Distributed Algorithms for Dynamic Coverage in Sensor Networks

Lan Lin and Hyunyoung Lee*
Department of Computer Science, University of Denver

Abstract

The coverage problem is of great interest for many sensor network applications, for example, detection of intruders in the sensor field. Topological changes in sensor networks may affect qualities of sensor coverage. In this paper, we present two suites of algorithms for dynamically maintaining the coverage and the measures of its qualities. Using only local knowledge, our algorithms capture the dynamic changes of network topology and efficiently maintain the coverage by updating the radii of sensors combined with limited sensor mobility. Our algorithms are fully distributed and have the advantages of low communication complexity with no need of a tight bound on message propagation delay.

1 Introduction

With sensory equipments on-board, a sensor can measure the environment within a certain range of itself. Networked sensors with their communication and computation capabilities act as monitors of the environment. As sensors are deployed in a field, the coverage of the field by the sensors is of great interest for various application-specific reasons, for example, detection of intruders in the field. Sensors often remain in a fixed location once deployed, in which case the topology of the sensor network remains static. But if we take into consideration scenarios such that sensors may be out of service due to battery depletions, or sensors are mobile in the field, the network topology becomes dynamic. How to maintain the coverage in a dynamic network topology is the problem we study in this paper.

We assume the omni-directionality of a sensor and model the area covered by the sensor as a disk centered at the sensor. The radius of the disk varies according to the power spent on sensory equipments. The union of all the areas covered by the networked sensors is the coverage of the sensor network.

In the first suite of our algorithms, we study the path computation problems for a mobile agent moving across a sensor field. Given a starting point, S, and a stopping point, T, in the sensor field, two types of path problems have been proposed [4], *maximum breach path* and *maximum support path*: In the maximum breach path problem, we want to find a path between point S and point T that stays as far away as possible from sensors. It evaluates the vulnerability of a sensor network, i.e., how well the sensors are placed and to what extent they can be breached. In the maximum support path problem, we want to find a path between S and T that stays as closely to sensors as possible. It measures the efficiency of the network coverage. Figure 1(Right) shows an example of maximum breach path, in which the path between S and T, denoted by the thick dotted line, is not covered by the union of sensor disks. Figure 1(Left) shows an example of maximum support path, in which a path between S and T, denoted by the thick dotted line, is completely covered by the union of sensor disks.

When a sensor fails, a breach may occur on the coverage, i.e., the maximum breach path and/or maximum support path may no longer hold true. In Figure 1(Right), the sensor with disk indicated by the dotted circle is out of service, in which case the current maximum breach path is no longer true since now there exist paths between S and T going through the area used to be covered by the dotted circle. The minimum distance from points on the paths to the sensor network is greater than the current maximum breach path. Similarly, the maximum support path is no longer true in the case of sensor failure shown in Figure 1(Left). The loss of the node with disk in dotted circle causes a division of the covered region into two disconnected regions. Paths between S and T are no longer completely covered.

The distance of a point p to the sensor network is defined as the smallest Euclidean distance from p to any of the sensor nodes. A maximum breach path is a path that the minimum distance from points on the path to the sensor network is maximized. This distance is called the *worst-case coverage distance* of the network. A maximum support path is a path that the maximum distance of points on the path to the sensor network is minimized. This distance is called the *best-case coverage distance* of the network. We propose distributed algorithms for dynamically maintaining the worst-case coverage distance and the best-case coverage distance. To the best of our knowledge, this is the first work that solves this problem using distributed algorithms. Our algorithms have the advantages of low communication complexity with no need of a tight bound on message propagation delay.

In the second suite of our algorithms, we study the coverage maintenance problem raised from node failures. We propose network reconfiguration schemes that maintain the coverage of a sensor field with limited total energy consumption thus prolong the lifetime of the network.

We define the coverage area that each sensor participates in coverage as the area covered by the sensor and its one-hop and two-hop neighbors. When a node failure is detected, sensors make no effort of recovery until a thresh-

*Contact author. E-mail: hlee@cs.du.edu. Phone: +1-303-871-7732, Fax: +1-303-871-3010.

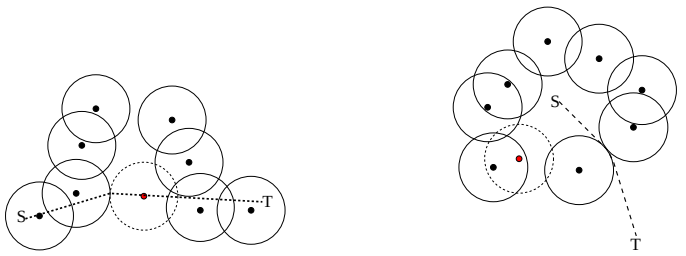


Figure 1: Left: example of best coverage radius. Right: example of worst coverage radius.

old of percentage of uncovered area is reached. In order to recover the area, sensors increase their radii and move within a limited range with limited energy cost. We design algorithms to determine the uncovered area and its percentage and to calculate the best strategy for recovering the area. Our algorithms use only local information and are fully distributed and highly scalable.

To migrate multi-hop neighbors of a failed node in order to recover the area has been proposed in [7]. Our approach combines mobility with different radii. We use probabilistic methods to measure the percentage of nodes alive, the total coverage deterioration, and the average migration distance, in time.

The rest of the paper is organized as follows. We present the two suites of algorithms in Sections 2, 3, and 4, followed by our simulation results in Section 5. We discuss some related works in Section 6 and conclude in Section 7.

2 Dynamic maintenance of best and worst coverage radii

In this section we assume a uniform sensor network in which all the sensors have the same sensing range and communication range. Sensors cannot detect any objects outside of the sensing range, neither can sensors have direct communication with any sensors outside of the communication range. Sensors are represented by a set of points P in $\mathbb{R}^2$. Let n be the total number of sensors, i.e., $|P| = n$. Let $D(P, r)$ be the union of all disks centered at points of P with radius r. Let $\overline{D(P, r)}$ be the complementary of $D(P, r)$. Given a pair of sensor nodes S and T: The *best coverage radius* [4] is defined as the minimum coverage radius r such that there exists a trajectory between S and T that is totally covered by $D(P, r)$. The *worst coverage radius* [4] is defined as the maximum coverage radius r such that there exists a trajectory between S and T that is covered by $\overline{D(P, r)}$. Examples of the radii are shown in Figure 1.

Theorem 2.1. *For best coverage radius, there exists at least one pair of nodes whose disks are tangent to each other.*

Proof. Suppose there does not exist a pair of nodes whose disks are tangent to each other. Then all pairs of nodes are either disconnected or overlapping. Since the best coverage radius completely covers a path between a pair of points S and T, the union of all the sensor disks covering the path is a connected region. Since the sensor disks are overlapping, we can decrease the radius such that two of the disks are tangent to each other. This contradicts the definition of best coverage radius. □

Similarly, we can prove the following.

Theorem 2.2. *For worst coverage radius, there exists at least one pair of nodes whose disks are tangent to each other.*

Network topological changes may cause changes to the best and/or worst coverage radii. Therefore, the maintenance of topological information of the network is the first step of the maintenance of the coverage radii. The second step is to determine the occurrences of breaches on the coverage. To restore the coverage, the topological information is employed to determine the new best and/or worst coverage radii. The coverage is restored when all the sensors start sensing and communicating using the new radii.

2.1 Maintenance of boundaries

The boundaries of the union of the sensor disks are composed of arcs of sensor disks that are not covered by any other sensors. For best coverage radius, there are two kinds of boundaries, the outer boundary of the area and the boundaries of the holes within the area. We identify the first kind with number 0 and the second kind with the identifiers the holes they belong to. Each hole is identified by a random number selected at the time the hole first appears. For worst coverage radius, there are three kinds of boundaries, the inner boundary and the outer boundary of the area, and the boundaries of the holes within the area. The inner boundary, identified as 1, is the boundary closest to the point enclosed by sensors, for example, point S in Figure 1. The outer boundary, identified as 0, is the boundary closest to the other point, for example, point T in Figure 1. The inner boundary and the outer boundary intersect at the tangent points of the pair of nodes that are tangent to each other.

Since we want to keep track of the boundaries, we only take into account the changes of network topologies that cause changes to the boundaries. There are four kinds of changes in terms of sensor dynamics: 1. a sensor on any of the boundaries fails or is out of service, 2. a sensor on any of the boundaries is moved to another location, 3. a node not on any of the boundaries fails or is out of service and causes a hole that is not covered by any sensors, 4. a sensor not on any of the boundaries is moved within the range of some boundary nodes.

These changes can be represented in two operations of the nodes, *insertion* and *deletion*. Insertion is to place a node at a location. Deletion is to remove a node from a location. Now, the four types of changes can be represented

as follows: 1. a boundary node is deleted, 2. a boundary node is deleted from a location and then is inserted at another location, 3. a non-boundary node is deleted, 4. a non-boundary node is deleted from a location and then is inserted at another location within the range of some nodes on the boundary.

In the first case, the neighbors each can determine if any section of their boundaries is now part of the boundary of the whole region, i.e., if they are on the boundary. This can be done by determining if any part of the arc used to be covered by the failed node is not covered by any of its own neighboring nodes. Those who become a boundary node update their status and notify their status changes to their neighbors. The first part of the second case runs the same as in the first case. In the second part, the neighboring nodes that are boundary nodes each determines if the new location of the moving node causes a change to their boundary status. This can be done by checking if the moving node completely covers any of its boundary curves. If it does, the node loses that arc as a boundary arc. Otherwise, it remains a boundary node. In the third case, the neighboring nodes of the failed node each determines if any part of the arc previously covered by the failed node is no longer covered by any of its neighboring nodes. If so, the node becomes a boundary node. The new boundary nodes select an identifier for this new region. The first part of the last case runs the same as in the third case. The second part runs the same as in the second part of the second case. Each case requires only constant number of messages with $O(\log n)$ bit complexity since each message contains a sender ID.

2.2 Dynamic best coverage radius

When a breach occurs, the neighboring nodes of the failed node elect two leader nodes from each of the disconnected sets. The leaders send out a message to neighboring boundary nodes notifying the disconnecting of the regions. Along with the message is the identifier of the region the leader belongs to. Clearly the boundary nodes receiving the message are all on the outer boundary, since the regions are now disconnected, as shown in Figure 2(Left), where the boundaries of the two disconnected regions are denoted by the thick curves. No boundary arc of the holes can cause disconnection before it first becomes an outer boundary.

The neighboring nodes of the failed node determine the temporary new radius by finding the shortest distance between the two disconnected sets. The leader nodes each sends the radius to neighboring boundary nodes together with the identifier of the region. The messages travel in the same direction.

To find the minimum increase of the radius, we only need to consider those nodes that are on the opposite sides of the breach. Upon receiving the radius, the boundary nodes increase their communication radius accordingly attempting to communicate with boundary nodes of the other side. There is no need to assume an approximately

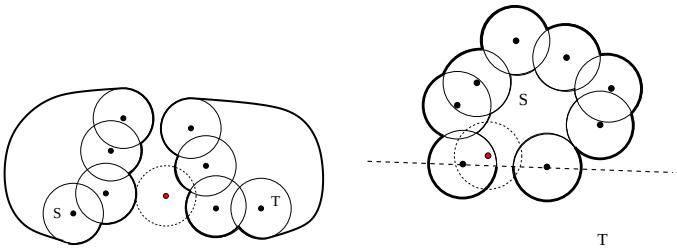


Figure 2: Left: Determining the best coverage radius. Right: Determining the worst coverage radius.

equal propagating speed on both sides. One side may propagate the radius at a faster pace, in which case the nodes of that side wait for any communication attempts from the other side. A node should be able to hear from a node of the other side if it is within its communication range. If a node hears from a node of the other side, the two nodes determine the new radius to be the half of the distance between them, i.e., when their disks are tangent to each other, as shown in Theorem 2.1. They then send this radius to their neighbors. If a node does not hear from any nodes of the other side within time ε, it concludes that the radius remains the same and sends it to its neighbors. This process repeats until the radius is received again by the two leaders. The radius is the new best coverage radius. The new radius is then propagated throughout the network.

We assume uniformly distributed sensors, thus the number of boundary nodes is $O(\sqrt{n})$. The message carrying the radius information is sent and received only once by each boundary node. Since each message carries the sender ID, which costs $O(\log n)$, the total message complexity for finding the best coverage radius is $O(\sqrt{n}\log n)$.

2.3 Dynamic worst coverage radius

The maintenance scheme for worst coverage radius is similar to the best coverage radius scheme. The leader nodes send the radius to neighboring boundary nodes together with the identifier of the region. The boundary nodes receiving the message from the leaders are all on either the inner or the outer boundary, for no boundaries of the holes can cause a breach before it is first on the inner or outer boundary, as shown in Figure 2(Right), where the inner boundary is denoted by the thin curves and the outer boundary by the thick curves.

To find the minimum increase of the radius, we only need to consider those nodes that are on the opposite sides of the breach. We need to treat the inner boundary and the outer boundary separately. The inner boundary propagates the radius to all of its nodes. The outer boundary only needs to propagate the radius to those nodes that are on the side of a *separator* opposite to S, a straight line shown in Figure 2(Right) as the dotted line that goes through the two leader nodes of the opposite sides. This is

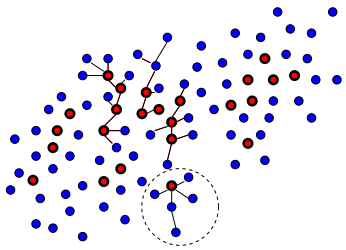


Figure 3: The nodes are divided into two sets: red (in thick circle) and non-red. A communication tree rooted at a red node is shown in the dotted circle.

because the inner boundary nodes on the other side of the separator are closer to each other than the outer boundary nodes of that side. We will not find a smaller radius between those outer boundary nodes.

Upon receiving the radius, a boundary node determines the new radius in the same way as in the dynamic best coverage radius case. Once the new worst coverage radius is found, it is propagated throughout the network. Finding the worst coverage radius requires $O(\sqrt{n}\log n)$ message complexity.

3 Dynamic coverage with migration of redundant nodes

Nodes in a sensor network can be divided into two sets: redundant nodes and non-redundant nodes. A *redundant node*, referred as *red node*, is a node whose sensing disk is completely covered by other sensors. A *non-redundant node*, referred as *non-red node*, is a node whose sensing disk is not completely covered by other sensors, i.e., part of the sensing area is covered by the node alone. An example of the two types of nodes is shown in Figure 3, with the redundant nodes as dots in circles (in red) and non-redundant nodes as dark dots (in blue). While node failure of a redundant node causes no change to coverage, a non-redundant node failure causes the loss of part of the coverage of the field. There has been research investigating in eliminating redundant nodes in order to save energy. Our scheme, however, uses the redundant nodes to recover the lost area from node failure. We assume that the energy cost of the movement for a sensor is ignorable, for example, when sensors are mounted on vehicles that use separate source of energy for movement.

We want to keep track the closest red node in distance to each non-red node such that when a non-red node fails the red node can be moved to the location of the non-red node to replace it. We call the red node the *recovery node* of the non-red node. In this scheme, we assume that sensors are unit disks and the distance between two sensors are Euclidean distance. Each non-red node determines its recovery node using a *connect* construction method as follows. If a non-red node has red nodes in its neighbors, it connects to the one closest and makes the red node its recovery node. Otherwise it connects to the neighboring non-red node whose closest red node is minimum distance to itself and makes the red node its recovery node. We call the structure a *communication tree*. An example of communication tree is in the dotted circle in Figure 3.

Property 3.1. *The communication tree is indeed a tree.*

Proof. We first prove that a node cannot connect to multiple nodes. This is true by the connect construction method. We then prove that the structure defined above has no loops. Suppose there exists a loop among k nodes. Then none of the k nodes is a red node since red nodes do not connect to other nodes. Thus all the k nodes are non-red nodes and do not connect to a red node. This contradicts the construction method. □

Lemma 3.2. *If there exists red nodes within range of a non-red node, one of the red nodes that the non-red node connects to is its recovery node.*

Proof. Suppose the statement is not true. Then there exists a red node that is closer to the non-red node in distance but is not among its neighboring red nodes. This cannot be true since the distance is Euclidean distance. □

Property 3.3. *The nodes of a communication tree is closer to the root node in distance than to any other red node.*

Proof. By Lemma 3.2 and the construction of the tree. □

When a node is inserted, for example, when a new sensor is deployed to the field, the communication tree is updated accordingly. If a node's sensing disk is not completely covered by neighboring nodes, it is deemed a non-redundant node. If the sensing disk is completely covered by its neighbors, it is a redundant node. When a non-red node is inserted, it communicates with neighboring nodes to find if any red nodes are within range or not. The non-red node picks the closest red node within range if it exists, otherwise it communicates with neighboring non-red nodes to find the locations of their closest red nodes, as shown in Figure 4(a). It picks the closest red node with minimum distance to itself. By this it adds itself to a communication tree of the red node, as shown in Figure 4(b). When a red node is inserted, it announces itself by broadcasting hello message containing its location information, as shown in Figure 5(a). A neighboring node receiving the message calculates the distance to the red node and decides whether it is the closest red node. If it is, it connects to the new red node, as shown in Figure 5(b).

When a non-red node fails, the neighboring nodes notify the closest red node of the non-red node to move to the location of the non-red node. Thus the lost area is completely recovered. The nodes previously connecting to the non-red node now connects to the red node which becomes non-red.

When a red node fails or is moved, nothing needs to be done to recover the area. But there are two kinds of changes to its neighboring nodes: (1) some neighboring

Figure 4: Insertion of a non-red node c (non-red node in circle, red node in solid dot).

Figure 5: Insertion of a red node c (non-red node in circle, red node in solid dot).

red nodes may become non-red nodes due to the loss of part of their coverage, and, (2) the non-red nodes having the red node as recovery node need to find a new red node and adjust their positions in the communication trees accordingly. In both cases, the non-red nodes on the communication tree of the lost red node need to find a new recovery node. It can be done as follows. The loss of the root of the tree is notified to all the nodes of the tree. Starting from the leaves of the tree, nodes broadcast message to their neighbors requesting their recovery node information. They pick the one closest and join the tree of that node. They then broadcast their new recovery node information. The non-leaf nodes decide their recovery node after they receive message of new recovery node. The original communication tree is dissolved in this manner. One example of it is shown in Figure 6.

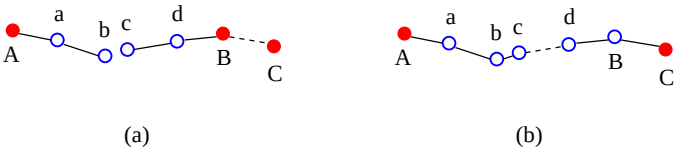


Figure 6: (a) Communication tree. (b) Node B becomes a non-red node and the subsequent change of node c joining another tree rooted in A. (non-red node in circle, red node in solid dot.)

4 Lazy coverage with different radii and limited mobility

Lazy failure detection. We define the energy cost of sensor movement E_m in terms of distance moved d times a constant k, $E_m = k \cdot d$, where k is determined by the particular vehicle. The energy cost of sensing E_s with radius r is $E_s = l \cdot r^2$ where l is a constant specified by the sensing equipment. The total cost of moving sensors to different locations and enlarging radii is $E_{total} = \sum_{i=1}^{n} E_{m,i} + \sum_{i=1}^{n} E_{s,i}$, where n is the total number of sensors. In our approach of combining mobility with different radii, we want the total cost to be minimized.

We assume that initially the field is completely covered by sensors. A sensor periodically exchanges beacons among its neighbors to maintain network connectivity. In addition, it also sends to its neighbors the locations of all its neighbors. Therefore each sensor knows its two-hop neighbors. A node failure is first detected by its neighbors and then broadcast to its two-hop neighbors. Thus each node knows the percentage of area covered within two-hop range. When the percentage of coverage falls below certain threshold α, a coverage maintenance scheme is employed to recover the lost area.

Assume that a failed node X has m neighbors, S_1, S_2, ..., S_m. Let the area covered by each node be $A(S_i)$, for $i = 1, 2, \ldots m$. Let the area covered by X be $A(X)$. We need to find the combination of movements of the neighbors to new positions P_1, P_2, ..., P_m and employments of different radii of sensing range for S_1, S_2, ..., S_m, such that the following conditions are fulfilled.

1. $A(X) \cap (\cup_{i=1}^{m} A(S_i))$ is maximized.
2. $E_{total} = \sum_{i=1}^{m} E_{m,i} + \sum_{i=1}^{m} E_{s,i}$ is minimized.

4.1 Dynamic coverage schemes

The complexity of the problem stated above is exponential. We propose recovery schemes under various constraints and assumptions. We use the lazy failure detection scheme to determine if a recovery is needed for a lost area caused by node failure. Since there may be multiple disjoint areas uncovered before a recovery scheme is performed and each uncovered area is monitored by multiple sensors, the recovery is a joint effort of multiple sensors. Figure 7 shows an example of two nodes in black with their one-hop (in solid line) and two-hop (in dotted line) neighbors. Two failed nodes in light (red) dots are two-hop neighbors of the black nodes. Each node is responsible for recovering its one-hop neighbors because two-hop neighbors are one-hop neighbors of some other nodes. In Figure 7, for example, recovering of the areas of the two red nodes is done by their one-hop neighbors a, b, and c.

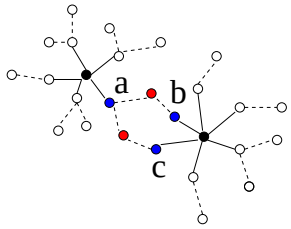


Figure 7: Dynamic recovery with one-hop neighbors.

4.1.1 Different radii without mobility

We propose a scheme that allows different radii for nodes without mobility of the nodes. Our scheme increases the radii of the one-hop neighbors of the failed node in $1+\varepsilon$ intervals. Each node keeps increasing its sensing and communication radii until the union of their sensing area completely recover the lost area. By increasing the radii the nodes also maintain the area they previously covered. The nodes communicate with each other to maintain the radii information of other nodes.

Different radii cause asymmetric communication, that is, node with smaller radius can hear from node with bigger radius but not vice versa. Thus node with bigger radius may not detect within its communication range the existence of nodes with smaller radii. In order to maintain connectivity we propose the clustering of nodes with smaller radii in a tree structure around the node with bigger radius. This can be done as follows. When a node with smaller radius hears from a node with a bigger radius outside of self communication range, it increases self radius in order to communicate with that node. This is based on the fact that the energy cost for increasing a smaller sensing range is less than the energy for increasing a bigger sensing range, derived from the following.

Let r_1 and r_2 be the two sensing ranges, $r_1 > r_2$. Then $E_{s,1} = l \cdot {r_1}^2$ and $E_{s,2} = l \cdot {r_2}^2$. If r_1 and r_2 are both increased by δ, $E'_{s,1} = l \cdot (r_1 + \delta)^2$ and $E'_{s,2} = l \cdot (r_2 + \delta)^2$. And $E'_{s,1} - E_{s,1} = l \cdot (2r_1\delta + \delta^2) > l \cdot (2r_2\delta + \delta^2) = E'_{s,2} - E_{s,2}$.

4.1.2 Different radii with mobility

In this scheme we recover the lost area by increasing the radii of one-hop neighbors of the failed node at $1+\varepsilon$ intervals while moving the nodes on the straight line toward the failed node. The scheme recovers the lost area while maintaining the coverage of the previously covered areas. Figure 8 illustrates the scheme. When the node (in thick dotted circle in the center) fails, its one-hop neighbors a, b, and c increase their radii while moving toward the failed node. The process stops when the thick dotted circle is completely covered by the union of the new disks of a, b, and c in thin dotted circles.

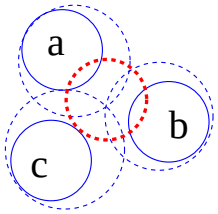


Figure 8: Different radii with mobility scheme.

For a node with radius r to cover a point d distance away from the node's disk, we can calculate the energy cost for the two schemes. For radius without mobility scheme, the energy cost $E_R = l \cdot (r + d)^2$. For radius with mobility scheme, the energy cost $E_M = l \cdot (r + \frac{d}{2})^2 + k \cdot \frac{d}{2}$. If $\frac{k}{l} > \frac{3}{2}d + 2r$, $E_M > E_R$. The node should simply increase its radius. If $\frac{k}{l} < \frac{3}{2}d + 2r$, $E_M < E_R$. The node should use an increased radius together with mobility.

5 Simulations

We simulated our algorithms in *ns-2*. The energy parameters we use are based on data presented in [6]. The transmitting power is set at 282 mW, the receiving power at 175 mW, the sensing power at 1.75 μW, and the idle power at 0 mW. We assume the migration energy to be 0.1 mJ/m. Each sensor is given the initial energy of $100J$. The lifetime of a node is modeled as an exponential distribution. The cumulative distribution function of an exponential distribution is $F(t) = 1 - e^{-\lambda t}$ for time t. For $F(t)$ we use a random number u from a uniform (0,1) distribution, which makes $u = 1 - e^{-\lambda t}$. Thus, $t = -\frac{log(1-u)}{\lambda}$. Since $(1-u)$ is another random number from uniform (0,1) distribution, we use $t = -\frac{log(u)}{\lambda}$ to model the lifetime of a sensor node.

We consider two types of network topology: grid and random distribution. For each topology, we deploy 100 sensors in a $350m$ x $350m$ field and 144 nodes in a $420m$ x $420m$ field. The initial communication radius of each node is $35m$. For our mobile redundant node scheme, we only test the random deployment since the initial configuration ensures that redundancy only occurs in random distribution topology. The energy cost for moving a redundant node is set to 0 J/m since our assumption is that the node has external source of energy for movement. To approximate the coverage of the whole field, we divide the field into $2m$ x 2 m cells and build a matrix with all the center points of the cells. The coverage of each node is modeled with sample points of 10-degree radial spacing and 10 points on each radius.

Each node keeps a timer for periodically broadcasting beacons to neighbors. We set the timer to be 10 seconds. At the end of each time period, messages are broadcast and node failures are calculated using the probability model. If a node fails and the failure is detected by neighboring nodes, the recovery scheme is applied by the neighbors. The simulation is stopped when only 5% of the nodes remain alive. Each simulation is run 7 times and the data presented are the average of the data collected.

Our simulation results are presented in Figure 9 and Figure 10. In Figure 9(a) and (b) we plot the coverage deterioration over time for network topologies of 100 node grid and 100 node randomly distributed. We compare the performance of two schemes, the different radii without mobility scheme (Radius only) and the different radii with mobility scheme (Radius w/ mobility). As a benchmark for worst-case coverage, we also simulate the scenario when no recovery scheme is employed and call it Static scheme. As expected, both the Radius only and the Radius with mobility schemes provide better coverage than the Static scheme. For both the grid and the random distribution

topology, the radii only scheme show better coverage percentage. This is because the energy consumption for moving is much greater than the energy cost for enlarging radii, i.e., the energy saved on a smaller radius is not enough to compensate the cost for the distance moved. The same is true in the 144 node scenarios, as shown in Figure 9(c) and (d). We also plot the coverage efficiency of the schemes by comparing the percentage of area covered by the node alive. In the grid and random distribution topologies for 100 nodes, shown in Figure 9(d) and (e), the Radius only scheme outperforms both the Radius with mobility and Static scheme. This is because by increasing the radius to a larger value the sensor covers some area previously not covered by any sensors thus increases the total coverage. The Radius with mobility scheme also increases the radius but to a smaller value than the Radius only scheme, therefore the total coverage increased is also smaller. This is true in the 144 node scenarios in Figure 9(g) and (h). Note also that the Radius only scheme outperforms the Radius with mobility scheme more in the random distribution topology than in the grid topology. This is because in the grid topology the area is completely covered initially. The performance of redundant node migration scheme is shown in Figure 10. For both 100 and 144 randomly distributed nodes, the scheme prolongs the coverage. We do not compare this scheme with the other two schemes, for the energy cost for mobility is zero in this scheme.

6 Related works

Meguerdichian et al. [4, 5] tackle the problems of finding the maximum breach path and the maximum support path in sensor networks. For the maximum breach path problem, they give an algorithm with complexity of $O(n^2 \log \Delta)$, where n is the number of sensors, and Δ is the difference between the highest and the lowest weight of an edge in the Voronoi Diagram of the sensor network. Their algorithm for the maximum support path has the time complexity same as the algorithm for the maximum breach path. Their algorithms are not only centralized but also heavily rely on geometric structures such as Voronoi Diagram and Delaunay triangulation of the network, which cannot be efficiently generated in a distributed manner.

Li et al. [3] prove the correctness of the algorithms in [4]. They also give a distributed algorithm for finding the best coverage distance and best coverage path in $O(n \log n)$ time with $O(n \log n)$ bit complexity. Their algorithm assumes only stationary sensors; no dynamic changes to the sensors and the locations of sensors are considered.

Huang et al. [2] discuss the problem of dynamically maintaining two measures of the quality of the coverage of a sensor network, the best-case coverage and worst-case coverage distances. They maintain a $(1+\varepsilon)$ approximation on the best-case coverage distance and a $(2+\varepsilon)$ approximation on the worst-case coverage distance of the network, for any fixed $\varepsilon > 0$. The algorithms have amortized or worst-case poly-logarithmic update costs. All their algorithms are centralized.

Zhang and Hou [9] prove that if the communication range of a sensor is at least twice its sensing range, a complete coverage of a convex area implies the connectivity among the working set of nodes. They derive the optimality conditions that a subset of working nodes is chosen for full coverage.

Huang and Tseng [1] give a $O(n^2 \log n)$ algorithm that determines if every point in a given area is covered by at least one sensor.

Wang et al. [8] present a Coverage Configuration Protocol that provides various degrees of connected coverage. They give a geometric analysis on the relation between connectivity and coverage.

7 Conclusions

We have proposed distributed algorithms to dynamically maintain two measures of quality of sensor coverage, the best coverage radius and the worst coverage radius. Using only local knowledge, our algorithms capture the dynamic changes of network topology and efficiently update the radii. Our algorithms are fully distributed with low communication cost. We have also proposed dynamic coverage maintenance schemes using only local knowledge. We have done performance evaluations of the schemes.

References

[1] C.F. Huang and Y.C. Tseng. The coverage problem in a wireless sensor networks. In *Proc. of the 2nd ACM Intl. Conf. on Wireless Sensor Networks and Applications*, pp. 115-121. 2003.

[2] H. Huang, A. W. Richa, and M. Segal. Dynamic coverage in ad-hoc sensor networks. *Mobile Networks and Applications* 10, pp. 9-17, Springer Science + Business Media. 2005.

[3] X.Y. Li, P.J. Wan, and O. Frieder. Coverage in wireless ad-hoc sensor networks, *IEEE Trans. Comput.* 52 (2003), pp. 1-11. 2003.

[4] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M.B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proc. of the 20th IEEE INFOCOM*, pp. 1380-1387, 2001.

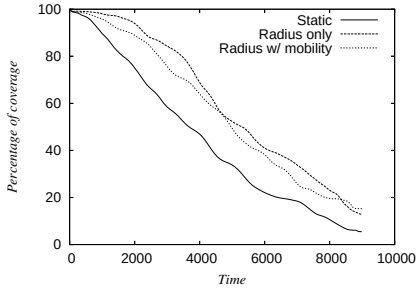
[5] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak. Exposure in wireless ad-hoc sensor networks. In *Proc. of the 7th ACM MOBICOM*, pp. 139-150. 2001.

[6] R. Min and A. Chandrakasan. Energy-Efficient Communication for Ad-Hoc Wireless Sensor Networks. *http://www.mit.edu/r̃min/research.* 2001.

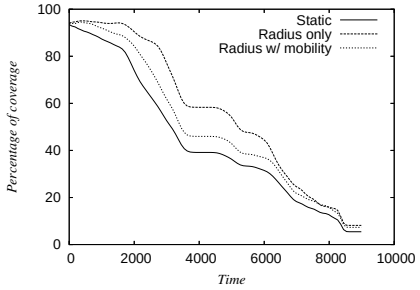
[7] A. Sekhar, B. S. Manoj, and C. S. R. Murthy. Dynamic Coverage Maintenance Algorithms for Sensor Networks with Limited Mobility. In *Proc. of the 3rd IEEE Intl. Conf. on Pervasive Computing and Communications (PerCom).* 2005.

[8] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C.D. Gill. Integrated coverage and connectivity configuration in wireless sensor networks. In *Proc. of the 1st ACM Conf. on Embedded Networked Sensor Systems.* 2003.

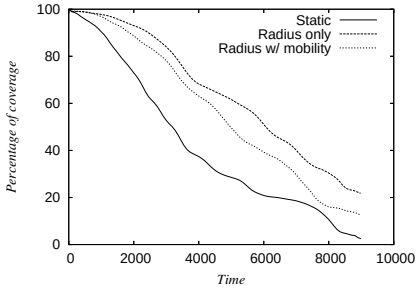
[9] H. Zhang and J. C. Hou. Maintaining sensing coverage and connectivity in large sensor networks. UIUCDCS-R-2003-2351. June 2003.



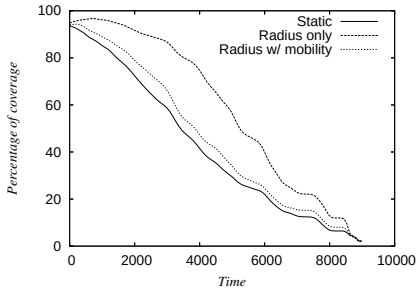
(a) Coverage vs. time: 100 node grid.



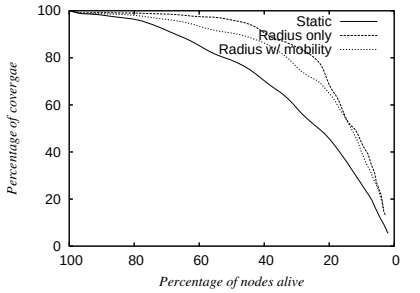
(b) Coverage vs. time: 100 nodes randomly distributed.



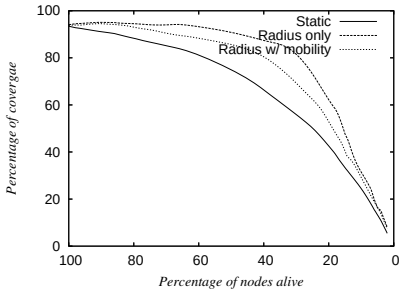
(c) Coverage vs. time: 144 node grid.



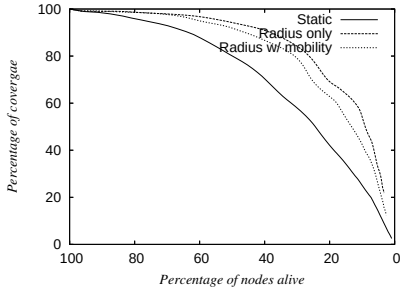
(d) Coverage vs. time: 144 nodes randomly distributed.



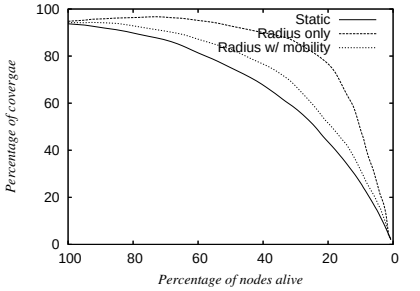
(e) Coverage efficiency: 100 node grid.



(f) Coverage efficiency: 100 nodes randomly distributed.



(g) Coverage efficiency: 144 node grid.



(h) Coverage efficiency: 144 nodes randomly distributed.

Figure 9: Simulation results for lazy coverage schemes.

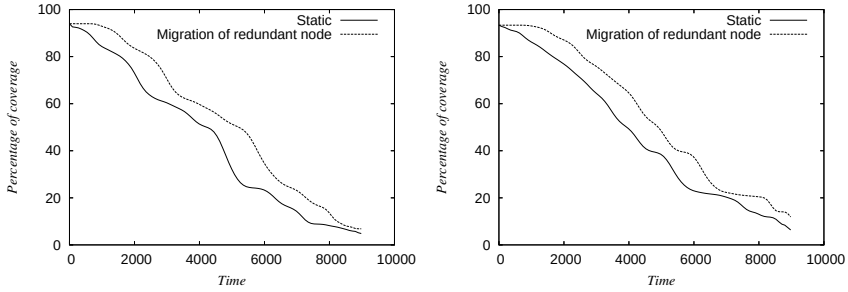


Figure 10: Simulation results for migration of redundant nodes. Left: 100 node. Right: 144 node.