Efficient Sensor Network Design for Continuous Monitoring of Moving Objects

Sotiris Nikoletseas * Paul G. Spirakis *

Abstract

We study the problem of localizing and tracking multiple moving targets in wireless sensor networks, from a network design perspective i.e. towards estimating the least possible number of sensors to be deployed, their positions and operation characteristics needed to perform the tracking task. To avoid an expensive massive deployment, we try to take advantage of possible coverage overlaps over space and time, by introducing a novel combinatorial model that captures such overlaps.

Under this model, we abstract the tracking network design problem by a combinatorial problem of covering a universe of elements by at least three sets (to ensure that each point in the network area is covered at any time by at least three sensors, and thus being localized). We then design and analyze an efficient approximate method for sensor placement and operation, that with high probability and in polynomial expected time achieves a $\Theta(\log n)$ approximation ratio to the optimal solution. Our network design solution can be combined with alternative collaborative processing methods, to suitably fit different tracking scenaria.

1 Introduction

Recent advances in micro-electromechanical systems (MEMS) and wireless communications have enabled the development of very small, smart, low cost sensing devices ([1, 19]) with sensing, data-processing and wireless transmission capabilities. They are meant to be pervasively deployed into forming ad-hoc wireless sensor networks that collect information from the ambient environment and make it available to the user. Some applications imply deployment in remote or hostile environments (battle-field, tsunami, earth-quake, isolated wild-life island, space exploration) to assist in tasks such as target tracking, enemy intrusion detection, forest fire detection, environmental or biological monitoring. Some other applications imply deployment indoors or in urban or controlled environments. Examples of such applications are industrial supervising, indoor micro-climate monitoring (e.g. to reduce heating cost by detecting poor building thermal isolation), smart-home applications, patient-doctor health monitoring or blind and impaired assisting. Sensor networks imply distributed and collaborative data-processing, because of the small utility of each sensor individually and the severe resource constraints, mainly with respect to energy but also memory and computation capabilities.

1.1 Problem Description

We wish to solve the problem of *localizing and continuously tracking mobile objects* moving in a domain described by a set of set of three-dimensional curves, S, over a period of time T i.e. we

*Computer Technology Institute (CTI) and Dept of Computer Engineering & Informatics, University of Patras, Greece. E-mails: {nikole, spirakis}@cti.gr. This work was partially supported by the IST/FET/Global Computing Programme of the European Union, under contact number IST-2005-15964 (AEOLUS).

want the wireless sensor network to be able to detect the position of any moving object, at any time t in T. We allow multiple targets that arise in the network area at random locations and at random times. The movement of each target can follow an arbitrary but continuous path i.e. we disallow the target to instantaneously "jump" to another location; still, we can handle such discontinuities as multiple targets.

In our setting, the set S of 3D curves is the set of possible trajectories of objects moving for some time within the period T. Such a moving object might follow a part of a curve in S, and possibly arrive to an intersection of curves and then follow another curve.

We can assume that each such curve in S is specified (for $t = 0, \ldots, T$) by an analytic equation (e.g. in x, y, z coordinates). Thus, we can compute a curve (route) *piece* τ_{ij} for the curve (route) R_i, via some criterion (e.g. to split R_i into pieces of equal length). We, then, wish to deploy some sensors (either standing or even moving for some time) in some (initial) places within S. A basic demand for such a deployment consists of the following rule:

(Rule R) For every point $\vec{p} = (x, y, z)$ in S and for every $t \in [0, T]$ there are at least three sensors, active at time t, whose sensing range includes the point $\vec{p}$. Here, for a sensor σ operating at time t, its *sensing range* is a sphere of some radius $R_\sigma(t)$ and with center the position of σ at time t.

Note that rule R guarantees localization of any moving target in S at any time t in T, via triangulation.

We follow a *network design approach* to this problem. Let us assume, hypothetically, that we could have an abundance of sensors, each characterized by an initial position, an operation period, and an initial available energy (battery) that allows a particular implementation of sensing ranges during the operation of the sensor. Then, the decision to actually select one sensor (with the initial position, operating period and available energy) has a certain *cost*. Our goal is to be able to select a subset of sensors that implement rule (R) and is of nearly minimal cost.

The problem we study is related (but different) to the problems of network coverage and tracking. In fact, we extend the well-studied coverage problems by being able to track the moving path, and by also taking time into account. On the other hand, to reduce the energy dissipation and overhead of our tracking solution, we avoid some of the collaborative information processing components (like which nodes should sense, which have useful information and should communicate, which should receive information and how often). Thus, we are not dealing directly with queries of the type "how many targets are in a certain region during a certain time interval". Still, our solution performs the collaborative processing tasks (triangulation by at least three sensors) that allow localization of the targets as they move in the network. Also, we discuss how alternative collaborative processing methods can be combined with our approach to provide full tracking.

1.2 Our Contribution

Our approach indeed tries to avoid the expensive, massive placement of all the time functioning sensors all over the monitoring area, *by exploiting possible overlaps of routes* at certain places in the network area, using sensors that can simultaneously monitor pieces of several (nearby) routes during their operation, or even exploit sensors on top of certain moving objects of our own that wander in the maze of routes. Thus, we somehow "multiplex" (both over space and time) the use of deployed sensors, since our approach identifies overlaps over space and time and thus deploys fewer sensors compared to the trivial approach.

In fact, the analysis of our method shows that *it is very efficient*, especially in very large domains, both with respect to computational time and the deployment cost, since it finds in poly-

nomial expected time a deployment solution which approximates the optimal solution within a logarithmic factor. Thus, we avoid expensive dense deployment of sensors, where the information about the target is simultaneously generated by multiple sensors; we instead achieve a low cost solution (by removing unnecessary redundancy) that still keeps tracking accuracy at high levels.

To be able to handle overlaps, *we propose a novel combinatorial model for possible routes* in the network domain. This model, although abstract, captures several of the technical specifications of real sensor devices, such as the energy spent as a function of the transmission range, the ability to vary this range to save energy or increase connectivity, the ability of sensors to employ power saving (sleep-awake) schemes to save energy etc. This combinatorial model allows us to reduce the tracking problem to a variation of a combinatorial problem of set covering (in particular to "at least 3 cover", i.e. having each point of the domain covered at any time by at least 3 sensors, and thus being localized). We feel that this combinatorial model is of independent interest and can (itself or its variations) be used in modeling other problems as well.

1.3 Related Work and Comparison

As discussed in the problem definition part, the problem we study is relevant to network coverage and tracking, that we discuss below.

1.3.1 Coverage. Sensor deployment strategies play a very important role in providing better QoS, which relates to the issue of how well each point in the sensing field is covered. However, due to severe resource constraints and hostile environmental conditions, it is nontrivial to design an efficient deployment strategy that would minimize cost, reduce computation, minimize node-to-node communication, and provide a high degree of area coverage.

Several deployment strategies have been studied for achieving an optimal sensor network architecture. Dhilon et al. ([5]) propose a grid coverage algorithm that ensures that every grid point is covered with a minimum confidence level. They consider a minimalistic view of a sensor network by deploying a minimum number of sensors on a grid that would transmit a minimum amount of data. Their algorithm is iterative and uses a greedy heuristic to determine the best placement of one sensor at a time. It terminates when either a preset upper limit on the number of sensors is reached or sufficient coverage of the grid points is achieved. However, the algorithm assumes line of sight of the target and the sensor. Also, since a complete knowledge of the terrain is assumed, the algorithm is not very applicable in cluttered environments, such as interior of buildings, because modeling obstacles becomes extremely difficult in those scenarios. Finally, a main difference of this approach (and in fact the other coverage methods as well) with ours, is that we explicitly take time into account.

In contrast to static sensor networks, nodes in mobile sensor networks are capable of moving in the sensing field. Such networks are capable of self-deployment starting from an initial configuration. The nodes would spread out such that coverage in the sensing field is maximized while maintaining network connectivity. A potential field-based deployment approach using mobile autonomous robots has been proposed to maximize the area coverage ([16]). Clearly, the assumptions of this method (and the one described below) are different to ours, since we do not use sensor mobility as an algorithmic design element (although we handle mobility when it appears).

Similar to the potential field approach, a sensor deployment algorithm in the presence of mobility based on virtual forces has been proposed in [22] to increase the coverage after an initial random deployment. A sensor is subjected to forces, which are either attractive or repulsive in nature. In this approach, obstacles exert repulsive forces, while areas of preferential coverage (sensitive areas where a high degree of coverage is required) exert attractive forces, and other sensors exert attractive or repulsive forces. A hard threshold distance is defined between two

sensors to control how close they can approach each other.

Other interesting network coverage approaches are discussed in the book chapter by A. Ghosh and S. Das in [18].

1.3.2 Tracking. Our method follows and extends the well-established line of research for a network architecture design for centralized placement/distributed tracking (see e.g. the book [18] for a nice overview). According to that approach, optimal (or as efficient as possible) sensor deployment strategies are proposed to ensure maximum sensing coverage with minimal number of sensors, as well as power conservation in sensor networks.

Centralized Approaches. In one of the methods ([4]), that focuses on deployment optimization, a grid manner discretization of the space is performed. Their method tries to find the grid point closest to the target, instead of finding the exact coordinates of the target. In such a setting, an optimized placement of sensors will guarantee that every grid point in the area is covered by a unique subset of sensors. Thus, the sensor placement problem can be modeled as a special case of the alarm placement problem described by Rao [17]. That problem is the following: given a graph G, which models a system or a network, one must determine how to place alarms on the nodes of G so that any single node fault can be diagnosed. It has been shown in [17] that the minimal placement of alarms for arbitrary graphs is an NP-complete problem. Clearly, their problem is easier than ours, since they relax the requirement to find exact coordinates of moving objects by just finding the nearest grid point. Since their problem is computationally difficult, this implies the inherently high complexity of our problem. Another indication of the hardness of the problem is the fact that, as shown in [3], the localization problem is NP-hard in sparse wireless sensor networks.

Also, our problem is related but different to the following other well known approach that focuses on power conservation: in [8] sleep−awake patterns for each sensor node are obtained during the tracking stage, to obtain power efficiency. The network operates in two stages: the surveillance stage during the absence of any event of interest, and the tracking stage, in response to the presence of moving targets. Each sensor initially works in the low-power mode when there are no targets in its proximity. However, it should exit the low-power mode and be active continuously for a certain amount of time when a target is sensed, or even better, when a target is shortly about to enter. Finally, when the target passes by and moves farther away, the node should decide to switch back to the low-power mode. Our approach is also power aware in the same sense (since we also affect the duration of sensors' operation), but additionally we also control the transmission range (and thus the power dissipation).

Another centralized approach ([9]), is "sensor specific", in the sense it uses some smart powerful sensors that have high processing abilities. In particular, this algorithm assumes that each node is aware of its absolute location via a GPS or a relative location. The sensors must be capable of estimating the distance of the target from the sensor readings.

Distributed Approaches. As opposed to centralized processing, in a distributed model sensor networks distribute the computation among sensor nodes. Each sensor unit acquires local, partial, and relatively coarse information from its environment. The network then collaboratively determines a fairly precise estimate based on its coverage and multiplicity of sensing modalities. Several such distributed approaches have been proposed. Although we are not comparing with them, we shortly discuss some of them, for completeness.

In [13], a cluster-based distributed tracking scheme is provided. The sensor network is logically partitioned into local collaborative groups. Each group is responsible for providing information on a target and tracking it. Sensors that can jointly provide the most accurate information on a target (in this case, those that are nearest to the target) form a group. As the target moves, the local region must move with it; hence groups are dynamic with nodes dropping out and others joining in. It is clear that time synchronization is a major prerequisite for this approach

to work. Furthermore, this algorithm works well for merging multiple tracks corresponding to the same target. However, if two targets come very close to each other, then the mechanism described will be unable to distinguish between them.

Another nice distributed approach is the dynamic convoy tree-based collaboration (DCTC) framework that has been proposed in [20]. The convoy tree includes sensor nodes around the detected target, and the tree progressively adapts itself to add more nodes and prune some nodes as the target moves. In particular, as the target moves, some nodes lying upstream of the moving path will drift farther away from the target and will be pruned from the convoy tree. On the other hand, some free nodes lying on the projected moving path will soon need to join the collaborative tracking. As the tree further adapts itself according to the movement of the target, the root will be too far away from the target, which introduces the need to relocate a new root and reconfigure the convoy tree accordingly. If the moving targets trail is known a priori and each node has knowledge about the global network topology, it is possible for the tracking nodes to agree on an optimal convoy tree structure; these are at the same time the main weaknesses of the protocol, since in many real scenaria such assumptions are unrealistic.

The interested reader is encouraged to refer to [21], the nice book by F. Zhao and L. Guibas, that even presents the tracking problem as a "canonical" problem for wireless sensor networks. Also, several tracking approaches are presented in [18].

A preliminary version of this work has appeared in [15].

2 The Model and its Combinatorial Abstraction

2.1 Sensors and Sensor Network

We abstract the most important technological specifications of existing wireless sensor systems. Each sensor in our model is a fully-autonomous computing and communication device, equipped with a set of monitors (e.g. sensors for temperature, humidity etc.) and characterized mainly by its available power supply (battery) and the energy cost of (1) sensing (i.e. receiving and processing) (2) data sending. The *sensing range* $R_1(t)$ is of particular importance here. We also assume that the sensor can transmit data (to nearly devices) within a range $R_2(t)$. Both these ranges may vary with time. This means that the power spending by the sensor can be set at various different levels. We also assume some law of energy consumption that is range-dependent (e.g. the order of the energy spent is quadratic in the transmitting distance; depending on environmental conditions and their harshness the exponent can be larger than 2).

Let n be the total number of *available* sensor devices for deployment. Let S be the set of *possible* deployment positions (i.e. the union of 3D curves as we described earlier). Let T be a period of time. For each sensor σ of the n available sensors, a placement act, A_σ, is a decision (i) either not to deploy σ or (ii) to deploy σ in an initial position $\vec{p}$ in S, for a period $T_\sigma \subseteq T$, with a pre-specified pattern of $R_1(t), R_2(t)$ $(t \in T_\sigma)$ and a possible trajectory of σ moving in S for all t in T_σ. All the placement acts, together, form a sensor network N. We assume here that N is capable of (somehow) reporting the sensing of local events (i.e. tracking events) to some set of sinks T_N.

2.2 A model for targets

In contrast to models that allow only a single moving target, we allow *multiple* targets. We assume that *the initial positions of all targets are arbitrary.* Also, we assume that *all targets arise in arbitrary times* during the network operation.

With respect to target mobility, we assume that each target follows an arbitrary path in S which is however continuous i.e. we disallow the target to instantaneously "jump" to another location. Furthermore, *we do not limit the movement speed of targets*; we only assume that when a target enters the sensing area of an (awake) sensor, it does not manage to leave this area before being sensed. This limit on motion speed is rather trivial, since sensing speed is very high i.e. practically the time needed for a target to be sensed is very close to zero.

2.3 The combinatorial model

Assume a given complicated 3D domain, S, with obstacles that disallow signal transmissions (e.g. a set of corridors in buildings or different shape obstacles in a mountain). By "complicated" we mean that the domain can be represented by *arbitrary* three-dimensional curves (see also the problem description subsection), i.e. the only modeling restriction on the curves is their continuity. We further assume that the domain can be represented by the union of λ routes $R_1, \ldots, R_\lambda$ (each can be realized by e.g. a moving robot or air-vessel). We are also given a period T of time.

We wish to equip each route with sensors (of varying capabilities e.g. varying transmission range and operation times) so that any moving object in the domain at any time t in T can be "seen" by at least three sensors (and, thus, its instantaneous position can be found by triangulation). If we can manage this, we can monitor the motion of any moving object within S during T.

A trivial, but very costly, solution is to equip *each route* with sensors (in various points of the route) each operating during the whole T and being able to track any motion in a part of the route. However, one could *exploit overlaps of routes at certain places*, sensors that can monitor pieces of several routes ("nearby") during their operation, or even sensors on top of certain moving objects of our own that wander in the maze of routes.

In order to argue about such economic tracking methods, we view each R_i partitioned into several "route pieces" r_{ij}. Let n_i be the number of the pieces of R_i. We also partition the period T into suitable *intervals* $\tau_1, \ldots, \tau_k$ so that $T = \tau_1 + \cdots + \tau_k$.

We call an *"element"* each pair (r_{ij}, τ_m) , for $1 \leq i \leq \lambda$, $1 \leq j \leq n_i$ and $1 \leq m \leq k$. There are $n = \left(\sum_{i=1}^{\lambda} n_i\right) \cdot k$ such elements.

We can then describe sensor placements (that work for a certain duration each) as relations (sets) between those elements. For example: (a) a sensor placed at r_{i5} can also "see" r_{j7}, r_{k30}. This sensor can operate for 3 intervals. If we start it at τ_1 then the element (r_{i5}, τ_1) "covers" the elements (r_{i5}, τ_2), (r_{i5}, τ_3) but also (r_{j7}, τ_1), (r_{j7}, τ_2), (r_{j7}, τ_3) and (r_{k30}, τ_1) (r_{k30}, τ_2) (r_{k30}, τ_3). (b) A sensor is attached to a moving object that moves from r_{11} to r_{12} to r_{13} and then r_{34}, r_{35}, r_{36}. The sensor lasts 6 intervals and our moving object starts at τ_5. Then, element (r_{11}, τ_5) "covers" elements (r_{12}, τ_6), (r_{13}, τ_7) and also (r_{34}, τ_8), (r_{35}, τ_9), (r_{36}, τ_{10}) (and itself, of course).

In general, each placement of a certain sensor activated at a certain time and operating for a certain time, corresponds to a *set of "covered" elements*. Each such "set" has *a certain cost* e.g. it is more expensive if the sensor's battery is such that the sensor lasts for a long time. Also it is more expensive if its sensing range is large and can "see" more route pieces.

Definition 1 A redundant monitoring design (RMD) D is a set of possible choices of $q \leq n$ sensors $\sigma_1, \ldots, \sigma_q$, each with a placement act $A(\sigma_i)$ of the form "deploy" and the associated cost of the placement act.

Each RMD results in a family of sets $\Sigma_1, \ldots, \Sigma_q$, each having a cost $c(\Sigma_i) > 0$ and each being a subset of our universal set of elements $U = \{e : e \text{ is a pair } (r_{ij}, \tau_m)\}$.

For *feasibility* of the RMD we can require:
(a) that the union of all Σ_i is U.
(b) that each e in U belongs to at least 3 Σ_i sets initially.
However, this is not necessary for our method, since the method itself will discover an infeasible RMD.

Definition 2 Given are an instance of an RMD of sets $\Sigma_1, \ldots, \Sigma_q$ on the universe U of elements related to the domain S and the period T, and also a cost $c(\Sigma_i) \geq 0$ for each Σ_i. Then, an *optimal* final monitoring decision (Optimal FMD) is a sub-collection, F, of sets $\{\Sigma_{t_1}, \ldots, \Sigma_{t_{q'}}\}$ (where $q' \leq q$ and each $t_i \in \{1, 2, \ldots, q\}$), whose total cost $c(F) = \sum_{j=1}^{q'} c(\Sigma_{t_j})$ is minimum, and such that (i) $\cup_i \sum_{t_i} = U$ (ii) each e in U belongs to at least 3 sets in the sub-collection F.

We note that we can construct several RMDs for each run of our Algorithm, get a close to optimal solution (FMD) for each and select the best among them.

3 A way to compute near optimal FMDs

From the above formulation, the Optimal FMD problem is actually the following "AT-LEAST-3-SET-COVER" problem:

AT-LEAST-3-SET-COVER ($\geq 3SC$): Given $D = (\Sigma_1, \ldots, \Sigma_q)$ where each $\Sigma_i \subseteq U$ (and the union of all Σ_i is U) and given the costs $c(\Sigma_i) \geq 0$, select a minimum total cost sub-collection F of D so that each element e in U belongs to (is "covered" by) at least 3 sets in F.

Important note: Note that geometry and geometric covers can not help here because the domain since S is a complicated 3D domain that can be highly irregular; also, the timing parts of the elements escape the Cartesian geometry; finally, the cost of each Σ_i is a complicated function of placement decisions of sensors of various capabilities.

The problem of $\geq 3SC$ is NP-hard. This is so, since the usual min-cost SET-COVER problem can be reduced to it by adding to any instance of SET-COVER two sets of zero cost, each covering all elements.

We now describe a formulation of the problem as an integer linear program and give an approximate solution based on randomized rounding. Note that our method can be extended to low cost "SET-COVER by a at least l sets" (let us denote this problem as $\geq l$-SET-COVER) if the redundancy of $l > 3$ is needed to make the *redundant monitoring decision* D easier to construct and fault-tolerant.

We remind the reader of the computational complexity of set covering problems: Lund and Yannakakis ([14]) showed in 1994 that set cover cannot be approximated in polynomial time to within a logarithmic factor, unless NP has quasi-polynomial time algorithms. Feige ([7]) improved their inapproximability lower bound, under the same assumptions, giving a slightly better bound that essentially matches the approximation ratio achieved by the greedy algorithm. Alon, Moshkovitz, and Safra established in [2] a larger lower bound, under the weaker assumption that P $\neq$ NP. The k-set cover problem is a variant in which every set is of size bounded by k. While k-set cover problem can be solved in polynomial time (via matchings) for $k = 2$, it is NP-complete and even MAX SNP-hard for $k \geq 3$. Greedy algorithms in that case achieve an approximation ratio of $\ln k + \Theta(1)$. Hardness results in [7] show that it is not approximable within $(1 - \epsilon) \ln n$, under strong complexity-theoretic evidence. For 3-set cover, besides linear programming techniques (fractional covers), also local and "semi-local" approximation techniques have been used e.g. in [10, 11, 6].

3.1 The randomized rounding method

For set Σ let $x(\Sigma)$ be 1 if Σ is selected and 0 else. We want to

$$\text{minimize} \sum_{\Sigma \text{ in } D} x(\Sigma) \cdot c(\Sigma)$$

subject to
(1) $x(\Sigma) \in \{0, 1\}$
(2) Let $E(e)$ be the collection of all Σ containing element e. Then, for each element e,

$$\sum_{\Sigma \text{ in } E_e} x(\Sigma) \geq 3$$

Let IP1 be the above integer linear program.

We relax the above integer program to the following linear program (LP1):

$$\text{minimize} \sum_{\Sigma \text{ in } D} x(\Sigma)c(\Sigma)$$

given that
(1)$\forall \Sigma, 0 \leq x(\Sigma) \leq 1$
(2) $\forall e, \sum_{\Sigma \text{ in } E(e)} x(\Sigma) \geq 3$

Let $\{p(\Sigma)\}$ be the optimal solution to the above (i.e. $x(\Sigma) = p(\Sigma)$ get the minimum). We can find $\{p(\Sigma)\}$ in polynomial time in the size n of the redundant monitoring decision D.

We then form a subcollection of sets as follows:

Initially $C = \emptyset$
Experiment E.

For each Σ in D, put Σ into the sub-collection C with probability $p(\Sigma)$, independently of the others.

The experiment E above outputs a sub-collection C. Clearly,

$$E(cost(C)) = \sum_{\Sigma \text{ in } D} c(\Sigma) \Pr\{\Sigma \text{ is picked}\} = \sum_{\Sigma \text{ in } D} c(\Sigma)p(\Sigma) = OPT_f$$

where OPT_f is the optimal value of the linear program LP1. The found collection C has a very nice expected cost (even better than what one can achieve in our original *integer* problem) but we have to examine feasibility.

Since we want to get a C where each e in U belongs to at least 3 sets of C, we must examine whether the (random) C obtained has this property.

Definition 3 *Let α be an element in U and C obtained by the experiment E. We denote by $p(\alpha, C)$ the probability that α belongs to at least 3 sets of C.*

Let w.l.o.g. $\Sigma_1, \ldots, \Sigma_\lambda$ be the sets of our RMD containing element α. Here, we must have $\lambda \geq 3$, else LP1 will report infeasibility. W.l.o.g. denote by p_i the probability $p\left(\sum_i\right)$ obtained via LP1, i.e. that Σ_i is chosen to be in C. Assuming that LP1 is feasible we get:

$$\gamma = p_1 + \cdots + p_\lambda \geq 3 \tag{1}$$

Let A_3, N_0, N_1, N_2 be the events:

A_3 = "α is covered by at least 3 sets in C"
N_0 = "α does not belong to any set in C"
N_1 = "α belongs to exactly one set in C"
N_2 = "α belongs to exactly two sets in C"
Now,

$$p(\alpha, C) = \Pr\{A_3\} = 1 - \Pr\{N_0\} - \Pr\{N_1\} - \Pr\{N_2\} \tag{2}$$

We now estimate $\Pr\{N_i\}$, for $i = 0, 1, 2$:
We will repeatedly use the following fact:
Fact (*) If the numbers $x_1, \ldots, x_\lambda$ are each in $[0, 1]$ and $x_1 + \cdots + x_\lambda \geq \gamma$, then $(1-x_1)\cdots(1-x_\lambda) \leq (1 - \frac{\gamma}{\lambda})^\lambda \leq e^{-\gamma}$ (it is assumed that $\gamma \leq \lambda$).
Fact (*) can be proved via an easy induction, or via the fact that the arithmetic mean is bigger or equal than the geometric one.
Proof of Fact (*) Let $y_i = 1 - x_i, i = 1, \ldots, \lambda$. Then, $\sum y_i = \lambda - \sum x_i \leq \lambda - \gamma$. So,

$$\frac{\sum y_i}{\lambda} \leq 1 - \frac{\gamma}{\lambda} \tag{3}$$

But, $\frac{\sum y_i}{\lambda} \geq \sqrt[\lambda]{\pi y_i}$ (arithmetic mean vs geometric mean), so:

$$\frac{\sum y_i}{\lambda} \geq \sqrt[\lambda]{(1-x_1)\cdots(1-x_\lambda)}$$

So, by Equation (3), $\sqrt[\lambda]{(1-x_1)\cdots(1-x_\lambda)} \leq 1 - \frac{\gamma}{\lambda}$. So, $(1-x_1)\cdots(1-x_\lambda) \leq (1-\frac{\gamma}{\lambda})^\lambda$ $\diamond$
We now have:
(a) $\Pr\{N_0\} = (1-p_1)\cdots(1-p_\lambda)$. By the Fact (*), then

$$\Pr\{N_0\} \leq \left(1 - \frac{\gamma}{\lambda}\right)^\lambda \leq \exp(-\gamma)$$

.
Also, by inclusion-exclusion,

$$\Pr\{N_1\} = p_1 \cdot \Pr\{\text{no cover of } \alpha \text{ in the trials of } \Sigma_2, \ldots, \Sigma_\lambda\} +$$

$$+(1-p_1)\Pr\{1 \text{ cover of } \alpha \text{ in the trials of } \Sigma_2, \ldots, \Sigma_\lambda\} \leq p_1 \cdot \frac{1}{e^{\gamma - p_1}} + (1-p_1)\Pr\{N_1\}$$

(because $p_2 + \cdots + p_\lambda = \gamma - p_1$ so $(1-p_2)\cdots(1-p_\lambda) \leq \left(1 - \frac{\gamma - p_1}{\lambda}\right)^\lambda = \exp(-(\gamma - p_1))$ and also because $\Pr\{1 \text{ success in the trials of } \Sigma_2, \ldots, \Sigma_\lambda\} \leq \Pr\{N_1\ \}$).
So,

$$\Pr\{N_1\} \leq p_1 \cdot \frac{1}{e^{\gamma - p_1}} + (1-p_1)\Pr\{N_1\}$$

so

$$\Pr\{N_1\} \leq \frac{1}{e^{\gamma - p_1}} \leq \frac{1}{e^{\gamma - 1}}$$

Similarly,

$$\Pr\{N_2\} = p_1 \cdot \Pr\{1 \text{ cover of } \alpha \text{ in the trials of } \Sigma_2, \ldots, \Sigma_\lambda\} +$$

$$+(1-p_1)\Pr\{2 \text{ covers of } \alpha \text{ in the trials of } \Sigma_2, \ldots, \Sigma_\lambda\ \} \leq p_1 \cdot \frac{1}{e^{\gamma - p_1}} + (1-p_1)\Pr\{N_2\}$$

also, so $\Pr\{N_2\} \leq \frac{1}{e^{\gamma - 1}}$.

Thus,

$$\Pr\{A_3\} = p(a,c) \geq 1 - \frac{1}{e^{\gamma}} - \frac{2}{e^{\gamma-1}}$$

and $\gamma \geq 3$. So, $\Pr\{A_3\} = p(a,c) \geq 1 - \frac{1}{e^3} - \frac{2}{e^2} > \frac{1}{2}$. Let $\xi = 1 - \frac{1}{e^3} - \frac{2}{e^2}$. Hence, we get the following:

Theorem 1

$$\Pr\{\alpha \text{ is covered by at least 3 sets in } C\} = p(\alpha, c) \geq 1 - \frac{1}{e^3} - \frac{2}{e^2} = \xi$$

We now repeat the experiment E (with the same $p(\Sigma_i)$) to get $r = c \log n$ such collections $C_1, C_2, \ldots, C_r$.

Let $V = C_1 \cup C_2 \cup \ldots \cup C_r$. By independence of the repeated experiments, if the event AV_3 is :

$AV_3 =$"element α is not covered by at least 3 sets in V" then

$$\Pr\{AV_3\} \leq (1 - \xi)^{c \log n}$$

We can always choose c so that $(1 - \xi)^c < 1/4$. Then,

$$\Pr\{AV_3\} \leq \left(\frac{1}{4}\right)^{\log n} \leq \frac{1}{n^2}$$

Thus the probability that there is an element in U not covered by at least 3 sets of V is bounded by above by $n \cdot n^{-2} = 1/n$.

So, we get the following:

Theorem 2 The collection V obtained satisfies: (i)

$$\Pr\{V \text{ covers each element by at least 3 sets}\} \leq 1 - \frac{1}{n}$$

and (ii)

$$E(cost(V)) \leq c \log n \; OPT_f \leq c \log n \; OPT$$

where OPT is the cost of the *optimal* FMD.

Let $\rho = c \log n$. Note that by the Markov inequality it is

$$\Pr\{cost(V) < 2\rho \, OPT_f\} \geq \frac{1}{2}$$

Thus, the probability that V is valid and has cost less than $2\rho OPT_f$ is at least

$$1 - \left(\frac{1}{n} + \frac{1}{2}\right) = \frac{1}{2} - \frac{1}{n}$$

We can then repeat the whole process an expected number of at most 2 times and get a V which is verified to be an almost optimal and valid FMD. Note that we have also shown:

Theorem 3 The problem $\geq 3SC$ can be approximated in polynomial expected time with an approximation ratio $\Theta(\log n)$.

4 The triangulation issue

The solution to $\geq 3SC$ of the last section selects a close to optimal FMD. Thus, it also specifies the initial positions of the associated (selected) sensors. However, no guarantee is provided that for any element e, the 3 elements "covering" e actually form a triangle (e.g. they may be on a line). We propose to handle this via a "post-processing" step as follows:

For each $e = (r_{ij}, \tau_m)$ let $\sigma_1(e), \sigma_2(e), \sigma_3(e)$ be the 3 sensors covering e (i.e. covering r_{ij} at τ_m). We now modify their placement act by perturbing the positions p_i^* of $\sigma_i(e)$ at τ_m by a random, small perturbation of center their p_i^* and radius $\epsilon > 0$, small enough so that the same sets are covered by them. We perform the perturbation act only for those e for which the $\sigma_i(e)$ are not forming a triangle.

At the end of the post-processing step, each e in U is covered by (at least) 3 sensors, whose positions (during τ_m) form a triangle with high probability. We can repeat the perturbation until the triangle is indeed formed.

This post-processing step can always be done, provided that S gives an infinitesimal free space around each of its points. This is safe to assume for any application. We note that our solution works for static sensors, and moving sensors whose motion is controlled by us, while it can not be applied to the case of sensors moving on their own.

5 Alternative collaborative processing methods for our approach

Our method is based on an easy to get RMD which is then "cleared" via the randomized rounding technique to get a low cost final monitoring decision. Its result is a selection of sensor placement acts, guaranteed to monitor each point $\overrightarrow{p}$ in S by at least 3 sensors for any time in the period T. When our localization method is combined with ability to distinguish all observed objects (e.g. by using unique IDs) then tracking of objects can be performed.

Clearly our method must be complemented by a way to report target positions and their associated times to some central facility. If the reporting delay is comparable to the speed of the target then the central facility can reconstruct the targets motion in real time. To this end, Delay and Disruption Tolerant Networking (DTN, see e.g. [12]) approaches can be useful, to improve network communication when connectivity is periodic, intermittent or prone to disruptions and when multiple heterogeneous underlying networks may need to be utilized to effect data transfers.

The sensor network may handle the localization reports (i.e. reports on target position at a certain time) distributedly. This reduces communication cost but the central facility must have a way to compare those reports into a consistent information about the trajectory of the target. The issues of local clocks and their synchronization is crucial for this and we do not solve it here. In fact, we view our approach as a building block for full tracking approaches.

6 Conclusions

To design low cost sensor networks able to efficiently localize and track multiple moving targets, we try to take advantage of possible coverage overlaps over space and time, by introducing an abstract, combinatorial model that captures such overlaps. Under this model, we abstract the localization and tracking problem by a combinatorial problem of set covering (by at least three sets, to ensure localization of any point in the network area, via triangulation). We then provide an efficient approximate method for sensor placement and operation, that with high probability and in polynomial expected time achieves monitoring decisions with a $\Theta(\log n)$ approximation ratio to the optimal solution.

We can start with redundant monitoring decisions D giving least frequency of covering an element which is very high. This allows a high flexibility in possible places, timings and motions of sensors to initially "over-cover" the domain. Then, we can run our algorithm to choose an economic implementation. Note that our solution is of low cost and also achieves continuous monitoring of any moving object in the period T. By extending our techniques, we can prove that the problem $\geq \lambda$-SET-COVER (λ any constant) has a (polynomial time) approximation ratio of $\Theta(\log n)$.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. Computer Networks, 371, 38:393422, March 2002.

[2] Noga Alon, Dana Moshkovitz, and Muli Safra, Algorithmic construction of sets for k-restrictions, ACM Transactions on Algorithms (TALG), v.2 n.2, p.153-177, April 2006.

[3] J. Aspnes, D. Goldberg and Y. Yang, On the computational complexity of sensor network localization. In: 1st International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS 2004), Springer-Verlag, Lecture Notes in Computer Science, LNCS 3121, pp. 3244, 2004.

[4] K. Chakrabarty, S. S. Iyengar, H. Qi, and E. Cho, Grid coverage for surveillance and target location in distributed sensor networks, IEEE Trans. Comput. 51(12) (2002).

[5] S. S. Dhilon, K. Chakrabarty, and S. S. Iyengar, Sensor placement for grid coverage under imprecise detections, Proc. 5th Int. Conf. Information Fusion (FUSION02), Annapolis, MD, July 2002, pp. 110.

[6] Rong-chii Duh and Martin Fuerer, Approximation of k -Set Cover by Semi-Local Optimization, in ACM STOC, 1997.

[7] Uriel Feige, A Threshold of $\ln n$ for Approximating Set Cover, Journal of the ACM (JACM), v.45 n.4, p.634 - 652, July 1998.

[8] C. Gui and P. Mohapatra, Power conservation and quality of surveillance in target tracking sensor networks, Proc. ACM MobiCom Conference, 2004.

[9] R. Gupta and S. R. Das, Tracking moving targets in a smart sensor network, Proc VTC Symp., 2003.

[10] M. M. Halldorsson, Approximating Discrete Collections via Local Improvements, in ACM SODA 1995.

[11] M. M. Halldorsson, Approximating k-Set Cover and Complementary Graph Coloring, in IPCO 1996.

[12] S. Jain, M. Demmer, R. Patra, K. Fall, Using Redundancy to Cope with Failures in a Delay Tolerant Network, in Proc. SIGCOMM 2005.

[13] J. Liu, J. Liu, J. Reich, P. Cheung, and F. Zhao, Distributed group management for track initiation and maintenance in target localization applications, Proc. Int. Workshop on Information Processing in Sensor Networks (IPSN), 2003.

[14] Carsten Lund and Mihalis Yannakakis, On the hardness of approximating minimization problems, Journal of the ACM (JACM), v.41 n.5, p.960-981, Sept. 1994

[15] S. Nikoletseas and P. Spirakis, Efficient Sensor Network Design for Continuous Monitoring of Moving Objects, in the Proceedings of the Third International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS 07), Springer Verlag, Lecture Notes in Computer Science (LNCS), Volume 4837, pp. 1831.

[16] S. Poduri and G. S. Sukhatme, Constrained coverage in mobile sensor networks, Proc. IEEE Int. Conf. Robotics and Automation (ICRA04), New Orleans, LA, AprilMay 2004, pp. 4050.

[17] N. S. V. Rao, Computational complexity issues in operative diagnosis of graph based systems, IEEE Trans. Comput. 42(4) (April 1993).

[18] Rajeev Shorey, A. Ananda, Mun Choon Chan, Wei Tsang Ooi, Mobile, Wireless, and Sensor Networks: Technology, Applications, and Future Directions, Wiley, 2006.

[19] B.Warneke, M. Last, B. Liebowitz, and K.S.J. Pister, Smart dust: communicating with a cubic-millimetre computer, Computer, 369 34:4451, January 2001.

[20] W. Zhang and G. Cao, Optimizing tree reconfiguration for mobile target tracking in sensor networks, Proc. IEEE InfoCom, 2004.

[21] Feng Zhao, Leonidas Guibas, Wireless Sensor Networks: An Information Processing Approach, Publisher: Morgan Kaufmann, 2004.

[22] Y. Zou and K. Chakrabarty, Sensor deployment and target localization based on virtual forces, Proc. IEEE InfoCom, San Francisco, CA, April 2003, pp. 1293 1303.