No. 1, pp.35–56.
- Wang, J. and Yu, Z. (2006) 'Reliable sensor arrangement for achieving wanted coverage lifetime with minimum cost', Manuscript.
- Wang, J. and Zhong, N. (2006) 'Efficient point coverage in wireless sensor networks', *Journal of Combinatorial Optimization*, Vol. 11, pp.291–305.

## Note

<sup>1</sup>This research was supported in part by NSF under grant CCF-04080261.