The Coverage Problem in a Wireless Sensor Network

Chi-Fu Huang
Department of Computer Science and
Information Engineering
National Chiao-Tung University
Hsin-Chu, 30050, Taiwan
cfhuang@csie.nctu.edu.tw

Yu-Chee Tseng
Department of Computer Science and
Information Engineering
National Chiao-Tung University
Hsin-Chu, 30050, Taiwan
yctseng@csie.nctu.edu.tw

ABSTRACT

One fundamental issue in sensor networks is the *coverage* problem, which reflects how well a sensor network is monitored or tracked by sensors. In this paper, we formulate this problem as a decision problem, whose goal is to determine whether every point in the service area of the sensor network is covered by at least k sensors, where k is a predefined value. The sensing ranges of sensors can be unit disks or non-unit disks. We present polynomial-time algorithms, in terms of the number of sensors, that can be easily translated to distributed protocols. The result is a generalization of some earlier results where only $k = 1$ is assumed. Applications of the result include: (i) positioning applications, (ii) situations which require stronger environmental monitoring capability, and (iii) scenarios which impose more stringent fault-tolerant capability.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Geometrical problems and computations, Routing and layout*

General Terms

Algorithms, Measurement, Reliability, Performance, Theory

Keywords

ad hoc network, computer geometry, coverage problem, ubiquitous computing, wireless network, sensor network

1. INTRODUCTION

The rapid progress of wireless communication and embedded micro-sensing MEMS technologies has made *wireless sensor networks* possible. Such environments may have many inexpensive wireless nodes, each capable of collecting, storing, and processing environmental information, and communicating with neighboring nodes. In the past, sensors are connected by wire lines. Today, this environment is combined with the novel *ad hoc* networking technology to facilitate inter-sensor communication [4, 15]. The flexibility of installing and configuring a sensor network is thus greatly improved. Recently, a lot of research activities have recently been dedicated to sensor networks, including design issues related to the physical and media access layers [13, 18, 20] and routing and transport protocols [2, 5, 6]. Localization and positioning applications of wireless sensor networks are discussed in [1, 3, 11, 12, 17].

Since sensors may be spread in an arbitrary manner, one of the fundamental issues in a wireless sensor network is the *coverage problem*. In general, this reflects how well an area is monitored or tracked by sensors. In the literature, this problem has been formulated in various ways. For example, the *Art Gallery Problem* is to determine the number of observers necessary to cover an art gallery (i.e., the service area of the sensor network) such that every point in the art gallery is monitored by at least one observer. This problem can be solved optimally in a 2D plane, but is shown to be NP-hard when extended to a 3D space [7]. Reference [8] defines a sensor coverage metric called *surveillance* that can be used as a measurement of quality of service provided by a particular sensor network, and centralized optimum algorithms that take polynomial time are proposed to evaluate paths that are best and least monitored in the sensor network. The work [9] further investigates the problem of how well a target can be monitored over a time period while it moves along an arbitrary path with an arbitrary velocity in a sensor network. Localized exposure-based coverage and location discovery algorithms are proposed in [10].

On the other hand, some works are targeted at particular applications, but the central idea is still related to the coverage issue. For example, sensors' on-duty time should be properly scheduled to conserve energy. Since sensors are arbitrarily distributed, if some nodes share the common sensing region and task, then we can turn off some of them to conserve energy and thus extend the lifetime of the network. This is feasible if turning off some nodes still provide the same "coverage" (i.e., the provided coverage is not affected). Reference [14] proposes a heuristic to select mutually exclusive sets of sensor nodes such that each set of sensors can provide a complete coverage the monitored area. Also targeted at turning off some redundant nodes, [19] proposes a probe-based *density control* algorithm to put some nodes in a sensor-dense area to a doze mode to ensure a long-lived, robust sensing coverage. A coverage-preserving node scheduling scheme is presented in [16] to determine when a node can be turned off and when it should be rescheduled to become active again.

In this work, we consider a more general sensor coverage problem. Given a set of sensors deployed in a target area, we want to determine if the area is sufficiently *k-covered*, in the sense that every point in the target area is covered by at least k sensors, where k is a predefined constant. As a result, the aforementioned works [16, 19] can be regarded as a special case of this problem with $k = 1$. Applications requiring $k > 1$ may occur in situations where the

WSNA'03, September 19, 2003, San Diego, California, USA.

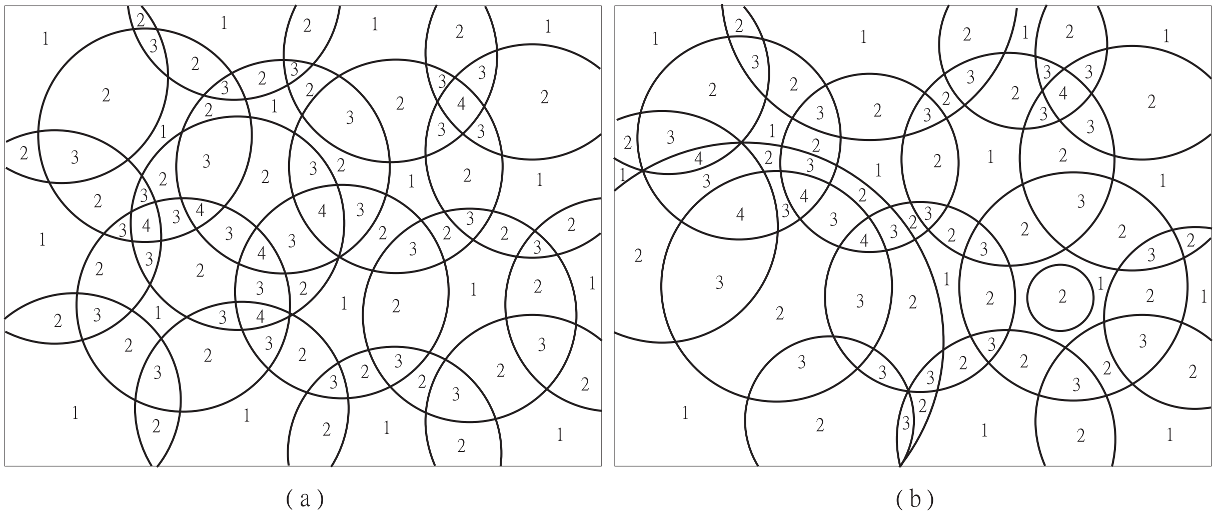

Figure 1: Examples of the coverage problem: (a) the sensing ranges are unit disks, and (b) the sensing ranges are non-unit disks. The number in each sub-region is its coverage.

stronger environmental monitoring is necessary, such as military applications. It also happens when multiple sensors are required to detect an event. For example, the triangulation-based positioning protocols [11, 12, 17] require at least three sensors (i.e., $k \geq 3$) at any moment to monitor a moving object. Enforcing $k \geq 2$ is also necessary for fault-tolerant purpose. In this paper, we propose a novel solution to determine whether a sensor network is k-covered. The sensing range of each sensor can be a unit disk or a non-unit disk. The solution can be easily translated to a distributed protocol where each sensor only needs to collect local information to make its decision. Instead of determining the coverage of each location, our approach tries to look at how the perimeter of each sensor's sensing range is covered, thus leading to an efficient polynomial-time algorithm. As long as the perimeters of sensors are sufficiently covered, the whole area is sufficiently covered.

The k-coverage problem can be further extended to solve several application-domain problems. In Section 4, we discuss how to use our results for discovering insufficiently covered areas, conserving energy, and supporting hot spots. At the end, we also show how to extend our results to situations where sensors' sensing regions are irregular.

This paper is organized as follows. Section 2 formally defines the coverage problems. Our solutions are presented in Section 3. Section 4 further discusses several possible extensions and applications of the proposed solutions. Section 5 draws our conclusions.

2. PROBLEM STATEMENT

We are given a set of sensors, $S = \{s_1, s_2, \ldots, s_n\}$, in a two-dimensional area A. Each sensor $s_i, i = 1..n$, is located at coordinate (x_i, y_i) inside A and has a sensing range of r_i, i.e., it can monitor any object that is within a distance of r_i from s_i.

DEFINITION 1. *A location in A is said to be covered by s_i if it is within s_i's sensing range. A location in A is said to be j-covered if it is within at least j sensors' sensing ranges.*

We consider two versions of the coverage problem as follows.

DEFINITION 2. *Given a natural number k, the $\underline{k}$-$\underline{N}$on-unit-disk $\underline{C}$overage (k-NC) Problem is a decision problem whose goal is to determine whether all points in A are k-covered or not.*

DEFINITION 3. *Given a natural number k, the $\underline{k}$-$\underline{U}$nit-disk $\underline{C}$overage (k-UC) Problem is a decision problem whose goal is to determine whether all points in A are k-covered or not subject to the constraint that $r_1 = r_2 = \cdots = r_n$.*

3. THE PROPOSED SOLUTIONS

At the first glance, the coverage problem seems to be very difficult. A naive solution is to find out all sub-regions divided by the sensing regions of all n sensors (i.e., n circles), and then check if each sub-region is k-covered or not, as shown in Fig. 1. Managing all sub-regions is a difficult and computationally expensive job in geometry because there could exist as many as $O(n^2)$ sub-regions divided by the circles. Also, it may be difficult to calculate these sub-regions.

3.1 The k-UC Problem

In the section, we propose a solution to the k-UC problem, which has a cost of $O(nd \log d)$, where d is the maximum number of sensors that may intersect a sensor. Instead of determining the coverage of each sub-region, our approach tries to look at how the perimeter of each sensor's sensing range is covered. Specifically, our algorithm tries to determine whether the perimeter of a sensor under consideration is sufficiently covered. By collecting this information from all sensors, a correct decision can be made.

DEFINITION 4. *Consider any two sensors s_i and s_j. A point on the perimeter of s_i is perimeter-covered by s_j if this point is within the sensing range of s_j.*

DEFINITION 5. *Consider any sensor s_i. We say that s_i is k-perimeter-covered if all points on the perimeter of s_i are perimeter-covered by at least k sensors other than s_i itself. Similarly, a segment of s_i's perimeter is k-perimeter-covered if all points on the segment are perimeter-covered by at least k sensors other than s_i itself.*

Below, we propose an $O(d \log d)$ algorithm to determine whether a sensor is k-perimeter-covered or not, where d is the number of sensors which have intersection with the former. Consider two sensors s_i and s_j located in positions (x_i, y_i) and (x_j, y_j), respectively. Denote by $d(s_i, s_j) = \sqrt{|x_i - x_j|^2 + |y_i - y_j|^2}$ the

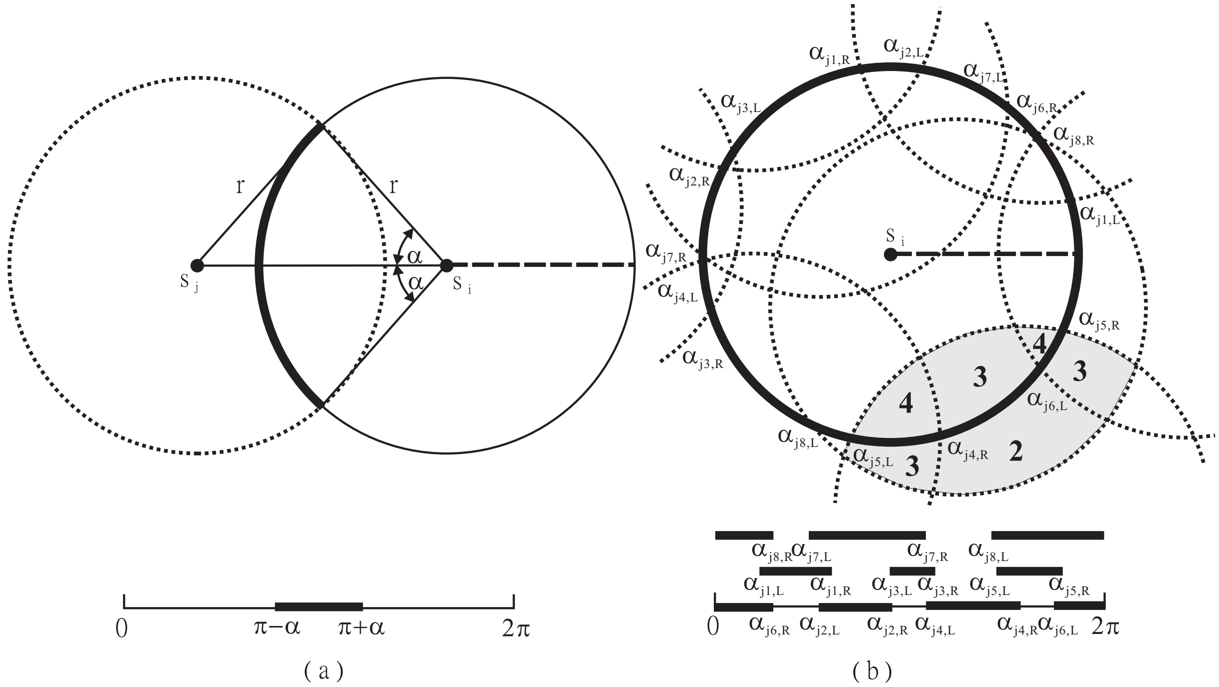

Figure 2: (a) Determining the segment of s_i's perimeter covered by s_j, and (b) determining the perimeter-coverage of s_i's perimeter.

distance between s_i and s_j. If $d(s_i, s_j) > 2r$, then s_j does not contribute any coverage to s_i's perimeter. Otherwise, the range of perimeter of s_i covered by s_j can be calculated as follows (refer to the illustration in Fig. 2(a)). Without loss of generality, let s_j be resident on the west of s_i (i.e., $y_i = y_j$ and $x_i > x_j$). The angle $\alpha = \arccos(\frac{d(s_i,s_j)}{2r})$. So the arch of s_i falling in the angle $[\pi - \alpha, \pi + \alpha]$ is perimeter-covered by s_j.

The algorithm to determine the perimeter coverage of s_i works as follows.

1. For each sensor s_j such that $d(s_i, s_j) \leq 2r$, determine the angle of $s_i' s$ arch, denoted by $[\alpha_{j,L}, \alpha_{j,R}]$, that is perimeter-covered by s_j.

2. For all neighboring sensors s_j of s_i such that $d(s_i, s_j) < 2r$, place the points $\alpha_{j,L}$ and $\alpha_{j,R}$ on the line segment $[0, 2\pi]$ and sort all these points in an ascending order into a list L. Also, properly mark each point as a left or right boundary of a coverage range.

3. (Sketched) Traverse the line segment $[0, 2\pi]$ by visiting each element in the sorted list L from the left to right and determine the perimeter-coverage of s_i.

Let d be the maximum number of sensors that are neighboring to a sensor ($d \leq n$). The complexities of steps 1 and 2 are $O(d)$ and $O(d \log d)$, respectively. The last step 3, though sketched, can be easily implemented as follows. Whenever an element $\alpha_{j,L}$ is traversed, the level of perimeter-coverage should be increased by one. Whenever an element $\alpha_{j,R}$ is traversed, the level of perimeter-coverage should be decreased by one. Since the sorted list L will divide the line segment $[0, 2\pi]$ into as many as $2d + 1$ segments, the complexity of step 3 is $O(d)$. So the overall complexity is $O(d \log d)$. An example is demonstrated in Fig. 2(b).

The above algorithm can determine the coverage of each sensor's perimeter at low cost. Below, we relate the perimeter-coverage property of sensors to the coverage property of the network area.

LEMMA 1. *Suppose that no two sensors are located in the same location. Consider any segment of a sensor s_i that divides two sub-regions in the network area A. If this segment is k-perimeter-covered, the sub-region that is outside s_i's sensing range is k-covered and the sub-region that is inside s_i's sensing range is $(k+1)$-covered.*

PROOF. The proof is directly from Definition 5. Since the segment is k-perimeter-covered, the sub-region outside s_i's sensing range is also k-covered by the continuity of the sub-region. The sub-region inside s_i's sensing range is $(k + 1)$-covered because it is also covered by s_i. □

For example, the gray areas in Fig. 2(b) illustrate how the above lemma works .

THEOREM 1. *Suppose that no two sensors are located in the same location. The whole network area A is k-covered iff each sensor in the network is k-perimeter-covered.*

PROOF. For the "if" part, observe that each sub-region inside A is bounded by at least one segment of a sensor s_i's perimeter. Since s_i is k-perimeter-covered, by Lemma 1, this sub-region is either k-covered or $(k + 1)$-covered, which proves the "if" part.

For the "only if" part, it is clear by definition that for any segment of a sensor s_i's perimeter that divides two sub-regions, both these sub-regions are at least k-covered. Further, observe that the sub-region that is inside s_i's sensing range must be covered by one more sensor, s_i, and is thus at least $(k + 1)$-covered. So excluding s_i itself, this segment is perimeter-covered by at least k sensors other than s_i itself, which proves the "only if" part. □

Below, we comment on several special cases which we leave unaddressed on purpose for simplicity in the above discussion. When two sensors s_i and s_j fall in exactly the same location, Lemma 1 will not work because for any segment of s_i and s_j that divides two sub-regions in the network area, a point right inside s_i's and s_j's sensing ranges and a point right outside their sensing ranges

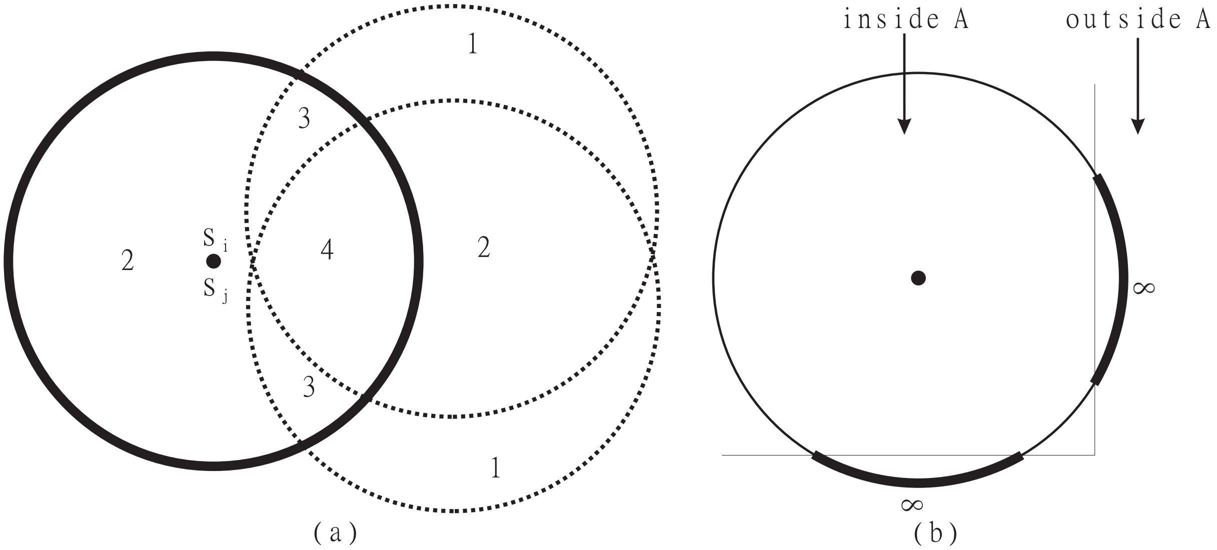

Figure 3: Some special cases: (a) two sensors falling in the same location (the number in each sub-region is its level of coverage), and (b) the sensing range of a sensor exceeding the network area A.

will differ in their coverage levels by two, making Lemma 1 incorrect (refer to the illustration in Fig. 3(a)). Other than this case, all neighboring sub-regions in the network will differ in their coverage levels by exactly one. Since in most applications we are interested in areas that are insufficiently covered, one simple remedy to this problem is to just ignore one of the sensors if both sensors fall in exactly the same location. Another solution is to first run our algorithm by ignoring one sensor, and then increase the coverage levels of the sub-regions falling in the sensor's range by one afterward. The other boundary case is that some sensors' sensing ranges may exceed the network area A. In this case, we can simply assign the segments falling outside A as as ∞-perimeter-covered, as shown in Fig. 3(b).

3.2 The k-NC Problem

For the non-unit-disk coverage problem, sensors' sensing ranges could be different. However, most of the results derived above remain the same. Below, we summarize how the k-NC problem is solved.

First, we need to define the how the perimeter of a sensor's sensing range is covered by other sensors. Consider two sensors s_i and s_j located in positions (x_i, y_i) and (x_j, y_j) with sensing ranges r_i and r_j, respectively. Again, without loss of generality, let s_j be resident on the west of s_i. We address how s_i is perimeter-covered by s_j. There are two cases to be considered.

Case 1: Sensor s_j is outside the sensing range of s_i, i.e., $d(s_i, s_j) > r_i$.

(i) If $r_j < d(s_i, s_j) - r_i$, then s_i is not perimeter-covered by s_j

(ii) If $d(s_i, s_j) - r_i \le r_j \le d(s_i, s_j) + r_i$, then the arch of s_i falling in the angle $[\pi - \alpha, \pi + \alpha]$ is perimeter-covered by s_j, where α can be derived from the formula:

$$r_j^2 = r_i^2 + d(s_i, s_j)^2 - 2r_i \cdot d(s_i, s_j) \cdot \cos(\alpha). \quad (1)$$

(iii) If $r_j > d(s_i, s_j) + r_i$, then the whole range $[0, 2\pi]$ of s_i is perimeter-covered by s_j.

Case 2: Sensor s_j is inside the sensing range of s_i, i.e., $d(s_i, s_j) \le r_i$.

(i) If $r_j < r_i - d(s_i, s_j)$, then s_i is not perimeter-covered by s_j

(ii) If $r_i - d(s_i, s_j) \le r_j \le r_i + d(s_i, s_j)$, then the arch of s_i falling in the angle $[\pi - \alpha, \pi + \alpha]$ is perimeter-covered by s_j, where α is as defined in Eq. (1).

(iii) If $r_j > d(s_i, s_j) + r_i$, then the whole range $[0, 2\pi]$ of s_i is perimeter-covered by s_j.

The above cases are illustrated in Fig. 4. Based on such classification, the same algorithm to determine the perimeter coverage of a sensor can be used. Lemma 1 and Theorem 1 still hold true (observe that in the corresponding proofs, we do not use any property about the absolute sensing ranges of sensors). So the k-NC problem can also be solved at a time complexity of $O(nd \log d)$, except that the neighbors of a sensor need to be redefined.

4. APPLICATIONS AND EXTENSIONS OF THE COVERAGE PROBLEM

The sensor coverage problem, although modeled as a decision problem, can be extended further in several ways for many interesting applications. The proposed results can also be extended for more realistic situations. In the following, we suggest several applications of the coverage problem and possible extensions of our results.

4.1 Discovering Insufficiently Covered Regions

For a sensor network, one basic question is whether the network area is fully covered. Our modeling of the k-UC and k-NC problems can solve the sensor coverage problem in a more general sense by determining if the network area is k-covered or not. A larger k can support a more fine-grained sensibility. For example, if $k = 1$, we can only detect in which sensor an event has happened. Using a larger k, the location of the event can be reduced to a certain intersection of at least k sensors. Thus, the location of the event can be more precisely defined. This would support more fine-grained location-based services.

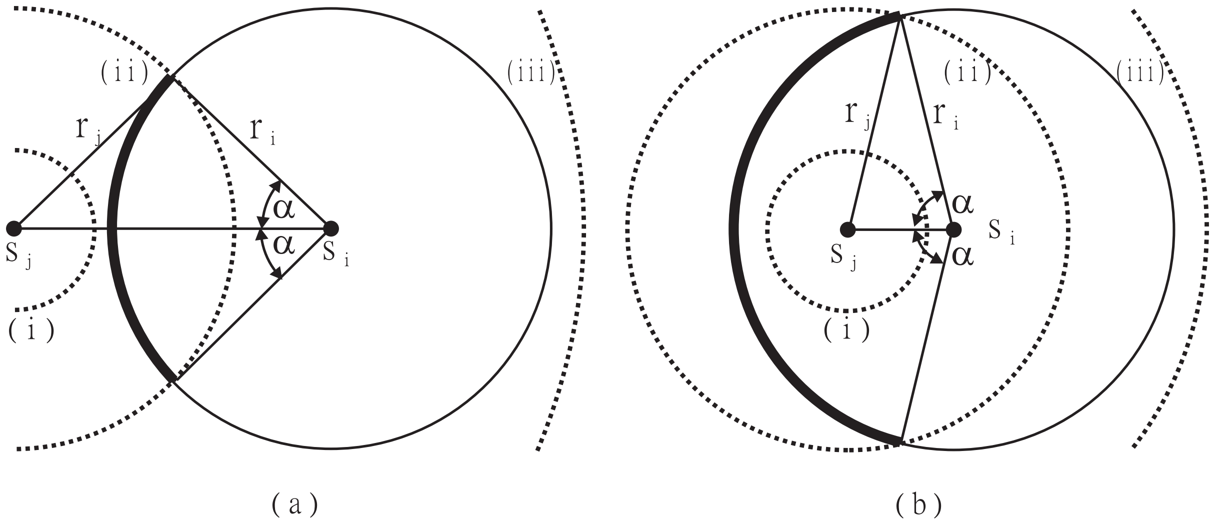

Figure 4: The coverage relation of two sensors with different sensing ranges: (a) s_j not in the range of s_i, and (b) s_j in the range of s_i.

To determine which areas are insufficiently covered, we assume that there is a central controller in the sensor network. The central controller can broadcast the desired value of k to all sensors. Each sensor can then communicate with its neighboring sensors and then determine which segments of its perimeter are less than k-perimeter-covered. The results (i.e., segments) are then sent back to the central controller. By putting all segments together, the central controller can precisely determine which areas are less than k-covered. Note that since Theorem 1 provides a necessary and sufficient condition to determine if an area in the network is k-covered, false detection would not happen.

Further actions can then be taken if certain areas are insufficiently covered. For example, the central controller can dispatch more sensors to these regions. However, the k-UC and k-NC problems are formulated as decision problems, which can only answer a yes/no question. A more general optimization problem is: how can we patch these insufficiently covered areas with the least number of extra sensors. This is still an open question and deserves further investigation.

Another interesting open question is the "granularity versus cost" issue. We would partition the network area A into sub-regions that are as fine-grained as possible by using as least sensors as possible. One possibility to capture the notion is to define a cost metric $C = n \times (\text{area of the largest sub-region})$ and the goal is to minimize C. This will be directed to our future research.

4.2 Power Saving in Sensor Networks

Contrary to the insufficient coverage issue, a sensor network may be overly covered by too many sensors in certainly areas. For example, as suggested in [16], if there are more sensors than necessary, we may turn off some redundant nodes to save energy. These sensors may be turned on later on when other sensors run out of energy. Reference [16] proposes a node-scheduling scheme to guarantee that the level of coverage of the network area after turning off some redundant sensors remains the same.

Based on our result, we can solve a more general problem as follows. First, those sensor nodes who can be turned off, called *candidates*, need to be identified. A sensor s_i is a candidate if all its neighbors are still k-perimeter-covered after s_i is removed. To do so, s_i can communicate with each of its neighbors and ask them to reevaluate their perimeter coverage by skipping s_i. If the responses from all its neighbors are positive, s_i is a candidate. After determining the candidates, each sensor can compete to enter the doze mode by running a scheduling scheme, such as that in [16], to decide how long it can go to sleep. Note that our scheme could find more candidates compared to that in [16]. Moreover, [16] only considers a special case of our results such that $k = 1$.

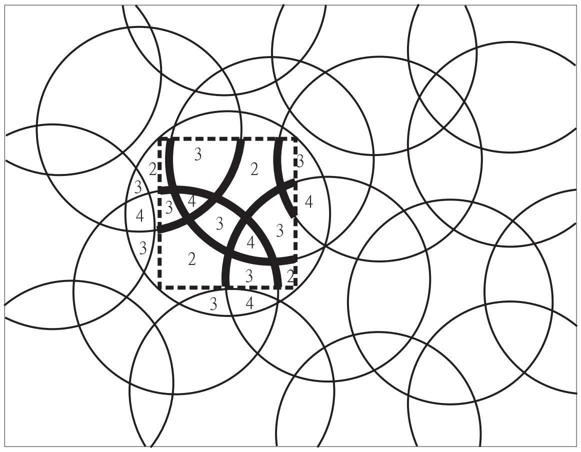

Figure 5: An example of verifying whether a hot spot (shown in dashed rectangle) is 2-covered or not.

4.3 Hot Spots

It is possible that some areas in the network are more important than other areas and need to be covered by more sensors. Those important regions are called *hot spots*. Our solutions can be directly applied to check whether a hot spot area is k-covered or not. Given a hot spot, only those sensors whose perimeters are within or have crossings with the hot spot need to be checked. So the central controller can issue a request by identifying the hot spot. Each sensor that is within the hot spot or has crossings with the hot spot needs to reevaluate the coverage of its perimeter segment that is within the hot spot. The results in Lemma 1 and Theorem 1 are directly

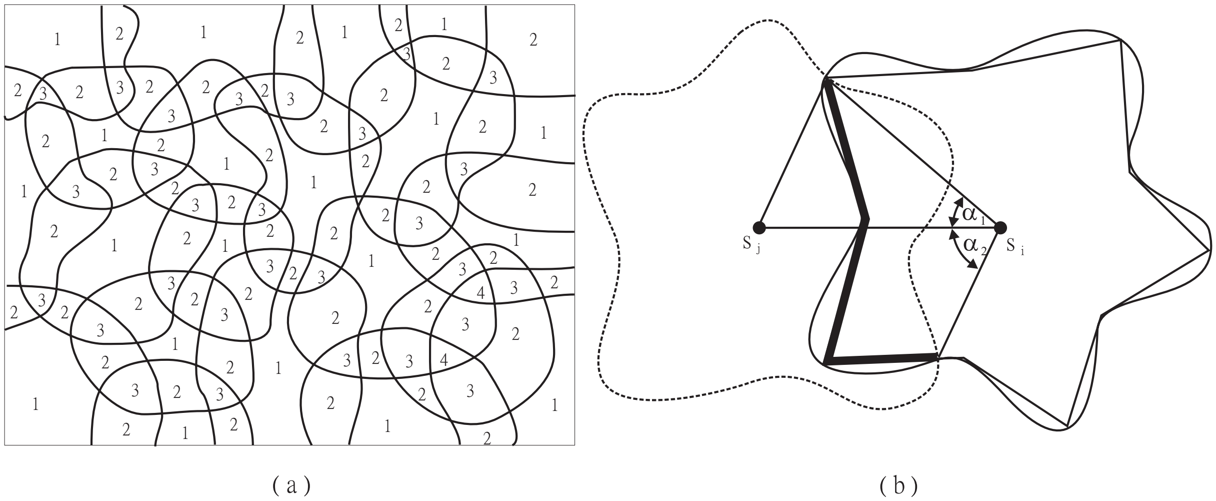

Figure 6: The coverage problem with irregular sensing regions: (a) coverage levels of irregular sub-regions, and (b) polygon approximation of a sensor's sensing region.

applicable. So a hot spot is k-covered if and only if all perimeter segments within this hot spot are k-perimeter-covered. An example to verify if a hot spot is 2-covered is shown in Fig. 5. Note that a hot spot can be defined in other shapes too.

4.4 Extension to Irregular Sensing Regions

The sensing region of a sensor is not necessarily a circle. In most cases, it is location-dependent and likely irregular.[1] Fortunately, our results can be directly applied to irregular sensing regions without problem, under the condition that each sensor's sensing region can be precisely defined. Observe that the sensing regions of sensors still divide the network area into sub-regions. Through Lemma 1, we can translate perimeter-covered property of sensors to area-covered property of the network. Then by Theorem 1, we can decide whether the network is k-covered. Fig. 6(a) shows an example.

Given two sensors' sensing regions that are irregular, it remains a problem how to determine the perimeter coverage relation of these two sensors. To do so, we may use polygon approximation. The idea is illustrated in Fig. 6(b).

5. CONCLUSION

In this paper, we have proposed solutions to two versions of the coverage problem, namely k-UC and k-NC, in a wireless sensor network. We model the coverage problem as a decision problem, whose goal is to determine whether each location of the target sensing area is sufficiently covered or not. Rather than determining the level of coverage of each location, our solutions are based on checking the perimeter of each sensor's sensing range. Although the problem seems to be very difficult at the first glance, our scheme can give an exact answer in $O(nd \log d)$ time. With the proposed techniques, we also discuss several applications (such as discovering insufficiently covered regions and saving energies) and extensions (such as scenarios with hot spots and irregular sensing ranges) of our results. We are currently working on these applications and extensions, and the related results will be reported in our future papers.

6. ACKNOWLEDGMENT

This work is co-sponsored by the MOE Program for Promoting Academic Excellence of Universities under grant numbers A-91-H-FA07-1-4 and 89-E-FA04-1-4.

7. REFERENCES

[1] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *IEEE INFOCOM*, pages 775–784, 2000.

[2] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. In *ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[3] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low cost outdoor localization for very small devices. *IEEE Personal Commun.*, 7(5):28–34, Oct. 2000.

[4] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. *Commun. ACM*, 43(5):51–58, May 2000.

[5] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly resilient, energy efficient multipath routing in wireless sensor networks. *ACM Mobile Comput. and Commun. Review*, 5(4):11–25, Oct. 2001.

[6] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocols for wireless microsensor networks. In *Hawaii Int'l Conf. on Systems Science (HICSS)*, 2000.

[7] J. O'Rourke. Computational geometry column 15. *Int'l Journal of Computational Geometry and Applications*, 2(2):215–217, 1992.

[8] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *IEEE INFOCOM*, pages 1380–1387, 2001.

[9] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak. Exposure in wireless ad-hoc sensor networks. In *ACM Int'l Conf. on Mobile Computing and Networking (MobiCom)*, pages 139–150, 2001.

[10] S. Meguerdichian, S. Slijepcevic, V. Karayan, and M. Potkonjak. Localized algorithms in wireless ad-hoc networks: location discovery and sensor exposure. In *ACM Int'l Symp. on Mobile Ad Hoc Networking and Computing (MobiHOC)*, pages 106–116, 2001.

[1] The sensing region of a sensor may even be time-varying, in which case frequent reevaluation of the sensing region would be necessary. This issue is beyond the scope of this work.

[11] D. Nicules and B. Nath. Ad-hoc positioning system (APS) using AoA. In *IEEE INFOCOM*, 2003.

[12] A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *ACM Int'l Conf. on Mobile Computing and Networking (MobiCom)*, pages 166–179, 2001.

[13] E. Shih, S.-H. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan. Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks. In *ACM Int'l Conf. on Mobile Computing and Networking (MobiCom)*, pages 272–287, 2001.

[14] S. Slijepcevic and M. Potkonjak. Power efficient organization of wireless sensor networks. In *IEEE Int'l Conf. on Communications (ICC)*, pages 472–476, 2001.

[15] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J. Pottie. Protocols for self-organization of a wireless sensor network. *IEEE Personal Commun.*, 7(5):16–27, Oct. 2000.

[16] D. Tian and N. D. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[17] Y.-C. Tseng, S.-P. Kuo, H.-W. Lee, and C.-F. Huang. Location tracking in a wireless sensor network by mobile agents and its data fusion strategies. In *Int'l Workshop on Information Processing in Sensor Networks (IPSN)*, 2003.

[18] A. Woo and D. E. Culler. A transmission control scheme for media access in sensor networks. In *ACM Int'l Conf. on Mobile Computing and Networking (MobiCom)*, pages 221–235, 2001.

[19] F. Ye, G. Zhong, S. Lu, and L. Zhang. PEAS: A robust energy conserving protocol for long-lived sensor networks. In *Int'l Conf. on Distributed Computing Systems (ICDCS)*, 2003.

[20] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *IEEE INFOCOM*, pages 1567–1576, 2002.