Scheduling Sensors Activity in Wireless Sensor Networks

Antonina Tretyakova, Franciszek Seredynski, Frédéric Guinand

Scheduling sensors activity in wireless sensor networks

Antonina Tretyakova[1], Franciszek Seredynski[1], and Frederic Guinand[12]

[1] Dep. of Mathematics and Natural Sciences
Cardinal Stefan Wyszynski University in Warsaw
Warsaw, Poland
a.tretyakova@uksw.edu.pl
[2] LITIS Laboratory, University of Le Havre
Le Havre, France

Abstract. In this paper we consider Maximal Lifetime Coverage Problem in Wireless Sensor Networks which is formulated as a scheduling problem related to activity of sensors equipped at battery units and monitoring a two-dimensional space in time. The problem is known as an NP-hard and to solve it we propose two heuristics which use specific knowledge about the problem. The first one is proposed by us stochastic greedy algorithm and the second one is metaheuristic known as Simulated Annealing. The performance of both algorithms is verified by a number of numerical experiments. Comparison of the results show that while both algorithms provide results of similar quality, but greedy algorithm is slightly better in the sense of computational time complexity.

Keywords: maximum lifetime coverage problem; metaheuristics; energy-efficient coverage preserving protocol

1 Introduction

Wireless Sensor Networks (WSNs) are one of the faster developing computer-communication technologies currently involving in many spheres of human activities, like healthcare, agriculture, industry environment, military (see, e.g. [1] [2]), etc. WSN is a set of a huge number of small devices, called sensors or sensor nodes, enabling to monitor surroundings, gather information about environment and perform many other tasks. For many missions, sensors are randomly distributed over the monitoring area in environments, where human access is limited or impossible. Therefore, batteries of sensors cannot be usually rechargeable or renewable. Such scenarios of WSN can be executed in deserts, forests, wilderness, mountain terrains and etc. Exhaustion of battery charge implies the change in topology of the WSN, quality of its work and reduction of its lifetime. In WSNs energy-efficient management is intrinsically important task.

One of the main tasks posed before wireless sensor networks is an area monitoring. According to the area applications a WSN should perform different functions, among which are sensing environmental characteristics, gathering data,

transmission data to a sink, etc. Due to their tiny construction WSN nodes have limitations on energy power, computing power, sensing range, transmission distance and bandwidth. These restrictions lead to a number of optimization problems, goals of which are to maximize lifetime of the system via effective managing the capabilities of the network. Limited energy sources of sensor devices demand to equip them by energy - efficient coverage preserving protocol. Such a kind of protocol: centralized or localized has to solve a variant of maximum lifetime coverage problem (MLCP) in WNSs. MLCP can be considered as a specific deterministic scheduling problem, where it is necessary to schedule sensors activity in time in a such way to maximize lifetime of the network maintaing at the same time some quality parameter like e.g. amount of covered by active sensors monitored area.

MLCP is known to be NP-hard problem [3] [4], therefore, one can relay on delivering rather approximate solutions instead of exact ones. Recently, a number of nature-inspired algorithms applied to optimization problems in WSNs have appeared in the literature. Among them are genetic algorithms [5], evolution strategies [6], particle-swarm optimization [7], etc. As we already mentioned, the quality of solutions and computational complexity are not satisfactory. Coverage problems were considered under different scenarios and types of WSNs, namely wireless multimedia sensor networks in dynamic environment [6], directional sensor networks [8]. These assumptions lead to different problems statement and, therefore, each approach should be modified to enable to solve coverage problem in another type of WSNs. Our research on a direct applying nature inspired metaheuristics of general purpose to solve MLCP [9] shows that they are not enough efficient. We believe that further improving the quality of approximate solutions and computational time complexity can be achieved by incorporation into a searching engine of an algorithm of a specific knowledge about the problem. In this paper we propose two knowledge based algorithms to solve a variant of MLCP. It is a greedy heuristic and an algorithm based on Simulated Annealing (SA).

The rest of the paper is organized as follows. In the next section the problem is stated. The two following sections present our greedy algorithm and SA-based algorithm. Sections 5 and 6 contain results of simulation experiments and conclusion remarks.

2 Problem statement

Let us consider a homogeneous sensor network $S = \{s_1, \ldots, s_N\}$ consisting of N sensor nodes randomly distributed over a given *target field* F, a two-dimensional rectangular area of $W \times H$ m^2. The target field F is uniformly divided on points of interest (POIs) with a step g, sensors are responsible for detection of an intruder (a target point) and sending an alarm message to the sink node. A sensor s_j is defined as a point of coordinates (x_j, y_j) in two-dimensional area, sensing range R_s and battery capacity b. An example of a sensor network randomly deployed over the target field is depicted in Figure 1. It is assumed that

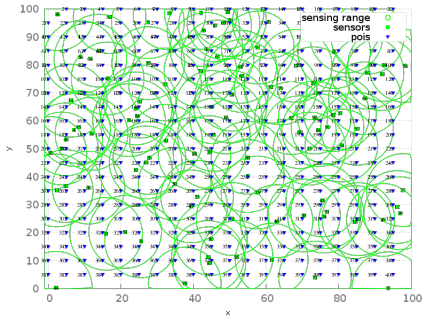

Fig. 1. An example of sensor network deployed over the target field

each sensor can work in two modes: *active mode* and *sleeping mode*. In active mode a sensor observes a circle area within its sensing range and can transmit or receive a signal. Let us denote the mode of i-th sensor during j-th time interval as $state(s_i, t_j)$, where $state(s_i, t_j) \in \{\text{ON, OFF}\}$. The value of $state(s_i, t_j)$ equals ON means that i-th sensor s_i during j-th time interval is in active mode, otherwise, $state(s_i, t_j) = \text{OFF}$.

Below we give a number of definitions concerning the problem statement.

Definition 1. *A sensor* $s_i(x_i, y_i)$ *covers a POI* p(x, y) *iff the Euclidean distance* d(s_i,p) *between them is less than the sensing range* R_s.

Let us denote a set of POIs covered by i-th sensor s_i as $POIs_{obs}(s_i)$ and call as *coverage area* of i-th sensor. All POIs covered by an active network during j-th time interval is denoted as $POIs_{obs}(t_j)$, i.e.

$$POIs_{obs}(t_j) = \cup_{i=1}^{N} POIs_{obs}(s_i)|_{state(s_i,t_j)=ON} \tag{1}$$

Definition 2. *Coverage of a target field* F *at* $j-th$ *time period* t_j *denoted as* $cov(t_j)$ *is a real number equal to a ratio of a number of POIs covered by an active network during* j*-th time interval* t_j *to all* $POIs$*, i.e.*

$$cov(t_j) = \frac{|POIs|_{obs}(t_j)}{|POIs|} \tag{2}$$

Let us denote a number of POIs covered by i-th sensor as $cov(s_i)$.

A sensor is assumed to consume energy for monitoring area and it depends on its sensing range R_s. Consider a homogeneous sensor network, where all sensors have the same sensing range, the energy consumption per time interval is constant. A potential solution is a schedule of a network S deployed over a target field prescribing states of activity for all sensors in the network during the whole period of time of the network operation.

Definition 3. *A schedule of a WSN is a binary* $T_{max} \times N$ *matrix denoted as* Sol*, i.e.*

$$Sol(S) = \{state_i^j\}, where\ i = 1, ..., N\ and\ j = 1, ..., T_{max}, \tag{3}$$

where $state_i^j \in [0,1]$ *is a state of* i*-th sensor during* j*-th time interval, 0 corresponds to OFF state and 1 is related to ON state.*

Each row of the matrix is related to one of the sensors and represents its schedule of activity over all period of network operation from t_1 till $t_{T_{max}}$. Let us assume a sensor to spend one unit of energy during one time unit of its activity.

Definition 4. *A schedule* $Sol(S)$ *is a feasible solution if the following equality is met:*

$$(\forall i)_{i=1,\ldots,N} | \sum_{j=1}^{T_{max}} state_i^j = b \tag{4}$$

Definition 5. *Coverage string is a set of real values, each of which corresponds to the coverage of a target field* F *during each time interval of the WSN operation, i.e.*

$$coverage\ string = \{cov(t_1), cov(t_2), ..., cov(t_{T_{max}})\} \tag{5}$$

Definition 6. *Lifetime of the WSN denoted as Lifetime(q) is defined as a sum of time intervals, during which the coverage requirement is met,*

$$Lifetime(q) = \sum_{i=1}^{T_{max}} i|_{cov(i) \geq q\}} \tag{6}$$

Maximal time of network performance is restricted by the characteristics of the network such as a number of sensors and their distribution, a distribution of POIs over the target field, sensing range, battery capacity value and the level of coverage required. The parameter T_{max} is a predefined number and should be set greater than the performance time $Lifetime(q)$ and less than the upper bound of network operation denoted as $Lifetime^{Up}$.

We consider Maximum Lifetime Coverage Problem (MLCP) as a scheduling problem applied to a WSN solving the area coverage problem in the discrete two-dimensional space. MLCP has as an objective to prolong lifetime of a WSN by minimizing a number of redundant sensors during each time interval in order to minimize energy consumption. The function $Lifetime(q)$ is maximized over the space of all feasible solutions. Coverage requirement is given by a coverage ratio q, which means that at least q-th part with small declination δ of all targets is covered by at least one sensor.

MLCP - specific knowledge A searching process conducted by both a greedy heuristic and SA-based algorithm (see, below) incorporates the MLCP specific knowledge and is based on the following classification of columns in a schedule. All columns of the schedule solution are divided on three groups called three subsequences: *Redundant Subsequence (RS)*, *Excellent Subsequence (ES)*, *Unsatisfactory Subsequence (US)*. Each subsequence groups time intervals such that a network of active sensors covers the target area with certain *coverage ratio*, i.e.

$$t_i \in RS,\ if\ cov(t_i) > q + \delta, \tag{7}$$

$$t_i \in ES,\ if\ |cov(t_i) - q| \leq \delta \tag{8}$$

$$t_i \in US,\ if\ cov(t_i) < q - \delta \tag{9}$$

Let us denote a number of elements in RS, ES and US as N_R, N_E and N_U respectively.

3 A greedy heuristic to solve MLCP

In this section, we present an iterative knowledge-based stochastic greedy heuristic to solve MLCP. The algorithm is based on constructing a tree of solutions. A root of the tree is a randomly created solution. In each iteration a solution called a predecessor is changed under two steps described below to form one more solution called a successor. The next iteration continues from the node corresponding to the best solution between a predecessor and its successor from the previous iteration. The pseudocode of the algorithm is presented in algorithm 1. At each iteration a schedule is a subject of two types procedures, pseudo codes of which are sketched in Procedure 1 (algorithm 1, lines 6-12) and Procedure 2 (algorithm 1, lines 18-21). The aim of procedure 1 is to improve a current solution

Algorithm 1 Pseudocode - Greedy algorithm for random initial solution.

```
 1: Input : WSN, Target field, N_I, q, δ, T_max
 2: initialize random sol_cur(N, T_max)
 3: k = 1
 4: for i ← 1 to N_I do
 5:    compute US,
 6:    for i ← 1 to N_U - k + 1 do
 7:       for j ← 1 to k do
 8:          modify i and i + j columns from US,
 9:          j = j + 1
10:       end for
11:       i = i +1
12:    end for
13:    compute Lifetime(q)
14:    if Lifetime(q) of the predecessor > Lifetime(q) of the successor then
15:       k = k + 1
16:    end if
17:    compute RS, US for the successor,
18:    for i ← 1 to N_R do
19:       modify the i − th column in RS in the solution,
20:       i = i +1
21:    end for
22:    compute Lifetime(q) for the successor,
23:    keep the best from the predecessor and its successor,
24:    i = i+1
25: end for
26: return sol
```

via joining active subnetworks during time intervals when necessary coverage is not achieved. This purpose can be obtained by multiple shifting several columns from US toward ES or RS. A schedule is changed under the Procedure 1 as follows. Two new columns are generated by applying boolean-valued functions OR and AND to a pair of two values from same row from US columns. The new first column contains the values resulted of OR operator, and the second column contains the results of AND operator applied. As the algorithm proceeds it may happen that a new solution is not improved in the sense of *Lifetime(q)*. In that case a parameter k (a number of use Procedure 1) is increased by 1. Initially, k is equal to 1. The solution obtained by the first modification (Procedure 1) is next changed by the Procedure 2. The aim of this stage is to reduce redundant consumption of energy, i.e. a randomly chosen sensor in active state from RS time interval is switched off and, next, is switched on during US time interval.

The Procedure 2 is executed on a current solution and consists of the two steps. Firstly, from the $i-th$ RS column a cell is randomly selected with probability p_i, where n_1 is a number of "1" cells in the column. Let us denote a row of the selected cell as j. Second, the "0" cell in the in first US column in $j-th$ row is changed on "1". Therefore, the first selected cell is equal to 1. The second selected cell is taken as the first "0" cell from US and from the same row as previous cell was. The selected cells swap their values. If there is not a cell with the value "0" in US, the predecessor solution coincides with its successor. The above mentioned two steps are repeated consequently N_R times for each RS column.

The predecessor schedule and its successor are evaluated by *Lifetime(q)* metric and the best one is saved as a current schedule for the next iteration to be applied. These steps are repeated until stop condition is met. The last saved schedule is a result of the algorithm.

4 Simulated annealing algorithm to solve MLCP

Simulated Annealing (SA) is one of the nature inspired metaheuristics based on the physical annealing process observed in glass manufacturing process and metallurgy.

The performance of SA depends on construction of a given solution neighbourhood. Generating the sequential solutions is based on a swap of a pair of opposite values in one row. A neighbouring solution is differed from the given solution by a number of bits changing comparatively with the given solution. Let us call this characteristic defined by a number of changing pairs of bits, as *neighbourhood size* and denoted as k_{neigh} - neighbourhood. In such a way, in case of changing two random cells, we obtain a solution in 1-neighbourhood from a given solution. The random neighbour is generated as follows: k_{neigh} times two opposite values chosen at random from the same row are swapped (see, Algorithm 2, lines 8-16). The additional information about solution is computed such as coverage of each time slot according which the time line is divided on RS, ES and US. The idea of knowledge-wise neighbourhood generating pro-

cedure is to switch off an active sensor from redundant subsequence in order to reduce a number of redundantly covered POIs and to switch it on in the first unsatisfactory subsequence with the aim of increasing coverage at the additional time interval. These steps may increase lifetime of the generated solution. The

Algorithm 2 Pseudocode of SA algorithm

```
1: Input : WSN, Target field, N_I, q, δ, T_max
2: Initialize unit Sol(N, T_max)
3: Maximal temperature L
4: k
5: while termination condition is not fulfilled do
6:    for i ← 1 to L do
7:       life = compute Lifetime(q) of Sol
8:       for i ← 1 to k_neigh do
9:          for j ← 1 to RS.size() do
10:            choose a random cell of "1" value from i − th RS column, the row of the
               gene let us denote as l
11:            find the first "0" gene from US and l − th row
12:            swap the values of the chosen pair
13:            j = j + 1
14:         end for
15:         i = i + 1
16:      end for
17:      lifeN = compute Lifetime(q) of SolN
18:      Δ = life - lifeN
19:      if Δ ≤ 0 then
20:         Sol = SolN
21:      else
22:         Sol = SolN with probability exp^(Δ/T)
23:      end if
24:      decrease T(i)
25:   end for
26:   update termination values
27: end while
28: Sol
```

pseudo code of SA solving MLCP is presented in Algorithm 2. SA algorithm works until termination condition meets performing cycles of searching solutions in the neighbourhood of the initial solution. Each cycle consists of several iterations characterized by the temperature. During one iteration a random solution within the current solution's neighbourhood is created, Lifetime(q) for the current solution and its neighbour is computed, the current solution is compared with the created neighbour. The better new solution always replaces the old one, while in case of worse neighbour it replaces the current solution with probability $\exp^{\frac{\Delta}{T}}$ so that the probability of acceptance of the new solution depends on the temperature value and difference in values of evaluation function between two

solutions. At the end of an iteration temperature level is increased according with the cooling scheme.

5 Experimental results

We consider WSN consisting of a number N of sensors equal to 100, 200 and 300, respectively. For each value of N, we created 3 instances, which differ by random allocation of sensors, so 9 instances were used in experimental study. Each instance is described as Instance{*indicator of network size*}{*order number of WSN instance*}, where N is equal to *indicator of network size* times 100. To give an example, Instance23 represents the third instance of WSN consisting of 200 sensors. The algorithm's parameters should be chosen as the set of the best

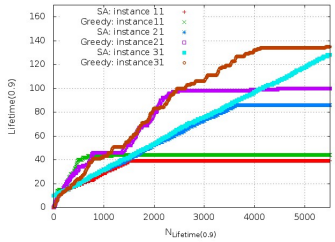

Fig. 2. An example of a typical run of two algorithms: *greedy* and SA for instance11, $R_s = 20$, $b = 10$.

values for each of the algorithms: greedy heuristic and SA. SA is defined by the following values. Temperature is cooled according to the logarithmic scheme with initial temperature 50, length of the temperature cycle 25, the frozen level 10 and maximal number of iterations 100. The termination condition is as follows: exceeding maximal number of iterations or achieving the frozen temperature level. Greedy heuristic needs to set a number of iterations equal to 150, after which the algorithm stops. The main component of computational cost of both algorithms is related to calculation of $Lifetime(q)$ function. An example of a typical run of these two algorithms is presented in Figure 2, which presents the dynamics of $Lifetime(0.9)$ obtained by greedy and SA for three instances consisting of 100, 200 and 300 nodes as a function of a number of computation the $Lifetime(q)$ function. From the figure one can observe that *greedy* converges quicker for all instances with a slightly better quality than SA. However, when the size of an instance is relatively large (instance 31) the *greedy algorithm* achieves a local optimum, while SA continuously improves quality of a solution.

Let us finally discuss the overall results on the MLCP input data. In order to present a broader view, WSN instances with different properties will be used. For testing purposes we will consider nine WSN instances with three types of densities and sensing coverage range 20 deployed over the same target field <100,

100, 5>. Coverage requirement is represented by three values: 0.85, 0.9 and 0.95. Battery capacity is equal to 10.

Table 1. Maximal, average with standard deviation values of Lifetime(q) obtained by two algorithms: *greedy* and SA for nine instances; $q \in = \{0.85, 0.9, 0.95\}$, $R_s = 20$, $b = 10$.

q	*algorithm*	Max	Avg $\pm$ σ	Max	Avg $\pm$ σ	Max	Avg $\pm$ σ
		*instance*11		*instance*12		*instance*13	
	greedy	**54**	**52 ± 3.32**	**58**	**56 ± 4.7**	**52**	**49 ± 4.36**
0.85	SA	51	49 ± 1.0	55	52 ± 1.73	49	47 ± 1.0
	greedy	**44**	**42 ± 3.88**	**47**	**44 ± 4.36**	**41**	**39 ± 3.88**
0.9	SA	42	38 ± 2.0	46	43 ± 1.41	38	36 ± 1.0
	greedy	**32**	**30 ± 3.47**	**35**	**33 ± 3.32**	**29**	**27 ± 4.36**
0.95	SA	29	26 ± 1.0	33	29 ± 1.73	25	23 ± 1.0
		*instance*21		*instance*22		*instance*23	
	greedy	109	107 ± 4.36	108	105 ± 5.48	109	107 ± 4.48
0.85	SA	**110**	106 ± 1.41	**108**	104 ± 2.23	**110**	**107 ± 1.41**
	greedy	**93**	**90 ± 4.7**	**93**	**88 ± 7.82**	**93**	**91 ± 4.48**
0.9	SA	88	85 ± 1.73	87	84 ± 1.41	90	86 ± 1.73
	greedy	65	60 ± 6.71	60	58 ± 3.75	63	61 ± 4.48
0.95	SA	64	**62 ± 0.0**	**61**	**59 ± 1.0**	**66**	**63 ± 1.41**
		*instance*31		*instance*32		*instance*33	
	greedy	166	162 ± 6.49	158	156 ± 4.8	165	160 ± 9.44
0.85	SA	165	162 ± 1.73	**163**	**158 ± 2.0**	164	161 ± 1.41
	greedy	**139**	**135 ± 6.33**	**136**	**129 ± 9.28**	**139**	**136 ± 4.36**
0.9	SA	137	130 ± 2.44	130	126 ± 2.23	135	131 ± 2.44
	greedy	97	92 ± 9.8	88	85 ± 5.39	98	93 ± 8.84
0.95	SA	**98**	**94 ± 1.73**	**94**	**89 ± 2.82**	**98**	**95 ± 1.0**

Table 1 presents results of systematic study of both algorithms conducted for representatives of three types of instances, which support our previous observations. The table contains maximal, average and standard deviation of the goal function values obtained by *greedy* and SA based on averaging of 10 runs. The remaining parameters are as follows: q-requirement is based on the set $\{0.85, 0.9, 0.95\}$, nine instances of WSN with $R_s = 20$, and $b = 10$. One can see from the table that the average and the maximal values of $Lifetime(q)$ are differed slightly. In the most cases, it is evident that *greedy* provides better solutions than SA, the results concerning instances of WSN consisting of 100 nodes can serve as an example. Meanwhile, with growth of the problem complexity, when N is equal to 200 or 300, SA finds better solutions, for instance, see q equal to 0.85 or 0.95. It should be notice that standard deviation computed for SA results in all cases are better than σ-values computed for the results provided by *greedy algorithm*. This indicates, that SA is a more stable than *greedy algorithm*. In such a way we can assume that in the case of bigger problem instance SA

enables to provide better solutions than *greedy*. To summarize aforementioned discussion it is shown that there are two different approaches solving MLCP providing good different solutions of the problem.

6 Conclusion

In this paper the problem of lifetime maximization in WSNs stated as MLCP with assumption of not full coverage defining by a coverage ratio requirement q was considered. The problem belongs to a class of NP-hard problems characterized by high computational complexity, what motivates to use algorithms provided approximate solutions.

To solve the problem we proposed and study two centralized knowledge-based algorithms: stochastic greedy heuristic and simulated annealing algorithm. All of the algorithms were studied on the same testbed and under the same assumptions. Results of experimental study of the algorithms shows that the greedy algorithm is efficient in both a quality of solutions and time complexity for a medium sizes of the problem instances. When the size of the problems becomes relatively large simulated annealing provides better quality of solutions, however it is achieved by increasing computational time of the algorithm.

References

1. S. Nesamony, M. Karky Vairamuthu, M. Orlowska and S. Sadiq, On Sensor Network Segmentation for Urban Water Distribution Monitoring, Lecture Notes in Computer Science, January 2006.
2. F. J. Pierce and T.V. Elliott, Regional and on-farm wireless sensor networks for agricultural systems in Eastern Washington, Computers and Electronics in Agriculture, pp. 32 - 43, 61(1), April 2008.
3. M. Cardei and J. Wu, Energy-efficient coverage problems in wireless ad-hoc sensor networks, Journal Computer Communications archive, pp. 413 - 420, Vol. 29, 2006.
4. M. R. Garey and D. S. Johnson, Computers and intractability: a guide to the theory of NP-completeness, W.H. Freeman and Co, 1979.
5. B. Sahoo, V. Ravu and P. Patel, Observation on using Genetic Algorithm for Extending the Lifetime of Wireless Sensor Networks, IJCA Special Issue on 2nd National Conference- Computing, Communication and Sensor Network, pp. 9 - 13, 2011.
6. H. Fayyazi, M. Sabokrou, M. Hosseini and A. Sabokrou, Solving heterogeneous coverage problem in Wireless Multimedia Sensor Networks in a dynamic environment using Evolutionary Strategies, ICCKE2011, Mashhad, Iran, October 13-14, 2011.
7. M. Abbasi, M. Shafie Abd Latiff, A. Modirkhazeni and M. H. Anisi, Optimization of Wireless Sensor Network Coverage based on Evolutionary Algorithm, IJCCN, Vol. 1(1), December 2011.
8. J. M. Gil and Y. H. Han, A Target Coverage Scheduling Scheme Based on Genetic Algorithms in Directional Sensor Networks, Sensors, pp. 1888-1906, Vol. 11(2), 2011.
9. A. Tretyakova and F. Seredynski, Application of Evolutionary Algorithms to Maximum Lifetime Coverage Problem in Wireless Sensor Networks, 27-th IEEE IPDPS, Boston, USA, 2013.