HELMUT-SCHMIDT-UNIVERSITÄT
UNIVERSITÄT DER BUNDESWEHR HAMBURG
LEHRSTUHL FÜR BETRIEBSWIRTSCHAFTSLEHRE,
INSBES. LOGISTIK-MANAGEMENT
Prof. Dr. M. J. Geiger

Arbeitspapier / Research Report
RR-11-02-01 · February 2011 · ISSN 2192-0826

The cover scheduling problem arising in wireless sensor networks

André Rossi[1,2], Marc Sevaux[1,3,*], Alok Singh[2] and Martin Josef Geiger[3]

[1]Université de Bretagne-Sud, Lab-STICC, Lorient, France.
[2]University of Hyderabad, Department of Computer and Information Sciences, Hyderabad, India.
[3]Helmut-Schmidt-University, Logistics Management Department, Hamburg, Germany.
[*]Corresponding author: marc.sevaux@univ-ubs.fr

Contents

Abstract

One critical problem in wireless sensor networks is to assemble the sensors in different covers in order to solve different objectives such as maximum duration of the network lifetime of minimizing the coverage breach. The output of these problems is a set of covers with a duration of usage. Another problem coming just after is how to schedule the covers. Most of the time, this part is left unsolved as the challenges appear less important than generating the covers. Our current work will describe the wireless sensor network cover scheduling problem (WSN-CSP), a non-linear mathematical model, a MILP model and a heuristic (CSH) that solves this problem. To improve the quality of the solutions, we also propose a memetic algorithm (CSMA). Both approaches will be tested on a large set of instances. Moreover, some complexity results and theorical properties are also given in the paper.

1 Introduction

The use of wireless sensor networks (WSNs) has been increasing at a rapid pace in remote or hostile environments for data gathering [1]. This includes battlefield surveillance, fire monitoring in forests, or undersea tsunami monitoring. In such environments, sensors are usually deployed in an *ad hoc* manner or at random when it is not possible to place them precisely. To compensate for this random deployment, a greater number of sensors are deployed than what is actually required. This also increases the fault tolerance as some targets are redundantly covered by multiple sensors.

Several objectives are usually aimed in WSN. The most two important are "minimizing the cover breach under bandwidth constraint" (MCBB) or its dual problem "maximize the network lifetime under bandwidth constraint" (MNLB). Both problems are very efficiently solved in a previous framework based on column generation [8]. Sensors are gathered into a number of subsets (not necessarily disjoint) such that sensors in each subset cover the targets. Such subsets are referred to as *covers*. Covers are activated sequentially in a mutually exclusive manner, *i.e.*, at any instant of time only sensors belonging to the active cover are used, whereas all other sensors are not. Using covers significantly increases network lifetime for two main reasons. First, sensors consume much more energy in an active state than in an inactive state [7]. Second, a sensor battery has been shown to last longer if it oscillates frequently between active and inactive states [2, 3].

The output of the previous framework [8] is the design of the covers and the time these covers should be used to satisfy one of the two objectives. Once the covers are determined, another problem appears and it consists in scheduling these covers. Most the time, this schedule part is left unanswered since producing the covers is much more important and difficult than scheduling them. While trying to solve the scheduling problem we observed that some of the targets where not covered for quite a significant time.

1.1 Example

In a small example depicted on figure 1, five covers have been generated. Gray boxes represent the coverage of the targets. In the current scheduling (in natural order), target T_4 is not covered for 8 consecutive time units. Of course, this can be improved by changing the scheduling order. Figure 2 gives, for the same example, another order for which the objective value is 5 (due to targets T_2 and T_3).

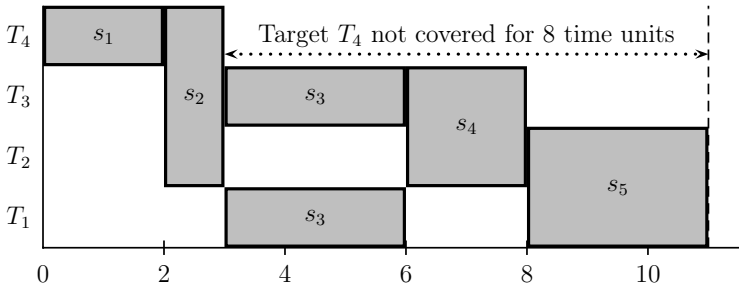

Figure 1: A first scheduling whose objective value is 8. Target T_4 is not covered for 8 consecutive time units because of successive scheduling of covers C_3, C_4 and C_5 (in any order).

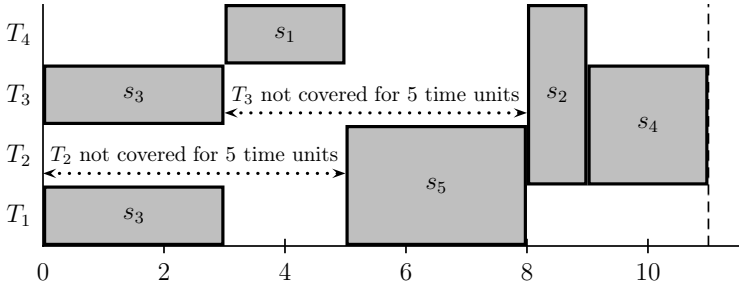

Figure 2: An improved scheduling of the same covers gives the objective value of 5. Two targets T_2 and T_3 are concerned.

1.2 Problem definition and notations

From this simple example, one can see that the problem is rather simple to understand. Given a set of covers, an usage duration of these covers, and the targets covered, find the best schedule that minimize the longest period of an uncovered target. This problem will be refereed as the wireless sensor network cover scheduling problem (WSN-CSP).

More formally, the WSN-CSP can be defined as follows. Let m be the number of targets, and q the number of covers. For all $j \in \{1, \ldots, q\}$ the duration of cover s_j is denoted by p_j, and we refer to it as a processing time like in the scheduling problems. For all $k \in \{1, \ldots, m\}$, C_k is the set of all covers that cover target k. For all $j \in \{1, \ldots, q\}$, D_j is the set of the targets that are covered by cover s_j. The cover scheduling problem is to schedule the covers so as to minimize Δ_{max}, the maximum period of time during which a target is not covered.

In our small example, $m = 4$, $q = 5$, $p = \{2, 1, 3, 2, 3\}$, $C_1 = \{s_3, s_5\}$, $C_2 = \{s_2, s_4, s_5\}$, $C_3 = \{s_2, s_3, s_4\}$ and $C_4 = \{s_1, s_2\}$. We have also $D_1 = \{4\}$, $D_2 = \{2, 3, 4\}$, $D_3 =$

$\{1,3\}$, $D_4 = \{2,3\}$ and $D_5 = \{1,2\}$.

1.3 Organization of the paper

The rest of the paper is organized as follows. Section 2 will formally define the problem as a non-linear programming formulation. Section 3 will present some complexity results of the problem. Section 4 will then give some important properties that can be stated. A MILP formulation of the problem is given in section 5. A heuristic together with a memetic algorithm will be presented in sections 6 and 7. Section 8 will report the numerical experiments and the last section will conclude the paper and give some perspectives.

2 A non-linear mathematical formulation for the WSN-CSP

For the sake of clarity, we introduce two dummy covers s_0 and s_{q+1} with a zero duration ($p_0 = p_{q+1} = 0$) and that are part of C_l for all $l \in \{1,\ldots,m\}$. s_0 starts at time 0, s_{q+1} ends at time LT. The sets Q, Q' and M will be also used in the model: $Q = \{0,\ldots,q+1\}$, $Q' = \{1,\ldots,q+1\}$ and $M = \{1,\ldots,m\}$.

The following variables will be used in the model: $x_{j,i}$ is a boolean variable that is set to 1 if and only if cover j is in position i in the solution (the solution is a semi-active single-machine schedule) for all $(j,i) \in Q \times Q$, $pseq_i$ is the duration of the cover in position i, $cseq_i$ is the completion time of the cover in position i, $d_{i,l}$ is the distance of the cover in position i to the next coverage occurrence of target l, for all $(i,l) \in Q \times M$ and Δ_{max} is the maximum duration over all the targets during which a target is not covered by a cover.

The model below will solve the WSN scheduling problem.

$$\text{Minimize} \quad \Delta_{max} \tag{1}$$

$$\sum_{i=0}^{q+1} x_{j,i} = 1 \qquad \forall j \in Q \tag{2}$$

$$\sum_{j=0}^{q+1} x_{j,i} = 1 \qquad \forall i \in Q \tag{3}$$

$$pseq_i = \sum_{j=0}^{q+1} x_{j,i} p_j \qquad \forall i \in Q \tag{4}$$

$$cseq_i - pseq_i \geq cseq_{i-1} \qquad \forall i \in Q' \tag{5}$$

$$d_{i,l} = \min_{z \in C_l | z \leq i-1} (cseq_i - pseq_i - cseq_z) \qquad \forall (i,l) \in Q' \times M \tag{6}$$

$$\Delta_{max} \geq d_{i,l} \qquad \forall (i,l) \in Q \times M \tag{7}$$

$$x_{0,0} = 1 \text{ and } cseq_0 = 0 \tag{8}$$

$$x_{q+1,q+1} = 1 \text{ and } cseq_{q+1} = LT \tag{9}$$

$$x_{j,i} \in \{0,1\} \qquad \forall (j,i) \in Q \times Q \tag{10}$$

$$pseq_i \geq 0 \qquad \forall i \in Q \tag{11}$$

$$cseq_i \geq 0 \qquad \forall i \in Q \tag{12}$$

$$d_{i,l} \geq 0 \qquad \forall (i,l) \in Q \times M \tag{13}$$

$$\Delta_{max} \geq 0 \tag{14}$$

Constraint (2) states that cover s_j is allocated exactly one position, for all $j \in Q$ and constraint (3) that each position i is allocated exactly one cover, for all $i \in Q$. The duration of the cover in position i, $pseq_i$ is defined by constraint (4). Constraint (5) states that the cover in position i starts after the completion of the cover in position $i-1$ for all i in Q'. For all $(i,l) \in Q' \times M$, $d_{i,l}$ is defined as the time elapsed between completion time of the last cover in C_l and the starting time of the cover in position i as presented in constraint (6). Constraint (7) imposes that Δ_{max} is larger than any $d_{i,l}$. Cover s_0 is in position 0 in the solution and starts at time 0 (Constraint (8)) and cover s_{q+1} is in position $q+1$ in the solution and starts at time LT, Constraint (9). Constraint (10), (11), (12), (13) and (14) fixes the integrality and signs of variables.

It can be seen that this model is non-linear because of constraint (6) that defines $d_{i,l}$. Due to this constraint, the model will not be used in practice to solve the WSN-CSP instances that are definitively too large.

3 Complexity analysis

3.1 The cover scheduling problem in a particular case

We consider the following particular case of the cover scheduling problem: $m = 2$ targets, the q sensors are partitioned in $O_1 + O_{12}$ where O_1 is the set of all covers that cover target t_1 only, and O_{12} is the set of all covers that cover targets t_1 and t_2. We assume that $O_1 \cup O_{12} = \{1, \ldots, q\}$, with $O_1 \cap O_{12} = \emptyset$. The cardinality of these sets is defined by $q_1 = |O_1|$ and $q_{12} = |O_{12}|$, so $q = q_1 + q_{12}$.

Moreover, the time during which cover j is used (it is denoted by p_j) is integer for all covers $j \in \{1, \ldots, q\}$, it is equal to 1 for all the covers in O_1, it is greater than or equal to 1 for the covers in O_{12}.

As a consequence, the network lifetime LT is also integer and is equal to

$$LT = q_1 + \sum_{j \in O_{12}} p_j$$

Target t_1 is covered by all the covers, but t_2 is covered by the covers in O_{12} only. The total amount of time during which t_2 is not covered is equal to q_1. This duration can be split in at most $q_{12} + 1$ time intervals. Since all the covers duration are integer, the minimum duration of the maximum time interval during which t_2 is not covered is $\left\lceil \frac{q_1}{q_{12}+1} \right\rceil$.

Lemma 1 *If we assume that the covers in O_{12} are numbered in $\{1, \ldots, q_{12}\}$, an optimal schedule to the covering scheduling problem in this particular case is returned by setting the starting time s_j of cover j in O_{12} as follows:*

$$s_j = j \left\lceil \frac{q_1}{q_{12}+1} \right\rceil + \sum_{z<j} p_z \qquad \forall j \in \{1, \ldots, q_{12}\}$$

The covers in O_1 are scheduled in between, in any order.

Proof 1 *$\left\lceil \frac{q_1}{q_{12}+1} \right\rceil$ is the minimum value for the maximum duration during which t_2 is not covered. It can be seen that the first cover in O_{12} stats at time $\left\lceil \frac{q_1}{q_{12}+1} \right\rceil$, and that the time elapsed between two consecutive covers in O_{12} is $\left\lceil \frac{q_1}{q_{12}+1} \right\rceil$ by construction. Now, we can checked that the time elapsed between the completion time of the last cover in O_{12} and LT is also less than or equal to $\left\lceil \frac{q_1}{q_{12}+1} \right\rceil$. This duration can be written as*

$$\begin{aligned} LT - q_{12}\left\lceil \frac{q_1}{q_{12}+1} \right\rceil + \sum_{z=1}^{q_{12}} p_z &= q_1 - q_{12}\left\lceil \frac{q_1}{q_{12}+1} \right\rceil \\ &= (q_{12}+1)\frac{q_1}{q_{12}+1} - q_{12}\left\lceil \frac{q_1}{q_{12}+1} \right\rceil \\ &= q_{12}\left(\frac{q_1}{q_{12}+1} - \left\lceil \frac{q_1}{q_{12}+1} \right\rceil\right) + \frac{q_1}{q_{12}+1} \end{aligned}$$

Since $\left(\frac{q_1}{q_{12}+1} - \left\lceil \frac{q_1}{q_{12}+1} \right\rceil\right)$ *is negative or zero, we have*

$$\begin{aligned} LT - q_{12}\left\lceil \frac{q_1}{q_{12}+1} \right\rceil + \sum_{z=1}^{q_{12}} p_z &\leq \frac{q_1}{q_{12}+1} \\ &\leq \left\lceil \frac{q_1}{q_{12}+1} \right\rceil \end{aligned}$$

□

Remark 1 *It may happen that the computed starting time of the last cover in* O_{12} *is equal to* LT*. In that case this cover should start at any time* $t \leq LT - p_{q_{12}}$ *provided it does not overlap another cover.*

Finally, as all the covers in O_1 *have a duration of one unit of time, they can be scheduled in the* $q_{12}+1$ *time intervals, that all have an integer duration that is less than or equal to* $\left\lceil \frac{q_1}{q_{12}+1} \right\rceil$.

3.2 The cover scheduling problem is $\mathcal{NP}$-hard in the strong sense

In order to show that the cover scheduling problem is $\mathcal{NP}$-hard in the strong sense, we consider the following instance: $m = 2$ targets, the covers in O_{12} all have a one duration, the network lifetime is $LT = q_{12} + (q_{12}+1)B$, where B is a given integer with $B > 1$. The covers in O_1 have an integer duration $p_j \leq B$ for all $j \in O_1$, such that $\sum_{j \in O_1} p_j = (q_{12}+1)B$. This implies that the $q_{12}+1$ time intervals during which t_2 is not covered have a minimum maximum duration of B.

Determining whether or not the covers in O_1 can be scheduled in those $q_{12}+1$ time intervals which duration is B is equivalent to answer the decision problem version of the bin packing problem: does there exists a partition of O_1 in $q_{12}+1$ sets such that the sum of the durations of the covers in each set does not exceed B?

Since a particular instance of the cover scheduling problem is equivalent to the bin packing problem, the WSN-CSP is $\mathcal{NP}$-hard in the strong sense.

4 Properties

4.1 Lower bounds

First, it can be observed that Δ_{max} is lower bounded by the duration of any cover that does not cover all the targets:

$$\Delta_{max} \geq \max_{\substack{j \in \{1,\dots,q\} \\ |D_j| < m}} p_j$$

This lower bound is likely to be efficient for instances in which the breach rate is low. We remind that the breach rate is the ratio of non-covered targets at any time in a solution. A breach rate equal to zero means that all the targets are covered.

Furthermore, for a specific target, the duration for which this target is not covered is equal to the lifetime of the network minus the time this target is covered. This maximum time can be split, using the covers of that target in an equally distributed separation. In Figure 1, target T_1 is not covered for 5 units of time. Hence, by positioning wisely covers s_3 and s_5, the maximum continuous time of non-covering T_1 is $5/3 \approx 1.67$ time units. If we repeat this computation for all target and take the maximum, we can deduce that Δ_{max} is lower bounded by

$$\Delta_{max} \geq \max_{k \in \{1,\dots,m\}} \left(\frac{LT - \sum_{j \in C_k} p_j}{|C_k| + 1} \right)$$

This bound is likely to perform well for instances in which the breach rate is high.

So in conclusion, Δ_{max} is lower bounded by $\overline{\Delta}$

$$\overline{\Delta} = \max \left(\max_{\substack{j \in \{1,\dots,q\} \\ |D_j| < m}} p_j, \max_{k \in \{1,\dots,m\}} \left(\frac{LT - \sum_{j \in C_k} p_j}{|C_k| + 1} \right) \right) \tag{15}$$

Obviously, this bound is not tight.

4.2 Inner symmetry property

Let *sol* be a solution to the WSN-CSP (*sol* is a permutation of $\{1, \dots, q\}$, and let sol_k be the index of cover in the k^{th} position in *sol*). Then, solution *sol'* where the covers have reversed positions has the same objective value as *sol*. Example: $sol = (1, 3, 2, 5, 4)$ and $sol = (4, 5, 2, 3, 1)$.

This symmetry can be avoided in the MILP described in section 5 by limiting the search to solutions into which the index of the first cover is less than the index of the last cover:

$$\sum_{j=1}^{q} j x_{j,1} \leq 1 + \sum_{j=1}^{q} j x_{j,q} \tag{16}$$

Then cover q cannot be allocated position 1, nor can cover 1 be allocated position q, with no inconvenience.

4.3 Full covers

A cover is said to be a *full cover* if and only if it covers all the targets. Let $F \subset \{1, \ldots, q\}$ be the set of full covers. F is defined by

$$F = \{j \in \{1, \ldots, q\}, |D_j| = m\}$$

For any instance of the cover scheduling problem, we can consider that there exist two additional dummy covers (out of the set $\{1, \ldots, q\}$) that are scheduled at time 0 and at time LT (they both have a zero duration). These two covers s_0 and s_{q+1} have been used in the NLP formulation of section 2.

In addition, it can be seen that the actual duration of full covers has no impact on the objective function value (even though the numerical value of LT depends on the duration of full covers) because Δ_{max} is set by non-full covers only.

Lemma 2 *If* $|F| \leq \left\lfloor \frac{q-1}{2} \right\rfloor$*, then there exists an optimal solution with no two consecutive full covers, and no full-cover in position 1 nor in position* q*.*

Proof 2 *As* $|F| \leq \left\lfloor \frac{q-1}{2} \right\rfloor$*, the number of non-full covers is large enough for building a solution with no consecutive full covers, and no full covers in first or last position. Let sol be an optimal solution to the cover scheduling problem with either two consecutive full covers, or a full cover in the first position, of a full-cover in the last position. Then, one of these full covers can be scheduled between two non-full covers. By performing this modification, the objective function value can only decrease, as moving a full cover cannot enlarge the period of time during which a target is not covered.* □

4.4 Highlighting three symmetry properties

Without loss of generality, we now assume that $1 \leq |F| \leq \left\lfloor \frac{q-1}{2} \right\rfloor$, and that the covers have been re-indexed so that cover s_j is a full cover if and only if $j \leq |F|$. In other words, $F = \{1, \ldots, |F|\}$. Since the duration of full covers has no impact on the solution, they can be interchanged freely.

Let *sol* be a solution to the cover scheduling problem. We assume that full covers have been interchanged so that $sol_j < sol_{j+1}$ for all $j \in \{1, \ldots, |F| - 1\}$. The solution is split into $|F| + 1$ contiguous intervals as follows:

Interval 1: covers in position $\{1, \ldots, sol_1\}$

Interval 2: covers in position $\{sol_1 + 1, \ldots, sol_2\}$

Interval 3: covers in position $\{sol_2 + 1, \ldots, sol_3\}$

...

Interval $|F|$: covers in position $\{sol_{|F|-1}+1,\ldots,sol_{|F|}\}$

Interval $|F|+1$: covers in position $\{sol_{|F|}+1,\ldots,sol_q\}$

The following three symmetry properties are operations that leave the objective function unchanged, and generate solutions that are all considered equivalent to *sol*:

Symmetry 1 Swap two full covers

Symmetry 2 Reverse the non-full covers of an interval

Symmetry 3 Swap two intervals

The Symmetry 2 property is the global symmetry property mentioned earlier in section 4.2.

These operations performed on *sol* can be seen as undesirable symmetry properties, that any optimization approach (whether exact of approximate) would attempt to avoid them for shortening the size of the solution space to explore.

5 A MILP formulation of the WSN-CSP

The solution to this problem can be modeled as a permutation of $\{1,\ldots,q\}$. Unlike in section 2, the set Q is re-defined such as $Q=\{1,\ldots,q\}$. For all $(j,i)\in Q\times Q$, let $x_{j,i}$ be a binary variable that is set to one if and only if cover s_j is in position i in the solution, and which is set to zero otherwise. The numerical value for Δ_{max} is fixed by the total duration of a subset of consecutive covers such that none of them is covering at least one target.

The WSN-CSP of finding a schedule (i.e. a permutation of $\{1,\ldots,q\}$) that minimizes Δ_{max} can be stated as follows:

$$\text{Minimize } \Delta_{max} \tag{17}$$

$$\sum_{i=1}^{q} x_{j,i} = 1 \qquad \forall j \in Q \tag{18}$$

$$\sum_{j=1}^{q} x_{j,i} = 1 \qquad \forall i \in Q \tag{19}$$

$$\Delta_{max} \geq \overline{\Delta} \tag{20}$$

$$\sum_{j\in S}\sum_{i=1}^{|S|} p_j x_{j,i+r} \leq \Delta_{max} \qquad \forall S\subseteq Q, |S|>1, \; |\cup_{j\in S} D_j| < m, \forall r\in\{0,\ldots,q-|S|\} \tag{21}$$

$$x_{j,i}\in\{0,1\} \qquad \forall (j,i)\in Q\times Q \tag{22}$$

$$\Delta_{max}\geq 0 \tag{23}$$

The objective function is to minimize Δ_{max}. Constraints (18), (19) and (22) are those of the assignment problem. They enforce that x is a valid permutation of $\{1, \ldots, q\}$. Constraint (20) is enforcing the lower bound on Δ_{max} defined in equation (15) and (23) is a (useless) non-negativity constraint on Δ_{max}.

Constraint (21) states that the duration of any subset of covers such that there exists at least one target that is not covered by any of them, and that are scheduled consecutively, is less than Δ_{max}. More precisely, any subset S of covers that do not cover at least one target and that are scheduled consecutively can occupy positions 1 to $|S|$, or 2 to $|S| + 1$ ... up to positions $q - |S|$ to q. Note that the actual order of the covers in S does not matter. In constraint (21), r is an offset that allows to track the starting position of the corresponding subset of consecutive covers in the solution: $i + r$ is in $\{1, \ldots, q - |S|\}$ for $i = 1$. The case where $|S| = 1$ is dealt with, with constraint (20). In short, this constraint removes all possible pairs, triplets, quadruplets, etc at any position in the sequence.

Finally, this formulation can be seen as a combination of the assignment problem along with an exponential set of constraints, which is a similar situation to what happens when addressing the Travelling Salesman Problem with a MILP formulation for example.

5.1 Breaking symmetry

We can attempt to break the three symmetry properties by adding the following constraints to the MILP formulation.

Symmetry 1 Swap two full covers

The full covers are indexed in $\{1, \ldots, |F|\}$, and we enforce that they appear in the same order in the solution:

$$\sum_{i=1}^{q} i x_{j,1} + 1 \leq \sum_{i=1}^{q} i x_{j+1,i} \quad \forall j \in \{1, \ldots, |F| - 1\} \tag{24}$$

Symmetry 2 Reverse the non-full covers of an interval

The following constraint, which is too global, may prevents such swapings for intervals 1 and $|F + 1|$ only.

$$\sum_{j=1}^{q} j x_{j,1} + 1 \leq \sum_{j=1}^{q} j x_{j,q} \tag{25}$$

Then cover q cannot be allocated position 1, nor can cover 1 be allocated position q, with no inconvenience.

Symmetry 3 Swap two intervals

We use the same idea as in graph coloring, where the stables are sorted by non decreasing cardinality.

- The number of non-full covers in the first interval is $\sum_{i=1}^{q} ix_{1,i} - 1$.
- The number of non-full covers in interval $j \in \{2, \ldots, |F|\}$ is given by $\sum_{i=1}^{q} ix_{j,i} - \sum_{i=1}^{q} ix_{j-1,i} - 1$.
- The number of non-full covers in the last interval (i.e. interval $|F| + 1$) is $q - \sum_{i=1}^{q} ix_{|F|,i} - 1$.

Then the constraints are as follows

The number of non-full covers in the first interval is less than in the second one.

$$\sum_{i=1}^{q} ix_{1,i} \leq \sum_{i=1}^{q} ix_{2,i} - \sum_{i=1}^{q} ix_{1,i} \tag{26}$$

The number of non-full covers in interval $j - 1$ is less than in interval j, for all $j \in \{2, \ldots, |F|\}$.

$$\sum_{i=1}^{q} ix_{j,i} - \sum_{i=1}^{q} ix_{j-1,i} \leq \sum_{i=1}^{q} ix_{j+1,i} - \sum_{i=1}^{q} ix_{j,i} \quad \forall j \in \{2, \ldots, |F|\} \tag{27}$$

The number of non-full covers in interval $|F|$ is less than in the last interval (i.e. interval $|F| + 1$).

$$\sum_{i=1}^{q} ix_{|F|,i} - \sum_{i=1}^{q} ix_{|F|-1,i} \leq q - \sum_{i=1}^{q} ix_{|F|,i} \tag{28}$$

Of course, if two intervals have the same cardinality, these constraints won't help in breaking that symmetry.

In total, Equation (17)-(28) can be used for solving the WSN-CSP. But in practice, this complete model cannot be addressed directly because of constraints (21) whose number is exponential.

5.2 Ideas for addressing the problem

As for the TSP, it is intractable to enumerate all the constraints in (21) and new strong inequalities or facets should be defined to be able to solve the problem to optimality. Nevertheless, it might be interesting to consider (from a theoretical point of view) the following approach.

1. The master problem is defined by equations (17)−(20), (22) and (23).
2. Solve the master problem. Let x be its optimal solution (it is a permutation) and Δ_{max} be its optimal objective value.
3. Let Δ_{max}^{x} be the actual value of the maximum period of time during which a target is not covered in solution x.

4. If $\Delta_{max} < \Delta^x_{max}$, then the current solution to the master problem is not optimal to the original problem, as some constraints of type (21) are missing. Generate the most violated constraint of type (21) and add it to the master problem. Go to step 2.

5. Otherwise (i.e. if $\Delta_{max} = \Delta^x_{max}$), and the current solution to the master problem is optimal. Return x, Δ_{max} and stop.

At step 3, the most violated constraint of type (21) is the one corresponding to the covers that are responsible for Δ^x_{max}, so this constraint is easy to find for a given solution x.

An additional constraint Let $\overline{\Delta_{max}}$ be the objective value of any feasible solution. Then it can be deduced that any subset S of covers which total duration is strictly greater than $\overline{\Delta_{max}}$ cannot be scheduled consecutively:

$$\sum_{j \in S} \sum_{i=1}^{|S|} x_{j,i+r} < |S| \quad \forall S \subseteq Q, |S| > 1, \sum_{j \in S} p_j > \overline{\Delta_{max}} \quad \forall r \in \{0, \ldots, q - |S|\} \qquad (29)$$

This new set of constraints (29) can be used in an interactive manner at step 4 of the previous method or with any other solution given by a heuristic or a metaheuristic (see section 6 and 7).

6 A heuristic approach (CSH)

The proposed cover scheduling heuristic (CSH) for WSN-CSP takes its inspiration from the lower bound introduced in Section 4. Targets are sorted by decreasing minimum non coverage time, i.e. the targets that are the most likely to be responsible for the longest non coverage period of time are processed first. In this heuristic approach, the non-overlapping constraint on covers is first relaxed. So the covers are allocated unfeasible starting times for minimizing the longest period of non coverage time of targets. More precisely, for each non-processed target k, the covers that cover it but that are still unscheduled are scheduled so as to minimize the longest non coverage duration. When all the covers have been scheduled, they are sorted by increasing (unfeasible) starting times, and are finally scheduled according to that sequence.

We consider the cover scheduling problem in its most general form. Let $Q = \{0, \ldots, q+1\}$ be the set of covers where two dummy covers s_0 and s_{q+1} have been added to the data. For all $j \in Q$ cover s_j is used $p_j > 0$ units of time ($p_0 = p_{q+1} = 0$). For all $j \in Q$ the starting time of cover s_j will be denoted by st_j (cover s_0 starts at time 0, and cover s_{q+1} starts at time LT). For all targets $k \in M = \{1, \ldots, m\}$, C_k is the set of covers that cover target k (The two dummy covers s_0 and s_{q+1} cover all the targets and their status is always *scheduled*).

First, the targets are sorted by decreasing uncovered time. For all targets $k \in M$, the uncovered time UT_k is the total time during which it is not covered divided by the number of covers that cover it minus one.

$$UT_k = \frac{\sum_{j \notin C_k} p_j}{|C_k| - 1} \quad \forall k \in M$$

Note that $|C_k| \geq 2$ for all k because of covers s_0 and s_{q+1}. Thus, $\max_{k \in \{1,\ldots,m\}} UT_k$ is a lower bound of the objective function.

Algorithm 1 will describe the Cover Scheduling Heuristic (CSH). We will need the following variables in the algorithm. Let $sched_k$ be the number of covers in C_k that have already been scheduled (for $k = 1$, $sched_1 = 2$ because of the two dummy covers). SC_k is the set of the scheduled covers in C_k. Initially for $k = 1$, $SC_1 = \{0, q+1\}$. UC_k is the set of the unscheduled covers in C_k. Initially for $k = 1$, $UC_1 = C_1 \backslash \{0, q+1\}$. C_k is then partitioned in two disjoint subsets $C_k = SC_k \cup UC_k$, but UC_k might possibly be empty with no inconvenience. By definition, $sched_k = |SC_k|$, for all $k \in M$.

In Algorithm 1, we need to compute w_z for all $z \in \{1, \ldots, sched_k - 1\}$. This value is given by equation (30) as stated below. Note also that $p_0 = p_{q+1} = 0$, $SC_k(0) = 0$ as cover 0 is always the very first scheduled cover. So $p_{SC_k(0)} = 0$. As $st_0 = 0$, $st_{SC_k(0)} = 0$ too. $SC_k(sched_k - 1)$ is the last element of SC_k, so it is cover $q + 1$ as this cover is the one having the maximum starting time. Consequently, $st_{SC_k(sched_k - 1)} = LT$. Moreover, Add_z is the set of all the covers that are to be added to time interval z, for all $z \in \{1, \ldots, sched_k - 1\}$ (initially, i.e. at the beginning of the process of target k, Add_z is empty for all z). Note that $st_{SC_k(z)}$ is the starting time of the cover in position z in the ordered set SC_k. If some covers are overlapping, some w_z will be negative, but it does not matter.

$$\begin{aligned} w_z &= st_{SC_k(z)} - \left(st_{SC_k(z-1)} + p_{SC_k(z-1)}\right) \\ &\quad - \sum_{g \in Add_z} p_g \qquad \forall z \in \{1, \ldots, sched_k - 1\} \end{aligned} \tag{30}$$

In Algorithm 1, in line 15, $Add_z(h)$ refers to the cover index of the element in position h in Add_z.

7 A memetic algorithm (CSMA)

This section introduces a genetic algorithm based approach for WSN-CSP. The proposed approach is referred to as CSGA (Cover Scheduling Genetic Algorithm) in the sequel.

CSGA uses permutation encoding, i.e., a chromosome consists of a linear permutation of covers. A permutation of covers specifies the order in which the covers are scheduled, i.e., a value of j at the position i in the permutation indicates that cover j will be the i^{th} cover in the schedule.

The fitness function is same as the objective function of WSN-CSP. In order to compute the fitness of a chromosome, we have to find the maximum duration for which a target

Algorithm 1: Cover Scheduling Heuristic (CSH)

Input: the set of covers Q and their duration
Output: a sequence of the covers

`// Initialization`
Compute UT_k for all targets
Sort the targets in decreasing values of UT_k and re-index
`// Main loop over all targets`
for $k = 1 \rightarrow m$ **do**
  Sort SC_k by increasing order of the starting times of these covers
  Sort UC_k by increasing duration of these covers
  while $UC_k \neq \emptyset$ **do**
    uc is the last cover index in UC_k `// i.e. with maximum duration`
    compute w_z as stated by equation (30)
    compute $r = \underset{z \in \{1,\ldots,sched_k - 1\}}{\arg\max} \left(\frac{w_z}{1 + |Add_z|} \right)$
    allocate $uc \in UC_k$ such that $Add_r \leftarrow Add_r \cup \{uc\}$
    $UC_k = UC_k \setminus \{uc\}$
  `// Schedule the covers in` $\boldsymbol{C_k}$ `by setting starting times`
  forall $z \in \{1, \ldots, sched_k - 1\}$ **do**
    forall $g \in \{1, \ldots, |Add_z|\}$ **do**
      $\sigma = Add_{z,g}$ `// i.e. the cover in position` g `in` Add_z
      $st_\sigma = \left(st_{SC_k(z-1)} + p_{SC_k(z-1)}\right) + g \frac{w_z}{1+|Add_z|} + \sum_{h=1}^{g-1} p_{Add_z(h)}$
    $SC_k \leftarrow SC_k \cup Add_z$
    $Add_z \leftarrow \emptyset$
`// Repairing phase`
Since some of the covers may overlap, sort them in increasing order of starting times
Output the final sequence (will be used for building a semi-active schedule of covers)

is uncovered in the schedule represented by that chromosome. The complexity of this fitness evaluation is $\mathcal{O}(mq)$.

The probabilistic binary tournament selection is used for selecting the two parents for crossover, where the candidate with better fitness is selected with probability p_b. The reason for using probabilistic binary tournament selection is that probabilistic binary tournament is similar to rank selection as far as selection pressure is concerned. At the same time, it is much more computationally efficient [5]. However, with very small probability p_r, we have selected a parent randomly instead of selecting it through probabilistic binary tournament selection. This has been done with the motive of increasing the diversity of the population.

The crossover operator used is the cycle crossover operator CX as initially described in [6]. This crossover operator preserves the absolute positions of the covers in the schedule from one parent or the other and causes a proper mix of the schedules of the two parents. Uniform order based crossover [4] was also tried, but, CX gave better results. Three swap mutations are used inside a single mutation operator to mutate the chromosomes. We have applied the crossover operator always to generate an offspring, whereas mutation operator has been used only with probability p_m.

CSGA relies on the steady-state [4] population replacement model instead of the commonly used generational model. Unlike generational replacement, where the entire parent population is replaced with an equal number of newly created children every generation, in the steady-state population replacement method a single child is produced in every generation and it replaces a less fit member of the population. In comparison to the generational method, the steady-state population replacement method generally finds better solutions faster. This is because of permanently keeping the best solutions in the population and the immediate availability of a child for selection and reproduction. Another advantage of the steady-state population replacement method is the ease with which duplicate copies of the same individuals are prevented in the population. In the generational approach, duplicate copies of the highly fit individuals may exist in the population. Within few generatio! ns, these highly fit individuals can dominate the whole population. When this happens, the crossover becomes totally ineffective and the mutation becomes the only possible way to improve solution quality. Under this situation, improvement in solution quality, if any, is very slow. Such a situation is called the premature convergence. In the steady-state approach we can easily avoid this situation by simply checking each newly generated child against current population members and discarding it if it is identical to any member.

Initial population is generated randomly with the restriction that each member of the initial population should be unique. The offspring created during each generation is checked for uniqueness with respect to the existing population members, and, if it is unique, it is inserted into the population replacing the worst member, otherwise it is discarded.

With the intent of improving the solution quality even further, we have applied a local search on the best solution obtained through genetic algorithm. This local search follows an iterative process. During each iteration, it tries to swap a cover involved in the longest breach with a cover which covers the target involved in the longest breach. If such a swap can decrease the longest breach then it is accepted, otherwise

original schedule is restored. This process is repeated till no swap move is possible that can decrease the longest breach. We have tried this local search inside the genetic algorithm but the resulting approach was found to be too slow. Actually, each swap move requires evaluating the fitness of the schedule from scratch which takes $\mathcal{O}(mq)$ time as mentioned above. This time complexity of a single swap move makes this local search inappropriate for use inside the genetic algorithm.

As far as parameter settings of the CSGA are concerned, the population size is fixed to 400 individuals, p_b is set to 0.8, p_r is set to 0.01, and p_m is taken to be equal to 0.05. The CSGA terminates when the best solution does not improve over 20,000 iterations. CSGA can also terminate when it fails to find a solution different from current population members in 20 consecutive trials. All these parameter settings are chosen empirically. Though these settings provide good results, they are in no way optimal for all instances.

8 Computational experiments

The set of instances used is taken from the output of the framework done in [8]. For the purpose of this paper, we solved the MNLB problem for different size and parameter values. The number of targets m is taken in $\{50, 100, 150, 200\}$. For each value of m a corresponding value of the number of sensor is given in $\{30, 60, 90, 120\}$ and the sensing range will be fixed to 150. Note that the number of sensors and the sensing range value are not parameters of the WSN-CSP, but will influence the number of covers and their duration. For each value of m, the bandwidth constraint will be either 5 of 10 or equal to the number of targets and the breach rate will be either 0.1 or 0.2. For each combination of the parameters, we generated 30 instances. The total set counts 720 instances in total.

All experiments were carried out on an Intel Core 2 Quad computer with 4 Gb of RAM running at 2.83 Ghz under Ubuntu 9.04. The code is developed in C and compiled with gcc version 4. The running times are not reported in details but the heuristic CSH running time is always less than 0.02s. The average and maximum CPU time for the CSGA are 5.83s and 40.61s when $\alpha = 0.1$ and 14.72s and 60.71s when $\alpha = 0.2$. All our approaches are executed once on each instance.

Table 1 presents the results obtained by the heuristic CSH and by the Memetic Algorithm CSMA. The table is organized as follows. The first column defines the instance parameters (namely, the number of targets and the bandwidth constraint). Each row corresponds to 30 instances for $\alpha = 0.1$ and 30 instances for $\alpha = 0.2$. The next five columns are dedicated to instances where the breach rate $\alpha = 0.1$ and the last five columns where $\alpha = 0.2$. For each group of five columns, the first figure is the deviation of CSH over LB in percentage, the second figure is the number of optimal solutions found by CSH over 30 instances (CSH=LB), the third figure is the deviation of CSMA over LB, the fourth figure the number of optimal solution found by CSMA over 30 instances (CSMA=LB) and the last figure is the improvement of the CSMA over the CSH. The last three lines of the table summarize these results.

Table 1: Results of CSH and CSMA

	$\alpha = 0.1$					$\alpha = 0.2$				
Parameters	% CSH	# CSH	% MA	# MA	Imp.	% CSH	# CSH	% MA	# MA	Imp.
$m = 50, w = 5$	66.2	2	11.0	21	31.1	107.4	0	15.4	12	42.2
$m = 50, w = 10$	75.3	0	8.2	22	36.1	97.0	0	14.6	14	39.8
$m = 50, w = 50$	61.5	2	7.9	23	30.8	110.3	0	14.4	14	42.6
$m = 100, w = 5$	77.4	0	1.0	28	41.2	136.1	0	10.3	7	51.1
$m = 100, w = 10$	89.4	1	1.1	26	43.4	127.7	0	8.8	10	50.9
$m = 100, w = 100$	78.4	0	0	30	41.9	132.3	0	7.5	10	50.9
$m = 150, w = 5$	97.8	0	1.2	25	46.9	160.0	0	13.3	3	54.2
$m = 150, w = 10$	102.3	0	1.1	26	47.9	165.5	0	19.6	2	53.8
$m = 150, w = 150$	95.6	0	0.9	27	46.6	164.2	0	12.3	4	54.5
$m = 200, w = 5$	107.1	0	1.3	26	49.4	183.1	0	22.7	0	56.2
$m = 200, w = 10$	133.9	0	0	27	55.9	201.7	0	16.4	3	59.6
$m = 200, w = 200$	131.4	0	1.3	24	54.9	188.6	0	18.0	2	58.2
Total opt. found		**5**		**305**			**0**		**81**	
Average Deviation to LB	**93.0%**		**2.9%**			**147.8%**		**14.5%**		
Average Imp. over CSH					**43.8%**					**51.2%**

From Table 1, one can observed there is a difference of the two approaches when $\alpha = 0.1$ or $\alpha = 0.2$. In fact, if we analyze the lower bound defined in section 4.1, we can easily see that this bound is not tight. The different gaps observed between the two subsets of instances might be due to the quality of the bound. This is comforted with the value in column "Imp." (the improvement of CSMA over CSH) which is rather constant for any subset of instances (around 50%). Nevertheless, even with a non-tight lower bound. We are able to prove the optimality of 386 instances over a total set of 720 instances.

9 Conclusion and perspectives

This paper, to our knowledge, is the first to properly address the Cover Scheduling Problem in Wireless Sensor Networks (WSN-CSP). This problem appears once the covers have been generated in a previous phase of the wireless sensor network problems (like in MCBB or MNLB). The paper introduces a NLP model, a MILP model with an exponential number of constraint and a theoretical study together with some interesting properties of the problem.

Our practical approach consists of an efficient heuristic CSH and a memetic algorithm CSMA. Both are tested on a large set of instances and give very convincing results. Our future work will be oriented in two directions. First we plan to improve the memetic algorithm with dedicated operators for the WSN-CSP and also look for different and more efficient lower bounds of the problem. We are also going to investigate a preemptive version of the problem where the covers can be scheduled for a duration shorter than the total duration, interrupted and continued later.

References

[1] I.A. Akylidiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey of sensor network. *IEEE Communication Magazine*, 40(8):102–116, 2002.

[2] L. Benini, G. Castelli, A. Macii, E. Macii, M. Poncino, and R. Scarsi. A discrete-time battery model for high-level power estimation. In *Proceedings of the IEEE Design Automation and Test in Europe conference*, pages 35–39, Paris, France, 2000.

[3] L. Benini, D.Bruni, A. Macii, E. Macii, and M. Poncino. Discharge current steering for battery lifetime optimization. *IEEE Transaction on computers*, 52(8):985–995, 2003.

[4] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, 1991.

[5] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Proceedings of the 1991 Conference on Foundations of Genetic Algorithms*, pages 69–93, Morgan Kaufmann, 1991.

[6] I. M. Oliver, D. J. Smith, and J. R. C. Holland. A study of permutation crossover operators on the travelling salesman problem. In *Proocedings of the second international conference on genetic algorithms*, pages 224–230, Erlbaum, 1987.

[7] V. Raghunathan, C. Schurgers, S. Park, and M.B. Srivastava. Energy aware wireless microsensor networks. *IEEE Signal Processing Magazine*, 19(2):1007–1023, 2002.

[8] A. Rossi, A. Singh, and M. Sevaux. Génération de colonnes et réseaux de capteurs sans fil. In *Proceedings of the ROADEF, 11ème congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision*, Toulouse, France, 24-26 February 2010.