A Near Optimal Algorithm for Lifetime Optimization in Wireless Sensor Networks

Karine Deschinkel and Mourad Hakem
DISC Laboratory, FEMTO-ST Institute - UMR CNRS, University of Franche-Comte, Belfort, France

Keywords: Target Coverage, Wireless Sensor Networks, Centralized Method, Disjoint Cover Sets, Lifetime Optimization.

Abstract: In this paper, we introduce an efficient energy-aware algorithm to enhance the lifetime of wireless sensor networks (WSN) by i) organizing/clustering the sensor nodes into disjoint cover sets where each cover set is capable of monitoring all the targets of the region of interest and ii) scheduling these cover sets successively/periodically. This study differs from previous works for the following reasons: i) it achieves near optimal solutions compared to the optimal ones obtained by the exact method and ii) unlike existing algorithms that construct gradually cover sets one after the other, our algorithm builds the different sets in parallel. Simulation results show the efficiency of our algorithm.

1 INTRODUCTION

Recent years have witnessed significant advances in wireless sensor networks which emerge as one of the most promising technologies for the 21st century (Akyildiz et al., 2002). In fact, they present huge potential in several domains ranging from health care applications to military applications. A sensor network is composed of a large number of tiny sensing devices deployed in a region of interest. Each device has processing and wireless communication capabilities, which enable to sense its environment, to compute, to store information and to deliver report messages to a base station. These sensor nodes run on batteries with limited capabilities. To achieve a long life of the network, it is important to conserve battery power. Therefore, lifetime optimisation is one of the most critical issues in wireless sensor networks.

In this paper we concentrate on the target coverage problem. We assume that sensors are randomly deployed for monitoring a set of targets with known locations and we also assume that sensors have location determination capabilities. We propose a centralized algorithm that selects mutually exclusive sets of sensor nodes, where the members of each of those sets together completely cover the monitored targets. The intervals of activity are the same for all sets, and only one of the sets is active at any time to provide continuous service while the remaining sets are scheduled to sleep. The objective, is to maximize the number of cover sets to increase the system's service life.

The major contribution of this paper is the design of a new near optimal algorithm for lifetime optimization in WSN. Unlike existing approaches that construct gradually cover sets one after the other, the proposed algorithm builds the different sets in parallel. Indeed, at each step of the clustering process, the algorithm attempts to add to each cover set a sensor capable of monitoring the most critical target. The choice of sensor to be clustered in each cover set is based on solving a linear assignment problem in polynomial time using the Hungarian algorithm. Our approach provides a lower bound K_{min} of the optimal number of disjoint cover sets K_{opt}. Intuitively, the upper bound K_{max} of the optimal value is given by the size of the smallest set of sensors covering a target. We deduce K_{opt} by performing a binary search between K_{min} and K_{max}. At each step of the search process we check if there exists a partition of the sensors in K cover sets by solving an integer programming problem.

The remainder of the paper is organized as follows. Section 2 reviews the related work in the field. Section 3 is devoted to the description of the target coverage problem and explains the basics of our algorithm. Then we express the time complexity of the algorithm. We also present how to compute the optimal number of disjoint cover sets by solving integer programming problems successively. Section 4 shows the simulation results, that fully demonstrate the usefulness of the proposed algorithm. Finally, we give some concluding remarks in Section 5.

Deschinkel K. and Hakem M..
A Near Optimal Algorithm for Lifetime Optimization in Wireless Sensor Networks.
DOI: 10.5220/0004272701970202
In *Proceedings of the 2nd International Conference on Sensor Networks* (SENSORNETS-2013), pages 197-202
ISBN: 978-989-8565-45-7

2 RELATED WORK

First algorithms proposed in the literature consider that the cover sets are disjoint: a sensor node appears in exactly one of the generated cover sets. For instance Slijepcevic and Potkonjak (Slijepcevic and Potkonjak, 2001) propose an algorithm which allocates sensor nodes in mutually independent sets to monitor an area divided into several fields. Their algorithm constructs a cover set by including in priority the sensor nodes which cover critical fields, that is to say fields that are covered by the smallest number of sensors. The time complexity of their heuristic is $O(n^2)$ where n is the number of sensors. (Cardei et al., 2002) present a graph coloring technique to achieve energy savings by organizing the sensor nodes into a maximum number of disjoint dominating sets which are activated successively. The dominating sets do not guarantee the coverage of the whole region of interest. In (Cardei and Du, 2005), the authors propose a heuristic to compute the disjoint set covers (DSC). In order to compute the maximum number of covers, they first transform DSC into a maximum-flow problem , which is then formulated as a mixed integer programming problem (MIP). Based on the solution of the MIP, they design a heuristic to compute the final number of covers. The results show a slight performance improvement in terms of the number of produced DSC in comparison to (Slijepcevic and Potkonjak, 2001) but it incurs higher execution time due to the complexity of the mixed integer programming resolution. Zorbas et al. (Zorbas et al., 2007) present B{GOP}, a centralized coverage algorithm introducing sensor candidate categorisation depending on their coverage status and the notion of critical target to call targets that are associated with a small number of sensors. The total running time of their heuristic is $0(mn^2)$ where n is the number of sensors, and m the number of targets. Compared to algorithm's results of Slijepcevic and Potkonjak (Slijepcevic and Potkonjak, 2001), their heuristic produces more cover sets with a slight growth rate in execution time.

In the case of non-disjoint algorithms (Chaudhary and Pujari, 2011), sensors may participate in more than one cover set. In some cases this may prolong the lifetime of the network in comparison to the disjoint cover set algorithms but designing algorithms for non-disjoint cover sets generally incurs a higher order of complexity. Moreover in case of a sensor's failure, non-disjoint scheduling policies are less resilient and less reliable because a sensor may be involved in more than one cover sets. For instance, Cardei et al. (Cardei et al., 2005) present a linear programming (LP) solution and a greedy approach to extend the sensor network lifetime by organizing the sensors into a maximal number of non-disjoint cover sets. Simulation results show that by allowing sensors to participate in multiple sets, the network lifetime increases compared with related work (Cardei and Du, 2005).

3 ALGORITHMS DESCRIPTION

We try to produce an adaptive scheduling which allows sensors to operate alternatively so as to prolong the network lifetime. For convenience, the notations and assumptions are described first.

3.1 Notations and Assumptions

- m : the number of targets
- n : the number of sensors
- K : maximal number of cover sets
- i : index of target ($i = 1..m$)
- j : index of sensor ($j = 1..n$)
- k : index of cover set ($k = 1..K$)
- T_0 : initial set of targets
- S_0 : initial set of sensors
- T : set of targets which are not covered by at least one cover set
- S : set of available sensors
- $S_0(i)$: set of sensors which cover the target i
- $T_0(j)$: set of targets covered by sensor j
- C_k : cover set of index k
- $T(C_k)$: set of targets covered by the cover set k
- $NS(i)$: set of available sensors which cover the target i
- $NC(i)$: set of cover sets which do not cover the target i
- $|.|$: cardinality of the set

We assume that the lifetimes of sensors are identical. Lifetime of a sensor is time duration when the sensor is in the active state all the time. In order to achieve lifetime extension, sensors must be divided into a number of subsets, called cover sets, where each cover set is capable of monitoring all the targets. Sensors belonging to a scheduled cover set are in active mode, while the others are in sleep mode. If the cover sets are disjoint, then each sensor is allowed to participate only in one cover set. The maximal number of possible disjoint cover sets is given by :

$$K_{max} = \min_{i=1..m} |S_0(i)| \tag{1}$$

since each cover set must cover all targets and each sensor can only be part of one cover set.

3.2 Near Optimal Algorithm

The main idea of the algorithm is to build simultaneously the cover sets by adding gradually the sensors capable of monitoring the most critical target. At each iteration of the algorithm we compute the critical rate (let call it $R(i)$) of a target i as follows :

$$R(i) = \frac{|NS(i)|}{|NC(i)|} \tag{2}$$

Algorithm 1: Near optimal algorithm.

Require: An initial set of targets T_0 and an initial set of sensors S_0
Ensure: A set of cover sets $C_1, .., C_{K_1}$
1: {INITIALIZATION}
2: **for all** cover sets $k = 1..K_{max}$ **do**
3: $C_k \leftarrow \emptyset$
4: $T(C_k) \leftarrow \emptyset$
5: **end for**
6: $S \leftarrow S_0$
7: $T \leftarrow T_0$
8: **for all** targets $i = 1..m$ **do**
9: $NC(i) \leftarrow \{C_1, .., C_{K_{max}}\}$
10: $NS(i) \leftarrow S_0(i)$
11: **end for**
12: {While there exists a target which is not covered by a cover set and the set of available sensors is not empty}
13: **while** $T \neq \emptyset$ and $S \neq \emptyset$ **do**
14: **for all** targets $i \in T$ **do**
15: Compute $R(i) = \frac{|NS(i)|}{|NC(i)|}$
16: **end for**
17: {Choose the most critical target}
18: $i^* = \min_{i \in T} R(i)$
19: **for all** $j \in NS(i^*)$ **do**
20: Compute $p_{jk} = |T_0(j) \cap \{T_0 \setminus T(C_k)\}|$ for each cover set $C_k \in NC(i^*)$
21: **end for**
22: Solve the linear assignment problem to assign one sensor $j \in NS(i^*)$ in each cover set $k \in NC(i^*)$ with costs p_{jk}
23: Update $C_k, T(C_k) \quad \forall k = 1..K_{max}$
24: Update T, S and $NS(i), NC(i) \quad \forall i \in T$
25: **end while**
26: $K_1 \leftarrow K_{max}$
27: **if** T is not empty **then**
28: **for all** $i \in T$ **do**
29: Delete the cover sets $C_k \in NC(i)$
30: $K_1 \leftarrow K_1 - 1$
31: **end for**
32: **end if**

Some targets may have the same critical rate. In this case, the choice of the most critical target is made randomly among these targets. When the most critical target has been identified, sensors monitoring this target have to be distributed in each cover set which does not already cover this target. This distribution can be made randomly or made soundly in order to cover a maximum number of targets in each cover set. That is why we give a cost p_{jk} for the possible assignment of the sensor j to a cover set C_k which represents the additional number of targets that the cover set C_k is able to monitor if the sensor j is added. We face a linear assignment problem where it is required to assign exactly one sensor to each cover set in such a way that the total cost of the assignment is maximized. In our case, the number of sensors to be assigned can be different to the number of cover sets. If the number of sensors is greater than the number of cover sets, we artificially add a new cover set and null costs between all sensors and this additional cover set to be reduced to a classical linear assignment problem. In contrast, if the number of sensors is less than the number of cover sets, we add a dummy sensor and null costs between this dummy sensor and the cover sets. During the process, if we face this second case, this means that some cover sets may not cover all the targets in the region of interest.

3.3 Algorithm's Analysis

We express the time complexity of our algorithm 1 by using the following notations :

$$v = max(|S|, |T|) = max(n, m) \tag{3}$$

and :

$$S_{max} = \max_{i=1..m} |S_0(i)| \tag{4}$$

The time complexity of our algorithm is at most

$$O\Big(v(|T| + K_{max}S_{max} + S_{max}^3 + K_{max} + |T|)\Big) \tag{5}$$

Proof. The main computational cost of the algorithm is spent in the while loop (lines 13 to 25). This loop is executed $v = \max(|S|, |T|)$ times. The inner loop (lines 14 to 16) cost at most $O(|T|)$ since the criticalness of all the targets in T need to be computed at each iteration. Line 18 takes $O(|T|)$ for finding the most critical target. The inner loop (lines 19 to 21) costs $|NS(i^*)| \times |NC(i^*)|, i^* \in T$, since the cost p_{jk} of each target $j \in |NS(i^*)|$ needs to be computed on each cover set $k \in NC(i^*)$. However, we can bound both $|NS(i^*)|$ and $|NC(i^*)|$ by S_{max} and K_{max} respectively. Thus, the cost of this loop is at most $O(S_{max}K_{max})$. Finally, the line statement 22 is equivalent to a linear assignment problem, which can be found in polynomial time, for instance using the Hungarian method (Harold W. Kuhn, 1955). Thereby, the linear assignment problem of line 22 takes at most $O(S_{max}^3)$, since we can bound both $|NS(i^*)|$ and $|NC(i^*)|$ by S_{max}. Updating steps (lines 23 and 24) require $O(K_{max} + |T|)$ iterations. Thus, summing up for the whole v loops/iterations, the algorithm's cost is at most $O\Big(v(|T|+K_{max}S_{max}+S_{max}^3+K_{max}+|T|)\Big)$. Since $K_{max} \leq S_{max} \leq |S| \leq v$, and $|T| \leq v$, we can derive the upper bound $O\left(v^4\right)$. □

3.4 Randomized Algorithm

Here, we give only a simple proposal of the algorithm which constructs in parallel the elementary cover sets as sketched in algorithm 2. Our goal is not to give the best implementation, but only a possible implementation in order to compare the lower bounds to the ones obtained by algorithm 1.

Unlike algorithm 1, the randomized algorithm chooses randomly a non covered target and tries to assign a sensor covering this target to each cover set if necessary. We use the same structure of algorithm 1. This choice is made to demonstrate the relevance of the selection/assignment policy of algorithm 1.

Algorithm 2: Randomized algorithm.

Require: An initial set of targets T_0 and an initial set of sensors S_0
Ensure: A set of cover sets $C_1,..,C_{K_2}$
{INITIALIZATION}
for all cover sets $k = 1..K_{max}$ **do**
  $C_k \leftarrow \emptyset$
  $T(C_k) \leftarrow \emptyset$
end for
$S \leftarrow S_0$
$T \leftarrow T_0$
for all targets $i = 1..m$ **do**
  $NC(i) \leftarrow \{C_1,..,C_{K_{max}}\}$
  $NS(i) \leftarrow S_0(i)$
end for
{While there exists a target which is not covered by a cover set and the set of available sensors is not empty}
while $T \neq \emptyset$ and $S \neq \emptyset$ **do**
  Choose randomly a target i^* in T
  Assign randomly one sensor $j \in NS(i^*)$ in each cover set $k \in NC(i^*)$
  Update $C_k, T(C_k) \quad \forall k = 1..K_{max}$
  Update T, S and $NS(i), NC(i), \quad \forall i \in T$
end while
$K_2 \leftarrow K_{max}$
if T is not empty **then**
  for all $i \in T$ **do**
    Delete the cover sets $C_k \in NC(i)$
    $K_2 \leftarrow K_2 - 1$
  end for
end if

3.5 Exact Method

In this section, we present how to deduce the optimal number of disjoint cover sets once we applied each of the previous algorithms. We obtain an approximated solution with algorithm 1 or with the randomized algorithm and we denote by $K_{min} = \max\{K_1, K_2\}$ the achieved number of disjoint cover sets, where K_1 corresponds to the number of disjoint cover sets obtained with algorithm 1 and K_2 to the number of disjoint cover sets obtained with the randomized algorithm. As the optimal number of disjoint cover sets K_{opt} is bounded by $K_{max} = \min_{1..m}|S_0(i)|$, we deduce that $K_{min} \leq K_{opt} \leq K_{max}$. The intuitive idea is to deduce K_{opt} by performing a binary search between K_{min} and K_{max}. At each step of the search process, we check if there exists a partition of the sensors in K cover sets. Once the number of disjoint cover sets is fixed to K, the distribution of sensors among the K cover sets is given by the resolution of the following integer programming (IP) problem:

$$\begin{cases} \min 1 \\ \text{subject to :} \\ \sum_{j \in S_0(i)} y_{jk} \geq 1, \quad \forall i \in T_0, \forall k \in K \\ \sum_{k=1..K} y_{jk} \leq 1, \quad \forall j \in S_0 \\ y_{jk} \in \{0,1\} \end{cases} \tag{6}$$

The decision variables y_{jk} are binary variables which are equal to 1 if the sensor j belongs to the cover set k and 0 otherwise. Note that there is no real objective function. We only try to find a distribution of the sensors over K cover sets which satisfies some constraints. The first constraint ensures that all targets are covered in each cover set $k = 1..K$. The second constraint forces each sensor to be in only one cover set. For a given number K of cover sets, it may happen that the problem has no solution. In this case, we decrement K and we solve again the problem (6). To compute K_{opt}, we perform a binary search as presented in algorithm 3.

Algorithm 3: Exact Method - binary search.

Require: A set of targets T_0, a set of sensors S_0
Ensure: A set of cover sets $C_1, C_2, ..., C_{K_{opt}}$
$K_{opt} \leftarrow K_{min}$
while $K_{min}! = K_{max}$ **do**
  $K \leftarrow \lceil \frac{K_{min}+K_{max}}{2} \rceil + 1$
  Solve (6)
  if (6) has no solution **then**
    $K_{max} \leftarrow K$
  else
    $K_{min} \leftarrow K$
    $K_{opt} \leftarrow K$
    Optimal solution y_{jk}^* of (6) are saved
  end if
end while
for all $k = 1..K_{opt}$ **do**
  $C_k \leftarrow \cup_{(s_j/y_{jk}^*=1)}\{s_j\}$
end for

Note that the integer programming problem (6) which is solved at each iteration of the binary search process, to reach the optimal solution, is known to be NP-hard (Garey and Johnson, 1990). We use a Branch-and-Bound method to solve it, and we have interest that algorithm 1 provides the best lower bound to avoid additional iterations for the resolution of the integer programming problem.

4 RESULTS

In this section we evaluate the performance of our algorithms by way of simulations. We simulate a network with sensor nodes and target points randomly located in a $500m \times 500m$ area. We assume the sensing range is equal for all the sensors in the network and is set to $150m$. In the different scenarios we vary the number of randomly deployed sensor nodes n between 50 and 200 with an increment of 50. The number m of targets to be covered varies between 30 and 120 with an increment of 30. The following requirements are satisfied: each sensor covers at least one target and each target is covered by at least one deployed sensor, the connectivity of the network is ensured and all sensors are capable of communicating with the base station. For a given number of sensors and targets, we generate 100 random topologies of network. Our experiments have been conducted on a regular Linux workstation with a AMD Athlon(tm) 64 X2 Dual Core Processor 4000+ of 2,1 GHz. The resolution of the integer programming problem is carried out by the Branch-and-Bound method implemented in GLPK (GNU linear Programming Kit) (Mahkorin, 2010) available in the public domain.

Note that for convenience and sake of simplicity, there is no need to conduct comparison with previous works in the literature since the obtained results are compared to the optimal ones achieved by the exact method.

4.1 Number of Disjoint Cover Sets

We measure the average number of disjoint cover sets $\overline{K_1}$, $\overline{K_2}$, $\overline{K_{opt}}$ over the 100 instances for the algorithm 1, the randomized algorithm and the exact method. As algorithm 1 and the randomized algorithm integrate a random part, they are executed 50 times for each network topology of each scenario. Table 1 summarizes the obtained results. Results of table 1 are consistent with those obtained in the literature (Cardei and Du, 2005). The number of disjoint cover sets increases with sensor density, and decreases as the number of targets goes up for a fixed number of sensors. This is explained by the fact that sensors are more requested. The obtained results show that our algorithm 1 has a very good behavior because it is able to achieve near optimal solution compared to the optimal one obtained by the exact method in almost all simulated cases. On the other side a simple randomized algorithm provides solutions which are far more than 15% of the optimum and this gap, computed as $\frac{\overline{K_{opt}}-\overline{K_2}}{\overline{K_{opt}}}$, grows when the number of sensors and targets increases to reach 38% with 200 sensors and 120 targets.

Table 1: Number of disjoint cover sets.

N	M	$\overline{K_2}$	$\overline{K_1}$	$\overline{K_{opt}}$
50	30	2.97	3.46	3.46
	60	2.44	2.96	2.96
	90	2.19	2.60	2.60
	120	2.06	2.49	2.49
100	30	7.16	8.84	8.84
	60	5.54	7.54	7.54
	90	4.97	6.99	6.99
	120	4.65	6.70	6.70
150	30	10.75	13.50	13.50
	60	8.80	12.03	12.03
	90	7.89	11.34	11.34
	120	7.42	10.91	10.91
200	30	14.53	19.03	19.03
	60	11.65	16.94	16.98
	90	10.44	16.11	16.11
	120	9.67	15.53	15.59

4.2 Comparison of the Execution Times

In this section, we compare and comment the CPU execution times of the different resolution methods. Table 2 gives the distribution of the execution times (in seconds) for the three methods over the 16 scenarios. Note that the method called ”exact method” consists in performing a binary search between K_{min} and K_{max} and in solving an integer programming problem at each iteration. To assess the efficiency of our algorithm 1, we distinguish two cases for the reporting execution times of the exact method. In the first case, K_{min} is given by the number of disjoint cover sets (K_2) obtained by the randomized algorithm, and in the second case, the binary search begins with $K_{min} = K_1$.

From table 2, we can see that the running times increase with both sensor and target density as expected (see the time complexity value in section 3.3). If the optimal value of the number of disjoint cover sets is reached with our algorithm 1, then no resolution of the integer programming is involved again. This leads to the execution times which are equal to zero (column 5 of table 2). Moreover, the resolution of inte-

Table 2: Execution times (in seconds) for the 3 methods.

N	M	Randomized algorithm 2	Heuristic algorithm 1	Exact method ($K_{min} = K_1$)	Exact method ($K_{min} = K_2$)
50	30	0.010	0.023	0.000	0.026
	60	0.019	0.031	0.000	0.053
	90	0.026	0.037	0.000	0.058
	120	0.036	0.046	0.000	0.030
100	30	0.026	0.145	0.000	1.445
	60	0.056	0.176	0.000	2.443
	90	0.086	0.205	0.000	3.540
	120	0.118	0.227	0.000	4.070
150	30	0.041	0.503	0.000	17.478
	60	0.100	0.653	0.000	25.733
	90	0.160	0.698	0.000	42.646
	120	0.222	0.768	0.000	58.182
200	30	0.063	1.270	0.000	123.600
	60	0.150	1.660	16.190	251.467
	90	0.250	1.730	0.000	345.877
	120	0.352	1.936	83.26	543.568

ger programming is time consuming and the involved binary search task to compute the optimal value by starting with random solution may require more than 9 minutes as the number of sensors/targets goes up.

5 CONCLUSIONS

In this paper, we have addressed the problem of lifetime optimization in wireless sensor networks. This is a very natural and important problem, as sensor nodes have limited resources in terms of memory, energy and computational power. To cope with this problem, an efficient centralized energy-aware algorithm is presented and analyzed. Our algorithm seeks to prolong the network lifetime by organizing sensors into disjoint cover sets which operate successively in order to monitor all targets.

The proposed algorithm involves a linear assignment problem to generate mutually disjoint cover sets containing a minimal number of sensors and covering a maximal number of targets at each iteration. Simulation results highlight the good behaviour of our algorithm, which provides near optimal solutions (equal to the optimal ones achieved by the exact method for almost all scenarios) with minimum time complexity. The quality of the obtained lower bound reduces significantly the search for the optimal number of cover sets, because this search process requires the resolution of an integer programming problem which is time-consuming.

REFERENCES

Akyildiz, I., Su, W., Y.Sankarasubramniam, and Cayirci, E. (2002). A survey on sensor networks. *IEEE Comm. Magazine*, pages 102–114.

Cardei, M. and Du, D.-Z. (2005). Improving wireless sensor network lifetime through power aware organization. *Wirel. Netw.*, 11:333–340.

Cardei, M., MacCallum, D., Cheng, M. X., Min, M., Jia, X., Li, D., and Du, D.-Z. (2002). Wireless sensor networks with energy efficient organization. *Journal of Interconnection Networks*, 3(3-4):213–229.

Cardei, M., Thai, M. T., Li, Y., and Wu, W. (2005). Energy-efficient target coverage in wireless sensor networks. In *INFOCOM*, pages 1976–1984.

Chaudhary, M. and Pujari, A. K. (2011). High-energy-first (hef) heuristic for energy-efficient target coverage problem. *International Journal of Ad hoc, Sensor and Uniquitous computing (IJASUC)*, 2(1).

Garey, M. and Johnson, D. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., NY, USA.

Harold W. Kuhn (1955). The Hungarian Method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97.

Mahkorin, A. (2010). *GNU Linear Programming Kit, Reference Manual*.

Slijepcevic, S. and Potkonjak, M. (2001). Power efficient organization of wireless sensor networks. In *IEEE Int. conf. on Communications*, pages 472–476.

Zorbas, D., Glynos, D., Kotzanikolaou, P., and Douligeris, C. (2007). B{GOP}: an adaptive coverage algorithm for wireless sensor networks. In *Proceedings of the 13th European Wireless Conference*, EW'07.