Open access • Journal Article • DOI:10.1007/S11036-005-1564-Y

The coverage problem in a wireless sensor network — Source link

Chi-Fu Huang, Yu-Chee Tseng

Institutions: National Chiao Tung University

Published on: 01 Aug 2005 - Mobile Networks and Applications (Springer-Verlag New York, Inc.)

Topics: Visual sensor network, Brooks–Iyengar algorithm, Wireless sensor network, Key distribution in wireless sensor networks and Mobile wireless sensor network

Mobile Networks and Applications 10, 519–528, 2005

The Coverage Problem in a Wireless Sensor Network

CHI-FU HUANG and YU-CHEE TSENG*
Department of Computer Science and Information Engineering, National Chiao-Tung University, 1001 Ta Hsueh Road, Hsin-Chu, 30050, Taiwan

Abstract. One of the fundamental issues 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 given parameter. 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 determining insufficiently covered areas in a sensor network, enhancing fault-tolerant capability in hostile regions, and conserving energies of redundant sensors in a randomly deployed network. Our solutions can be easily translated to distributed protocols to solve the coverage problem.

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 [13,17]. The flexibility of installing and configuring a sensor network is thus greatly improved. Recently, a lot of research activities have been dedicated to sensor networks, including design issues related to the physical and media access layers [15,22,24] and routing and transport protocols [3,5,7]. Localization and positioning applications of wireless sensor networks are discussed in [2,4,11,14,19].

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 [12]. Reference [8] proposes polynomial time algorithms to find the *maximal breach path* and the *maximal support path* that are least and best monitored in the sensor network. How to find the *minimal and maximal exposure path* that takes the duration that an object is monitored by sensors is addressed in [9,20]. 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 may be arbitrarily deployed, 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). Slijepcevic and Potkonjak [16] proposes a heuristic to select mutually exclusive sets of sensor nodes such that each set of sensors can provide a complete coverage of the monitored area. Also targeted at turning off some redundant nodes, Ye et al. [23] 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 [18] 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 given parameter. As a result, the aforementioned works [18,23] can be regarded as a special case of this problem with $k = 1$. Applications requiring $k > 1$ may occur in situations where a stronger environmental monitoring capability is desired, such as military applications. It also happens when multiple sensors are required to detect an event. For example, the triangulation-based positioning protocols [13,14,19] require at least three sensors (i.e., $k \geq 3$) at any moment to monitor a moving object. Enforcing $k \geq 2$ is also desirable for fault-tolerant purpose. The work [21] also considers the same coverage problem combined with the

A preliminary version of this paper has appeared in the Workshop on Wireless Sensor Networks and Applications, 2003, San Diego, CA, USA.
*Corresponding author.

communication connectivity issue. However, it incurs higher computational complexity to determine a network's coverage level as compared to the solution proposed in this paper. The *arrangement* issue [1,6], which is widely studied in combinatorial and computational geometry, also considers how a finite collection of geometric objects decomposes a space into connected elements. However, to construct arrangements, only *centralized* algorithms are proposed in the literature, whilst what we need for a wireless sensor network is a distributed solution. The solutions proposed in this paper can be easily translated to distributed protocols where each sensor only needs to collect local information to make its decision.

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. Rather than 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. Note that this step can be executed by each sensor based on location information of its neighbors. This can lead to an efficient distributed solution. 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 5, 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 presents our simulation results and demonstrates a tool that we implemented to solve the k-coverage problem. Section 5 further discusses several possible extensions and applications of the proposed solutions. Section 6 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, $i = 1, \ldots, n$, is located at coordinate (x_i, y_i) inside A and has a sensing range of r_i, i.e., it can monitor any point 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.

Definition 2. A *sub-region* in A is a set of points who are covered by the same set of sensors.

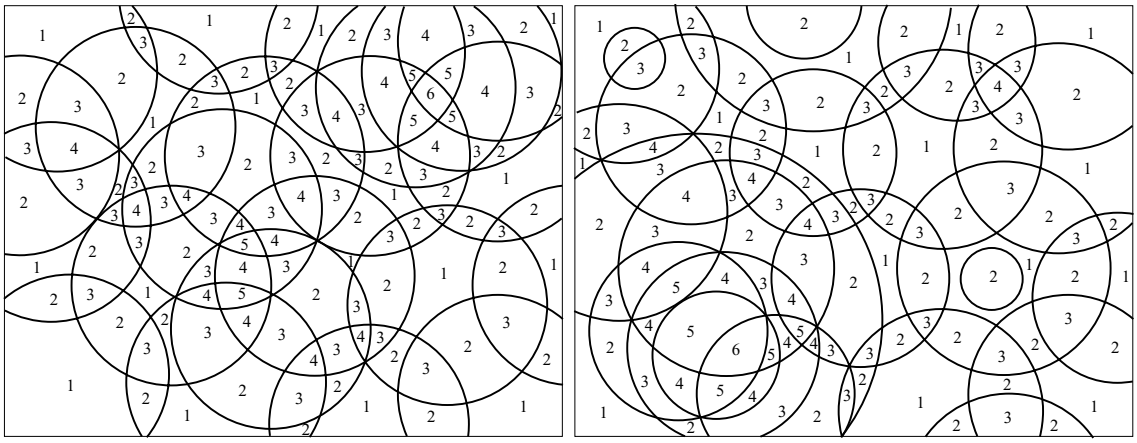
We consider two versions of the coverage problem as follows.

Definition 3. Given a natural number k, the *k-Non-unit-disk Coverage (k-NC) Problem* is a decision problem whose goal is to determine whether all points in A are k-covered or not.

Definition 4. Given a natural number k, the *k-Unit-disk Coverage (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. One naive solution is to find out all sub-regions divided by the sensing boundaries of all n sensors (i.e., n circles), and then check if each sub-region is k-covered or not, as shown in figure 1. Managing all sub-regions could be a difficult and computationally expensive job in geometry. There may exit as many as $O(n^2)$ sub-regions divided by the circles. Also, it may be difficult to calculate these sub-regions.



(a) (b)

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.

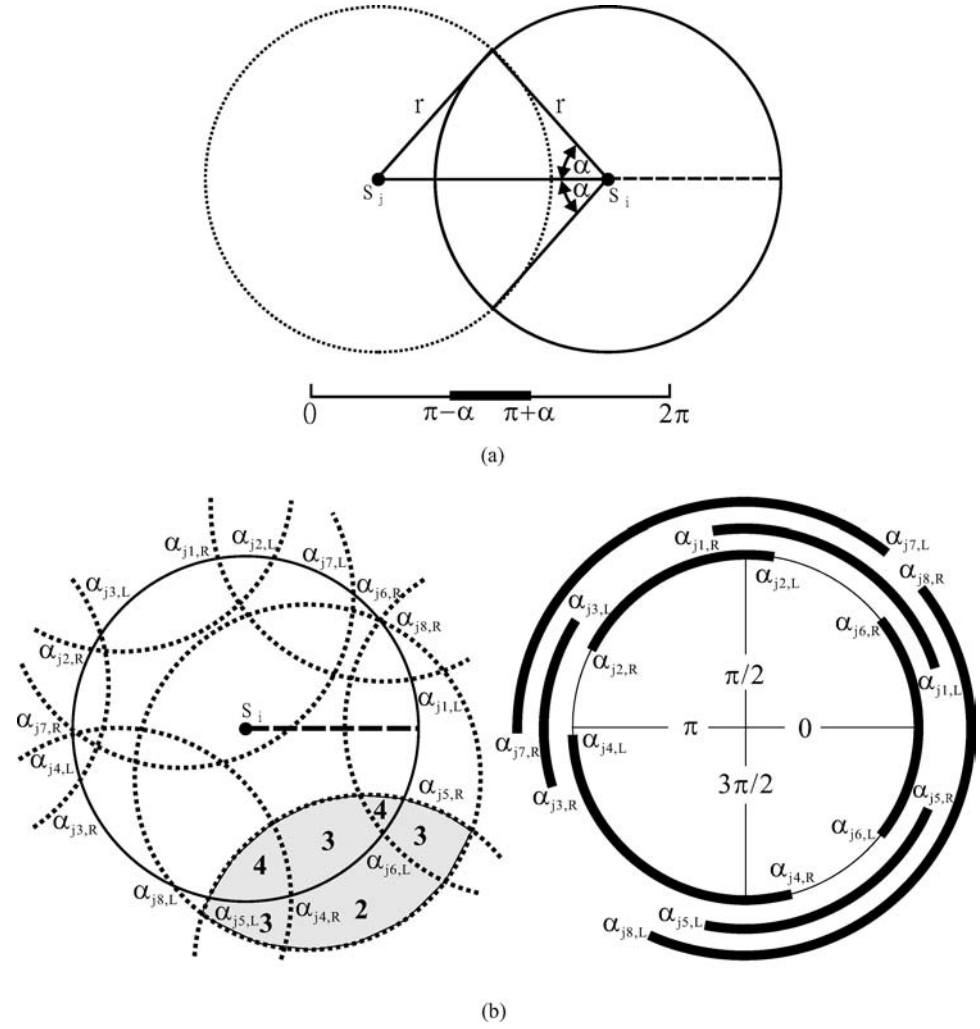


Figure 2. Determining: (a) the segment of s_i's perimeter covered by s_j, and (b) the perimeter-coverage of s_i's perimeter.

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 whose sensing ranges may intersect a sensor's sensing range. 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 answer can be obtained.

Definition 5. 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 6. 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. 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 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 figure 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.

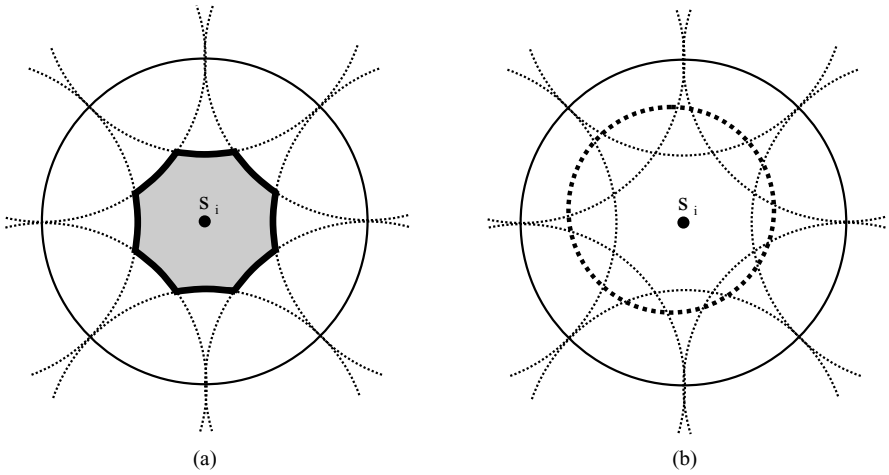


Figure 3. Some examples to utilize the result in Theorem 1.

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

The above algorithm can determine the coverage of each sensor's perimeter efficiently. 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 6. Since the segment is k-perimeter-covered, the sub-region outside s_i's sensing range is also k-covered due to 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. □

An example is demonstrated in figure 2(b). The gray areas in figure 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. □

Note that Theorem 1 is true when *all* sensors are claimed to be k-perimeter-covered. When a specific sensor s_i is k-perimeter-covered, it only guarantees that each point right outside s_i's perimeter is k-covered, and each point right inside s_i's perimeter is $(k + 1)$-covered. However, it does not guarantee that *all* points inside s_i's perimeter is $(k + 1)$-covered. An example is shown in figure 3. In figure 3(a), sensor s_i is 2-perimeter-covered since each segment of its perimeter is covered by two sensors. This only implies the coverage levels of the points nearby the perimeter of s_i. The gray area, which is outside the coverage of s_i's neighboring sensors, is only 1-covered. In fact, the segments that bound the gray area are only 1-perimeter-covered. If we add another sensor to cover these segments (shown in thick dotted line) as shown in figure 3(b), then s_i's sensing region will be 2-covered.

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 will differ in their coverage levels by two, making Lemma 1 incorrect (refer to the illustration in figure 4(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

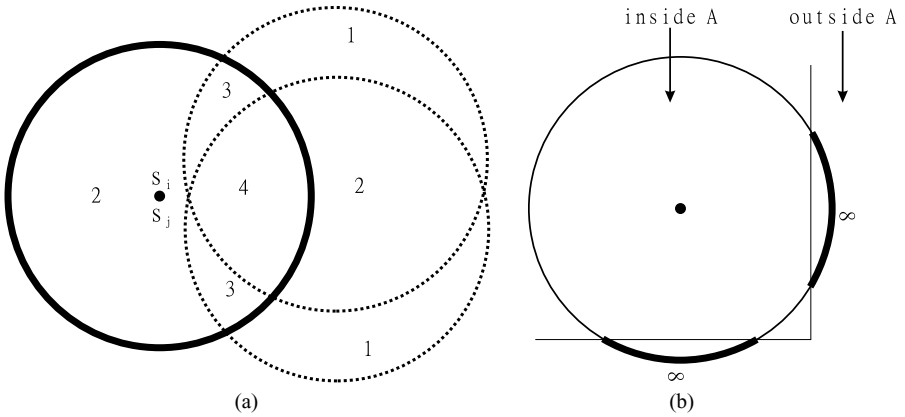


Figure 4. Some special cases: (a) two sensors falling in the same location (the number in each sub-region is its level of coverage), and (b) the sensing range of a sensor exceeding the network area A.

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 ignored 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 ∞-perimeter-covered, as shown in figure 4(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 that 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 equation (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 figure 5. Based on such classification, the same algorithm to determine the perimeter coverage of a sensor can be used. Lemma 1 and Theorem 1

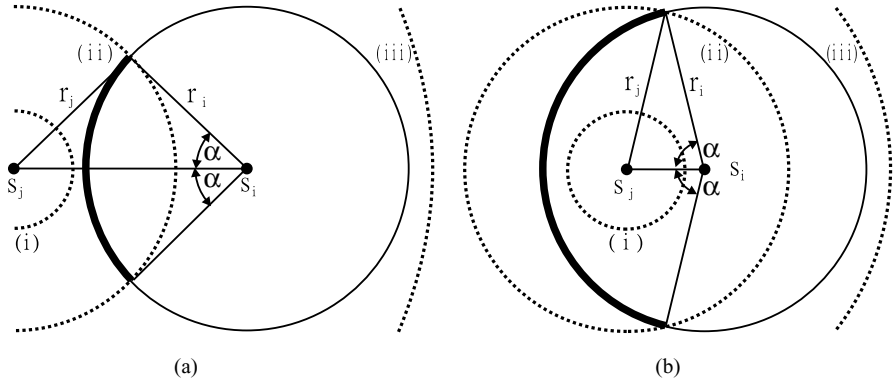


Figure 5. The coverage relation of two sensors with different sensing ranges: (a) s_j not in the range of s_i, and (b) s_j in the range of s_i.

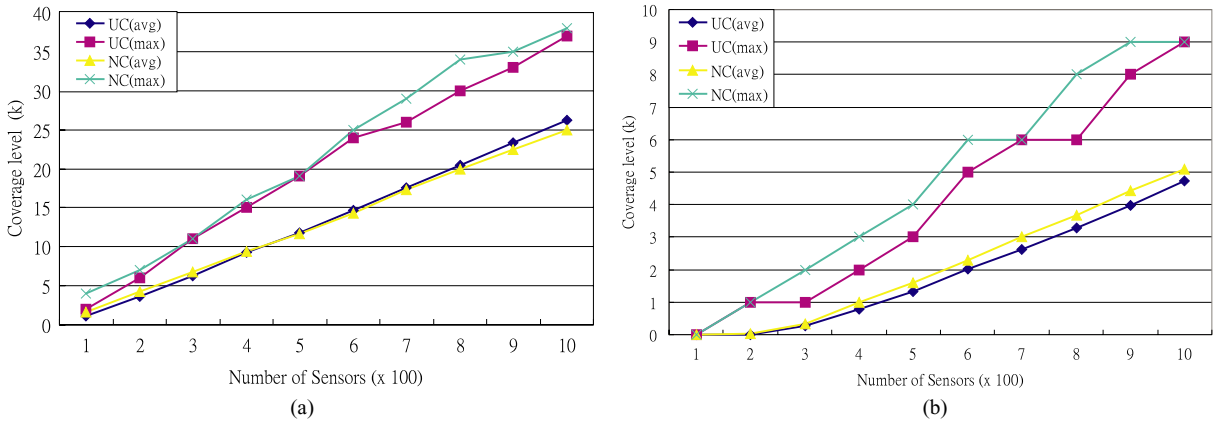


Figure 6. Number of sensors v.s. coverage level for sensor fields of sizes: (a) 500 × 500, and (b) 1000 × 1000.

still hold true (observe that in the corresponding proofs, we do not use any property about the absolute sensing ranges of sensors).

3.3. Complexity analysis

Consider the algorithm in Section 3.1. 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 complexity to determine a sensor's perimeter coverage is $O(d \log d)$. The overall complexity for the k-UC problem is thus $O(nd \log d)$. The k-NC problem can also be solved with complexity $O(nd \log d)$, except that the neighbors of a sensor need to be redefined. The work [21] also proposes a solution to determine the coverage level of a sensor network. It looks at how intersection points between sensors' sensing ranges are covered. Since there are as many as $O(n^2)$ intersection points in the network and the calculation of the coverage level of each intersection point takes time $O(n)$, the overall complexity is $O(n^3)$.

4. Simulation results and a sensor coverage toolkit

We have developed a simulator and implemented a toolkit based on the proposed algorithms. Square sensor fields are simulated with randomly placed nodes. There are two settings of sensing ranges: unit-disc sensing range and non-unit-disc sensing range. All results presented below are from the average of at least 1000 runs.

First, we investigate the level of coverage (i.e., k) that can be achieved by using different numbers of sensors. Sensor fields of sizes 500 × 500 and 1000 × 1000 are simulated with 100–1000 nodes. The unit-disc sensing range is 100 units and the non-unit-disc sensing range falling uniformly between 50–150 units. Both the average and the maximum levels of coverage are evaluated. The results are in figure 6. As can be seen, the average value of k grows about linearly as the number of sensors increases.

Next, we investigate the level of coverage that can be achieved by setting different sensing ranges of sensors. Sensor fields of sizes 500 × 500 and 1000 × 1000 are simulated with 500 nodes. For the unit-disc case, the sensing range is fixed from 50 to 150 units. For the non-unit-disc case, we first pick an average sensing range avg, and the sensors' sensing ranges are uniformly distributed between $avg - 50$ and $avg + 50$. The results are in figure 7. The average value of k grows as the average sensing range of sensors increases.

We have also implemented a toolkit based on the proposed algorithms to determine the coverage level of a given sensing field. Figure 8 shows the user interface of the toolkit. In the drawing area, one can easily deploy sensors by pointing out their locations and dragging their sensing ranges. By clicking on the "Deploy" button, the deployment of sensors will be fed into our program. There are three major functions of this toolkit, as described below.

1. *Compute the Level of Coverage*: By clicking on the "Compute Coverage" button and then the "Display Coverage" button, the system will calculate and return the current coverage level of the whole area, as illustrated in figure 9(a).
2. *Color the Drawing Area*: By clicking on the "Paint the drawing area" button, the drawing area will be colored based on each region's coverage level. The coloring speed can also be modified, which will reflect on the coloring quality. An example is shown in figure 9(b).
3. *Display Insufficiently Covered Segments*: One can first select the desired value of k followed by clicking on the "Commit" button to feed k into the system. Clicking on the "Get Low Coverage Segments" button will generate an output file which contains all segments that are

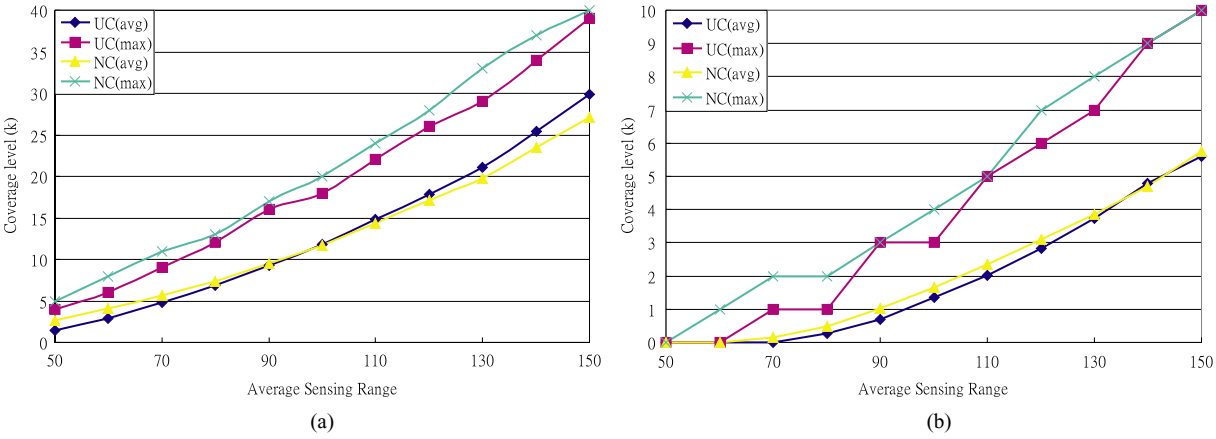


Figure 7. Sensing range v.s. coverage level for sensor fields of sizes: (a) 500 × 500, and (b) 1000 × 1000.

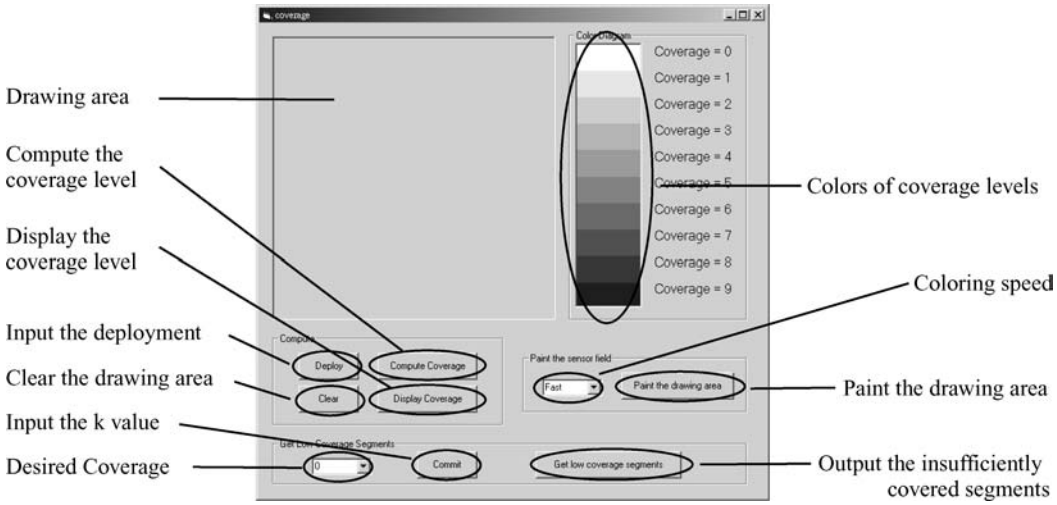


Figure 8. Functional descriptions of the toolkit.

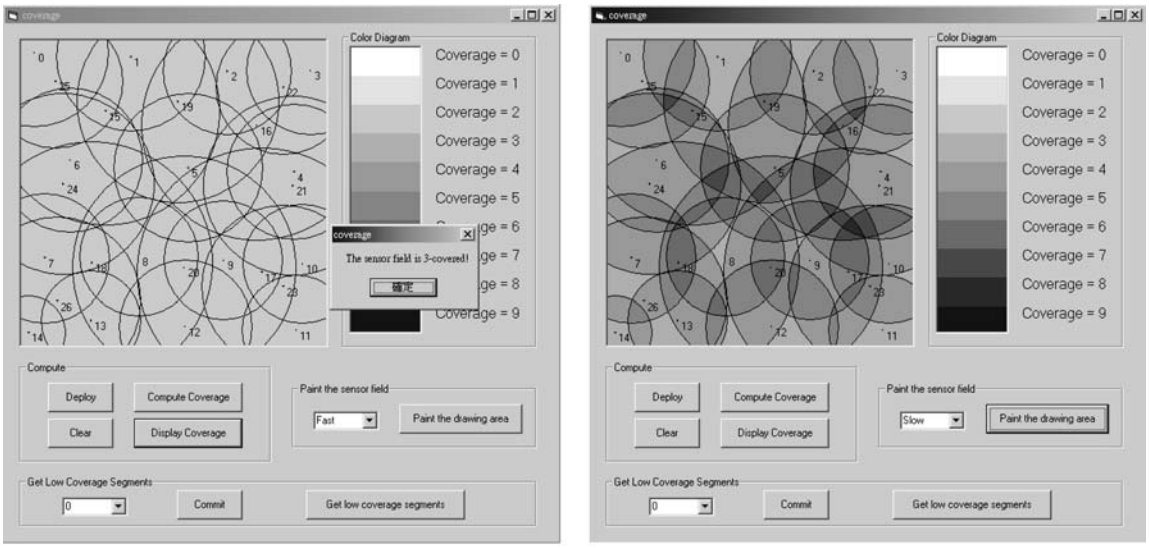


Figure 9. Execution results of the toolkit: (a) coverage level, and (b) painting results.

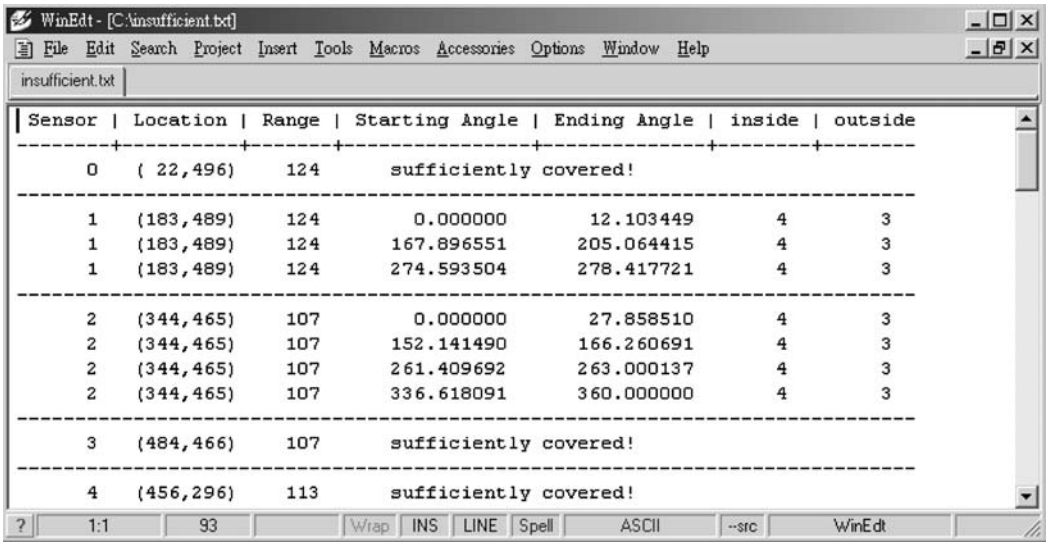


Figure 10. Insufficiently 4-perimeter-covered segments for the example in figure 9.

insufficiently k-perimeter-covered, as shown in the figure 10. Each line in the file is a segment of one sensor's perimeter that is insufficiently covered. Fields in a line include: sensor ID, location, sensing range, starting and ending angles of the corresponding segment, and the levels of coverage inside and outside this segment.

This toolkit is publicly downloadable from http://hscc.csie.nctu.edu.tw/download/coverage.zip.

5. 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.

5.1. 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.

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., insufficiently covered 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. An 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.

5.2. Power saving in sensor networks

Contrary to the insufficient coverage issue, a sensor network may be overly covered by too many sensors in certain areas. For example, as suggested in [18], if there are more sensors than necessary, we may turn off some redundant nodes to save energy. These sensors may be turned on later when other sensors run out of energy. Tian and Georganas [18] 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 of 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 [18], to decide how long it

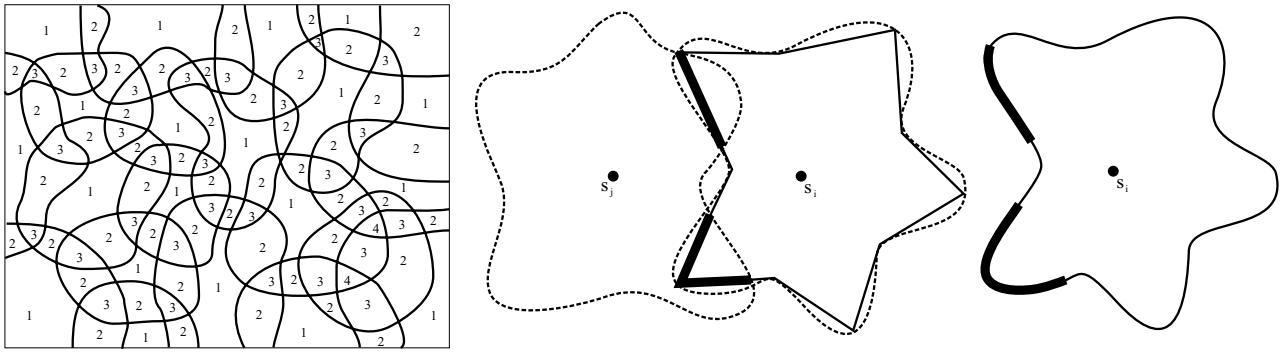


Figure 11. The coverage problem with irregular sensing regions: (a) coverage levels of irregular sub-regions, (b) polygon approximation of sensor s_i's sensing region, and (c) covered segments of s_i.

can go to sleep. However, [18] only considers a special case of our results with $k = 1$.

5.3. 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 applicable. So a hot spot is k-covered if and only if all perimeter segments within this hot spot are k-perimeter-covered. Note that a hot spot can be defined in other shapes too.

5.4. 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, assuming 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. figure 11(a) shows an example.

Given two sensors' sensing regions that are irregular, it remains a problem how to determine the intersections of their perimeters. One possibility is to conduct polygon approximation. The idea is illustrated in figure 11(b), which can give the perimeter coverage in figure 11(c).

[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.

6. Conclusions

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. A software tool that implements the proposed algorithms is available on the web (http://hscc.csie.nctu.edu.tw/download/coverage.zip) for free download.

Acknowledgments

The authors would like to thank Li-Chu Lo for implementing the simulation and toolkit mentioned in Section 4. Y.C. Tseng's research is co-sponsored by the MOE Program for Promoting Academic Excellence of Universities, by NSC of Taiwan under grant numbers NSC92-2213-E009-076 and NSC92-2219-E009-013, by Computer and Communications Research Labs., ITRI, Taiwan, by Intel Inc., by the Institute for Information Industry and MOEA, R.O.C, under the Handheld Device Embedded System Software Technology Development Project, by the Lee and MTI Center of NCTU, and by Chung-Shan Institute of Science and Technology under contract number BC93B12P.

References

[1] P.K. Agarwal and M. Sharir, Arrangements and their applications. in: *Handbook of Computational Geometry*, eds. J.-R. Sack and J. Urrutia, (Elsevier, North-Holland, New York, 2000) pp. 49–119.

[2] P. Bahl and V.N. Padmanabhan, RADAR: an in-building RF-based user location and tracking system, in: *IEEE INFOCOM* (2000) pp. 775–784
[3] D. Braginsky and D. Estrin, Rumor routing algorithm for sensor networks, in: *ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA)* (2002).
[4] N. Bulusu, J. Heidemann and D. Estrin, GPS-less low cost outdoor localization for very small devices, IEEE Personal Commun. 7(5) (2000) 28–34.
[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) (2001) 11–25.
[6] D. Halperin, Arrangements, in: *Handbook of Discrete and Computational Geometry*, eds. J.E. Goodman and J. O'Rourke, chapter 21, (CRC Press LLC, Boca Raton, FL, 1997) pp. 389–412.
[7] 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).
[8] S. Meguerdichian, F. Koushanfar, M. Potkonjak and M.B. Srivastava, Coverage problems in wireless ad-hoc sensor networks. in: *IEEE INFOCOM* (2001) pp. 1380–1387.
[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)* (2001) pp. 139–150
[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)* (2001) pp. 106–116.
[11] D. Nicules and B. Nath, Ad-hoc positioning system (APS) using AoA, in: *IEEE INFOCOM* (2003).
[12] J. O'Rourke, Computational geometry column 15, Int'l Journal of Computational Geometry and Applications 2(2) (1992) 215–217.
[13] G.J. Pottie and W.J. Kaiser, Wireless integrated network sensors, Commun. ACM 43(5) (2000) 51–58.
[14] 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)* (2001) pp. 166–179.
[15] 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)* (2001) pp. 272–287.
[16] S. Slijepcevic and M. Potkonjak, Power efficient organization of wireless sensor networks, in: *IEEE Int'l Conf. on Communications (ICC)* (2001) pp. 472–476.
[17] K. Sohrabi, J. Gao, V. Ailawadhi and G.J. Pottie, Protocols for self-organization of a wireless sensor network, IEEE Personal Commun. 7(5) (2000) 16–27.
[18] 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).
[19] 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).
[20] G. Veltri, Q. Huang, G. Qu and M. Potkonjak, Minimal and maximal exposure path algorithms for wireless embedded sensor networks. in: *ACM Int'l Conf. on Embedded Networked Sensor Systems (SenSys)* (2003) pp. 40–50.
[21] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless and C. Gill, Coverage and connectivity configuration in wireless sensor networks. in: *ACM Int'l Conf. on Embedded Networked Sensor Systems (SenSys)* 2003 pp. 28–39.
[22] 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)* (2001) pp. 221–235.
[23] 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).
[24] W. Ye, J. Heidemann and D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in: *IEEE INFOCOM* (2002) pp. 1567–1576.

Chi-Fu Huang received his B.S. and M.S. degrees both in Computer Science and Information Engineering from the Feng-Chia University and the National Central University in 1999 and 2001, respectively. He obtained his Ph.D. in the Department of Computer Science and Information Engineering from the National Chiao-Tung University in September of 2004. He is currently a Research Assistant Professor at the Department of Computer Science and Information Engineering, National Chiao-Tung University, Taiwan. His research interests include wireless communication and mobile computing, especially in ad hoc and sensor networks.
E-mail: cfhuang@csie.nctu.edu.tw

Yu-Chee Tseng received his B.S. and M.S. degrees in Computer Science from the National Taiwan University and the National Tsing-Hua University in 1985 and 1987, respectively. He worked for the D-LINK Inc. as an engineer in 1990. He obtained his Ph.D. in Computer and Information Science from the Ohio State University in January of 1994. He was an Associate Professor at the Chung-Hua University (1994–1996) and at the National Central University (1996–1999), and a Full Professor at the National Central University (1999–2000). Since 2000, he has been a Full Professor at the Department of Computer Science and Information Engineering, National Chiao-Tung University, Taiwan. Dr. Tseng served as a Program Chair in the *Wireless Networks and Mobile Computing Workshop*, 2000 and 2001, as a Vice Program Chair in the *Int'l Conf. on Distributed Computing Systems (ICDCS)*, 2004, as a Vice Program Chair in the *IEEE Int'l Conf. on Mobile Ad-hoc and Sensor Systems (MASS)*, 2004, as an Associate Editor for *The Computer Journal*, as a Guest Editor for *ACM Wireless Networks* special issue on "Advances in Mobile and Wireless Systems", as a Guest Editor for *IEEE Transactions on Computers* special on "Wireless Internet", as a Guest Editor for *Journal of Internet Technology* special issue on "Wireless Internet: Applications and Systems", as a Guest Editor for *Wireless Communications and Mobile Computing* special issue on "Research in Ad Hoc Networking, Smart Sensing, and Pervasive Computing", as an Editor for *Journal of Information Science and Engineering*, as a Guest Editor for *Telecommunication Systems* special issue on "Wireless Sensor Networks", and as a Guest Editor for *Journal of Information Science and Engineering* special issue on "Mobile Computing". He is a two-time recipient of the Outstanding Research Award, National Science Council, ROC, in 2001–2002 and 2003–2005, and a recipient of the Best Paper Award in Int'l Conf. on Parallel Processing, 2003. Several of his papers have been chosen as Selected/Distinguished Papers in international conferences. He has guided students to participate in several national programming contests and received several awards. His research interests include mobile computing, wireless communication, network security, and parallel and distributed computing. Dr. Tseng is a member of ACM and a Senior Member of IEEE.
E-mail: yctseng@csie.nctu.edu.tw