Adaptive *k*-Coverage Contour Evaluation and Deployment in Wireless Sensor Networks

JANG-PING SHEU, National Tsing Hua University
GUEY-YUN CHANG, National Central University
SHAN-HUNG WU, National Tsing Hua University
YEN-TING CHEN, National Central University

The problem of coverage is a fundamental issue in wireless sensor networks. In this article, we consider two subproblems: *k*-coverage contour evaluation and *k*-coverage rate deployment. The former aims to evaluate, up to *k*, the coverage level of any location inside a monitored area, while the latter aims to determine the locations of a given set of sensors to guarantee the maximum increment of *k*-coverage rate when they are deployed into the area. For the *k*-coverage contour evaluation problem, a nonuniform-grid-based approach is proposed. We prove that the computation cost of our approach is at most the square root of existing solutions. Based on our *k*-coverage contour evaluation scheme, a greedy *k*-coverage rate deployment scheme (*k*-CRD) is proposed, which is shown to be an order faster than existing studies for *k*-coverage rate deployment. The *k*-CRD can incorporate two different heuristics to further reduce its running time. Simulation results show that *k*-CRD with these heuristics can be significantly more time efficient without causing much degradation in the coverage rate of final deployment.

Categories and Subject Descriptors: C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network monitoring*

General Terms: Measurement, Management, Algorithms

Additional Key Words and Phrases: Coverage problem, coverage contour, grid scan, wireless sensor networks

ACM Reference Format:
Sheu, J.-P., Chang, G.-Y., Wu, S.-H., and Chen, Y.-T. 2013. Adaptive *k*-coverage contour evaluation and deployment in wireless sensor networks. ACM Trans. Sensor Netw. 9, 4, Article 40 (July 2013), 31 pages.
DOI: http://dx.doi.org/10.1145/2489253.2489257

1. INTRODUCTION

Coverage is one of the most important issues in wireless sensor networks (WSNs), which is concerned with how well a specified area is monitored by sensors [Bai et al. 2006; Cardei and zhu Du 2005; Dhillon et al. 2002; Du and Lin 2005; Gallais et al. 2007; Heo and Varshney 2004; Huang and Tseng 2005; Krishnamachari and Iyengar 2004; Kumar et al. 2004, 2005; Liu et al. 2002; Lorincz and Welsh 2007; Megerian et al. 2005; Sanli and Cam 2005; Sheu and Lin 2007; Sun et al. 2005; Wan and Yi 2006; Zhang and Hou 2005a; Zou and Chakrabarty 2005]. Degree of coverage is often used as a measurement of the quality of service of WSNs. There are two critical subproblems in the coverage issue: coverage determination (or coverage evaluation) and deployment. The former aims to evaluate the degree of coverage, while the latter aims to determine the minimum number of sensors required and their locations to guarantee that the degree of coverage of monitored area meets application requirements.

Corresponding author email: J.-P. Sheu, sheujp@cs.nthu.edu.tw.

DOI: http://dx.doi.org/10.1145/2489253.2489257

Coverage level may be the most popular metric of degree of coverage. An area is said to have coverage level k, or k-covered, if every point in the area is within sensing radii of k distinct sensors. Due to the limited lifetime of sensors and the importance of monitored areas, many applications require $k > 1$ to tolerate sensor failure [Huang and Tseng 2005; Liu and Towsley 2005; Meguerdichian et al. 2002], as it may not be feasible to replace failure sensors, especially when sensors are deployed in harsh areas. In coverage evaluation, the k-coverage evaluation problem, which aims to evaluate whether the monitored area is k-covered, has been widely studied [Huang and Tseng 2005; Lin and Chiu 2005; Meguerdichian et al. 2002; Shakkottai et al. 2005; Zou and Chakrabarty 2003]. In deployment, numerous efforts have been made [Chakrabarty et al. 2002; Dhillon and Chakrabarty 2003; Kar and Banerjee 2003; Lin and Chiu 2005; Sun et al. 2005; Wang et al. 2003, 2005] for the k-coverage deployment problem, which aim to determine the minimum number of sensors required and their locations to guarantee that monitored areas are k-covered.

Most of the existing studies are based on the coverage level metric. While this metric gives a yes/no answer to the k-coverage of the whole monitored area, in many situations, it is desirable to know what percentage of the area is covered if the answer is negative. For example, an application may want to know the percentage of the monitored area that is not k-covered in order to evaluate the cost of redeployment. Shen et al. [2006] consider the *coverage rate* metric, that is, the ratio of the area achieving the coverage requirement of the target application (e.g., k-coverage) relative to the whole monitored area. A k-coverage rate evaluation scheme, called Grid Scan, is proposed based on the deployment/redeployment algorithm in Shen et al. [2006].

In this article, we focus on a more general *coverage contour* metric, which identifies the degree of coverage at any location inside the monitored area. Clearly, the k-coverage contour provides more detailed coverage information. By the aid of this metric, one is able to infer both the coverage level and coverage rate metrics easily. Additionally, there are many reasons why the coverage may change after initial deployment. These include node failure (due to, say, energy exhaustion, material fatigue, or other lifetime metrics [Dietrich and Dressler 2009; Zhang and Hou 2005b]), communication errors, environmental hazards, or deliberate attacks, etc. Knowing the coverage contour allows one to specifically identify where in the monitored sensor redeployment are required to cope with the preceding situations. We investigate two problems: k-coverage contour evaluation and k-coverage rate deployment. The former aims to determine the coverage level, up to k, at any location inside the monitored area, while the latter aims to determine the location of a given set of sensors to guarantee the maximum increment of k-coverage rate when they are deployed into a monitored area (either with or without pre-deployed sensors). The k-coverage rate deployment is particularly useful for small-scale redeployment, where specific placement of each sensor is possible.

One naive approach to supporting coverage contour is to extend Grid Scan [Shen et al. 2006]. The basic idea is to partition the monitored area into uniform-sized grids and then evaluate the coverage level of each grid. Given any location inside the monitored area, the coverage level of that location is answered by using the coverage level of the containing grid. The main drawback of Grid Scan is that the grid size is hard to decide. Too large a size will result in poor accuracy of contour evaluation, yet too small a size will incur high computation cost due to the huge number of grids. This drawback propagates to the k-coverage deployment scheme, as the center of each grid is examined to find the best location for each sensor.

This article makes two main contributions. First, for the k-coverage contour evaluation problem, a nonuniform-grid-based approach is proposed for areas with arbitrary sensor deployments. In our scheme, each grid can have a large initial size to save computation cost. A grid is further divided into subgrids only if the division benefits the accuracy of k-coverage contour evaluation. We prove that for the same level of accuracy, the computation cost of our approach is at most the square root of that of Grid Scan. Second, based on our k-coverage contour evaluation scheme, a greedy k-coverage rate deployment scheme, called k-CRD, is proposed. We prove that the k-CRD is an order faster than the Grid Scan-based deployment. This scheme also removes a restriction in Grid Scan, where the sensors are required to deploy at the center of grids. In addition, the k-CRD can incorporate two different heuristics to further reduce the computation cost. Simulation results show that both these heuristics can significantly reduce the running time of k-CRD without causing much degradation in the coverage rate of final deployment.

The rest of this article is organized as follows. Section 2 reviews previous works and gives necessary assumptions and notations. A k-coverage contour evaluation scheme is proposed in Section 3. We elaborate the greedy k-coverage rate deployment scheme in Sections 4 and 5. Simulation results are given in Section 6. Finally, we conclude with remarks in Section 7.

2. PRELIMINARIES

In this section, we review existing works on the coverage problem in sensor networks. Some important assumptions and notations are introduced as well that will be used throughout this article.

2.1. Related Work

Coverage problems have attracted lots of attentions during the past few years. For coverage evaluation, the k-coverage evaluation problem, which aims to evaluate whether the monitored area is k-covered, has been widely studied. For example, Zou and Chakrabarty [2003] introduce a virtual force model for coverage evaluation. Shakkottai et al. [2005] propose a one-coverage evaluation scheme for grid deployments. Lin and Chiu [2005] discuss three-coverage evaluation for the case of strip-shaped sensor fields. Huang and Tseng [2005] proposed a distributed k-coverage evaluation protocol. Meguerdichian et al. [2002] investigated a Voronoi diagram-based, worst-case k-coverage evaluation.

For deployment, early efforts have been made [Dhillon and Chakrabarty 2003; Kar and Banerjee 2003; Lin and Chiu 2005; Wang et al. 2003, 2005] for one-coverage deployment. Xingyu and Hongyi [2006] formulate the redeployment problem as a 0-1 programming model and propose some heuristics to solve the problem. Yang and Cardei [2007] address the redeployment problem in a network where sensor nodes are movable and improve the coverage through relocating sensor nodes. Unlike this study, where the coverage is improved by changing the initial deployment so that the problems of deployment and redeployment are correlated, we assume that the sensor nodes are static. The coverage is improved by deploying new sensor nodes to the monitored area without affecting the coverage of the existing nodes. Therefore, our redeployment schemes are compatible with any initial deployment algorithm. There are other works [Bartolini et al. 2009, 2012; Zhou et al. 2009] which focus on achieving a desirable coverage level given that sensor nodes can have different sensing radii. Bartolini et al. [2009] introduced a mobile, heterogeneous, sensor-deployment algorithm, called Vorlag. Vorlag is a generalization of the Voronoi approach based on Laguerre geometry. In addition, an extended version of Vorlag is introduced for environments which require varying coverage density. Bartolini et al. [2012] proposed a Voronoi Laguerre-based method for the joint problem of dynamically scheduling the activation of different subsets of sensor nodes and of tuning their sensing radii (if their technology allows) for prolonging the network lifetime while ensuring the maximum achievable coverage. Zhou et al. [2009] address the problem of selecting a minimum energy-cost connected sensor cover, when each sensor node can vary its sensing and transmission radii. They design a centralized algorithm which performs within an $O(\log n)$ factor of the optimal solution—where n is the size of the network—and a distributed algorithm based on Voronoi diagrams. All these works achieve one-coverage only, and it remains unclear how to perform the sensor redeployement with the Voronoi Laguerre metric.

Recently, importance is attached to the study of multilevel coverage deployment [Gallais et al. 2007; Wang and Tseng 2008; Zhou et al. 2005]. Sun et al. [2005] proposed a k-coverage hexagon-like deployments scheme. Chakrabarty et al. [2002] considered a k-coverage grid deployment scheme for sensors with inhomogeneous sensing capabilities. Most of these studies are based on the coverage level metric, giving only a yes/no answer. To give more detailed information about the monitored area, Shen et al. [2006] consider the coverage rate metric. Grid Scan is proposed for k-coverage rate evaluation, which aims to determine the ratio of the k-covered area relative to the whole monitored area.

Kasbekar et al. [2011] aim to maximize coverage and network lifetime simultaneously. The network lifetime is an important metric, as many parameters can be reduced to the lifetime consideration. These parameters include application characteristics (e.g., distribution of tasks and destination of data packets), quality of service (e.g., coverage and connectivity), heterogeneity (e.g., various sensor radii and various processing power), and node mobility, etc. [Dietrich and Dressler 2009; Zhang and Hou 2005b]. Kasbekar et al. [2011] design a polynomial-time distributed algorithm for guaranteeing coverage and maximizing the lifetime of the network, and prove that its lifetime is at most a factor $O(\log n \log nB)$ lower than the maximum possible lifetime, where n is the number of sensors and B is an upper bound of the initial energy of each sensor. In this article, we focus on the problem of ensuring k-coverage of the monitored area rather than on extending the lifetime of the network or individual sensor nodes. There are many reasons why the coverage may change after initial deployment, such as the lifetime metrics previously stated, node failure (due to, say, material fatigue), communication errors, environmental hazards, or deliberate attacks, etc. The network or nodes may still be alive, but some regions are not observable anymore. Ensuring k-coverage allows the data to be collectable (or the service to be available) continuously in these situations.

The Coverage Contour Metric. Despite the coverage level and coverage rate metrics, there are few studies (if any) focusing on a more detailed coverage contour metric that identifies the degree of coverage at any position inside the monitored area. With the aid of this metric, the sensor redeployment cost can be significantly lowered (as we will discuss in Section 4), since one can know the exact regions in the monitored area that are not k-covered and can, therefore, redeploy sensors economically to these regions only.

One naive approach to supporting the coverage contour is to extend the Grid Scan [Shen et al. 2006]. The basic idea is to partition the monitored area into uniform-sized grids, as shown in Figure 1(a), and then evaluate the coverage level of each grid. A grid is evaluated as k-covered if the grid's center is k-covered. Given any location inside the monitored area, the coverage level of that location is answered by using the coverage level of the containing grid. For the k-coverage rate deployment problem, a sensor can be deployed to the grid center at which the deployment of a sensor most increases the number of k-covered grids. This grid center can be found using the coverage contour information. After adding a sensor to that location and updating the contour information, one can deploy the next sensor by repeating the preceding steps until all given sensors' locations are determined.

However, the grid size is hard to decide in this native approach. The schemes can have either low accuracy, if the grid size is too large, or high computation cost, if the grid size is too small.

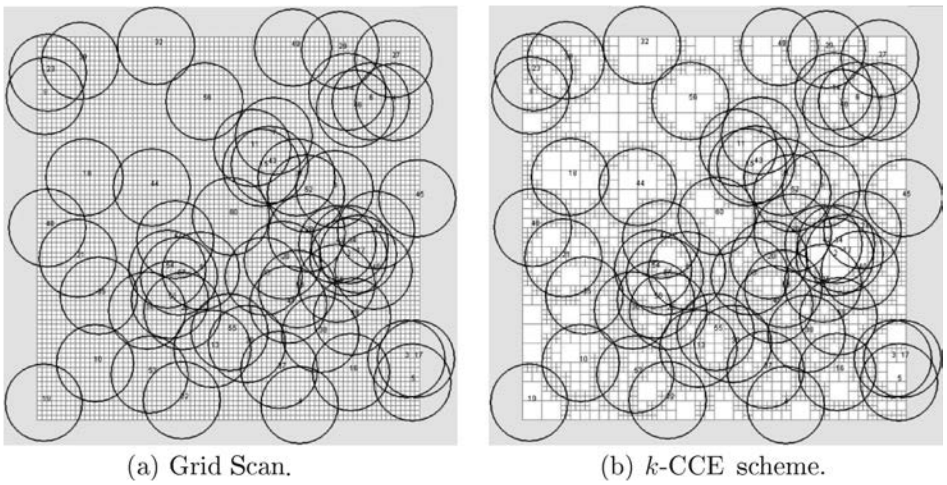

Fig. 1. Grid division for $k = 4$.

2.2. Target Environments

In this article, sensors are assumed to have ideal sensing ranges (i.e., circles) and the same sensing radii (denoted by r) and to be static after deployment [Wang et al. 2003; Wang and Tseng 2008; Zou and Chakrabarty 2003; Zhou et al. 2005]. Besides, sensors are aware of their own locations through either the global positioning system (GPS) [Hofmann-Wellenhof et al. 1997] or other localization techniques [Bulusu et al. 2000; Hu and Evans 2004]. The position error depends on the used localization techniques. Sensors report their location information to the base station. With the aid of sensor location information, the base station performs the k-coverage contour evaluation. We also assume that the communication radius of a node is at least twice that of the sensing radius; hence the network is connected as long as its coverage is guaranteed. If this is not true, then relay nodes [Cheng et al. 2008; Hou et al. 2005] that forward data between sensing nodes are required to ensure network connectivity.

3. *K*-COVERAGE CONTOUR EVALUATION SCHEME (*K*-CCE)

This section is devoted to evaluating the k-coverage contour of a monitored area. Our main idea is to divide the monitored area into nonuniform-sized grids, as depicted in Figure 1(b). Unlike the Grid Scan (Figure 1(a)), where the grid size needs to be determined in advance, we start from coarse grids and divide a grid into subgrids only if the division benefits the coverage contour evaluation. The division is performed recursively to a subgrid until the evaluation error is less than a user tolerance. Since the fine-granularity grids are generated only in an on-demand manner, the number of grids considered in our contour evaluation can be much smaller than that in the Grid Scan, thereby achieving high accuracy without incurring high computation cost.

3.1. Grid Division

We give the following definitions first. An area A' is said to be *fully covered* by a sensor s if each point in A' is covered by s. A' is said to be *partially covered* by s if some points in A' are covered by s and some are not. If A' is not fully covered or partially covered by any sensor, then A' is *uncovered*. For simplicity, an area fully covered by exactly n distinct sensors is called to be *exactly n-fully covered* (*n-fully covered* for short), and an area partially covered by exactly n distinct sensors is called to be *exactly n-partially covered* (*n-partially covered* for short). An illustrative example is shown in Figure 2.

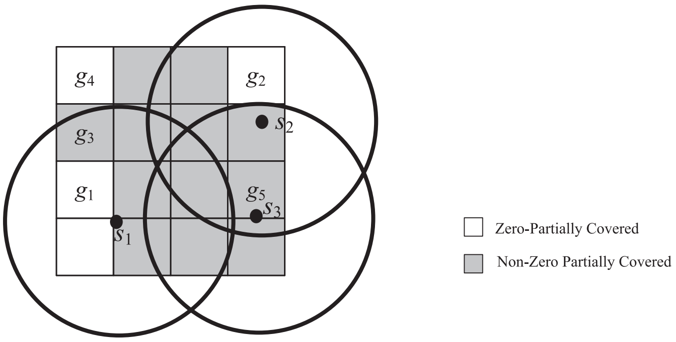

Fig. 2. An example of fully covered, partially covered, and uncovered grids. Each grid has side length $r/2$.

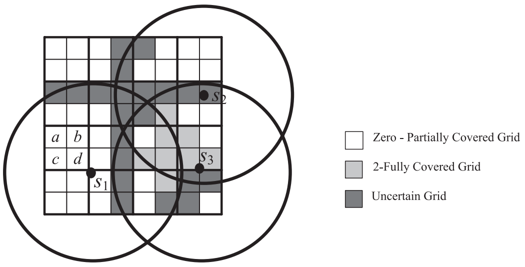

Fig. 3. An example of grids with side length $r/4$.

Three circles denote the sensing ranges of three sensors s_1, s_2, and s_3. The grid g_5 is two-fully covered by sensors s_2 and s_3 and one-partially covered by sensor $s1$.

Our k-coverage contour evaluation algorithm is proposed based on the insights into when a grid should be divided and when the division should be terminated, as explained next.

3.1.1. On-Demand Division. When a grid, say g, is partially covered by s, evaluating what percentage of g is covered by s requires complex computation. The matter gets worse as grids are partially covered by more than one sensor. Instead of applying complex computation, we can divide the grid into subgrids to obtain more precise coverage information. However, two kinds of grid division bring no benefits to coverage contour evaluation.

—Dividing zero-partially covered grids.
—Dividing k-fully covered grids.

Consider grid g_1 in Figure 2. Grid g_1 is one-fully covered and zero-partially covered. When g_1 is further divided into four subgrids a, b, c, and d, as shown in Figure 3, no further information about coverage contour is obtained. In fact, no matter how many subgrids g_1 is divided into, the situation cannot be improved. Dividing k-fully covered grids also gives zero information gain, as we already have satisfactory information for k-coverage contour evaluation.

Hence, division shall be performed on those grids which are partially covered by at least one sensor and fully covered by less than k distinct sensors. Since the coverage situations of these grids are uncertain, we call them *uncertain grids* in the rest of this article. Referring to Figure 2 again, assume two-coverage contour evaluation problem is considered, that is, $k = 2$. Only uncertain grids are divided into subgrids, as shown in Figure 4.

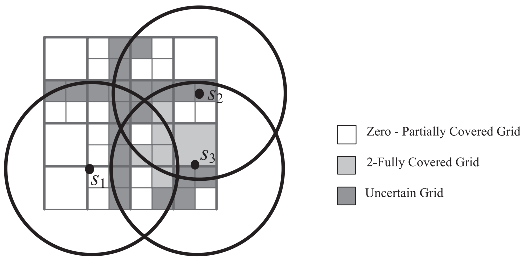

Fig. 4. An example of nonuniform-sized grids.

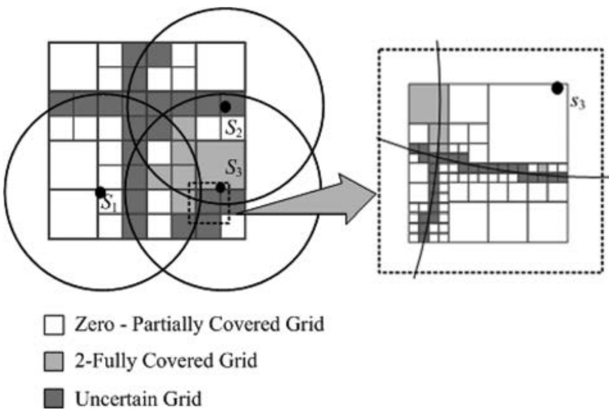

Fig. 5. An example of grid division.

Definition 3.1 (*Uncertain Grid*). A grid g is called an uncertain grid for k-coverage contour evaluation if the following two conditions hold: (1) g is partially covered by some sensors; (2) g is fully covered by less than k distinct sensors.

After an uncertain grid is divided into subgrids, for the purpose of k-coverage contour evaluation, we try to acquire coverage information of each subgrid, that is, how many sensors fully cover or partially cover it. This coverage information can be efficiently acquired, and the details are described in Section 3.2.

3.1.2. Terminating Grid Division. Next, we propose the criterion for terminating the gird division. Sometimes grid division is of little worth, even if more coverage information is obtained from the new generated subgrids. Figure 5 illustrates grid division for two-coverage contour evaluation. In order to acquire more coverage information of the grid in the dotted rectangle, this grid is further divided into many subgrids, as shown in the right side of Figure 5. However, some subgrids on the right side are still uncertain grids. Even if no uncertain grids are on the right side of Figure 5, the contribution of coverage information of these subgrids to coverage contour evaluation is limited because these subgrids are so tiny. Based on this fact, we introduce a criterion for terminating division. Let the *maximum evaluation error* (MEE) be the ratio of the uncertainly covered area relative to the whole monitored area, that is, $MEE = \sum_{g \in U} |g|/|A|$, where U denotes the set of uncertain grids, and $|g|$ and $|A|$ denote the area size of g and A, respectively. In essence, the MEE denotes the probability that the coverage of a point, which is uniformly drawn from the monitored area, cannot be correctly evaluated. We also define the *maximum tolerable evaluation error* (MTEE) to be the maximum evaluation error that is permitted for a target application.

ALGORITHM 1: The Simplified k-Coverage Contour Evaluation (k-CCE) Scheme

```
Global Variables:
  typedef struct
    grid_id num_fully_covered num_partially_covered;   // number of sensors partially
    covering this grid
  as Grid;
  r; // sensing radius
  m; // initial dividing factor, must be positive
  U; // a set of uncertain grids
  K; // a set of k-covered grids
  C; // a map from grid_ids to the generated grids
1  Create an initial grid enclosing the monitored area and put it into U ; // an uncertain
   grid
2  repeat
3      foreach grid x in U whose side length is larger than $r/m$ do
4          pop x from U Divide x into sub-grids of side length $r/m$ and put them into
           C foreach sub-grid g of x do
5              Evaluate num_fully_covered and num_partially_covered for g if g is k-fully
               covered then
6                  put g into K
7              else if g is not zero-partially covered then
8                  put g into U
9              end
10         end
11     end
12     Calculate maximum evaluation error $m = m \times 2$
13 until maximum evaluation error < maximum tolerable evaluation error;
14 return C and K
```

Note that although the MEE is a relative measure to the monitored area, one can obtain comparable absolute error (i.e., the total area of uncertain grids) for networks of different sizes by issuing a smaller MTEE for a larger network. For example, given two networks of sizes 10×10 and 100×100 m^2, we can issue inverse-proportional MTEEs x and $0.01x$, $0 < x < 1$, for these networks, respectively, to expect the same total area of uncertain grids at termination.

3.2. Acquiring Coverage Information

The k-Coverage Contour Evaluation (k-CCE) scheme is outlined in Algorithm 1. Note the steps are simplified for the sake of intuitiveness. More detailed steps can be found in Algorithms 6 and 7 in the Appendix.

Denote r as the sensing range of a node and m as the *dividing factor*, a positive integer given by the user that controls the side length of a grid. Basically, the scheme divides the monitored area into grids of side length r/m and calculates the coverage information of each grid, that is, the numbers of sensors that fully and partially cover this grid, which are stored in the *num_fully_covered* and *num_partially_covered* fields, respectively, in a grid **struct**. For grids that are neither k-fully covered nor zero-partially covered, the division helps, and these grids are kept in a set U. The scheme then doubles the dividing factor m and divides all the grids in U into subgrids of side length r/m. The coverage information for each subgrid is evaluated, and those subgrids that are neither k-fully covered nor zero-partially covered are kept in U again. This process repeats until the current division results in satisfactory MEE.

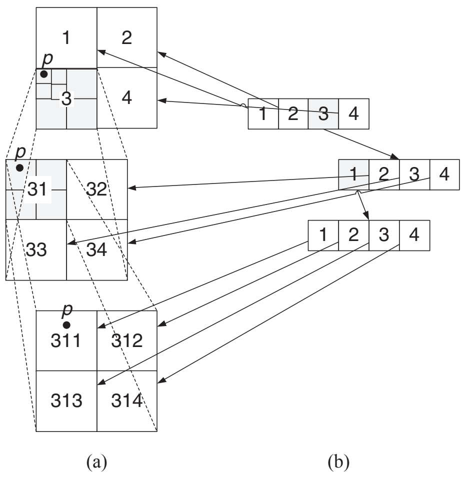

Fig. 6. (a) The grid IDs. (b) The structure of C.

The k-CCE scheme returns two sets, C and K, of grids. The set C is a map from grid IDs to the grid **struct**s containing the coverage information, and K is a set of all k-fully covered grids. By the aid of K, the k-coverage rate of the whole monitored area can be evaluated as follows.

$$k\text{-coveragerate} \simeq \sum_{g \in K} |g| / |A|.$$

This allows the k-CRD algorithms (to be described in Section 4) to check if more sensors should be deployed.

In addition to the k-coverage rate of the entire monitored area, the k-CCE scheme can answer the coverage contour, which gives coverage information at any specific point by the aid of C. Note that the ID of a grid is geo-encoded, as depicted in Figure 6(a). At the top, the monitored area is divided into grids 1, 2, 3, and 4. At the middle, the subgrids of grid 3, which are generated during the second iteration of the k-CCE scheme, have IDs 31, 32, 33, and 34, respectively. At the bottom, the subgrids of grid 31 generated in the third iteration have IDs 311, 312, 313, and 314, respectively. Other grids obtain their IDs similarly. The structure of C, as shown in Figure 6(b), is an ordered prefix tree (known as *a trie*) keyed on grid IDs, where each internal node points to either the grid **struct**s (i.e., leaf nodes) if the grids have no subgrids or otherwise to other internal nodes having a common prefix associated with this node. Given any point p inside the monitored area (Figure 6(a)), one can easily derive the ID of p's containing grid at a specific iteration. The coverage at p can be evaluated by

$$\text{Coverage of } p = g'.num\_fully\_covered,$$

where the grid **struct** g' can be looked up by traversing the tree C down from the root and determining the ID of the grid covering p at each level, until reaching the leaf.

Next, we describe the details of checking whether a grid is fully covered, partially covered, or uncovered by a sensor.

Fully-Covered Grids. A grid is fully covered by a sensor if four corners of the grid are covered by the sensor. In other words, the grid is fully covered by the sensor if the distance between each corner of the grid and sensor is within the sensing radius of the sensor. In Figure 7(a), the grid g is fully covered by sensor s.

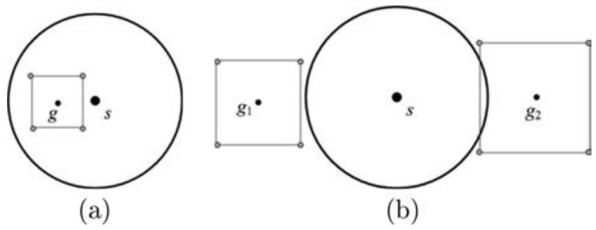

Fig. 7. Fully covered and uncovered grids. (a) Grid g is fully covered by s; (b) grids g_1 and g_2 are uncovered and partially covered by s, respectively.

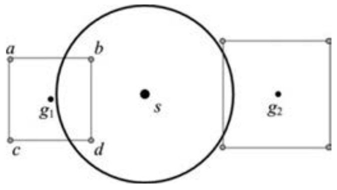

Fig. 8. Both grids g_1 and g_2 are partially covered by s.

Uncovered Grids. A grid is uncovered if (1) for each sensor, the distance from each corner of the grid to the sensor is larger than the sensing radius of the sensor; and (2) the distance from the center of the grid to the sensor is larger than the summation of the sensing radius of the sensor and half the side length of the grid. An illustrative example is shown in Figure 7(b). There, the four corners of grid g_1 are not covered by sensor s, and grid g_1 satisfies the second condition just mentioned, so grid g_1 is uncovered. On the other hand, grid g_2 is not uncovered because grid g_2 does not satisfy the second condition.

Partially-Covered Grids. According to the preceding discussion, a grid is partially covered by a sensor if it is neither fully covered nor uncovered by sensors. Formally, a grid is partially covered by a sensor if one of the two following conditions is satisfied: (1) Some corners of the grid are covered by the sensor and some are not, that is, there exist two corners, c_1 and c_2, of the grid such that the distance between c_1 (c_2) and the sensor is larger (smaller) than the sensing radius of the sensor. In Figure 8, grid g_1 is partially covered by sensor s because corner b and corner d are covered by sensor s and corner a and corner c are not covered by the sensor, s. (2) The four corners are not covered by the sensor, and the distance between the center of the grid and this sensor is smaller than the summation of the sensing radius of the sensor and half the side length of the grid. Consider grid g_2 in Figure 8, g_2 is partially covered by sensor s.

3.3. Complexity Analysis

In the following, we derive an upper bound of grids generated in the k-CCE scheme. The following lemma shows an upper bound on the number of grids partially covered by a single sensor.

LEMMA 3.2. *Given a dividing factor m' such that each grid has side length r/m', there are less than $8m' + 8$ grids partially covered by a single sensor.*

PROOF. Consider a sensor s which is deployed in the monitored area. Recall that the sensing range of each sensor is assumed to be a circle. Without loss of generality, assume the position of s is at the origin of the Euclidean plane. Then, each point (x, y) inside the sensing range of s should satisfy $x^2 + y^2 \leq r^2$. First, we prove that there are at most $2m' + 2$ grids partially covered by s in the first quadrant. Obviously, each

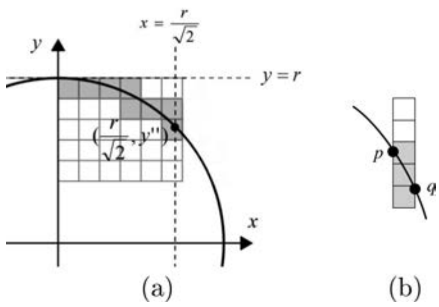

Fig. 9. Gray grids intersect the segment of the periphery of the sensing range of s in $0 \leq x \leq r/\sqrt{2}$.

grid g is partially covered by s if and only if the periphery of the sensing range of s intersects grid g. For example, in Figure 9(a), gray grids are partially covered by s, and each of them intersects the periphery of the sensing range of s. For ease of the following discussion, we consider two subsegments of the periphery of the sensing range of s—segment of periphery in $0 \leq x \leq r/\sqrt{2}$ and segment of periphery in $r/\sqrt{2} < x \leq r$—in the first quadrant.

Case 1. For the segment of periphery in $0 \leq x \leq r/\sqrt{2}$, each point (x, y) on this segment has that $r/\sqrt{2} \leq y < r$ and $-1 \leq \partial y/\partial x \leq 0$. We first claim that for each column c intersecting this periphery segment, there are at most two grids in c intersecting the segment, as shown in Figure 9(a). Otherwise, there exist two points, $p = (x_p, y_p)$ and $q = (x_q, y_q)$, on this periphery segment, as shown in Figure 9(b), such that they are on column c but on distinct nonadjacent rows, which implies that the slope of line segment $\overline{pq}$ is smaller than -1, that is, $y_q - y_p/x_q - x_p < -1$, a contradiction to $-1 \leq \partial y/\partial x \leq 0$. Next, we claim that there are at most $\lceil m' - m'/\sqrt{2} \rceil$ columns, each of which has two grids intersecting the segment. Suppose conversely that there are at least $\lceil m' - m'/\sqrt{2} \rceil + 1$ such columns. Since $-1 \leq \partial y/\partial x \leq 0$ implies that the segment is monotonically decreasing, for every two columns c_1 and c_2, there exists at most one row r' such that both grids at (r', c_1) and (r', c_2) intersect the segment. In other words, for two columns, each of which has two grids intersecting the segment, their grids intersecting with the segment are spread on three or more rows. For example, in Figure 9(a), there are two columns, each of which has two grids intersecting the segment, and there are three rows intersecting the segment. So, there are at least $\lceil m' - m'/\sqrt{2} \rceil + 2$ rows intersecting the segment. However, this implies that the far-right point $(r/\sqrt{2}, y'')$ on this segment satisfying $y'' \leq r - [(\lceil m' - m'/\sqrt{2} \rceil + 2) - 2] \cdot (r/m') \leq r/\sqrt{2}$, where r/m' is the side length of a grid, contradicts $r/\sqrt{2} \leq y < r$.

Notice that there are at most $\lceil \frac{r/\sqrt{2}}{r/m'} \rceil = \lceil m'/\sqrt{2} \rceil$ columns for $0 \leq x \leq r/\sqrt{2}$. So there are at least $\lceil m'/\sqrt{2} \rceil - \lceil m' - m'/\sqrt{2} \rceil$ columns, each of which has exactly one grid intersecting the segment. Then, there are at most $\lceil m' - m'/\sqrt{2} \rceil \cdot 2 + (\lceil m'/\sqrt{2} \rceil - \lceil m' - m'/\sqrt{2} \rceil) \cdot 1 = m' - \lfloor m'/\sqrt{2} \rfloor + \lceil m'/\sqrt{2} \rceil$ grids intersecting the segment, that is, there are at most $m' - \lfloor m'/\sqrt{2} \rfloor + \lceil m'/\sqrt{2} \rceil$ partially covered grids for $0 \leq x \leq r/\sqrt{2}$.

Case 2. For the segment of periphery in $r/\sqrt{2} < x \leq r$, that is, $0 \leq y < r/\sqrt{2}$, each point (x, y) on this segment has $-1 < \partial y/\partial x < 0$. Symmetrically, there are at most $\lceil m' - m'/\sqrt{2} \rceil$ rows, each of which has two grids intersecting the segment, and there are at most $\lceil \frac{r/\sqrt{2}}{r/m'} \rceil = \lceil m'/\sqrt{2} \rceil$ rows for $0 \leq y < r/\sqrt{2}$. So, we have that there are at most $m' - \lfloor m'/\sqrt{2} \rfloor + \lceil m'/\sqrt{2} \rceil$ grids intersecting the segment, that is, there are at most $m' - \lfloor m'/\sqrt{2} \rfloor + \lceil m'/\sqrt{2} \rceil$ partially covered grids for $0 \leq y < r/\sqrt{2}$.

Summarizing the preceding discussion, there are at most $(m' - \lfloor m'/\sqrt{2} \rfloor + \lceil m'/\sqrt{2} \rceil) \cdot 2 \leq 2m' + 2$ grids partially covered by s in the first quadrant. Notice that when $\lfloor m'/\sqrt{2} \rfloor \neq \lceil m'/\sqrt{2} \rceil$, there are some partially covered grids double-counted in these two cases. So, there are less than $2m' + 2$ grids partially covered by s in the first quadrant. Taking the four quadrants into account, there are less than $8m' + 8$ grids partially covered by s. $\square$

We next derive an upper bound on the number of uncertain grids generated at the ith iteration. Suppose that the monitored area A is divided into equal grids with side length r/m at the first iteration and there are n_s sensors deployed in monitored area A. Then there are a total of $|A|m^2/r^2$ grids at the first iteration. It is not difficult to see that when all grids are further divided into four subgrids, there are $|A|(2m)^2/r^2$ grids. Let $N_i = |A|(2^{i-1}m)^2/r^2$ be the number of grids after repeating the process of dividing all grids into four subgrids $i-1$ times.

LEMMA 3.3. *There are less than* $\min\{(2^{2+i}m + 8) \cdot n_s, N_i\}$ *uncertain grids at the ith iteration.*

PROOF. Initially, since there are n_s sensors deployed in the monitored area and each grid has side length r/m, by Lemma 3.2, there are less than $\min\{(8m + 8) \cdot n_s, N_1\}$ partially covered grids. Recall that uncertain grids are partially covered grids. When a new iteration is executed, each uncertain grid is divided into four subgrids, that is, the side length of each subgrid is half the side length of the original grid. Again by Lemma 3.2, we have that there are less than $\min\{(16m + 8) \cdot n_s, N_2\}$ uncertain grids at the second iteration. Then, it is not difficult to check that there are less than $\min\{(2^{2+i}m + 8) \cdot n_s, N_i\}$ uncertain grids at the ith iteration. $\square$

Without loss of generality, assume that maximum evaluation error is lower than or equal to maximum tolerable evaluation error after the ith iteration. Then we have the following theorem.

THEOREM 3.4. *There are a total of* $O(2^i)$ *grids generated from the first iteration to the ith iteration in our k-coverage contour evaluation scheme.*

PROOF. Recall that each uncertain grid is further divided into four subgrids. By Lemma 3.3, there are less than $\min\{(2^{2+i-1}m + 8) \cdot n_s, N_{i-1}\}$ uncertain grids at the $(i-1)$th iteration. Hence, there are less than $\min\{(2^{2+i-1}m + 8) \cdot n_s, N_{i-1}\} \cdot 4 = \min\{(2^{3+i}m + 2^5) \cdot n_s, N_i\}$ new grids generated at the ith iteration. The total number of grids generated from the first iteration to the ith iteration is less than $|A|m^2/r^2 + \sum_{1 \leq j \leq i} \min\{(2^{3+j}m + 2^5) \cdot n_s, N_i\} = \min\{2^{i+4}mn_s - 2^5mn_s + 2^5(i-1)n_s + |A|m^2/r^2, (|A|m^2/r^2) \cdot (2^{2i} - 1/3)\} = O(2^i)$. $\square$

It is important to note that although the k-CCE has exponential complexity in terms of i, our simulations show that the k-CCE runs in polynomial time in practice. This is because, given an MTEE, the number of iterations i is proportional to $O(\lg(1/MTEE))$ only, as the MEE is halved during each iteration. Suppose, at the ith iteration, we have $MEE = \sum_{g \in U} |g|/|A|$, where U denotes the set of uncertain grids, and $|g|$ and $|A|$ denote the area size of an uncertain grid and the monitored area, respectively. We can see from Lemma 3.2 that $MEE \leq n_s(8m' + 8)|g|/|A|$, where m' is the dividing factor at the ith iteration. At the $(i + 1)$th iteration, the dividing factor m' is doubled, but the area $|g|$ becomes one-fourth of the previous one. The MEE at the $(i + 1)$th iteration is less than half of the MEE at the ith iteration. This implies that given any MTEE, the k-CCE algorithm requires at most $O(\lg(1/MTEE))$ iterations to obtain a satisfactory result. Therefore, the actual running time is polynomial. This leads to the following theorem.

THEOREM 3.5. *Given an MTEE, the computation cost of the k-coverage contour evaluation scheme is $O(1/MTEE)$.*

PROOF. We consider the following two cases separately: the first iteration and the remaining iterations. Recall that at the first iteration, each sensor is regarded as having the possibility of partially covering the uncertain grid, that is, the whole monitored area (steps 2 to 6 of Algorithm 6 in the Appendix). Since there are $|A|m^2/r^2$ subgrids (because each subgrid has side length r/m) at the first iteration (Algorithm 6, step 1) and each subgrid should check which sensors fully/partially cover it (Algorithm 7, steps 8 to 15), the computation cost of the first iteration is $O(n_s \cdot |A|m^2/r^2)$.

At remaining iterations, a sensor is regarded as having the possibility of partially covering a subgrid of grid g if it partially covers grid g, that is, sensors which fully cover grid g should fully cover subgrids of g. Referring to Algorithm 7 in the Appendix, sensors which partially cover an uncertain grid $U[*].grid\_id$ are maintained in $U[*].partially\_covered$. Since each uncertain grid $U[*].grid\_id$ is dividing into four subgrids and each subgrid of $U[*].grid\_id$ should check whether it is partially/fully covered by sensors maintained in $U[*].partially\_covered$, the computation cost of iteration i, $i > 1$, is $O(4n_p)$, where $n_p = \sum_{1 \le j \le \alpha} U[j].num\_partially\_covered$ denotes the maximum number of uncertain grids which are partially covered by the same sensor, $U[j].num\_partially\_covered$ is the number of sensors partially covering the grid j, and α is the number of uncertain grids at iteration $i-1$. By Lemma 3.2, we have $n_p \le 8m' + 8$ (i.e., there are less than $8m' + 8$ grids partially covered by a single sensor) when the side length of each grid is r/m'. Since the smallest grids at iteration $i-1$ have a side length of $r/(2^{i-2}m)$, we have that $n_p \le (8 \cdot 2^{i-2}m + 8) \cdot n_s = (2^{i+1}m + 8) \cdot n_s$, implying that the total computation cost of the remaining iterations is $O(\sum_{2 \le j \le i} 4 \cdot (2^{j+1}m + 8) \cdot n_s) = O(2^i)$. Taking into account that $n = O(\lg(1/MTEE))$, the computation cost of k-CCE is $O(2^{\lg(1/MTEE)}) = O(1/MTEE)$. □

On the other hand, consider the Grid Scan scheme proposed in Shen et al. [2006]. In Grid Scan, the monitored area is divided into uniform-sized grids. If grids in Grid Scan have side lengths equal to the smallest grids' side lengths at the nth iteration of k-CCE (i.e., $r/2^{n-1}m$), then there are a total of $N_n = |A|(2^{n-1}m)^2/r^2 = \theta(2^{2n})$ grids in Grid Scan. For each sensor, $\frac{\pi r^2}{r/2^{n-1}m} = 2^{2n-2}\pi m^2$ grids fall within the sensor's sensing range. So the computation cost of Grid Scan is $\theta(n_s \cdot 2^{2n-2}\pi m^2) = \theta(2^{2n}) = \theta(1/MTEE^2)$. In summary, the computation cost of k-CCE is at most the square root of that of Grid Scan.

4. *K*-COVERAGE RATE DEPLOYMENT SCHEME (*K*-CRD)

In addition to evaluating the k-coverage contour of a monitored area, we propose the k-coverage rate deployment scheme to improve the k-coverage rate if the required k-coverage rate is not achieved and there are additional sensors available for deployment. The basic idea of our scheme is to deploy sensors to locations that increase the total area of k-fully covered grids most economically (in terms of the number of sensors used). Our scheme works iteratively, that is, one location for deployment is determined at each iteration, so it accepts any number of available sensors.

4.1. Deployment for Fully Covering a Grid

Given a grid g, we define a *deployment region* with respect to g, denoted by $DR(g)$, as an area within which a deployed sensor can fully cover g. An example of a deployment region with respect to g is shown in Figure 10. The hollow circles denote the sensing

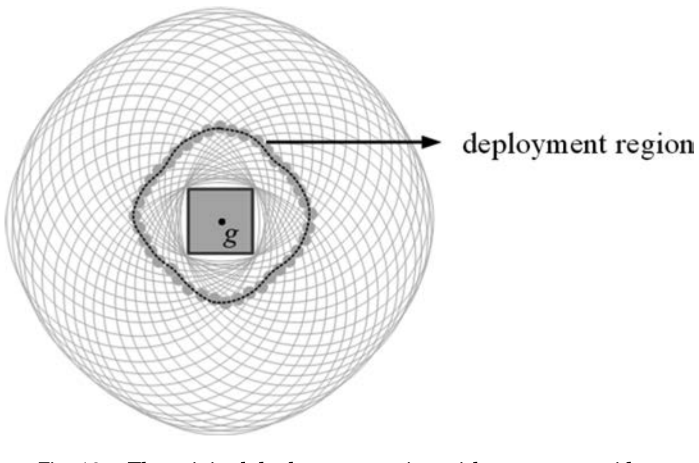

Fig. 10. The original deployment region with respect to grid g.

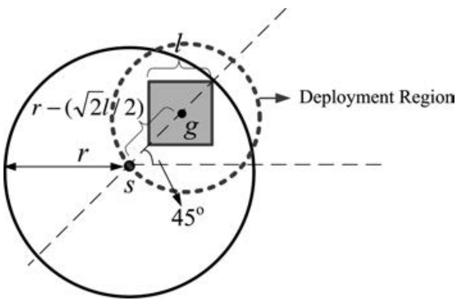

Fig. 11. The dashed circle is a simplified deployment region with respect to grid g.

ranges of s when s is deployed at the solid gray circles. The region enclosed by a dashed line denotes $DR(g)$. We can see that the shape of the deployment region is not a simple geometric shape. For simplicity, in the rest of this article, $DR(g)$ is simplified to be the maximal circle enclosed by the true deployment region. Since the sensing range of each sensor is assumed to be a circle, it is not difficult to check that $DR(g)$ is the circle of radius $r - \sqrt{2}l/2$ centered at the center of g, where l is the side length of g. Figure 11 shows an example.

Notice that points on the periphery of $DR(g)$ also belong to $DR(g)$. That is, if s is deployed at a point on the periphery of $DR(g)$, s can fully cover g. For example, in Figure 12(a), if s is located at points a or b, s can fully cover both g_1 and g_2.

4.2. Greedy Deployment Strategy

We employ the following two heuristics for deploying sensors economically (in terms of the number of sensors used). First, consider $\lambda = \max\{i \mid \text{there exists some grid } g \text{ that is } i\text{-fully covered and } i < k\}$. It is clear that deploying sensors to fully cover the λ-fully covered grids improves the k-coverage rate economically. Second, define a *candidate grid* to be a λ-fully covered grid. Among all candidate grids, deploying sensors to fully cover ones with the largest area is an even more economic way. Define the *grid weight* of grid g, $GW(g)$, to be $|g|$ if g is a candidate grid and 0 otherwise. And let the *coverage weight* of point p, $CW(p)$, which is located in the intersection of $DR(g_i)$s be the summation of $GW(g_i)$s. Consider a set of points $\{p_a, p_b, p_c, p_d, p_e, p_f, p_g\}$ in Figure 12(b): the coverage weight of points p_b, p_c, and p_e is

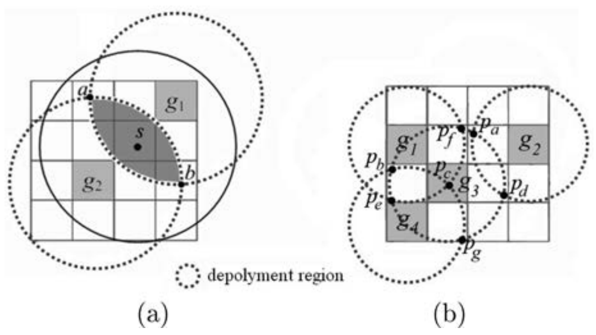

Fig. 12. Intersection of deployment regions. (a) Intersection of DR(g_1) and DR(g_2); (b) points p_b, p_c, and p_e could be chosen as the best fit.

the same and equals $GW(g_1) + GW(g_3) + GW(g_4)$, and the coverage weight of points p_a and p_d is $GW(g_2)+GW(g_3)$. Intuitively, s should be deployed at a point with the highest coverage weight.

4.3. Highest Coverage-Weight Points

For a candidate grid g, we define *fit* with respect to g, $Fit(g)$, to be a specific point in $DR(g)$ having the highest coverage weight among all points in $DR(g)$. If there is no other candidate grid whose deployment region intersects $DR(g)$, then an arbitrary point in $DR(g)$ can be chosen as $Fit(g)$. Otherwise, $Fit(g)$ can be chosen from those intersection points of peripheries of deployment regions. Consider grid g_1 in Figure 12(b). Points p_b, p_c, p_e, and p_f are intersection points of peripheries of deployment regions in $DR(g_1)$. Since p_b, p_c, and p_e have the highest coverage weight among the four intersection points, it is not difficult to see that one of the three can be chosen as $Fit(g_1)$. Also, we define a $best\_fit$ to be a fit with the highest coverage weight among all fits with respect to candidate grids. For example in Figure 12(b), either p_b, p_c, or p_e can be chosen as the best fit, since no other grid has a fit with a coverage weight higher than these points. Clearly, deploying s at a $best\_fit$ is an efficient solution. So, the main idea of our k-coverage rate deployment scheme, named k-CRD, is to deploy $(k - \lambda)$ sensors at a $best\_fit$ and repeat the process until no sensor remains. Note that deploying $(k - \lambda)$ sensors at a time is effectively the same as deploying one sensor to the $best\_fit$ at each step of the iteration, because if a sensor is deployed to a candidate point p inside the deploy region of a candidate grid g, then g becomes $(\lambda + 1)$-fully covered. The coverage weight of p will remain the same if $\lambda+1 < k$, and we can see that in the next iteration, p will still have the highest coverage weight. This implies that a sensor will be deployed to p again and so forth in the following iterations until g becomes k-fully covered. So deploying $(k - \lambda)$ sensors at a time simply saves the iteration. The detailed steps of k-CRD scheme are given in Algorithm 2.

In our k-CRD scheme, candidate grids are first determined and stored in C (Algorithm 2, step 3). For any two candidate grids g_i and g_j, determine the intersection points of peripheries of deployment regions of g_i and g_j (Algorithm 2, steps 4 to 12), and store these intersection points in P. Notice that at line 7, we identify at most two intersection points for each pair of deployment regions. As shown in Figure 10, a deployment region is not a simple geometric shape, so the number of intersection points is not necessarily two. However, in this article, we simplify $DR(g)$ of a grid g to be the maximal circle enclosed by the true deployment region, so the number of intersection points is at most two.

ALGORITHM 2: *k*-CRD Scheme

```
   Global Variables:
     Set $P$; // set of intersection points of peripheries of two deployment region
     Set $C$; // set of candidate grids
15 repeat
16     if $C$ is empty then // initialization
17         Execute $k$-Coverage Contour Evaluation Scheme and determine $\lambda$ and $C$ from its
           return foreach candidate grid $g_i$ in $C$ do
18             foreach candidate grid $g_j \neq g_i$ do
19                 if intersection of peripheries of $DR(g_i)$ and $DR(g_j)$ are not empty then
20                     Put the (at most two) intersection points into $P$
21                 else
22                     Put an arbitrary point on the periphery of $DR(g_i)$ into $P$
23                 end
24             end
25         end
26         Call Initial_Weight_Determination($P$, $C$)
27     end
28     Choose $best\_fit$ as the point (in $P$) with the highest coverage weight Deploy $k - \lambda$
       sensors at $best\_fit$ foreach candidate grid $g_i$ in $C$ do
29         if $best\_fit$ is in $DR(g_i)$ then
30             Remove $g_i$ from $C$ foreach $p_j$ in $P$ do
31                 if $p_j$ is in $DR(g_i)$ then
32                     $CW(p_j) = CW(p_j) - GW(g_i)$ if $CW(p_j)$ equals to 0 then Remove $p_i$
                       from $P$
33                 end
34             end
35         end
36     end
37     Execute $k$-Coverage Contour Evaluation Scheme
38 until the required $k$-coverage rate is satisfied or there exist no more sensors for deployment;
39 return $k$-coverage rate of the monitored area
```

ALGORITHM 3: The Initial_Weight_Determination Prcedure

```
   Data: $P$, $C$
40 foreach $p_i$ in $P$ do $CW(p_i) = 0$ foreach $p_i$ in $P$ do
41     foreach grid $g_j$ in $C$ do
42         if $p_i$ is in $DR(g_j)$ then
43             $CW(p_i) = CW(p_i) + CW(g_j)$
44         end
45     end
46 end
```

The coverage weight of each intersection point is determined by the aid of the procedure Initial_Weight_Determination and stored in CW (Algorithm 2, step 13 and Algorithm 3, steps 1 to 8). Choose one of the highest coverage weight points (in P) as the $best\_fit$ (Algorithm 2, step 15). Since candidate grids are λ-fully covered, for the purpose of increasing the k-coverage rate, deploy $k-\lambda$ sensors at the $best\_fit$ (Algorithm 2, step 16). Then every candidate grid with $best\_fit$ falling in its deployment region becomes k-fully covered, that is, they are not candidate grids anymore (Algorithm 2, steps 17 to 19). Besides, coverage weights of some points in P should be recomputed, because some candidate grids become k-covered (Algorithm 2, steps 20 to 25).

After the deployment, the k-coverage rate of the monitored area is evaluated again to check whether the required k-coverage rate is satisfied (Algorithm 2, step 28). If not, the process described is executed again, and so forth. Note that in k-CRD, C and P are pruned gradually (through steps 17 to 27 of Algorithm 2) and used only for deployment. The calling to the k-coverage contour evaluation at line 28 does not pass C. Therefore, the grids created during the evaluation have no impact on C. C and P are re-calculated from scratch from the return of the k-coverage contour evaluation only when they become empty (Algorithm 2, step 3).

4.4. Complexity Analysis

Denote the total number of grids generated by the k-CCE scheme as N. We have the number of candidate grids $|C| = O(N)$ and $N = O(2^n)$, where n is the number of iterations executed in k-CCE. The k-CRD scheme takes $O(N^3)$ time in determining a $best\_fit$.

Steps 4 to 12 of Algorithm 2 can be completed in $O(N^2)$. Recall that the deployment region of a given grid is a circle which is centered at the grid's center and has a radius smaller than r. It is not difficult to see that if a grid's deployment region intersects another grid's deployment region, the distance between these two grids' centers is less than $2r$, and vice versa. Suppose that there are at most $\hat{g}$ grids falling in a sensor's sensing range in k-CCE. Given a grid, consider all nearby grids within the range of radius $2r$: there are at most $4\hat{g} = O(\hat{g})$ deployment regions intersecting the deployment region of the grid. Hence, the number of intersection points $|P| = O(\hat{g}N)$ and step 13 of Algorithm 2 can be completed in $O(\hat{g}N^2)$ computation time. Notice that we have $\hat{g} = O(N)$. Then it is not difficult to check that our k-CRD scheme takes $O(N^3)$ time to determine a $best\_fit$.

On the other hand, consider the Grid Scan-based deployment scheme [Shen et al. 2006]. Recall that in Grid Scan, $\theta(2^{2n}) = \theta(N^2)$ grids fall in a sensor's sensing range and that the total number of grids generated by Grid Scan is $\theta(N^2)$ (see Section 3). So, Grid Scan-based deployment takes $\theta(N^4)$.

5. PRACTICAL CONSIDERATIONS AND IMPROVEMENTS

In this section, we discuss practical considerations and improvements of the k-CCE and k-CRD schemes.

5.1. Monitoring Areas with Arbitrary Shapes

So far, we assume that the monitored area is a square. It is clear that in practice, the monitored area could have an arbitrary shape. For example, in a vehicle detection application, the sensors may only be deployed along the roads.

The proposed k-CCE and k-CRD schemes can be readily extended to support areas of arbitrary shapes. We begin by enclosing the monitored area with a minimum rectangle, called the *minimum bounding rectangle* (MBR). In k-CCE, the MBR can be iteratively divided into grids as usual, except that during each iteration, we consider only those grids, called the *grids of interest*, that overlap (either fully or partially) the monitored area (see Figure 13). On the other hand, the k-CRD scheme can be extended as follows to deploy sensors to the grids of interest only. First, we assign zero grid weight (see Section 4.2) to grids out of interest (i.e., girds that do not overlap the monitored area). Second, the coverage level of grids out of interest is set to k-covered. Finally, we add an additional criteria for fit (see Section 4.3)—that a fit should be inside the monitored area (excluding obstacles, if any).

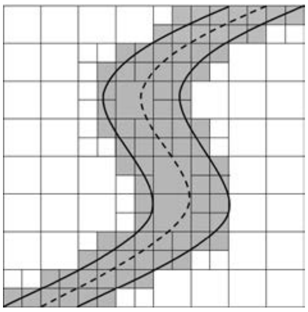

Fig. 13. The minimum bounding rectangle of a road, where the grids of interest are shaded.

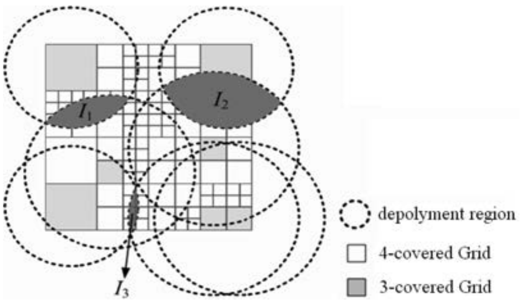

Fig. 14. The first three candidate grids are at the upper-left, right, and lower-left corners.

5.2. *k*-CRD1

Notice that if a sensor's sensing range is much smaller than the monitored area (i.e., $\hat{q}$ is much smaller than N), our simulations show that the k-CRD scheme takes less time than Grid Scan-based deployment. In the following, we propose two heuristics to further reduce the computation cost of k-CRD. Two algorithms modified from k-CRD based on the heuristics, called k-CRD1 and k-CRD2, are introduced. Simulation results show that k-CRD1 and k-CRD2 provide almost the same coverage rate increment and require much less time as compared to Grid Scan-based deployment.

Recall that a $best\_fit$ is a fit with the highest coverage weight among all fits. As we previously saw, determining such a fit results in high computation cost. In order to reduce the computation cost in k-CRD1, fit with respect to lower grid-weight grids are not determined. The main idea is based on the observation that there is a high possibility that a $best\_fit$ is a fit with respect to a higher grid-weight grid. Let C_α denote the set of first α candidate grids, sorted by grid weight, where α is a constant. In k-CRD1, a fit whose coverage weight is the highest among those fits with respect to grids in C_α is chosen as an approximate $best\_fit$. An example of k-CRD1 with $k = 4$ and $\alpha = 3$ is illustrated in Figure 14. Dotted circles denote deployment regions with respect to candidate grids (i.e., three-fully covered grids). The first α candidate grids (and their deployment regions) are at the upper-left, upper-right, and lower-left corners, respectively. Clearly, fits with respect to these grids are in I_1, I_2, and I_3, respectively. Besides, fit with respect to the grid at the lower-left corner has the highest weight, so we deploy a sensor in fit with respect to the grid at the lower-left corner.

Details of the k-CRD1 scheme are summarized in Algorithm 4. In the k-CRD1 scheme, candidate grids are determined and stored in C (step 3), and candidate grids with the highest α coverage weight are stored in C_α (steps 4 to 8). For any candidate grid g_i in C_α and any candidate grid g_j in C, determine the intersection of peripheries of

ALGORITHM 4: *k*-CRD1 Scheme

```
Global Variables:
  int $\alpha$; // a constant
  Set $P$; // set of intersection points of peripheries of two deployment region
  Set $C$; // set of candidate grids
47 repeat
48   if $C$ is empty then // initialization
49     Execute $k$-Coverage Contour Evaluation Scheme and determine $\lambda$ and $C$ from its
       return if $|C| \geq \alpha$ then
50       $C_\alpha$ = the set of first $\alpha$ candidate grids, sorted by grid weight
51     else
52       $C_\alpha = C$
53     end
54     foreach candidate grid $g_i$ in $C_\alpha$ do
55       foreach candidate grid $g_j \neq g_i$ do
56         if intersection of peripheries of $DR(g_i)$ and $DR(g_j)$ are not empty then
57           Put the (at most two) intersection points into $P_\alpha$
58         else
59           Put an arbitrary point on the periphery of $DR(g_i)$ into $P_\alpha$
60         end
61       end
62     end
63     Call Initial_Weight_Determination($P_\alpha$, $C$)
64   end
65   Choose best_fit as the point (in $P_\alpha$) with the highest coverage weight Deploy $k - \lambda$
     sensors at best_fit foreach candidate grid $g_i$ in $C$ do
66     if best_fit is in $DR(g_i)$ then
67       Remove $g_i$ from $C$ foreach $p_j$ in $P_\alpha$ do
68         if $p_j$ is in $DR(g_i)$ then
69           $CW(p_j) = CW(p_j) - GW(g_i)$ if $CW(p_j)$ equals to 0 then Remove $p_i$
             from $P_\alpha$
70         end
71       end
72     end
73   end
74   Execute $k$-Coverage Contour Evaluation Scheme
75 until the required k-coverage rate is satisfied or there exist no more sensors for deployment;
76 return $k$-coverage rate of the monitored area
```

$DR(g_i)$ and $DR(g_j)$ (steps 9 to 17); these intersection points are maintained in P_α. Notice that the intersection points of the deployment regions of any two candidate grids in $C - C_\alpha$ are not determined. Hence, the computation cost could be reduced. Similarly, coverage weight of these intersection points are determined in the procedure Initial_Weight_Determination and stored in CW (step 18). Choose one of the highest coverage weight intersection points in P_α as the approximate *best_fit* (step 20). Since candidate grids are λ-fully covered, for the purpose of increasing the *k*-coverage rate, deploy $k - \lambda$ sensors at the approximate *best_fit* (step 21). The remaining steps are similar to those of *k*-CRD.

Our *k*-CRD1 takes $O(\alpha\hat{g}^2 + \alpha N) - O(\hat{g}^2 + N) = O(N^2)$ time in determining an approximate *best_fit*, where $\hat{g}$ is an upper bound on the number of grids falling in a sensor's sensing range and N is the total number of grids generated by *k*-CCE.

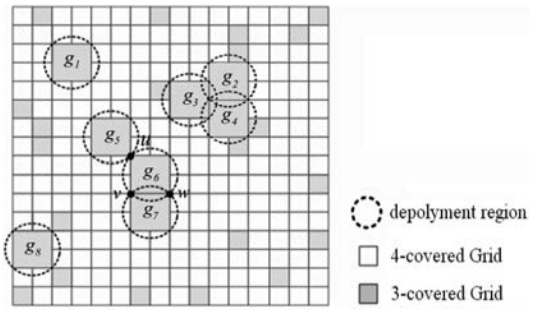

Fig. 15. An example of 4-CRD2.

5.3. *k*-CRD2

In order to further reduce the computation cost, the main motivation of scheme k-CRD2 is to avoid high computation cost of determining fits. In k-CRD2, only the highest grid-weight candidate grids are considered. Our k-CRD2 scheme is described as follows. Let C_1 denotes the set of highest grid-weight candidate grids. Randomly choose a candidate grid g from C_1. Deploy $k - \lambda$ sensors at a point p satisfying the following.

(1) p is located in $DR(g)$.
(2) The maximal number of grids in C_1 can be fully covered.

For example, in Figure 15, grids $g_1, g_2, \ldots, g_8$ constitute C_1. Randomly choose a grid from C_1, say g_6. Then deploy $(4 - 3)$ sensors at point u, because the maximum number of grids (i.e., g_5 and g_6) in C_1 can be fully covered.

Details of the k-CRD2 scheme are summarized in Algorithm 5. Candidate grids with the highest coverage weight are considered and stored in C_1 (step 4). For any two candidate grids g_i and g_j in C_1, determine their intersection points and maintain these intersection points in P_1. Notice that the intersection points of deployment regions of any two candidate grids in $C - C_1$ are not determined. Hence the computation cost could be much reduced. Similarly, coverage weights of these intersection points are determined in the procedure Initial_Weight_Determination and stored in CW. Choose one of the highest coverage weight intersection points in P_1 as the approximate $best\_fit$ (step 15), and deploy $k - \lambda$ sensors at the approximate $best\_fit$ (step 16). The remaining steps are similar to those of k-CRD.

As we can see, k-CRD2 takes $O(i\hat{g} + N) = O(N \log N)$ time, where i is the number of iterations executed in k-CCE, $\hat{g}$ is an upper bound on the number of grids falling in a sensor's sensing range, and N is the total number of grids generated by k-CCE.

6. SIMULATION RESULT

A simulator is implemented in Java language to evaluate the performance of our schemes. All simulations are executed on a personal computer with Intel Core 2 Duo E6400 2.13G/2M, 1G RAM and Windows XP operation system. There are two major experiments: the first concerns k-coverage contour evaluation and the second concerns k-coverage rate deployment. The assumptions for these two experiments are summarized as follows. Pre-deployed sensors are randomly and uniformly deployed in a 100×100 m monitored area. The sensing radius of each sensor is ten meters. All algorithms are executed at the base station, so some other issues, such as MAC-layer protocol and routing overhead, are all ignored in our simulator. Each experiment is

ALGORITHM 5: k-CRD2 Scheme

```
Global Variables:
    int $\beta$; // a constant
    int $n$; // number of iterations executed in $k$-CCE
    Set $P$; // set of intersection points of peripheries of two deployment region
    Set $C$; // set of candidate grids
77 repeat
78     if $C$ is empty then // initialization
79         Execute $k$-Coverage Contour Evaluation Scheme and determine $\lambda$ and $C$ from its
           return $C_1$ = the set of candidate grids with the highest grid weight Randomly
           choose a grid $g$ in $C_1$ foreach candidate grid $g_i \neq g$ and $|P_1| \leq \beta \cdot n$ do
80             if intersection of peripheries of $DR(g)$ and $DR(g_i)$ are not empty then
81                 Put the (at most two) intersection points into $P_1$
82             else
83                 Put an arbitrary point on the periphery of $DR(g)$ into $P_1$
84             end
85         end
86         Call Initial_Weight_Determination($P_1$, $C_1$)
87     end
88     Choose $best\_fit$ as the point (in $P_1$) with the highest coverag weight Deploy $k-\lambda$ sensors
       at $best\_fit$ foreach candidate grid $g_i$ in $C$ do
89         if $best\_fit$ is in $DR(g_i)$ then
90             Remove $g_i$ from $C$ foreach $p_j$ in $P_1$ do
91                 if $p_j$ is in $DR(g_i)$ then
92                     $CW(p_j) = CW(p_j) - GW(g_i)$ if $CW(p_j)$ equals to 0 then Remove $p_i$
                       from $P_1$
93                 end
94             end
95         end
96     end
97     Execute $k$-Coverage Contour Evaluation Scheme
98 until the required k-coverage rate is satisfied or there exist no more sensors for deployment;
99 return $k$-coverage rate of the monitored area
```

repeated 20 times with distinct sensor pre-deployments. The experiment results of our scheme are compared with those of the Grid Scan scheme [Shen et al. 2006]. To the best of our knowledge, k-CCE and Grid Scan are currently the only two schemes that support k-coverage contour evaluation.

There is no discussion on how to decide the proper size of grids in Grid Scan. In this article, we adopt an approach similar to that of k-CCE: in the beginning, the side length of each grid is set to r/m, where m is the initial dividing factor decided by users. If the MEE returned by the Grid Scan is higher than MTEE, every grid is divided into four subgrids, and MEE is calculated again. The division repeats until the MEE given by the Grid Scan is lower than MTEE, and then the final grids are used for evaluation.

Note that by the definition of MEE, the MEE given by Grid Scan will be identical to that given by k-CCE if the grid size used by Grid Scan is the minimum grid size in k-CCE. This is because in k-CCE, a grid having size larger than the minimum size cannot be an uncertain grid, thus has no impact on MEE. In addition, grids with minimum size in k-CCE will have one-to-one corresponding grids in Grid Scan.

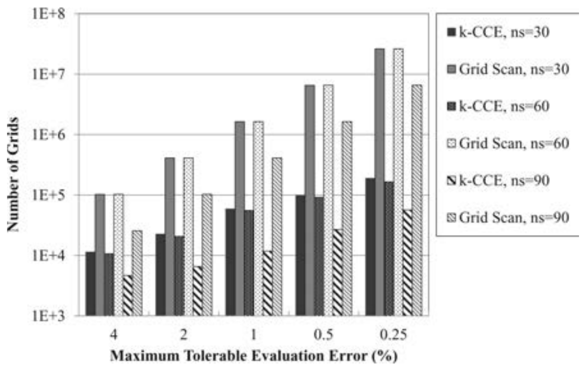

Fig. 16. Number of grids versus maximum tolerable evaluation error for $k = 1$. Red blocks on the top of the bars represent the 95% confidence intervals.

6.1. Simulation Result of *k*-Coverage Contour Evaluation

Clearly, in the k-coverage contour evaluation experiment, the number of grids is a basic metric for computation of cost estimation. The number of grids generated in k-CCE is compared with that of Grid Scan for different numbers of pre-deployed sensors (n_s), maximum tolerable evaluation error (MTEE), and values of k.

According to the experiment results, much fewer grids are needed in the k-CCE scheme than in Grid Scan under the same number of pre-deployed sensors, maximum tolerable evaluation error, and value of k. A typical example was shown in Figure 1. Figures 1(a) and 1(b) show grid division for Grid Scan and our scheme, respectively, under $MTEE = 0.1$, $k = 4$, and $n_s = 60$.

Effect of Maximum Tolerable Evaluation Error on Number of Grids. Figure 16 shows that the difference in the number of grids between k-CCE and Grid Scan increases as the maximum tolerable evaluation error decreases for $k = 1$. Notice that the number of grids for $n_s = 90$ is the smallest no matter which scheme is considered and what the value of MTEE is. This is because most of the area is one-fully covered when $n_s = 90$. That is, few grid divisions are required. Besides, in Grid Scan, both the number of grids for $n_s = 30$ and $n_s = 60$ are the same, no matter the value of MTEE, the smallest grid sizes in these cases are the same.

Effect of k on Number of Grids. Figures 17, 18, and 19 show the number of grids of k-CCE and Grid Scan for $k = 2$, 3, and 4, respectively. According to experiment results, the number of grids of these two schemes increases as k increases, and no matter what value k is, the number of grids in Grid Scan is at least ten times the number of grids in k-CCE.

We observe that the number of grids is largest when the k-coverage rate is between 50% and 65%. This is because there are fewer uncertain grids when the k-coverage rate is higher than 65% or lower than 50%. In Figure 17, the number of grids of the k-CCE scheme for $n_s = 60$ is greater than for $n_s = 90$ because the two-coverage rate is about 70% for $n_s = 90$ but 60% for $n_s = 60$. In Figures 18 and 19, the number of grids of the k-CCE scheme for $n_s = 60$ is smaller than for $n_s = 90$ because the three-coverage rate and four-coverage rate are less than 50% for $n_s = 60$.

Execution Time of k-CCE and Grid Scan. Figure 20 shows the execution time of the k-CCE and Grid Scan for $MTEE = 0.25\%$ and $n_s = 90$. The execution time of the k-CCE scheme is the total execution time from the first iteration to the last iteration, while the execution time of Grid Scan is the total time to evaluate the k-coverage rate of

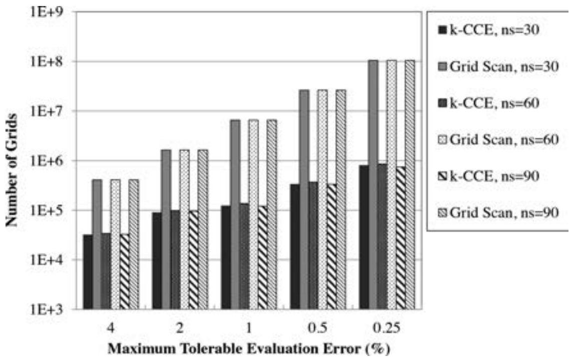

Fig. 17. Number of grids versus maximum tolerable evaluation error for $k = 2$. Red blocks on the top of the bars represent the 95% confidence intervals.

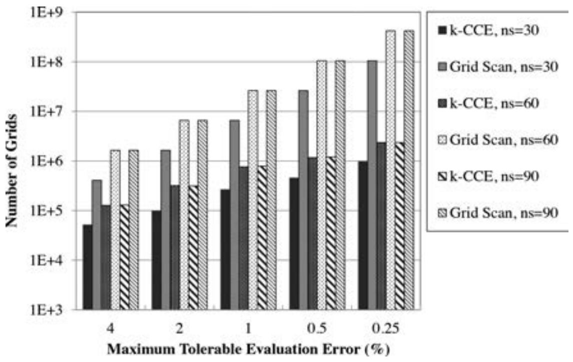

Fig. 18. Number of grids versus maximum tolerable evaluation error for $k = 3$. Red blocks on the top of the bars represent the 95% confidence intervals.

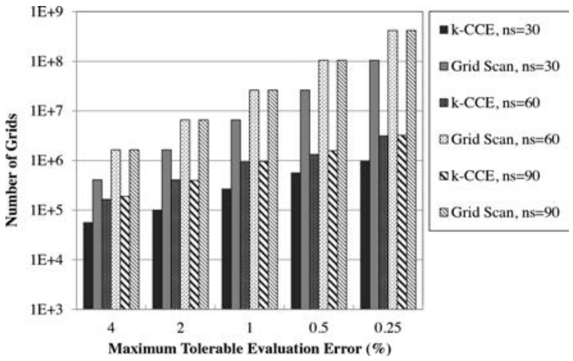

Fig. 19. Number of grids versus maximum tolerable evaluation error for $k = 4$. Red blocks on the top of the bars represent the 95% confidence intervals.

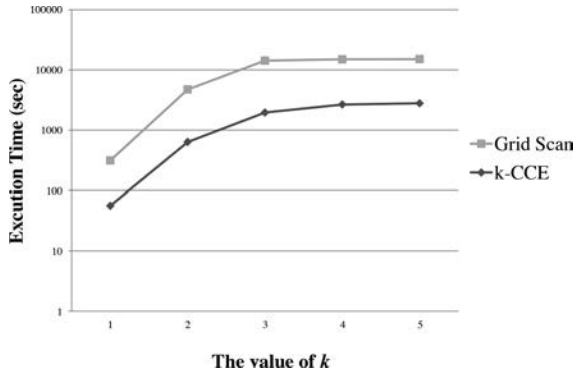

Fig. 20. Execution time versus value of k for $n_s = 90$ and $MTEE = 0.25\%$.

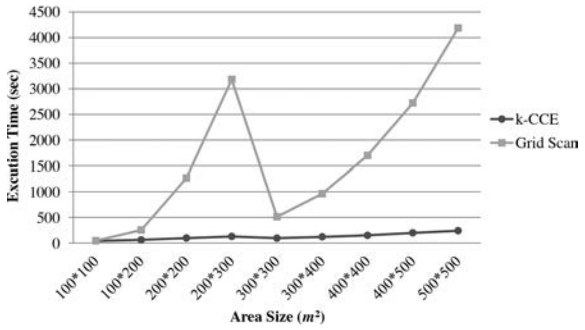

Fig. 21. Execution time versus network size for $n_s = 60$ and $MTEE = 0.25\%$.

the monitored area when its grids are of equal size to the size of the smallest grids in k-CCE. According to the simulation results, the execution time of k-CCE is much better than that of Grid Scan for $1 \leq k \leq 4$. Recall that the computation cost of k-CCE and Grid Scan are $O(N)$ and $O(N^2)$ (see Section 3), respectively, where N is the total number of grids generated by k-CCE. The simulation result also illustrates the same trend.

Effect of Network Size. Figure 21 shows the execution time of k-CCE and Grid Scan for $MTEE = 0.25\%$ and $n_s = 60$ under different network sizes varying from 100×100 to 500×500 m^2. As we can see, the execution time of the k-CCE scheme is far less than that of Grid Scan. Also, the execution time of the k-CCE scheme is relatively stable to changes in network size because the k-CCE scheme divides the grids only when needed, so the number of grids processed in each iteration is far less then the number of all possible grids in that iteration. Besides, based on Lemma 3.3, if the network is not too dense, it is likely that the number of grids at the ith iteration is contributed by the $(2^{2+i}m + 8) \cdot n_s$ part rather than by $N_i = {}^{|A|(2^{i-1}m)^2}/{}_{r^2}$ in the minimum and therefore is independent with respect to $|A|$. Notice that the execution time of Grid Scan fluctuates dramatically. We believe there are two main reasons for this. One is that the number of sensors overlapping a grid drops suddenly when the network density falls below a certain threshold. Recall that we fix the number of sensor nodes. The network density drops as the network size increases. When the network size increases from 100×100 m^2, the execution time increases proportionally to the total number of grids in the network. However, at the network size where the number of sensors overlapping a grid drops suddenly, the execution time shrinks, since the time spent in processing each grid (which depends on how many sensors overlapping it) becomes

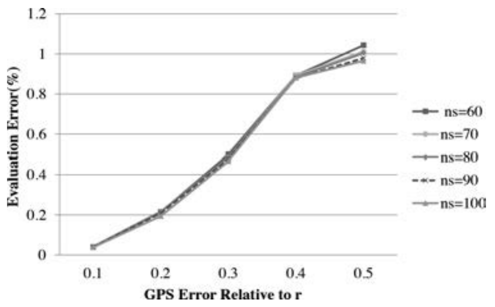

Fig. 22. Evaluation errors versus localization error given different numbers of sensor nodes.

much faster. After this point, the execution time increases again proportionally to the total number of grids in the network. Another reason is that both k-CCE and Grid Scan terminate with larger-size grids after this threshold, as the number of intersecting points between the coverage peripheries of sensor nodes drops at this threshold, and therefore, the final grid size needs not be as fine as those in denser networks in order to yield a satisfactory MEE. However, since Grid Scan is sensitive to the grid size, its execution time is much higher right before this threshold.

Effect of Localization Error. In terms of the impact of localization (e.g., GPS) error, we conducted a set of simulations, and their results, are shown in Figure 22. To be general, we do not focus on the error of a particular localization technique. Instead, we consider different error levels proportional to the sensing radius r of a node. The error is modeled using a normally distributed random variable with standard deviation xr, where $x = 0.1, 0.2, \ldots, 0.5$ as indicated by the x-axis. Define the *intrinsic evaluation error* as the difference between the union of areas covered by sensors at their correct locations and the union of areas covered by sensors at their localized (e.g., GPS) locations. This error will be inherited by any algorithm evaluating one-coverage (and above). The y-axis of Figure 22 shows the ratio (in percentage) of the intrinsic evaluation error to the monitored area.

Clearly, the larger the localization error, the higher the intrinsic evaluation error. However, even at a high localization error rate ($0.5r$), the intrinsic evaluation error is still small (1.2% at most). In addition, we can see that the impact of localization error is mitigated as the number of nodes increases, because when the number of nodes is large, the two union areas become larger, making their difference (which occurs at their borders) less significant. A dense network can be even more resistant to the localization error.

6.2. Simulation Result of *k*-Coverage Rate Deployment

In k-coverage rate deployment experiments, coverage rate increment and execution time are two metrics for judging performance. Similarly, experiments are performed with 60 pre-deployed sensors (i.e., $n_s = 60$), $k = 1$, and several values of maximum tolerable evaluation error. First, we consider the case in which there is only one sensor for deployment.

Effect of MTEE. Table I reveals a similar result to the analysis on time complexity of Grid Scan-based deployment (i.e., $\theta(N^4)$), k-CRD (i.e., $O(N^3)$), k-CRD1 (i.e., $O(N^2)$), and k-CRD2 (i.e., $O(N \log N)$) shown in Section 4, where N is the total number of grids generated by k-CCE. It shows that both k-CRD1 and k-CRD2 provide almost the same coverage rate increment and take much less execution time compared to Grid

Table I. Maximum Tolerable Evaluation Error, Coverage Rate Increment, and Execution Time ($n_s = 60$ and $k = 1$)

	Coverage Rate Increment (%)			
MTEE (%)	k-CRD	k-CRD1	k-CRD2	Grid Scan
4	2.67	2.56	1.96	2.68
2	2.97	2.69	2.07	2.79
1	2.83	2.74	2.18	2.83
0.5	2.83	2.78	2.25	2.86

	Execution Time (sec.)			
MTEE (%)	k-CRD	k-CRD1	k-CRD2	Grid Scan
4	2852	13.6	0.0063	2985
2	27282	92	0.015	36519
1	214212	598	0.043	589407
0.5	1682756	3698	0.096	N/A*

Table II. Coverage Rate Increment versus k ($n_s = 60$ and $MTEE = 4\%$)

	Coverage Rate Increment (%)			
k	k-CRD	k-CRD1	k-CRD2	Grid Scan
1	2.67	2.56	1.96	2.68
2	2.35	2.3	1.99	2.59
3	2.23	2.2	1.96	2.38
4	2.21	2.15	2.11	2.36

Scan-based deployment. It is also observed that when MTEE is small, Grid Scan-based deployment takes too much time to figure out the location for deploying sensors.

Effect of k. Next, we study the effect of k on the coverage rate increment. Given $n_s = 60$ and $MTEE = 4\%$, the results are shown in Table II. As we can see, both k-CRD and k-CRD1 offer comparable coverage rate increment compared with Grid Scan. k-CRD2 is left behind when $k = 1$. However, as k increases, the coverage rate increment offered by k-CRD2 becomes comparable with that of other schemes. This implies that at a higher k, there is a high possibility that the $best\_fit$ comes from the fit with respect to the grid with the highest grid weight. Hence, k-CRD2 is able to achieve significant reduction in execution time without causing too much coverage rate degradation when k is large.

Effect of Network Size. Table III summarizes the execution time of respective schemes given $n_s = 60$ and $MTEE = 4\%$ under different network sizes varying from 100×100 to 500×500 m^2. As we can see, the low execution-time advantage offered by k-CRD, k-CDR1, and k-CRD2 remains as the network size is changed. Notice that the execution time of all schemes increase sharply at 300×300 m^2, because, as we have seen in Figure 21, the number of sensors overlapping a grid drops suddenly when the network density falls below a certain threshold. Right before this threshold, many sensors partially overlap the grids, resulting in a sudden increase in the number of fits and, hence, a rise in execution time. After this point, the execution time decreases again due to the drop of overlapping sensors.

Effect of the Number of Re-deployed Sensor Nodes. We consider the case in which there are five sensors for deployment. The values of MTEE and k are assumed to be 2% and 1, respectively. Figure 23 shows that k-CRD1 provides almost the same coverage rate increment as Grid Scan-based deployment, and there is only small difference,

Table III. Execution Time versus Network Size ($n_s = 60$ and $MTEE = 4\%$)

	Execution Time (sec)			
Area Size (m^2)	k-CRD	k-CRD1	k-CRD2	Grid Scan
100×100	2,852	13.6	0.0063	2,985
200×200	34,960	8.9	0.005	39,245
300×300	59,457	11.2	0.012	247,233
400×400	19,365	6.4	0.006	52,897
500×500	32,665	8.4	0.009	130,389

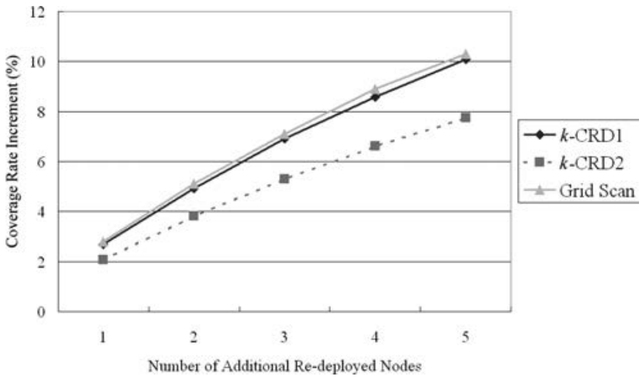

Fig. 23. Coverage rate increment versus number of deployed sensors for MTEE = 2%.

2.55%, in the coverage rate increment between k-CRD2 and Grid Scan-based deployment after deploying five sensors. Although the coverage rate increment for k-CRD2 is little less than for k-CRD1 and Grid Scan-based deployment, it takes much less execution time.

7. CONCLUSIONS

In this article, a k-coverage contour evaluation scheme and a k-coverage rate deployment scheme are proposed. In our k-coverage contour evaluation scheme, the monitored area is divided into nonuniform grids. Each grid is further divided into subgrids if more coverage information can be obtained from these subgrids. With the aid of coverage information of grids, an evaluation of the k-coverage rate of the monitored area is available. On the other hand, in order to avoid dividing many grids for acquiring little coverage information, another criterion—maximum tolerable evaluation error—for terminating grid division is also introduced. Based on our k-coverage contour evaluation scheme, we propose a deployment scheme, called k-CRD, based on the notion of deployment regions to increase the k-coverage rate of the monitored area. Two approximation algorithms for k-CRD, which require much less computational time, are also discussed.

To make our work applicable to a wider range of real-world scenarios where the sensing rage of a sensor may be affected by environments or the remaining power of sensor, in the future, we will target relaxing the assumptions that each sensor has a circular sensing range and that all sensors have a uniform sensing range. In addition, in order to avoid system bottlenecks and single points of failure, we will also study how the proposed schemes can be executed in a distributed manner.

APPENDIX

Algorithms 6 and 7 elaborate the detailed steps of our k-coverage contour evaluation (k-CCE) scheme. We summarize this scheme next.

Initially, the monitored area is regarded as an uncertain grid (Algorithm 6, steps 2 and 3), and each sensor which is deployed in the network is regarded as having the possibility of partially covering every uncertain grid (Algorithm 6, steps 5 and 6). In Coverage_Inf_ Acquaring, uncertain grids are divided into equal subgrids of side length l (Algorithm 7, step 3). Note that the initial l is obtained by dividing the sensing radius r by a positive integer m. We call r/l the *dividing factor*, and m is the initial dividing factor decided by users. The coverage information of subgrids is obtained (Algorithm 7, steps 4 to 24). Coverage information of uncertain subgrids are temporarily stored in array U_temp (Algorithm 7, steps 5 to 15), while subgrids, which are fully covered by at least k distinct sensors, are maintained in array K (Algorithm 7, steps 17 to 20). For each uncertain subgrid g, g's ID, the number of sensors which partially cover g, IDs of sensors which partially cover g, and the number of sensors which fully cover g are stored in $U\_temp[*].grid\_id$, $U\_temp[*].num\_partially\_covered$, $U\_temp[*].partially\_covered$, and $U\_temp[*].num\_fully\_covered$, respectively. For each subgrid which is at least fully covered by k distinct sensors, its ID is stored in array K. The hash table C_temp stores all grids created during the zth iteration. Finally, coverage information of uncertain subgrids and contour at the zth iteration are stored in array U and $C[z]$, respectively (Algorithm 7, steps 26 to 28). After completing procedure Coverage_Inf_Acquaring, the maximum evaluation error is also available with the aid of array U. If the maximum evaluation error is higher than the maximum tolerable evaluation error, the dividing factor is doubled (Algorithm 6, step 13) and further grid division will be performed. Otherwise, the arrays C and K are returned.

ALGORITHM 6: *k*-Coverage Contour Evaluation Scheme

```
Global Variables:
  int $max\_len$; // predefined maximum length of an array
  typedef struct
  |  long $grid\_id$ int $num\_partial\_covered$ long[$max\_len$] $partial\_covered$ int
  |  $num\_fully\_covered$
  as $Grid$;
  double $r$; // sensing radius
  int $m$; // initial dividing factor, must be positive
  int $z$; // iteration count
  double $l$; // side length of a grid at the $z$th iteration
  Grid[$max\_len$] $U$; // array of uncertain grids
  int $u\_size$; // number of uncertain grids
  long[$max\_len$] $K$; // array of $k$-covered grids
  int $k\_size$; // number of $k$-covered grids
  HashTable<long, Grid> $C$; // array of hash tables mapping $grid\_id$s to Grids
100 $l = r/m$; // initialization
101 $z = u\_size = k\_size = 1$; // let the monitored area be an uncertain grid
102 $U[u\_size].grid\_id = 1$; // row-major index of the monitored area
103 $U[u\_size].num\_partially\_covered$ = number of sensors deployed in the network for $i = 1$ to
    $U[u\_size].num\_partially\_covered$ do
104 |  $U[u\_size].partially\_covered[i]$ = the $i$th sensor deployed in the network
105 end
106 $C[z].put(U[u\_size].grid\_id, U[u\_size])$ repeat
107 |  Call Coverage_Inf_Acquaring() Calculate maximum evaluation error $z$++ $l = l/2$
108 until maximum evaluation error > maximum tolerable evaluation error;
109 return $C$ and $K$
```

ALGORITHM 7: The Coverage_Inf_Acquaring Procedure

```
Local Variables at the $z$th Iteration:
    Grid[$max\_len$] $U\_temp$; // array of uncertain grids
    int $u\_temp\_size$; // number of uncertain grids
    HashTable<long,Grid> $C\_temp$; // hash table mapping $grid\_id$s to Grids
110 $u\_temp\_size = 0$ for $i = 1$ to $u\_size$ do
111     Divide grid $U[i].grid\_id$ into sub-grids with side length of $l$ foreach sub-grid $g$ of grid $U[i].grid\_id$ do
112         $u\_temp\_size$++ $U\_temp[u\_temp\_size].grid\_id = g$ $U\_temp[u\_temp\_size].num\_fully\_covered = U[i].num\_fully\_covered$ for $j = 1$ to $U[i].num\_partially\_covered$ do
113             if $g$ is partially covered by $U[i].partially\_covered[j]$ then
114                 $p = $ ++$U\_temp[u\_temp\_size].num\_partially\_covered$ $U\_temp[u\_temp\_size].partially\_covered[p] = U[i].partially\_covered[j]$
115             else if $g$ is fully covered by $U[i].partially\_covered[j]$ then
116                 $U\_temp[u\_temp\_size].num\_fully\_covered$++
117             end
118         end
119         $C\_temp.put(g, U\_temp[u\_temp\_size])$ if $U\_temp[u\_temp\_size].num\_fully\_covered \geq k$ then
120             $K[k\_size] = U\_temp[u\_temp\_size].grid\_id$ $k\_size$++
121         end
122         if $U\_temp[u\_temp\_size].num\_partially\_covered = 0$ or $U\_temp[u\_temp\_size].num\_fully\_covered \geq k$ then
123             $u\_temp\_size$–
124         end
125     end
126 end
127 $U = U\_temp$ $u\_size = u\_temp\_size$ $C[z] = C\_temp$
```

REFERENCES

BAI, X., KUMAR, S., XUAN, D., YUN, Z., AND LAI, T. H. 2006. Deploying wireless sensors to achieve both coverage and connectivity. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 131–142.

BARTOLINI, N., CALAMONERI, T., LA PORTA, T., MASSINI, A., AND SILVESTRI, S. 2009. Autonomous deployment of heterogeneous mobile sensors. In *Proceedings of the ACM International Conference on Network Protocols*. 42–51.

BARTOLINI, N., CALAMONERI, T., LA PORTA, T., PETRIOLI, C., AND SILVESTRI, S. 2012. Sensor activation and radius adaptation (sara) in heterogeneous sensor networks. *ACM Trans. Sen. Netw. 8,* 3.

BULUSU, N., HEIDEMANN, J., AND ESTRIN, D. 2000. Gps-less low-cost outdoor localization for very small devices. *IEEE Per. Commun. 7,* 5, 28–34.

CARDEI, M. AND ZHU DU, D. 2005. Improving wireless sensor network lifetime through power aware organization. *ACM Wirel. Netw. 11*, 333–340.

CHAKRABARTY, K., MEMBER, S., IYENGAR, S. S., QI, H., AND CHO, E. 2002. Grid coverage for surveillance and target location in distributed sensor networks. *IEEE Trans. Comput. 51*, 1448–1453.

CHENG, X., DU, D., WANG, L., AND XU, B. 2008. Relay sensor placement in wireless sensor networks. *Wirel. Netw. 14,* 3, 347–355.

DHILLON, S. S. AND CHAKRABARTY, K. 2003. Sensor placement for effective coverage and surveillance in distributed sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference*. 1609–1614.

DHILLON, S. S., CHAKRABARTY, K., AND IYENGAR, S. S. 2002. Sensor placement for grid coverage under imprecise detections. In *Proceedings of the IEEE International Conference on Information Fusion*. 1581–1587.

Dietrich, I. and Dressler, F. 2009. On the lifetime of wireless sensor networks. *ACM Trans. Sen. Netw. 5,* 1, 1–39.

Du, X. and Lin, F. 2005. Improving sensor network performance by deploying mobile sensors. In *Proceedings of the IEEE Performance, Computing, and Communications Conference*. 67–71.

Gallais, A., Carle, J., Simplot-Ryl, D., and Stojmenovic, I. 2007. Ensuring area k-coverage in wireless sensor networks with realistic physical layers. In *Proceedings of the IEEE International Conference on Sensors*. 880–883.

Heo, N. and Varshney, P. 2004. Energy-efficient deployment of intelligent mobile sensor networks. *IEEE Trans. Syst. Man Cybernet. Part A: Systems and Humans 35,* 1, 78–92.

Hofmann-Wellenhof, B., Lichtenegger, H., and Collins, J. 1997. *Global Positioning System: Theory and practice*. Springer Verlag, Berlin Heidelberg.

Hou, Y., Shi, Y., Sherali, H., and Midkiff, S. 2005. On energy provisioning and relay node placement for wireless sensor networks. *IEEE Trans. Wirel. Commun. 4,* 5, 2579–2590.

Hu, L. and Evans, D. 2004. Localization for mobile sensor networks. In *Proceedings of the ACM International Conference on Mobile Computing and Networking*. 45–57.

Huang, C. and Tseng, Y. 2005. The coverage problem in a wireless sensor network. *Mobile Netw. Appl. 10,* 4, 519–528.

Kar, K. and Banerjee, S. 2003. Node placement for connected coverage in sensor networks. In *Proceedings of the Workshop on Modeling Optimization in Mobile Ad Hoc and Sensor Systems (WiOpt)*.

Kasbekar, G., Bejerano, Y., and Sarkar, S. 2011. Lifetime and coverage guarantees through distributed coordinate-free sensor activation. *IEEE/ACM Trans. Netw. 19,* 2, 470–483.

Krishnamachari, B. and Iyengar, S. 2004. Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks. *IEEE Trans. Comput.* 241–250.

Kumar, N., Gunopulos, D., and Kalogeraki, V. 2005. Sensor network coverage restoration. *Distrib. Comput. Sens. Syst.* 409–409.

Kumar, S., Lai, T., and Arora, A. 2005. Barrier coverage with wireless sensors. In *Proceedings of the ACM International Conference on Mobile Computing and Networking*. 284–298.

Kumar, S., Lai, T., and Balogh, J. 2004. On k-coverage in a mostly sleeping sensor network. In *Proceedings of the ACM Intnational Conference on Mobile Computing and Networking*. 144–158.

Lin, F. and Chiu, P. 2005. A near-optimal sensor placement algorithm to achieve complete coverage-discrimination in sensor networks. *IEEE Commun. Lett. 9,* 1, 43–45.

Liu, B. and Towsley, D. 2005. A study of the coverage of large-scale sensor networks. In *Proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems*. 475–483.

Liu, T., Bahl, P., and Chlamtac, I. 2002. Mobility modeling, location tracking, and trajectory prediction in wireless atm networks. *IEEE J. Select. Areas Commun. 16,* 6, 922–936.

Lorincz, K. and Welsh, M. 2007. Motetrack: A robust, decentralized approach to rf-based location tracking. *Pers. Ubiq. Comput. 11,* 6, 489–503.

Megerian, S., Koushanfar, F., Potkonjak, M., and Srivastava, M. 2005. Worst and best-case coverage in sensor networks. *IEEE Trans. Mobile Comput.* 84–92.

Meguerdichian, S., Koushanfar, F., Potkonjak, M., and Srivastava, M. 2002. Coverage problems in wireless ad-hoc sensor networks. In *Proceedings of the IEEE International Conference on Computer and Computer Communications*. Vol. 3, 1380–1387.

Sanli, H. and Cam, H. 2005. Energy efficient differentiable coverage service protocols for wireless sensor networks. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops*. 406–410.

Shakkottai, S., Srikant, R., and Shroff, N. 2005. Unreliable sensor grids: Coverage, connectivity and diameter. *Ad Hoc Netw. 3,* 6, 702–716.

Shen, X., Chen, J., and Sun, Y. 2006. Grid scan: A simple and effective approach for coverage issue in wireless sensor networks. In *Proceedings of the IEEE International Conference on Communications*. 3480–3484.

Sheu, J. and Lin, H. 2007. Probabilistic coverage preserving protocol with energy efficiency in wireless sensor networks. In *Proceedings of the IEEE International Conference on Wireless Communications and Networking Conference*. 2631–2636.

Sun, T., Chen, L., Han, C., and Gerla, M. 2005. Reliable sensor networks for planet exploration. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control*. 816–821.

Wan, P. and Yi, C. 2006. Coverage by randomly deployed wireless sensor networks. *IEEE/ACM Trans. Netw. 14,* SI, 2658–2669.

WANG, X., XING, G., ZHANG, Y., LU, C., PLESS, R., AND GILL, C. 2003. Integrated coverage and connectivity configuration in wireless sensor networks. In *Proceedings of the ACM International Conference on Embedded Networked Sensor Systems*. 28–39.

WANG, Y., HU, C., AND TSENG, Y. 2005. Efficient deployment algorithms for ensuring coverage and connectivity of wireless sensor networks. In *Proceedings of the IEEE International Conference on Wireless Internet*. 114–121.

WANG, Y. AND TSENG, Y. 2008. Distributed deployment schemes for mobile wireless sensor networks to ensure multilevel coverage. *IEEE Trans. Parallel Distrib. Syst. 19,* 9, 1280–1294.

XINGYU, P. AND HONGYI, Y. 2006. Redeployment problem for wireless sensor networks. In *Proceedings of the IEEE International Conference on Communication Technology*. 1–4.

YANG, Y. AND CARDEI, M. 2007. Movement-assisted sensor redeployment scheme for network lifetime increase. In *Proceedings of the ACM Symposium on Modeling, Analysis, and Aimulation of Wireless and Mobile Systems*. 13–20.

ZHANG, H. AND HOU, J. 2005a. Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc Sen. Wirel. Netw. 1*, 89–124.

ZHANG, H. AND HOU, J. 2005b. On the upper bound of α-lifetime for large sensor networks. *ACM Trans. Sen. Netw. 1,* 2, 272–300.

ZHOU, Z., DAS, S., AND GUPTA, H. 2005. Connected k-coverage problem in sensor networks. In *Proceedings of the IEEE International Conference on Computer Communications and Networks*. 373–378.

ZHOU, Z., DAS, S., AND GUPTA, H. 2009. Variable radii connected sensor cover in sensor networks. *ACM Trans. Sen. Netw. 5,* 1.

ZOU, Y. AND CHAKRABARTY, K. 2003. Sensor deployment and target localization based on virtual forces. In *Proceedings of the IEEE International Conference on Computer and Communications*. 1293–1303.

ZOU, Y. AND CHAKRABARTY, K. 2005. A distributed coverage-and connectivity-centric technique for selecting active nodes in wireless sensor networks. *IEEE Trans. Comput. 54,* 8, 978–991.

Received February 2011; revised February, June 2012; accepted June 2012