A Bidding Protocol for Deploying Mobile Sensors

Guiling Wang, Guohong Cao, and Tom LaPorta
Department of Computer Science & Engineering
The Pennsylvania State University,
University Park, PA 16802
Email: {guiwang,gcao,tlp}@cse.psu.edu

Abstract

In some harsh environments, manually deploying sensors is impossible. Alternative methods may lead to imprecise placement resulting in coverage holes. To provide the required high coverage in these situations, we propose to deploy sensor networks composed of a mixture of mobile and static sensors in which mobile sensors can move from dense areas to sparse areas to improve the overall coverage. This paper presents a bidding protocol to assist the movement of mobile sensors. In the protocol, static sensors detect coverage holes locally by using Voronoi diagrams, and bid for mobile sensors based on the size of the detected hole. Mobile sensors choose coverage holes to heal based on the bid. Simulation results show that our algorithm provides suitable tradeoff between coverage and sensor cost.

1. Introduction

Wireless sensor networks are expected to be intensively utilized in the future since they can greatly enhance our capability of monitoring and controlling the physical environment. Sensor networks are revolutionizing the traditional methods of data collection, bridging the gap between the physical world and the virtual information world [8, 11, 14, 16]. Due to the inextricable relation with the physical world, the proper deployment of sensors is very important for the successful completion of the sensing tasks issued [6, 17].

In many potential working environments, such as remote harsh fields, disaster areas and toxic urban regions, sensor deployment cannot be performed manually. To scatter sensors by aircraft is one possible solution. However, using this technique, the actual landing position cannot be controlled because of the existence of wind and obstacles, such as trees and buildings. Consequently, the coverage may be inferior to the application requirements no matter how many sensors are dropped. Moreover, in many cases, such as during in-building toxic-leaks [9, 10], chemical sensors must be placed inside a building from the outside. In these scenarios, it is necessary to make use of mobile sensors, which can move to the correct places to provide the required coverage.

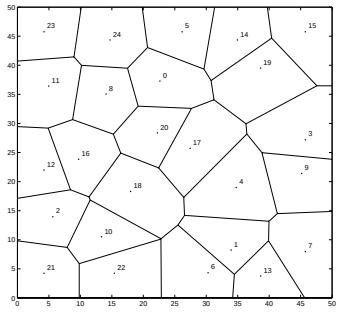
In our previous work [18], we designed three distributed self-deployment protocols that assist mobile sensors in moving from densely covered areas to sparsely covered areas, and demonstrated that these protocols can provide good coverage within a short time while keeping a low deployment cost in terms of moving distance and message complexity. Simulation results also showed that our algorithms can work under all initial deployments. However, to equip each sensor with a motor increases the network cost and is unnecessary when the coverage requirement is not very strict, or if sensors can be scattered in the target field relatively uniformly. To achieve a balance between sensor cost and coverage, we can deploy a mixture of mobile sensors and static sensors to construct sensor networks.

In this paper, we propose a new distributed *bidding* protocol for the deployment of mobile sensors, especially designed for sensor networks in which only a subset of deployed sensors are mobile. In our protocol, mobile sensors are treated as *servers* to heal *coverage holes*, which are locations not covered by any sensor. Each mobile sensor has a certain *base price* for serving one hole in the sensing field. The price is related to the size of any new hole generated by their movement. Static sensors will detect the coverage holes locally, estimate their sizes as *bids*, and *bid* the mobile sensors with a *base price* lower than their *bids*. Mobile sensors choose the highest bids and move to heal the largest coverage holes. This process iterates until no static sensor can give a bid higher than the base price of any mobile sensor and the process terminates naturally. Simulation results show our $bidding$ protocol can achieve high sensor coverage at low cost.

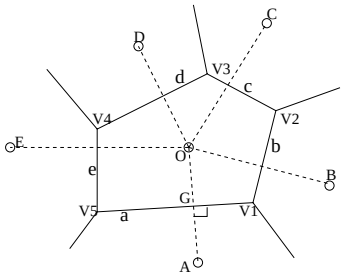
The rest of the paper is organized as follows. Section 2 introduces the technical preliminary on Voronoi diagram,

which is an important data structure used to detect the coverage holes. Section 3 analyzes the partial-mobile problem from the theoretical perspective. A detailed description of the bidding protocol is given in section 4. Section 5 describes our evaluation methodology and presents the simulation results. Section 6 concludes this paper.

2. Technical Preliminary On Voronoi Diagram



(a) Voronoi diagram



(b) Voronoi Cell

Figure 1. Voronoi diagram

The Voronoi diagram [5, 12, 13] is an important data structure in computational geometry. It represents the proximity information about a set of nodes. The Voronoi diagram of a collection of geometric nodes partitions the space into cells, each of which consists of the points closer to one particular node than to any others. Figure 1(a) is an example of the Voronoi diagram, and Figure 1(b) is an example of a Voronoi cell, which is used to introduce some technical terms about Voronoi diagram. A, B, C, D, and E are the Voronoi neighbors of node O. $lines$ a, b, c, d, and e are the Voronoi edges and the polygon constructed with these lines is the Voronoi cell of O. V_1, V_2, V_3, V_4, and V_5 are the Voronoi vertices of O, which are the intersection points of its Voronoi edges. O's Voronoi edges are the vertical bisectors of the line passing O and its Voronoi neighbors, e.g., $line$ a is $line$ OA's vertical bisector. All the points inside O's Voronoi cell are closer to O than any other nodes.

Our sensor deployment protocols are based on Voronoi diagrams. As shown in Figure 1(a), each sensor, represented by a number, is enclosed by a Voronoi cell. These polygons together cover the target field. The points inside one polygon are closer to the sensor inside this polygon than the sensors positioned elsewhere. If this sensor cannot detect the expected phenomenon, no other sensor can detect it. As a result, each sensor is responsible for the sensing task in its Voronoi cell and we choose the Voronoi diagram as our data structure to examine the coverage holes. In this way, each node can examine the coverage hole locally, and only needs to monitor a small area around it. Also, to construct the Voronoi cell, each sensor needs only to know the existence of its Voronoi neighbors, which reduces the communication complexity since it is sufficient to communicate locally with its Voronoi neighbors.

3. Theoretical Analysis

3.1. Problem Statement

When a portion of deployed sensors are mobile, the deployment problem can be described as follows: given a target field covered by a number of circles (the sensing circle of the static sensors), but still having some uncovered areas, how to place a certain number of additional circles (the sensing circle of the mobile sensors) to maximize the overall coverage.

3.2. Reducing to the Set-Covering Problem

This problem is at least as hard as the set-covering problem, which is NP-hard. The set-covering problem [4, 15] can be stated as follows. Given a ground set X and a collection F of its subsets satisfying

$$X = \bigcup_{S \in F} S,$$

find a minimum-size subset $C \subseteq F$, whose elements cover all of X:

$$X = \bigcup_{S \in C} S.$$

In our deployment problem, we can partition the target field into a large number of extremely small grid squares and assume that the mobile sensors can only be located in the center of the grid squares. If the partition is fine enough, choosing the grid squares in which to place the mobile sensors is the same as our deployment problem. The squares are assigned two colors, *white* and *black*, as shown in Figure 2. The black squares are those covered by the static sensors and the white squares are those not covered. We can view the area not covered by any static sensors in the target field as the ground set X, with the white grid squares as its elements. A sensing circle encompasses a set of white grid

squares, which form a subset (S) of X. The elements of S are determined by the center of the sensing circle, which is the location of the mobile sensor. We can say subset S is determined by the grid square where the mobile sensor is located. Then, we have a collection of n subsets, where n is the number of the white grid squares. Obviously, the union of these n subsets is equal to X. Finding the smallest number of sensors to completely cover the target field is equivalent to finding the smallest number of subsets whose union is X, which is a set-covering problem. Our deployment problem of getting the maximum coverage with N mobile sensors, is *linear-time reducible* to this set-covering problem with the following reduction function:

```
for i = 1, ..., N
    calculate the max coverage with i mobile sensors
    if the coverage is equal to 1
        stop and the answer is i
end
```

The set-covering problem is a strongly NP-hard problem. Current hardness results [7] show that there is no polynomial time approximation algorithm for set-covering whose performance is better than $(1 + o(1)) \ln n$ unless each problem in NP can be solved in $O(nO(lnln(n)))$ time.

3.3. Greedy Approximation Algorithm

Although our problem is a fundamentally difficult problem, and there is no optimal solution, we can still find some practical solutions to approximate the optimal solution based on heuristics. Similar to the greedy algorithm, which is a commonly used heuristic for the set covering problem, our deployment problem can be solved as follows:

```
for i = 1, ..., N
  1. assign each white square a weight, which is the
       number of white squares covered if a mobile
       sensor is placed inside.
  2. sort the white squares by their weights
  3. place a sensor in the white square with
       the highest weight, change the color of
       the newly covered squares to black.
end
```

4. The Bidding Protocol

4.1. General Idea of the Bidding Protocol

According to the greedy heuristic for this NP-hard problem, mobile sensors should move to the area where the most additional coverage can be obtained. After a mobile sensor leaves its original location to cover (heal) another coverage hole, it may generate a new hole in its original location. Thus, a mobile sensor only moves to heal another hole if its leaving will not generate a larger hole than that to be healed. However, due to lack of global information, mobile sensors may not know where the coverage hole exists. Even with the location of the coverage hole, it is still a big challenge to find the target position inside the coverage hole, which can bring the most additional coverage when a mobile sensor is placed there compared to other positions. We propose to let the static sensors detect the coverage holes locally, estimate the size of these holes, and determine the target position inside the hole. Based on the properties of the Voronoi diagram, static sensors can find the coverage holes locally and provide a good way to estimate the target location of the mobile sensors.

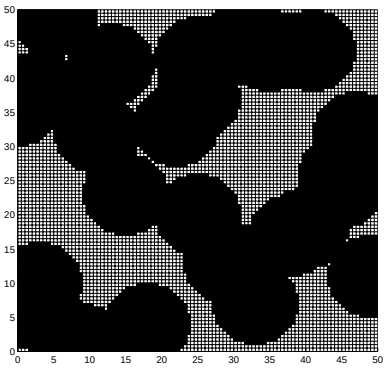


Figure 2. Reducing to the set-covering problem

The roles of mobile and static sensors motivate us to design a bidding protocol to assist the movement of the mobile sensors. We view a mobile sensor as a *hole healing server*. Its service has certain base price, which is the estimate of generated coverage hole after it leaves the current place. Static sensors are the bidders of the coverage hole healing services. Their bids are the estimated sizes of the holes they detect. Mobile sensors choose the highest bids and move to the target locations provided by the static sensors.

The bidding protocol runs round by round after the initialization period. During the initialization period, all static sensors broadcast their locations and identities locally. We choose the broadcast radius to be two hops, with which sensors can construct the Voronoi diagram in most cases. After the initialization period, static sensors broadcast this information again only when new mobile sensors arrive and need this information to construct their own Voronoi cells.

Each round consists of three phases: *service advertisement*, *bidding*, and *serving*. In the advertisement phase, mobile sensors broadcast their base prices and locations in a local area. The base price is set to be zero initially. In the bidding phase, static sensors detect coverage holes locally by examining their Voronoi cells. If such holes exist, they cal-

Notations:

$bid\langle h\_loc, h\_size\rangle$: target location of the mobile sensor and the estimated additional coverage
$list_b$: list of $bid\langle h\_loc, h\_size\rangle$
$mobile\langle id, m\_loc, base\_price\rangle$: the location of a mobile sensor and its base price
$list_m$: list of $mobile\langle id, m\_loc, base\_price\rangle$
$static\langle id, s\_loc\rangle$: the id and location of a static sensor
$list_s$: list of $static\langle id, s\_loc\rangle$

At static sensor N_i

(1) *Initialization:*
  (1.1) set $timer$ to be $(init\_interval + advertise\_interval)$, enter *Bidding phase* upon timeout
  (1.2) broadcast $static\langle i, s\_loc_i\rangle$
(2) *Upon entering Bidding phase*
  (2.1) set $timer$ to be $round\_interval$, enter *Bidding phase* upon timeout
  (2.2) construct Voronoi cell considering $list_s$ and $list_m$ whose $base\_price > 0$
  (2.3) compose $bid\langle h\_loc, h\_size\rangle$ if a coverage hole exists
  (2.4) find the closest mobile sensor N_j from $list_m$ so that $(base\_price_j < h\_size)$
  (2.5) send $bid\langle h\_loc, h\_size\rangle$ to N_j if N_j is found
(3) *Upon receiving $mobile\langle j, m\_loc_j, base\_price_j\rangle$ from N_j:*
  (3.1) add $mobile\langle j, m\_loc_j, base\_price_j\rangle$ to $list_m$
  (3.2) if N_j is a newly arriving sensor, broadcast $static\langle i, s\_loc_i\rangle$
(4) *Upon receiving $static\langle j, s\_loc_j\rangle$ from N_j:*
  (4.1) add $static\langle j, s\_loc_j\rangle$ to $list_s$

At mobile sensor N_i

(1) *Initialization:*
  (1.1) set $timer$ to be $init\_interval$, enter *Service-Advertisement phase* upon timeout
  (1.2) set $base\_price_i$ to be zero
(2) *Upon entering Service-Advertisement phase*
  (2.1) set $timer$ to be $advertise\_interval$, enter *Bidding phase* upon timeout
  (2.2) Broadcast $mobile\langle i, m\_loc_i, base\_price_i\rangle$
(3) *Upon entering Bidding phase*
  (3.1) set $timer$ to be $bidding\_interval$, enter *Serving phase* upon timeout
  (3.2) if $base\_price_i \neq 0$
    (3.2.1) Construct Voronoi cell considering $list_s$ and $list_m$ whose $base\_price > 0$
    (3.2.2) Compose $bid\langle h\_loc, h\_size\rangle$ if a coverage hole exists
    (3.2.3) find the closest mobile sensor N_j from $list_m$ so that $(base\_price_j < h\_size)$
    (3.2.4) send $bid\langle h\_loc, h\_size\rangle$ to N_j if N_j is found
(4) *Upon entering Serving phase:*
  (4.1) set $timer$ to be $serving\_interval$, enter *Service-Advertisement phase* upon timeout
  (4.2) if $length(list_b) > 0$
    (4.2.1) search $list_b$, get $bid\langle h\_loc, h\_size\rangle$ with largest h_size
    (4.2.2) move to h_loc
    (4.2.3) set $base\_price = h\_size$
    else
    (4.2.4) do duplicate healing detection
      set $base\_price = 0$ if duplicate healing happens
    (4.2.5) do local adjustment if no duplicate healing happens
(5) *Upon receiving $mobile\langle j, m\_loc_j, base\_price_j\rangle$ from N_j:*
  (5.1) add $mobile\langle j, m\_loc_j, base\_price_j\rangle$ to $list_m$
(6) *Upon receiving $static\langle j, s\_loc_j\rangle$ from N_j:*
  (6.1) add $static\langle j, s\_loc_j\rangle$ to $list_s$

Table 1. The Bidding Protocol

culate the bids and the target locations for the mobile sensors. Based on the received information from the mobile sensors, the static sensor can find a closest mobile sensor whose base price is lower than its bid, and sends a bidding message to this mobile sensor. In the serving phase, the mobile sensor chooses the highest bid and moves to heal that coverage hole. The accepted bid will become the new base price of the mobile sensor. After the serving phase, another new round can start after the mobile sensors broadcast their new locations and new base prices. As the base price increases monotonically, when no static sensors can give out a bid higher than the base price of the mobile sensors, the protocol terminates.

Before getting into the technical details of the bid protocol, we first use an example to show how the protocol works. As shown in Figure 3, the circles with stripped shadow are the sensing coverage of the static sensors, and the circles with grid shadow are that of the mobile sensors. Initially, 40 sensors are randomly placed in a $50m \times 50m$ flat field, among which 30% are mobile sensors. The initial coverage is 82%. The protocol terminates in the fifth round. The sixth round has the same topology as the fifth round and the coverage reaches 93%. The formal description of the algorithm is given in Table 1. We discuss the technical details in the following subsections.

4.2. Bid Estimation

In the bidding message, static sensors give out the estimated coverage hole size and the target location to which the mobile sensor should move. This information is calculated based on their Voronoi cells. Static sensors construct Voronoi cells considering only static neighbors and mobile neighbors which are not likely to move. These mobile sensors are detected by examining their base prices. If the base price of a mobile sensor is zero, this sensor has not moved yet and most likely it will move to heal some coverage holes soon. Thus, when detecting coverage holes, static sensors do not consider those mobile sensors which are about to leave. To construct its Voronoi cell, each sensor first calculates the bisectors of the considered sensors and itself based on the location information. These bisectors form several polygons. The smallest polygon encircling the sensor is the Voronoi cell of this sensor.

Having constructed the Voronoi cells, static sensors examine these cells. If there exists a coverage hole, the static sensor chooses the farthest Voronoi vertices as the target location of the coming mobile sensor. Inside one coverage hole, there are many positions that a mobile sensor can be located. If the mobile sensor is placed at the position farthest from any nearby sensors, the gained coverage is the highest since the overlap of the sensing circles between this new coming mobile sensor and existing sensors is the lowest. As shown in Figure 4, sensor N_a chooses its farthest Voronoi vertex O as the target location of the mobile sensor it bids for.

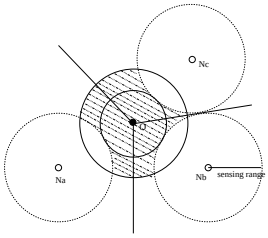


Figure 4. Bid estimation

From the global point of view, using the greedy heuristic to choose the largest coverage hole may not be optimal in some cases. As shown in Figure 5, A is the farthest Voronoi vertex of N_a. Although a high additional coverage can be obtained by placing a mobile sensor at A, it is not globally optimal since it leaves some scattered coverage holes which are hard to cover by placing additional mobile sensors. To deal with this problem, we propose an optimization which puts a limit on the maximum distance between the calculated target location and the bidder. As shown in Figure 5, by setting this maximum distance, a mobile sensor will be placed at B so that another mobile sensor can move to point C, to achieve better coverage.

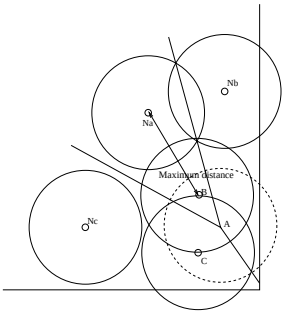
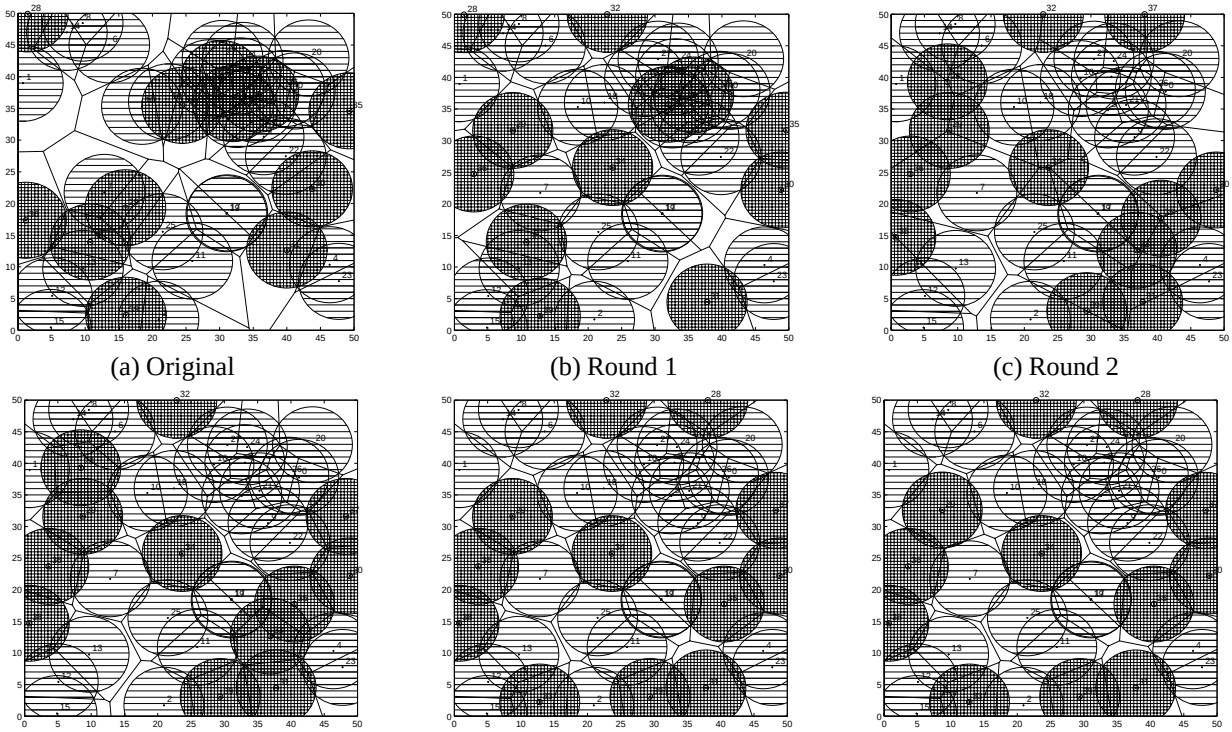


Figure 5. Optimize the greedy heuristic

Having determined the target location of the mobile sensor it bids, static sensors calculate the bid as: $\pi * (d - sensing\_range)^2$, where d is the distance between the bidder and the target location. As shown in Figure 4, N_a's bid is the area of the inner circle centered at O, which is not the actual additional coverage to be obtained. The actual additional coverage is the shadow area, which is difficult to calculate since it involves the union of circles. Using the inner circle as the bid simplifies the calculation, and can be used to approximate the actual additional coverage, which is the sensing circle minus the overlapping area of the sensing circles. The larger the overlapping area, the smaller the inner



(a) Original (b) Round 1 (c) Round 2

(d) Round 3 (e) Round 4 (f) Round 5

Figure 3. Snapshot of the execution of the bidding protocol

circle. Thus, the bid used can represent the relative size of the coverage holes.

The property of the Voronoi diagram guarantees that the shadow area is always the additional coverage. This can be explained as follows. The points inside one Voronoi cell are closest to the sensor in this cell. The points in the Voronoi edge are closest to these two sensors besides this edge. The Voronoi vertex is the point closest to the sensors which contribute to the existence of this vertex. The sensing circle centered at the Voronoi vertex must only overlap with the sensing circles of the sensors which contribute to the construction of this Voronoi vertex. Thus we guarantee that the shadow area shown in Figure 4 is always the additional coverage brought by placing a mobile sensor at O.

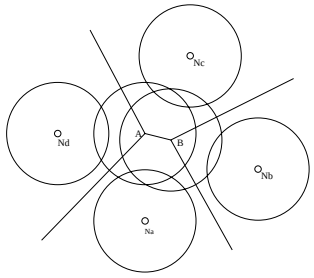
In addition to the static sensors, mobile sensors with a base price larger than zero also act as bidders. This is necessary because mobile sensors with a relatively larger price are essential acting as static sensors. At this point, they can assist the movement of other mobile sensors.

4.3. Duplicate Healing Detection

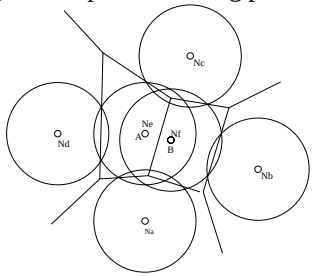
A **duplicate healing** occurs when two mobile sensors move to heal the same coverage hole. Figure 6(a) shows one example. A is the farthest Voronoi vertex of N_a and B is the farthest Voronoi vertex of N_b. Both N_a and N_b bid mobile sensors to their farthest Voronoi vertices. It is possible that both biddings are successful, resulting in duplicate healing. We propose a self-detection algorithm for mobile sensors to solve this problem. In the advertisement phase, mobile sensors broadcast their locations and base prices. If a mobile sensor hears that another mobile sensor in its neighborhood has a higher base price than its own, it will run the detection algorithm to check whether a duplicate healing has occurred. If yes, the mobile sensor reduces its base price to zero and it will likely to move to cover a different hole. This duplicate healing can be easily extended to the case that more than two mobile sensors move to heal the same hole since the detection algorithm will move all the sensors except that with the largest price (or the largest id when the base prices are the same) away to other holes.

In the detection algorithm, the detecting mobile sensor calculates a **detecting threshold**, equal to $\pi * (d_{min} - sensing\_range)^2$, where d_{min} is the distance to its closest neighbor. If the detecting threshold is smaller than its new base price, or d_{min} is smaller than the sensing range, a duplicate healing has occurred, since without duplicate healing, the calculated value should be the same as its new base price. As shown in Figure 6(b), N_e and N_f, located in A and B respectively, are the mobile sensors bid by N_a and N_b. N_f's new base price, the bid put forward by N_b, is calculated without considering N_e, which

is $\pi * (d_{b,f} - sensing\_range)^2$, where $d_{b,f}$ is the distance between N_b and N_f. Without a duplicate healing, $d_{b,f}$ is just d_{min}, and the calculated detecting threshold should be the same as the new base price. If duplicate healing has occurred, $d_{e,f}$ is d_{min}, which is smaller than $d_{b,f}$, and the detecting threshold is smaller than the new base price.



(a) The duplicate healing problem



(b) Fix the duplicate healing problem

Figure 6. Duplicate healing

4.4. Local adjustment with the VEC algorithm

It is possible that the target location of the mobile sensors calculated by the static sensors is not accurate. As shown in Figure 7, the mobile sensor should not be centered at A, since moving to B can achieve a better coverage. The process of moving the mobile sensor from A to B is called *local adjustment*.

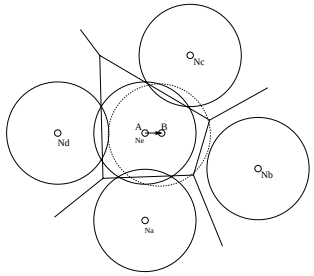


Figure 7. Local adjustment

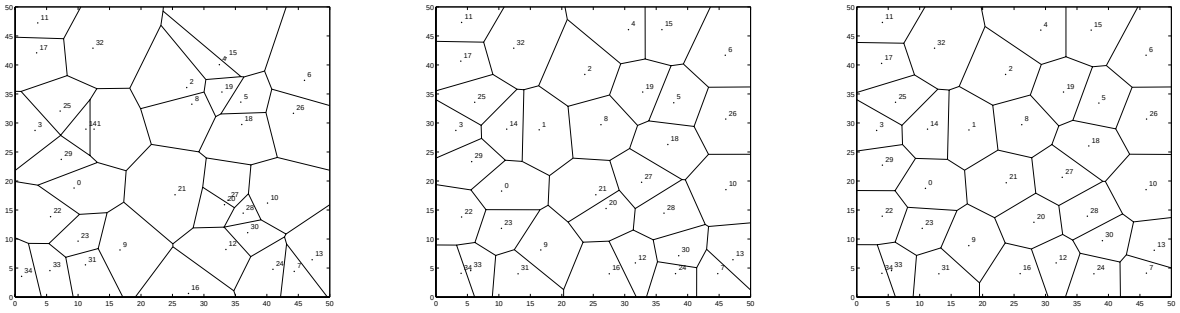
We use the VEC (VECtor-based) algorithm [18] to perform local adjustment. The VEC algorithm was proposed in [18] for sensor deployment when all sensors are mobile. VEC is motivated by the attributes of electro-magnetic particles: when two electro-magnetic particles are too close to each other, there will be a force to push them apart. Assume $d_{i,j}$ is the distance between N_i and N_j, and d_a is the average distance between two sensors when the sensors are evenly distributed into the working area. d_a can be calculated beforehand since we will be able to know the target area and the number of sensors to be deployed. The virtual force between two single sensors N_i and N_j are calculated as: $d_a - d_{i,j}$. This force will push the nodes to move $\frac{d_a - d_{i,j}}{2}$ away from each other. In case one sensor covers its Voronoi cell completely and should not move, the other sensor will move $d_a - d_{i,j}$. The final force is the summation of the forces from all its Voronoi neighbors. In this way, the virtual forces "pushes" sensors from the densely covered area to sparsely covered area. The detailed description of VEC can be found in [18]. Figure 8 shows how VEC works. Round 0 is the initial random deployment of 35 sensors in a 50m by 50m flat space, with the sensing range of 6 meters, and the coverage of 75.7% initially. After Round 1 and Round 2, the coverage is improved to 92.2% and 94.7% respectively.

As shown in Figure 7, since the mobile sensor is too close to N_d, the force between them pushes the mobile sensor to B by using the VEC algorithm.

5. Performance Evaluations

We evaluate our bidding protocol from two aspects: the *deployment quality* which is measured by the coverage and deployment time, and the *deployment cost* which is measured by the cost of the sensors and the energy consumption. Sensor coverage is the primary concern of our algorithm. Deployment time is a function of the number of rounds needed to achieve certain coverage and the time of each round. The duration of each round is primarily determined by the moving speed of the sensors, which is a mechanical attribute. Thus, we use the number of rounds to measure the deployment time.

Both mechanical moving and electrical communication consume energy, but mechanical movement is the dominant factor. Therefore, we use moving distance as the evaluation metric for energy consumption. Sensor cost is determined by the total number of sensors used and the percentage of mobile sensors among them. As more mobile sensors are used, a better coverage can be obtained, but the sensor cost will be increased. Thus, we also evaluate the tradeoff between cost and coverage.



(a) Round 0 (b) Round 1 (c) Round 2

Figure 8. Snapshot of the execution of VEC

5.1. Evaluation Methodology

The bidding protocol is implemented in the ns-2 (version 2.1b9a) simulator. Unless otherwise stated, we initially deploy 40 sensors randomly in the field and run 10 independent experiments. The results presented are the average of these experiments. For the network components, the physical layer is based on the RF MOTE from Berkeley, with 916.5MHZ OOK 5kbps as the bandwidth and 20 meters as the transmission range. We use 802.11 as the MAC layer protocol and DSDV as the routing protocol. We also did evaluations based on AODV and DSR. The obtained coverage and moving distance are similar. Based on the information from [3], we set the *sensing range* to be 6 meters. This is consistent with other current sensor prototypes, such as Smart Dust (U.C.Berkeley), CTOS dust, Wins (Rockwell)[2], and JPL[1].

5.2. Simulation Results

5.2.1. Tradeoff between cost and coverage In order to evaluate the tradeoff between cost and coverage, we compare three sensor deployment algorithms: random deployment, the VEC algorithm, and the bidding protocol. Random deployment is used for sensor deployment when all sensors are static; the VEC algorithm is used when all sensors are mobile; and the bidding protocol is used when both mobile and static sensors exist. Figure 9 shows the total number of sensors needed to reach certain coverage with these three algorithms. In this figure, random deployment is used when the percentage of mobile sensors is 0%, and the VEC algorithm is used when the percentage of the mobile sensors is 100%. The bidding protocol is used when the percentage of mobile sensors varies from 10% to 50%.

As shown in Figure 9, to reach a certain coverage, random deployment needs the most number of sensors, and the VEC algorithm needs the least number of sensors. As the percentage of mobile sensors increases, the required number of sensors to reach a certain coverage decreases. Compared to random deployment, the bidding protocol can significantly reduce the number of sensors required to reach a certain coverage. For example, to reach a 90% coverage, with only 10% of mobile sensors, the bidding protocol needs 30% fewer sensors; when 50% of the sensors are mobile, the required number of sensors is reduced by 50%. Compared to the VEC algorithm where all sensors are mobile, to reach 90% coverage, the bidding protocol requires 40% fewer mobile sensors, although it requires 20% more sensors in total with 50% mobile sensors. Note that the cost of static sensors will typically be cheaper than the mobile sensors, so the overall cost of using both static and mobile sensors will be reduced.

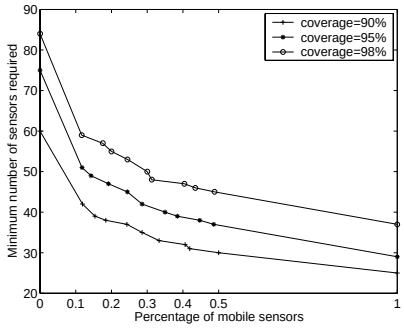
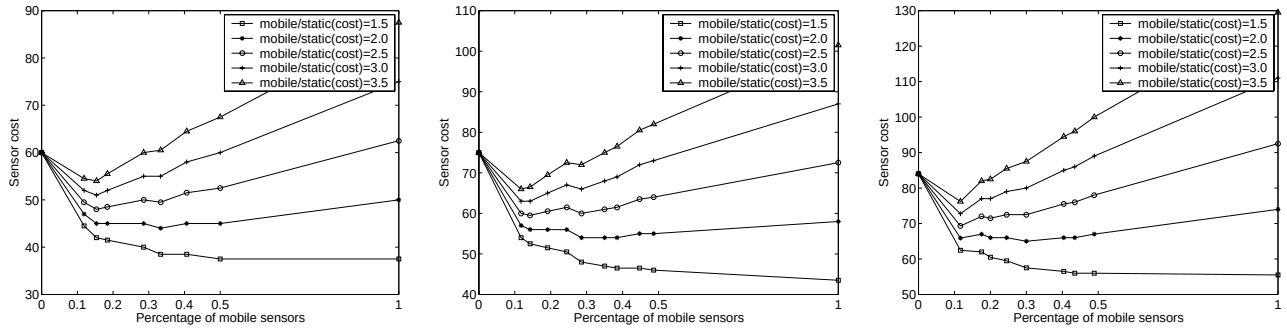


Figure 9. The number of sensors needed to reach certain coverage under different mobile percentage

Figure 10 shows the sensor cost of these three algorithms to reach a certain sensor coverage. Based on the *cost ratio* between the mobile sensor and the static sensor, the overall sensor cost of these three algorithms may be different. Intuitively, if the cost ratio is low (e.g., 1.5), increasing the percentage of mobile sensors can reduce the overall sensor cost. On the other hand, if the mobile sensors are very expensive, using only static sensors may have the lowest sensor cost (not shown in the figure). When the cost ratio



(a) To reach 90% coverage (b) To reach 95% coverage (c) To reach 98% coverage

Figure 10. The cost of sensors to reach certain coverage

is somewhere in the middle, the bidding algorithm which has a mix of mobile and static sensors can achieve the lowest sensor cost. For example, when the cost ratio is 3.5, to reach 95% coverage, the bidding algorithm has the lowest cost when 10% sensors are mobile. Based on this figure, we can see there is a tradeoff between cost and coverage. The bidding protocol can achieve a balance between these two most of the time.

5.2.2. Sensor Coverage Figure 11 shows the coverage reached after each round with different percentages of mobile sensors in our bidding protocol. It's easy to see that our bidding protocol is stable and the coverage monotonically increases as the number of rounds increases. Also, the deployment time is relatively short. The figure only shows the number of rounds needed to terminate in the worst case. On average, the number of rounds needed to terminate is 2.3, 3.9, 4.3, 5.6, 6.6 when 10%, 20%, 30%, 40% and 50% of the sensors are mobile respectively.

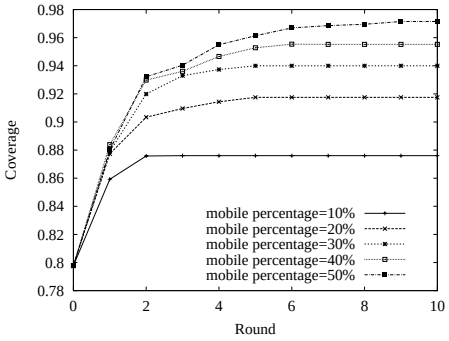


Figure 11. Sensor Coverage (n=40)

Generally speaking, as the percentage of mobile sensors increases, the sensor coverage also increases, since more mobile sensors can move to increase the coverage. However, at the end of the first round, the coverage is very close when the percentage of mobile sensors varies from 10% to 50%. Further, the coverage is 88.4% when 40% of the sensors are mobile, but the coverage is only 88% when 50% of the sensors are mobile. This phenomenon is because of duplicate healing and can be further explained by Figure 12. From the figure, we can see that duplicate healing occurs more often when the percentage of mobile sensors increases. For example, at the end of the first round, with 50% mobile sensors, there are 6.4 duplicate healings, which is about twice of that with 40% mobile sensors. This duplicate healing effectively limits the benefit of having additional mobile sensors in the first several rounds. After several rounds, duplicate healing disappears or significantly drops. This explains why the coverage is mainly decided by the percentage of mobile sensors after several rounds in Figure 11.

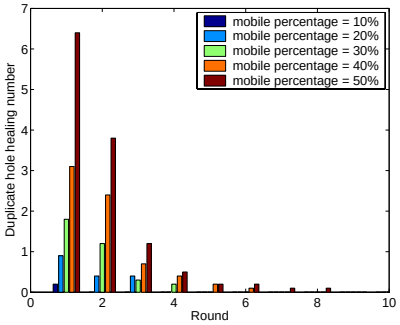


Figure 12. Duplicate healing number(n=40)

5.2.3. Moving Distance and Efficiency Figure 13 shows the average moving distance of mobile sensors with different percentages of mobile sensors. When the percentage of mobile sensors is low (e.g., 10% and 20%), the moving distance is relatively long, as expected. The counter-intuitive result is that when the percentage of mobile sensor is 50%, the average moving distance is longer than that with 30% mobile sensors. The duplicate healing problem is the major reason. When there are a large number of mobile sensors,

duplicate healing occurs more often and many sensors have to move more than once. If a sensor moves once and heals a coverage hole effectively, it will typically move a relatively short distance. From this figure, we conclude that the moving distance is the lowest when 30% of the sensors are mobile.

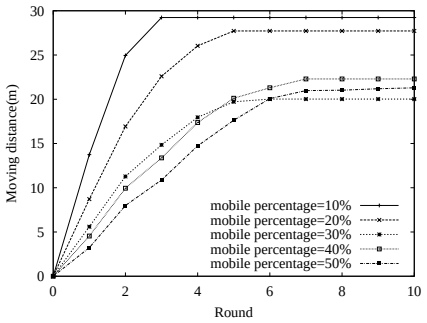


Figure 13. Average moving distance of mobile sensors (n=40)

6. Conclusions and Future Work

In this paper, we proposed to deploy a mixture of mobile and static sensors to construct sensor networks to provide the required uniform sensing service in harsh environments while maintaining a relatively low cost. We proved this is a NP-hard problem and designed a bidding protocol within which an optimized greedy approximation algorithm was used to deploy mobile sensors. Performance evaluation shows that our bidding protocol can increase the coverage significantly with low communication overhead, low computation complexity and limited movement. We also calculate the sensor costs to reach certain coverage by using all static sensors, all mobile sensors and the mix type with our bidding protocol, and find that the mix type can achieve a good balance between coverage and sensor cost.

To our knowledge, this is the first paper on deploying a mix of static and mobile sensors to meet the coverage requirement. We believe that this work will stimulate further research along this line. As future work, we will study how obstacles on the field affect the performance, and how to deal with non-uniform sensing coverage. We will also study other approximation algorithms and look at the tradeoff between mechanical movement and electrical communication.

Acknowledgments

We are grateful to Long Chen, Dr. Piotr Berman and Haoying Fu for their helpful discussions. We especially thank Dr. Martin Furer for his suggestions and comments. This work was supported in part by the National Science Foundation CAREER Award CCR-0092770.

References

[1] http://sensorwebs.jpl.nasa.gov.

[2] http://wins.rsc.rockwell.com.

[3] http://www-bsac.eecs.berkeley.edu/ shollar.

[4] *Introdution to Algorithms*. McGraw-Hill Book Company, 1994.

[5] F. Aurenhammer. Voronoi Diagrams – A Survey of a Fundamental Geometric Data Structure. *ACM Computing Surveys*, 1991.

[6] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan and K. k. Saluja. Sensor Deployment Strategy for Target Detection. *First ACM International Workshop on Wireless Sensor Networks and Applications*, 2002.

[7] U. Feige. A threshold of ln n for approximating set cover. *Proc. 28th ACM Annual Symp. on Theory of Computing, 314-318, 1996*.

[8] W. R. Heinzelman, J. Kulik and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Network. *Mobicom'99*, August 1999.

[9] A. Howard, M. J. Mataric and G. S. Sukhatme. An Incremental Self-Deployment Algorithm for Mobile Sensor Networks. *Autonomous Robots, Special Issue on Intelligent Embedded Systems*, September 2002.

[10] A. Howard, M. J. Mataric and G. S. Sukhatme. Mobile Sensor Networks Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem. *the 6th International Symposium on Distributed Autonomous Robotics Systems*, June 2002.

[11] C. Intanagonwiwat, R. Govindan and D. Estrin. Directed Diffusion: A Scalable and Robust Communication. *MobiCOM '00*, August 2000.

[12] S. Meguerdichian, F. Koushanfar, G. Qu and M. Potkonjak. Exposure In Wireless Ad-Hoc Sensor Networks. *Mobicom*, 2001.

[13] S. Meguerdichian, F. Koushanfar, M. Potkonjak and M. B. Srivastav a. Coverage Problems in Wireless Ad-hoc Sensor Network. *IEEE INFOCOM'01*, April 2001.

[14] G. J. Pottie and W. J. Kaiser. Wireless Integrated Network Sensors. *Communications of the ACM*, May 2000.

[15] P. Slavik. Approximation Algorithms for Set Cover and Related Problems. *Ph.D Thesis, State University of New York at Buffalo*, 1998.

[16] K. Sohrabi, J. Gao, V. Ailawadhi and G. J. Pottie. Protocols for self-Organization of a Wireless Sensor Network. *IEEE Personal Communication*, October 2000.

[17] S. Tilak, N. B. Abu-Ghazaleh and W. Heinzelman. Infrastructure Tradeoffs for Sensor Networks. *First ACM International Workshop on Wireless Sensor Networks and Applications*, 2002.

[18] G. Wang, G. Cao and T. La Porta. Movement-assisted sensor deployment. *Technical Report CSE-03-008, Dept. of Computer Science and Engineering, Pennsylvania State University, 2003. Also available at http://www.cse.psu.edu/ gcao/paper/movement.pdf*, 2003.