Wireless sensor network redeployment under the target coverage constraint

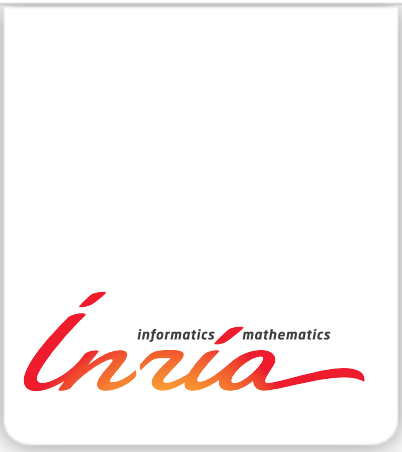
Dimitrios Zorbas, Tahiry Razafindralambo

Wireless sensor network redeployment under the target coverage constraint

Dimitrios Zorbas, Tahiry Razafindralambo

RESEARCH REPORT
N° 7818
November 2011
Project-Teams POPS

ISSN 0249-6399 ISRN INRIA/RR--7818--FR+ENG



Wireless sensor network redeployment under the target coverage constraint

Dimitrios Zorbas, Tahiry Razafindralambo

Project-Teams POPS

Research Report n° 7818 — November 2011 — 13 pages

Abstract: A critical problem of wireless sensor networks is the efficient handling of the nodes energy under the target coverage constraint. The sensors are randomly deployed in the field covering a set of targets. Due to the randomness of the deployment some targets are covered by a few only sensors. Thus, the maximum achieved network lifetime is upper bounded by the energy of the sensors that cover the most poorly covered target. To tackle this problem one could move some sensors to these poorly covered areas from areas that are covered by many sensors. In this paper we present centralised and localised solutions that can be used to redeploy the sensing nodes and balance the amount of energy between the sensors that cover each target. We simulate our algorithms and our findings show an over 100% increase of the amount of energy of the sensors that cover the most poorly covered target in the network.

Key-words: WSN, deployment, coverage

RESEARCH CENTRE
LILLE – NORD EUROPE

Parc scientifique de la Haute-Borne
40 avenue Halley - Bât A - Park Plaza
59650 Villeneuve d'Ascq

Redéploiement de réseau de capteurs contraint par la couverture de cible

Résumé :

Mots-clés : Réseau de capteurs, déploiment, couverture

1 Introduction

Wireless sensors networks consist of hundreds of tiny nodes with limited battery lifetime. The nodes are used to monitor a number of events that periodically occur in some areas of the field called the "targets". The nodes use their sensing range to monitor the events that may occur in a proximate target. Moreover, they use their communication range to exchange data with other nodes in the network.

Because of the randomness of the deployment, other targets may be covered by a small number of sensors and others by a large number of nodes. In applications where there is the need of full coverage, the target that is covered by the lowest number of nodes, sets an upper bound on the maximum achieved network lifetime [8]. In other words, the sensors that cover poorly covered targets exhaust their energy faster than the other sensing nodes in the network. This observation leads to at least one uncovered target, while other targets still remain covered by many nodes. Since the monitoring process terminates when at least one target cannot be covered anymore, the energy of the sensors that cover other targets cannot be used.

To tackle the problem above, we present two algorithms, a centralised algorithm and a localised one, that redeploy sensors between target areas, such that all the targets will be covered by about the same number of sensors. Specifically, our algorithms take nodes from targets that are covered by many sensors and move them to more sparsely covered targets. By this way, they distribute the available energy among all the targets in the network and increase the total amount of energy that is related to the most poorly covered target.

The redeployment takes place before other operations of the network, such as the monitoring operation or the routing operation. Apparently, during the redeployment, the nodes' movement consumes energy, so the network after the redeployment consists of less energy than before. However, this amount of energy is spent by the sensors that cover richly covered targets and without the redeployment process their energy would remain useless.

An important requirement of a sensor network is the connectivity with the sink. Making the assumption that initially all the sensing nodes are connected with the sink using neighbouring relay nodes, the final network will remain connected as well. This statement holds true, since the nodes are moved only from a target to another target and no relay node changes position. In the case where there is an insufficient number of relay nodes, redeployment cannot guarantee connectivity and a relay node redeployment is needed. This paper focuses on sensing node redeployment leaving this special case for consideration in our future work.

The rest of the paper is organised as follows. In Section 2, we present the related work in the fields of target coverage and mobile sensor networks. In Section 3, we provide a formal description of our problem, while in Section 4 we present our two solutions. In Section 5 we evaluate our methods and we compare them to the case where the nodes are non-mobile. Finally, Section 6 concludes the paper.

2 Related work

Most of the works about target coverage in WSNs deal with the problem of dividing the available sensors in sets, such that only one set is active at any time [2, 7, 8]. By successively activating the sets, the network lifetime can be increased. In [8] a special care is taken for the sensors cover poorly covered targets, when the monitoring data of each sensing node are forwarded to the sink. However, these works assume static nodes and the maximum achieved network lifetime still depends on the initial number of sensors cover the most poorly covered target in the network.

From the other hand, the works that deal with sensor redeployment consider only the case of the uniform coverage of a large monitored area [6, 5, 3, 4]. The algorithms presented in these works are basically based on the use of virtual forces or voronoi diagrams, where each node pushes its neighbours to more sparsely covered areas. However, these algorithms cannot be used in the target coverage problem, since they require a continuous deployed area (not only the area around each target) in order to provide a uniform area coverage.

3 The Target Coverage Redeployment Problem

Let $T_0 = \{t_1, t_2, \ldots, t_k\}$ be the set of targets and $S_0 = \{s_1, s_2, \ldots, s_n\}$ the set of sensors. Initially, each target in T_0 is covered by at least one sensor in S_0 if it is in a sensor's sensing range R_s. Each sensor has a maximum communication range R_c and an initial battery lifetime equal to l_0. A sensor may spend a part of this energy to move itself to another location. This amount of energy depends on the distance and it is equal to α_m. Moreover, let l_j be the energy of a sensor s_j throughout the redeployment process.

Set $N = \{N_1, N_2, \ldots, N_k\}$ contains the sets of sensors that each target is covered by. n_{avg} is the average number of sensors that cover a target among all the targets and it is given by:

$$n_{avg} = \frac{|N_1| + |N_2| + \ldots + |N_k|}{k}, \tag{1}$$

where $|N_i|$ is the cardinality of N_i, $i \in [1, k]$. Let N' contain the sets of sensors that each target is covered by after the redeployment process.

The objective of the redeployment algorithm is to solve the Target Coverage Redeployment Problem (TCRP) by achieving $\min(|N'_1|, |N'_2|, \ldots, |N'_k|) = \lfloor n_{avg} \rfloor$, and maximise $\sum_{j=1}^{|N'_{min}|} l_j$, subject to $l_j > 0 \ \forall \ s_j \in S_0$, where t_{min} is the target covered by the lowest number of sensors at the end of the redeployment.

In N' the most poorly covered target is defined as the target that the sum of the energy of the sensors covering this target is less than or equal to the sum of the energy of the sensors covering each of the other targets in the network.

If we consider a constant energy consumption per meter, the maximum distance that a node is able to be moved is $d_{max} = \frac{l_0}{\alpha_m}$. Since a sensor is moved from a target area to another target area, the maximum distance that two targets can be found, in order to exchange sensors, is equal to d_{max}. This practically means that if a sensor is equipped with two AA-type batteries and consumes

100 J/m to move itself, then the maximum distance between two targets will be 141 meters. In other words, the maximum allowed size of a square-shaped terrain would be 20,000m^2.

4 Solutions for TCRP

In this section we present two solutions to solve TCRP. The first one is a centralised algorithm that has a complete knowledge of the parameters of the network; the location of the nodes, the location of the targets and the distances between them. The second one is a localised solution where the nodes decide if, who and where a sensor may be moved. According to this solution each node does not have a complete knowledge of the network, but it initially knows its own location and the coordinates of the targets.

Greedy-TCR (see Algorithm 1) is a centralised algorithm that begins by computing n_{avg} and the sharing factor (i.e. sf) for each target in the network (Step 1). The sharing factor shows how many sensors may offer or receive a particular target by other targets in order to decrease or increase the number of sensors that is covered by. In Step 2, each target that can receive nodes computes a number of distances to sensors that cover targets with redundant nodes. This number of sensors depends on how many nodes the target-donator is covered by. Greedy-TCR keeps these distance in a temporary set D along with the target-receiver, the target-donator and the sensor that may be moved. All these distances are examined in Step 3 and the shortest distances are selected. The sharing factors of the targets are updated respectively.

Since no other node can be moved, Greedy-TCR moves the selected sensors to the appropriate location. Each sensor moves on the straight line between its initial location and the target-receiver. It stops when the distance to the target-receiver is equal to R_s. Greedy-TCR checks if the target-receiver is the only target that can be covered from this location. If not, Greedy-TCR avoids to cover multiple targets using a single sensor, a fact that could affect the sharing factor of some other targets and could lead to infinite number of movements. Thus, the node is moved to a place that can cover only the target-receiver. If the energy consumption due to the movement is higher than the available energy of the sensor, the algorithm abandons this movement and continues with the next sensor.

Even if no other sensor can be moved due to the restrictions on the selection of the distances (e.g. a node has no available energy), some targets may still remain covered by a higher number of sensors than n_{avg}. Greedy-TCR, in Step 4, finds the target surrounded by the minimum amount of energy among all the targets and moves towards it the closest node to this target. These nodes had not been tested during Step 2, because other sensors where closer to the target-receivers. Greedy-TCR ends by returning set N'.

Greedy-TCR may not provide always the optimal solution (i.e. the minimum total traveling distance). In order to find the optimal result an algorithm should check among $\frac{n}{2}!$ possible solutions inheriting a prohibitive complexity. The longest run of Greedy-TCR appears when all the targets take part in Step 2 of the algorithm. Thus, the complexity of Greedy-TCR is $O(k^2n)$, where k is the number of targets and n the number of sensors.

Local-TCR is a distributed and localised algorithm that operates in rounds.

Algorithm 1: Greedy-TCR

require: $S_0 \neq \emptyset$, $N_i \neq \emptyset$, $\forall\ t_i \in T_0$, $T_0 \neq \emptyset$, $l_0 > 0$
// Step 1
$n_{avg} = \frac{|N_1|+|N_2|+\ldots+|N_k|}{k}$;
foreach $t \in T_0$ **do**
  if $|N_t| > n_{avg}$ **then**
    $sf_t = \lfloor |N_t| - n_{avg} \rfloor$;
  else
    $sf_t = \lceil n_{avg} - |N_t| \rceil$;
// Step 2
$D = \emptyset$;
foreach $t \in T_0$ **do**
  if $sf_t < 0$ **then**
    foreach $t' \in T_0$ **do**
      if $sf_{t'} > 0$ **then**
        select the $sf_{t'}$ shortest distances from t to a sensor s_j that covers t';
        keep these distances in D along with t, t' and s_j;
// Step 3
$moving\_nodes = \emptyset$;
while *no node can be moved* **do**
  foreach $d \in D$ **do**
    select the shortest d;
    $sf_t = sf_t + 1$;
    $sf_{t'} = sf_{t'} - 1$;
    mark s_j as a moving node;
foreach $s_j \in moving\_nodes$ **do**
  move s_j to the appropriate location and update l_j;
// Step 4
Repeat Step 1;
foreach $t \in T_0$ **do**
  if $sf_t > 0$ **then**
    compute target t' surrounded by the minimum amount of energy;
    find the sensor s_j that covers t and it is in the shortest distance to t';
    move s_j to t';
return N';

In each round (see Algorithm 2) the sensing nodes perform a number of operations to decide which nodes will be moved from a target area to another. During the first operation each node covering a target discovers the neighbouring nodes that also cover this particular target. Each node keeps this set of neighbouring sensors in its memory, excluding those that cover other targets at the same time. Along with the neighbour set, a node is able to build n_i for each t_i it monitors. n_i is named "target cardinality" and represents the number of sensors that cover t_i (i.e. n_i is equal to $|N_i|$).

In Figure 1 a network consists of four targets (squares) and a number of sensing nodes (dots) covers these targets. All the nodes that are between the circle and the target, can monitor this particular target and consist n_i of target t_i (i.e. $n_A = 5$, $n_B = 8$, $n_C = 2$ and $n_D = 4$). The radius of the circle is equal to R_s.

Since all nodes know their neighbours and how many other sensors cover their covered targets, they can compute their sharing factor. To do this, each particular node receives the corresponding cardinalities of the neighbouring nodes; the nodes that are in its communication range and cover different targets than what the particular node covers. In Figure 1, supposing that the communication range is about 5 times the sensing range, a node that covers target "B" can

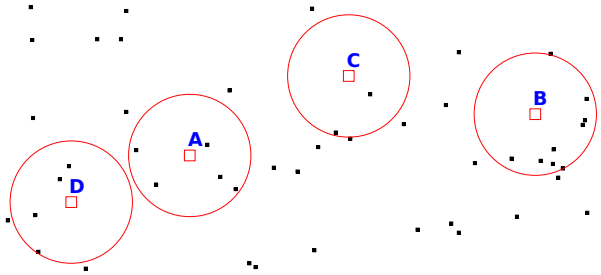


Figure 1: A network with 4 targets

directly communicate with nodes covering target "C" and nodes covering target "A", but not with nodes covering target "D".

Each particular node computes a local sharing factor of the cardinalities of the targets that it covers along with the cardinalities of the targets that the neighbouring nodes cover. The cardinalities that are larger or equal to the average cardinality of the targets that the particular node covers, are not taken into account in this computation. Depending on the value of the local average, a node can offer (supporter) or receive nodes (receiver) for the targets it covers. Each supporter keeps in its memory the closest receivers for each target it supports. In Figure 1, a node that covers target "A" computes a sharing factor equal to $\lfloor n_A - \frac{n_A+n_D+n_C}{3} \rfloor = 1$. n_B is not taken into account, because $n_B > n_A$. So, nodes covering "A" can be supporters for this round.

During the next phase, each supporter receives the sharing factor of all the other supporters in the network. This operation is needed in order to avoid having two different supporters sending sensors to the same receiver. To avoid flooding of the network, one or more head-nodes are assigned for each target and a backbone network is built. This backbone network consists of head-nodes that may be supporters or receivers. A supporter can be head of a target t_i if there is no other node in N_i closer to a neighbouring target. Supporters that are not head-nodes can receive the sharing factors of the other supporters in the network through their target head-node. The lines illustrated in Figure 2 correspond to the backbone paths.

Since all supporters have the sharing factor of the other ones, they compare it with their own. The node with the highest sharing factor is allowed to decide who, where and how many nodes will be moved. This node is called the "leader". In order to avoid having two leaders in the same round, nodes multiply their sharing factor with a random value in $(1, 1.01]$ before send it to the other supporters. For example, a node covering target "B" will be the leader in Figure 1.

During the last phase, the leader finds the receiver that needs the highest number of sensors and computes how many nodes could offer it. The number of nodes that will be moved, let say f, is equal to $\lfloor \frac{\max(n_{s_1}, \ldots, n_{s_k}) - \min(n_{r_1}, .., n_{r_l})}{2} \rfloor$, where $t_{s_1}, \ldots, t_{s_k}$ are the targets covered by the supporter and $t_{r_1}, \ldots, t_{r_l}$ are the targets covered by the receiver. The leader looks in the set of neighbouring nodes for the f closest nodes to the target-receiver. In case that its own distance is shorter than at least one of the other f nodes, it can move itself excluding one of the other f nodes. Since the moving nodes have been discovered, the leader

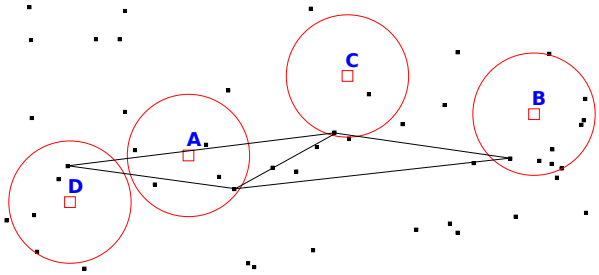


Figure 2: Head-nodes and backbone paths of the network

computes their final destination similar to the centralised algorithm. The next round starts when the nodes have reached their final destination. In the example of Figure 1, the two left nodes of target "B" will be moved, since they are closer to target "C". The new location of these two nodes is indicated by a circle in Figure 3.

Theorem 1. *Local-TCR always balances the number of monitoring nodes between two neighbouring targets when the number of nodes covering the two targets differs by two or more sensors.*

Proof. Since all the nodes covering these two targets have built the set of neighbouring nodes and the cardinality sets, they are ready to compute their sharing factor. Let t_μ and t_λ be the two targets, where $n_\mu - n_\lambda \geq 2$. Let also s_μ and s_λ be two sensors that cover t_μ and t_λ, respectively. The sharing factors of the two sensors will be:
$sf_{s_\mu} = \lfloor n_\mu - \frac{n_\mu - n_\lambda}{2} \rfloor = \lfloor \frac{n_\mu - n_\lambda}{2} \rfloor$, and
$sf_{s_\lambda} = \lfloor n_\lambda - \frac{n_\lambda}{1} \rfloor = 0$.
This practically means that s_μ will move at most $f = \lfloor \frac{n_\mu - n_\lambda}{2} \rfloor$ nodes towards t_λ in Phase 5. In order to have a balanced final network, it should be true that $n'_\mu - n'_\lambda \leq 1$. It is true that $n'_\mu = n_\mu - f$ and $n'_\lambda = n_\lambda + f$. It follows that $n_\mu - f - n_\lambda - f \leq 1$ which is equal to $n_\mu - n_\lambda \leq 1 + 2f$. Using $n_\mu - n_\lambda \geq 2$, it follows that $f \geq \frac{1}{2}$, which is true because f is integer and $f \geq 1$. □

In the very special case where three successive targets are covered by k_1, k_2 and k_3 sensors respectively, and $k_1 - k_2 = 1$ and $k_2 - k_3 = 1$, Local-TCR will fail to balance the number of sensors between t_1 and t_3. This problem could be solved by allowing the sensors to receive the coverage status of their 2-hop neighbouring nodes in the network during Phase 2. However, this would increase the total overhead of the network.

5 Evaluation and discussion of the results

In this section we simulate the proposed algorithms and we compute the amount of energy of the sensors cover the most poorly covered target. We compare the results to the case where the nodes are static and cannot be moved.

We assess the algorithms in 50 topologies with random and uniform target and sensor deployments and we compute the average energy value of these 50

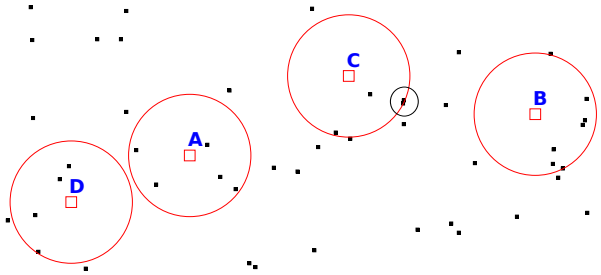


Figure 3: New location of the nodes moved

topologies. The 95% confidence intervals are shown in each figure. The communication range of the nodes is $50m$ and their sensing range is $10m$. We assume that two targets cannot be found in a distance shorter than the sensing range. The battery of the sensors is initially equal to 20K J (this is a typical energy capacity of one C-type or two AA-type batteries [1]). A sensor consumes 100 *Joules* of energy in order to travel one meter distance, except from the last scenario where the energy consumption varies.

5.1 Simulation results

The first scenario involves 60 deployed sensors and 15 targets scattered on a terrain with variable size. Varying the terrain size we assess the algorithms on the network density. The results presented in Figure 4 show an over 100% increase on the energy corresponds to the most poorly covered target in comparison to the case where the nodes cannot be moved (i.e. Wo redeployment). As it was expected, the minimum energy decreases as the terrain size increases and less sensors are able to monitor the targets. The centralised algorithm performs a little bit better than the localised algorithm, as it has a complete knowledge of the network and can move sensors between distant targets minimising the total traveling distance.

In Figure 5 is shown the performance of the algorithms in a scenario with constant number of sensors and fixed terrain size. We gradually increase the number of targets from 5 to 30. The two algorithms improve a lot the total amount of energy of the sensors that cover the most poorly covered target, but the centralised algorithm performs better than the localised when many targets are deployed. As in the previous scenario, this occurs due to the fact that the localised algorithm moves sensors only between neighbouring targets and, thus, some sensors travel longer distances to move to a distant target. When the number of targets is higher, the number of movements is increased and the nodes consume more energy.

In the next scenario, we evaluate the algorithms in the case where the number of targets remains constant in a fixed-size terrain with variable number of sensors. The findings are illustrated in Figure 6 and show a high increase in cases where the number of sensing nodes is above 40. When a few only nodes are deployed the improvement is small due to the fact that the most of the targets are covered by one or two nodes. However, in all the cases, the localised algorithm remains very close to the centralised approach.

Finally, in the last scenario, we vary the energy consumption yielded by a

Algorithm 2: Local-TCR

require: $S_0 \neq \emptyset$, $T_0 \neq \emptyset$, $l_0 > 0$
// Phase 1
foreach $s \in S_0$ **do**
  $neighbouring\_set_s = \emptyset$;
  foreach s' *in communication range of* s **do**
    if $s' \in S_0$ *and covers the same targets* **then**
      $neighbouring\_set_s = neighbouring\_set_s \cup \{s'\}$;
foreach $s \in S_0$ **do**
  foreach t *in sensing range of* s **do**
    $n_t = 1$;
    foreach s' *in communication range of* s **do**
      if $s' \in S_0$ *and covers* t **then**
        $n_t = n_t + 1$;
// Phase 2
foreach $s \in S_0$ **do**
  foreach t *in sensing range of* s **do**
    $sf_s = sf_s + n_t$;
  $sf'_s = sf_s/(\#$ of targets in sensing range of $s)$;
  foreach s' *in communication range of* s **do**
    if $s' \in S_0$ **then**
      foreach t' *in sensing range of* s' **do**
        if $n_{t'} < sf'_s$ **then**
          $sf_s = sf_s + n_{t'}$;
  $sf_s = sf_s/(\#$ of covered and examined targets$)$;
  $sf_s = \lfloor(\max(n_{s_1}, ..., n_{s_k}) - sf_s)\rfloor$;
  if $sf_s > 0$ **then**
    s is supporter;
  else if $sf_s = 0$ **then**
    s is receiver;
// Phase 3
foreach *supporter* s **do**
  $can\_support_s = \emptyset$;
  foreach *receiver* s' *in communication range of* s **do**
    $can\_receive_{s'} = \emptyset$;
    foreach t *in sensing range of* s' **do**
      if s' *is in the closest distance to* s **then**
        $can\_support_s = can\_support_s \cup \{t\}$;
        $can\_receive_{s'} = can\_receive_{s'} \cup \{s\}$;
      if *no other node in* $neighbouring\_set_s$ *is closer to* s' **then**
        make s and s' head-nodes;
// Phase 4
foreach *supporter* s **do**
  receive the sharing factor of the other supporters (multiplied by a random number in (1,1.02]);
  compute the max sharing factor sf_{max} that you have received;
  if $sf_s > sf_{max}$ **then**
    s is the leader;
// Phase 5 (operations by leader s)
$f = \lfloor \frac{\max(n_{s_1}, \ldots, n_{s_k}) - \min(n_{r_1}, .., n_{r_l})}{2} \rfloor$;
find the f closest nodes to the target-receiver;
move this nodes to the target-receiver if they have available energy;

node's movement. We keep constant the other parameters of the network. The results illustrated in Figure 7 show that even in the case where the energy consumption is large, both Greedy-TCR and Local-TCR can increase the amount

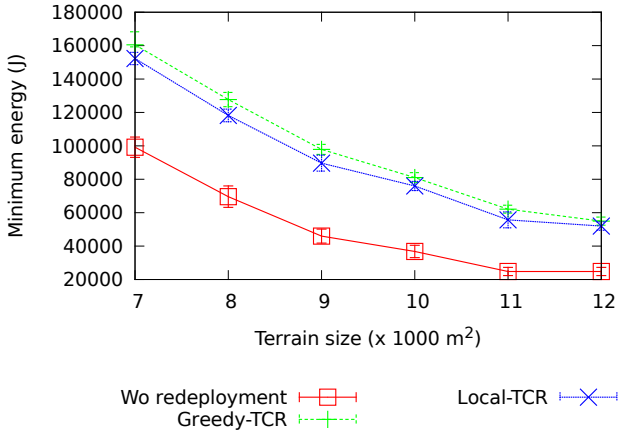


Figure 4: Total amount of energy of the sensors covering the most poorly covered target (minimum energy) in a scenario with 60 nodes, 15 targets and a variable terrain size

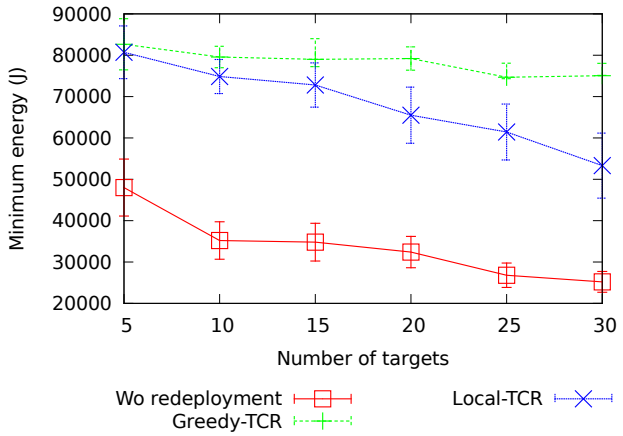


Figure 5: Total amount of energy of the sensors covering the most poorly covered target (minimum energy) in a scenario with 60 nodes, 10K m^2 terrain size and variable number of targets

of energy that corresponds to the most poorly covered target.

6 Conclusion and future work

In this paper we dealt with the redeployment of a set of sensors that cover a number of targets in the field. Since the nodes are randomly deployed, sensors that cover the most poorly covered target set an upper bound on the network lifetime. To tackle this problem, we presented two algorithms that balance the number of sensors between the targets. The first algorithm works centralised and has a complete knowledge of the network parameters. The second one is a localised solution that is based on the neighbouring information. The evaluation results showed that the energy of the sensors that cover the most poorly covered

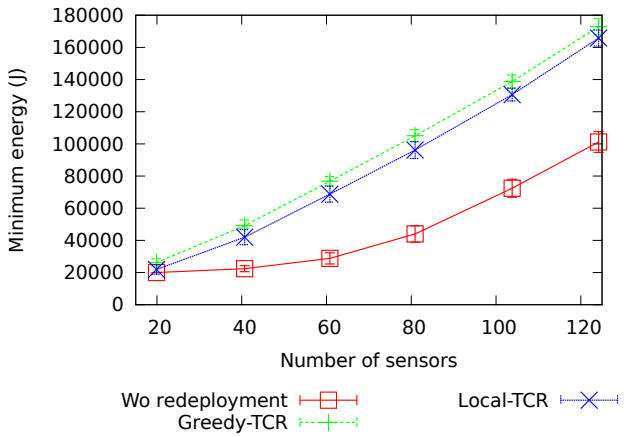


Figure 6: Total amount of energy of the sensors covering the most poorly covered target (minimum energy) in a scenario with variable number of nodes, 15 targets and 10K m^2 terrain size

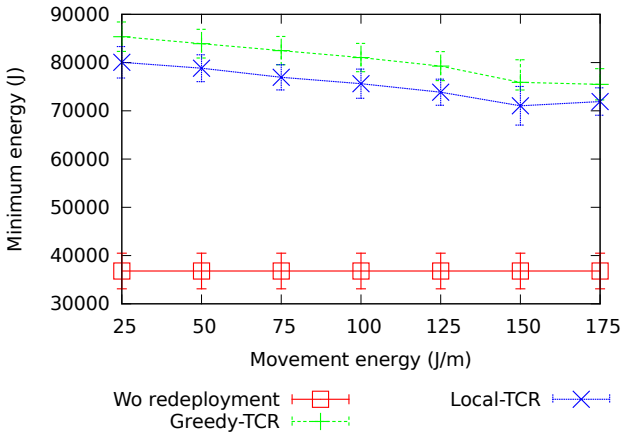


Figure 7: Total amount of energy of the sensors covering the most poorly covered target (minimum energy) in a scenario where we vary the movement energy consumption

target can be doubled by applying one of the previous redeployment approaches. Since connectivity is an important requirement of a sensor network, we are working on providing a scheme that ensures connectivity, even in the case where the network is not initially connected.

References

[1] Battery energy storage. `http://www.allaboutbatteries.com/Energy-tables.html`.

[2] Ionut Cardei and Mihaela Cardei. Energy efficient connected coverage in wireless sensor networks. *Int. J. Sen. Netw.*, 3(3):201–210, 2008.

[3] Nojeong Heo and P.K. Varshney. Energy-efficient deployment of intelligent mobile sensor networks. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(1):78 – 92, jan. 2005.

[4] Foued Kribi, Pascale Minet, and Anis Laouiti. Redeploying mobile wireless sensor networks with virtual forces. In *Proceedings of the 2nd IFIP conference on Wireless days*, WD'09, pages 254–259, Piscataway, NJ, USA, 2009. IEEE Press.

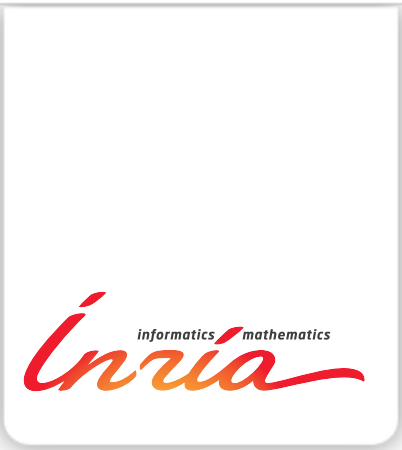
[5] Guiling Wang, Guohong Cao, and T. La Porta. Proxy-based sensor deployment for mobile sensor networks. In *Mobile Ad-hoc and Sensor Systems, 2004 IEEE International Conference on*, pages 493 – 502, oct. 2004.

[6] Guiling Wang, Guohong Cao, and Thomas F. La Porta. Movement-assisted sensor deployment. *IEEE Transactions on Mobile Computing*, 5:640–652, June 2006.

[7] Qun Zhao and Mohan Gurusamy. Lifetime maximization for connected target coverage in wireless sensor networks. *IEEE/ACM Trans. Netw.*, 16(6):1378–1391, 2008.

[8] Dimitrios Zorbas and Christos Douligeris. Connected coverage in wsns based on critical targets. *Computer Networks*, 55(6):1412 – 1425, 2011.

Contents



RESEARCH CENTRE
LILLE – NORD EUROPE

Parc scientifique de la Haute-Borne
40 avenue Halley - Bât A - Park Plaza
59650 Villeneuve d'Ascq

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399