A Coverage Inference Protocol for Wireless Sensor Networks

Chi Zhang, *Student Member*, *IEEE*, Yanchao Zhang, *Member*, *IEEE*, and
Yuguang Fang, *Fellow*, *IEEE*

Abstract—After a wireless sensor network (WSN) is deployed, sensor nodes are usually left unattended for a long period of time. There is an inevitable devolution of the connected coverage of the WSN due to battery exhaustion of sensor nodes, intended physical destruction attacks on sensor nodes, unpredictable node movement by physical means like wind, and so on. It is, therefore, critical that the base station (BS) learns in real time how well the WSN performs the given sensing task (i.e., what is the current connected coverage) under a dynamically changing network topology. In this paper, we propose a coverage inference protocol (CIP), which can provide the BS an accurate and in-time measurement of the current connected coverage in an energy-efficient way. Especially, we show that the scheme called BOND, which our CIP requires to be implemented on each sensor node, enables each node to locally self-detect whether it is a boundary node with the minimal communication and computational overhead. The BOND can also be exploited to seamlessly integrate multiple functionalities with low overhead. Moreover, we devise extensions to CIP that can tolerate location errors and actively predict the change of the connected coverage based on residual energy of sensor nodes.

Index Terms—Connected coverage, self-monitoring, wireless sensor network.

✦

---

1 INTRODUCTION

A *wireless sensor network* (WSN) is a large collection of densely deployed, spatially distributed, autonomous devices (or nodes) that communicate via wireless and cooperatively monitor physical or environmental conditions [1], [18]. In such networks, sensor nodes are deployed over a geographic area (called the *region of interest* or ROI) by aerial scattering or other means. Each sensor node can only detect events within some very limited distance from itself, called the *sensing range*. In addition, sensor nodes normally have fairly limited transmission and reception capabilities so that sensing data have to be relayed via a multihop path to a distant *base station* (BS), which is a data collection center with sufficiently powerful processing capabilities and resources. After being deployed, sensor nodes are usually left unattended for a very long period of time so that they may fail over time due to various reasons such as battery exhaustion and physical destructions by attackers. They may also be moved away from where they were deployed by animals, winds, or other environmental means. As a consequence of node failures, node movements, and other unpredictable factors, the network topology may change with time. It is, therefore, critical that the BS learns in real time *how well the WSN performs the given sensing task under dynamically changing network topology.*

- C. Zhang and Y. Fang are with the Department of Electrical and Computer Engineering, University of Florida, PO Box 116130, Gainesville, FL 32611-6130. E-mail: zhangchi@ufl.edu, fang@ece.ufl.edu.
- Y. Zhang is with the Department of Electrical and Computer Engineering, Newark College of Engineering, New Jersey Institute of Technology, 213 Electrical & Computer Engineering Center (ECEC), University Heights, Newark, NJ 07102. E-mail: yczhang@njit.edu.

From the BS's (or user's) point of view, a position in the ROI is really under the surveillance of the WSN if and only if this position is within the sensing range of at least one sensor node connected to the BS. We define the collection of all these positions in the ROI as the *connected coverage* (or *coverage* in short). Obviously, it is one of the most important performance metrics measuring the quality of surveillance a WSN can provide. The BS also should have the ability to monitor the coverage status in real time.

Although much research [2], [9], [10], [12], [16], [20], [21] has been conducted to ensure high network coverage and connectivity for the WSN, none of them addresses how to help the BS infer the coverage boundary when coverage holes emerge. Possible causes leading to coverage holes include energy depletion of sensor nodes, intended attacks on sensor nodes, and so on. In many WSN applications, especially security-sensitive applications, it is a must to accurately detect the coverage boundary. The protocol developed in this paper can affirmatively answer this open challenging issue.

On the other hand, problems related to the self-monitoring of a WSN have been studied in the literature for various applications and purposes. For example, Chessa and Santi [30] propose a single time-out scheme to monitor the system-level fault diagnosis. In [43], a residual energy scan is designed to approximately depict the remaining energy distribution within a WSN. In addition, a self-monitoring mechanism for detecting node failures is proposed in [15]. However, All these schemes cannot be directly used for the coverage inference, as they are either centralized schemes or assume that each individual sensor in the WSN needs to be monitored. This is not true for our case because the BS only needs to ensure that a certain percentage of the sensors are functioning, especially when the WSN is densely deployed.

Generally, we can distinguish two basic types of WSNs: proactive and reactive. Proactive WSNs involve a periodic data delivery between sensor nodes and the BSs. By

comparison, in reactive WSNs, packets are sent only when some event of interest occurs and is sensed. Although for proactive WSNs, each node can simply help the BS infer the connected coverage by piggybacking its status information on data traffic, it is well known that proactive WSNs are energy inefficient and not scalable [8], [18]. Therefore, in this paper, we focus on providing coverage inference for the BS in reactive WSNs. However, due to the very nature of WSNs, this task is far from triviality. The most significant challenge is due to the strict resource limitation of sensor nodes (battery power, memory, computational capability, etc.), which highlights the need for a localized solution. Although there is plenty of work [9], [11], [13], [16], [36], [42] on the localized coverage boundary detection, none of them is adequate because some of them such as polygon-based approaches [9], [11], [36] are not truly localized solutions and suffer from possibly significant communication overhead, while others such as perimeter-based schemes [13], [16], [42] are inefficient when the node density is high (cf., Section 5). Generalization of such schemes to all the situations we are interested in is not trivial.

This paper makes the following contributions: First, we present a *Coverage Inference Protocol* (CIP), which can provide the BS an accurate and in-time measurement of the current connected coverage in an energy-efficient way. Second, we show that the major component of our CIP—*BOundary Node Detection* (BOND) scheme—can be reused to provide many other functionalities for WSNs, such as topology control, efficient routing, sleeping scheduling, and spatial aggregation. Therefore, our schemes can be exploited to seamlessly integrate multiple functionalities with low overhead. The performance of our BOND-based CIP compared with other possible CIPs is also investigated. Moreover, we devise extensions to CIP which can tolerate location errors and actively predict the change of the connected coverage by utilizing the information of residual energy information of sensor nodes.

The BOND scheme was originally proposed in our previous work [38], which was further extended to BOND-based CIP in [39]. Other coverage concepts like barrier coverage [18] were also investigated, and the corresponding boundary node self-detection scheme like neighbor embracing polygon (NEP) scheme was proposed in [41]. In this paper, we still concentrate on traditional area coverage problem [18], and extend our previous work by showing the optimality of our CIP protocol with theoretical proofs and extensive simulations. Recent works on coverage in WSNs include the problem of coverage in three-dimensional space [3], [4] and reducing the position information needed in boundary node detection, see [5] and [41].

The rest of this paper is organized as follows: Section 2 introduces the network model and the design goals of our CIP. Then we detail the core component of our solution called BOND in Section 3. This is followed by presenting the complete CIP in Section 4 and its comparison with other possible alternatives in the design space in Section 5. We then present several extensions to CIP in Section 6 and end with conclusions and future work.

2 PRELIMINARIES

2.1 Network Model

Throughout this paper, we assume that any two sensor nodes can directly communicate via bidirectional wireless links if their euclidean distance is not greater than r_c, the *communication range*; and a position in the plane can be perfectly monitored (or covered) by a sensor node if their euclidean distance is not greater than r_s, the *sensing range*. Similar to [2], [21], [37], we also assume that sensor nodes are homogeneous in the sense that r_c and r_s are the same for all nodes, and keep constant during each node's lifetime.

Instead of considering all the possible combinations of r_c and r_s, we focus on the case of $r_c = 2r_s$ in this paper. There are two reasons for doing so. First, as pointed out in [42], the specification of $r_c \geq 2r_s$ holds for most commercially available sensors such as Berkeley Motes and Pyroelectric infrared sensors. Second, as shown in Section 3.6, for arbitrary spatial distributions of sensor nodes, $r_c \geq 2r_s$ is the sufficient and necessary condition for the existence of local boundary node detection algorithms.[1] Therefore, we set $r_c = 2r_s$ to reduce communication energy consumption and interference. However, it should be noted that our algorithms are still applicable to the scenarios of $r_c > 2r_s$ without any changes. We also present a scheme called SAB to deal with the case of $r_c < 2r_s$ (cf., Section 5.2.2).

For simplicity, we assume that the ROI is a 2D square planar field hereafter. Our results, however, can be easily extended to 2D or 3D ROIs of arbitrary shapes. For $l > 0$, let A_l denote the square ROI of side length l centered at the origin, i.e., $A_l = [-l/2, l/2]^2$, and ∂A_l be the *border* of A_l. We examine a large-scale WSN consisting of hundreds or even thousands of stationary sensor nodes,[2] and denote the sensor nodes deployed in the ROI to be $V = \{s_1, \ldots, s_i, \ldots, s_n\}$ ($s_i \in A_l$, for $1 \leq i \leq n, i \in \mathbb{N}$), where s_i represents the position of node i and n is the total number of sensor nodes (or *network size*).

In general, no assumption should be made about the distribution of the sensor nodes in the environment. Our schemes are designed to work correctly under arbitrary node distributions. However, in the performance evaluation of the schemes proposed in this paper, we utilize homogeneous *Spatial Poisson Point Process* (SPPP) as the node distribution model to facilitate the theoretical analysis and simulations. It is well known that this model is a good approximation of the distribution of sensor nodes massively or randomly deployed (e.g., via aerial scattering or artillery launching) and can be easily extended to characterize the process that nodes fail dynamically.

2.2 Design Goals

In this paper, we intend to design a coverage inference protocol that can provide the BS an *accurate* and *timely* measurement of the current connected coverage in an *energy-efficient* way. Specifically, our design goals include:

Effectiveness and robustness. The BS should be able to have a timely and accurate view of the connected coverage, regardless of arbitrary network topologies, location errors of sensor nodes, and error-prone wireless channels such that it can instruct necessary and quick actions, e.g., adding new sensor nodes to enlarge the connected coverage.

---

1. The formal definition of "boundary nodes" and "local algorithms" will be given in Sections 3.1 and 3.6, respectively.
2. Stationary nodes here do not imply that the topology of the WSN is static. Instead, the WSN may have highly dynamic topology changes due to node failures, new node additions, or nodes switching their states between active and sleeping modes to save energy. One advantage of our schemes lies in the efficiency to handle topology changes in WSNs (cf., Section 3.5).

Truly localized and distributed properties. As compared to previous approaches, the coverage inference protocol is intended to be a truly localized and distributed solution in which each sensor node can self-determine whether it is on the area coverage boundary by a few simple computations on information only from one-hop neighbors. These nice properties will enable the protocol to have low computational and communication overhead, high energy efficiency, and excellent network scalability.

Universal applicability. Although WSNs are often said to be highly application dependent [1], [18], the coverage inference protocol is designed to work with arbitrary applications and network topologies and to be independent of all the other components of the network protocol stack.

Versatility. For resource-constrained sensor networks, it is highly desirable that some basic protocol operations for implementing one functionality can be reused in providing other necessary functionalities. Otherwise, the protocol stack of sensor nodes will be too complicated to have high operational efficiency [18]. The design of coverage inference protocol will take this requirement into consideration so that many of its basic operations can be highly reused in realizing important functionalities other than network health diagnosis, as we will show soon.

3 BOUNDARY NODE DETECTION SCHEME

This section presents BOND that enables each node to locally self-detect whether it is a boundary node. We begin with several important definitions, followed by the illustration of BOND.

3.1 Boundary Node and Its Detection Algorithm

We say that nodes s_i and s_j ($i \neq j$ and $s_i, s_j \in V$) are *neighbors* or there exists a *direct wireless link* between them if the euclidean distance between them is no larger than r_c, i.e., $\|s_i - s_j\| \leq r_c$. We also denote by $Neig(s_i)$ the neighbors of node s_i (not including s_i). In addition, two nodes s_i and s_j are said to be *connected* if there is at least one *path* consisting of direct wireless links between them, similarly a set of nodes is called *connected* if at least one path exists between each pair of nodes in the set. The fundamental connected unit of WSNs is called a *cluster*:

Definition 1 (cluster). *A connected set of nodes is said to be a* ***cluster*** *if the inclusion of any other node not in this set will break the connectedness property.*

We write $Clust(s_i)$ for the cluster containing node s_i. Based on the sensing model, the *sensing disk* (or coverage) of node s_i can be given by

$$Disk_i = Disk(s_i, r_s) = \{u \in \mathbb{R}^2 : \|u - s_i\| \leq r_s\}. \quad (1)$$

Specifically, let $\mathbf{0}$ indicate the origin, we have $Disk_\mathbf{0} = Disk(\mathbf{0}, r_s)$. Then the coverage corresponding to a cluster can be defined as follows:

Definition 2 (coverage of a cluster). *Given $Clust(s_i)$, we refer to the set of all points in the monitored field that are within radius r_s from any node of $Clust(s_i)$ as the set* ***covered*** *by cluster $Clust(s_i)$. Denoting this set by $Cover(s_i)$, we have*
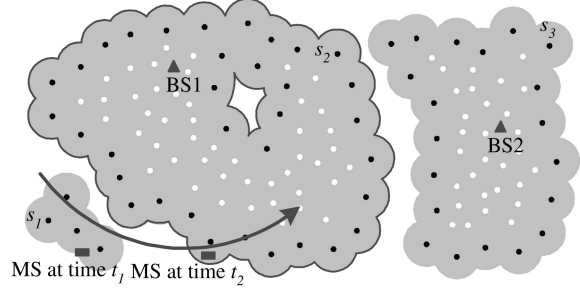

Fig. 1. An exemplary WSNs. There are three sensor clusters: $Clust(s_1)$, $Clust(s_2)$, and $Clust(s_3)$, two static BSs: $BS1$ and $BS2$, and one mobile BS: MS. Shaded area, solid dots, and open dots represent the coverage of sensors, boundary nodes, and interior nodes, respectively. The boundary of $Cover(s_2)$ is depicted by the curves.

$$Cover(s_i) = \left(\bigcup_{u \in Clust(s_i)} (u + Disk_\mathbf{0}) \right) \bigcap A_l. \quad (2)$$

Obviously, $Cover(s_i)$ is uniquely determined by its boundary $\partial Cover(s_i)$.

Definition 3 (boundary and interior node). *We define* ***boundary nodes*** *of $Clust(s_i)$ as those whose minimum distances to $\partial Cover(s_i)$ are equal to r_s, i.e.,*

$$BN(s_i) = \{u \in Clust(s_i) : \min \|u - v\| = r_s \\ \text{for } v \in \partial Cover(s_i)\}. \quad (3)$$

Accordingly, we call all the other nodes in $Clust(s_i)$ as ***interior nodes***, *i.e.,*

$$IN(s_i) = \{u \in Clust(s_i) : u \notin BN(s_i)\}. \quad (4)$$

The minimum information needed to describe $\partial Cover(s_i)$ is r_c and $BN(s_i)$. We denote the position of the base station as BS, then the *connected coverage* of the BS is[3]

$$Cover(BS) = \left\{ \bigcup_{1 \leq i \leq n} Cover(s_i) : \\ BS \bigcap \left(Clust(s_i) \oplus Disk(\mathbf{0}, r_c) \right) \neq \emptyset \right\}. \quad (5)$$

Obviously, the problem of finding the boundary of connected coverage, i.e., $\partial Cover(BS)$, is equivalent to detecting the boundary nodes of clusters with connections to the BS. Based on this observation, it is possible to design a distributed CIP if we can first find a localized way to detect boundary nodes.

Note that our definition of boundary nodes is uniquely based on the cluster they belong to and is unrelated to the position of the BS. In addition, our definition of connected coverage is applicable to WSNs with multiple or mobile BSs. For example, in the WSN given in Fig. 1 where there are three BSs (two static and one mobile), from (5), we can directly obtain the connected coverage of the WSN at time instances t_1 and t_2 as $Cover(s_1) \bigcup Cover(s_2) \bigcup Cover(s_3)$ and $Cover(s_2) \bigcup Cover(s_3)$, respectively. Since for the multiple-BS cases, the connected coverage of the WSN is just the union of the connected coverage regarding each individual BS, hereafter we focus on the single-BS case for the ease of presentation.

---

3. Note that $A \oplus B = \{u + v : u \in A, v \in B\}$ for $A, B \subset \mathbb{R}^2$.

3.2 Localized Voronoi Polygons

Our BOND scheme is based on two novel geometric concepts called *Localized Voronoi Polygon* (LVP) and *Tentative LVP* (TLVP), which are nontrivial generalization of *Voronoi Polygons* (VPs) [25] from computational geometry. We must point out that a similar concept called *Localized Voronoi Diagrams* (LVDs) is introduced as the dual of *Localized Delaunay Triangulations* (LDTs) in the literature [17], [23]. The edge complexity of LDT is analyzed in [17] and its applications in topology control and routing for wireless networks are discussed in [23]. However, there is no indication on how to relate this concept to the coverage problems in WSNs. Moreover, unlike our work, there is no description on how to efficiently construct LVDs given in [17], [23]. Furthermore, the idea of using TLVP to reduce the overhead of the detection algorithm in this paper is completely new. Finally, and most important, our scheme BOND only uses the local information to detect boundary instead of global information commonly used in either VP or DT.

We first define VPs, LVPs, and TLVPs in terms of half planes. For two distinct points $s_i, s_j \in V$, the *dominance region* of s_i over s_j is defined as the set of points, which is at least as close to s_i as to s_j, i.e.,

$$Dom(s_i, s_j) = \{v \in \mathbb{R}^2 : \|v - s_i\| \leq \|v - s_j\|\}. \quad (6)$$

Obviously, $Dom(s_i, s_j)$ is a half plane bounded by the perpendicular bisector of s_i and s_j, which separates all points in the plane closer to s_i than those closer to s_j.

Definition 4 (VP, LVP, and TLVP). *The **VP** associated with s_i, denoted by $Vor(s_i)$, is the subset of the plane that lies in all the dominance regions of s_i over other points in V, namely*

$$Vor(s_i) = \bigcap_{s_j \in V - \{s_i\}} Dom(s_i, s_j). \quad (7)$$

*In the same way, the **LVP**, denoted by $LVor(s_i)$, and the **TLVP**, denoted by $TLVor(s_i)$, associated with s_i are defined as:*

$$LVor(s_i) = \bigcap_{s_j \in Neig(s_i)} Dom(s_i, s_j), \quad (8)$$

$$TLVor(s_i) = \bigcap_{s_j \in SubNeig(s_i)} Dom(s_i, s_j), \quad (9)$$

where $SubNeig(s_i)$ is a proper subset of $Neig(s_i)$, i.e., $SubNeig(s_i) \subset Neig(s_i)$.

The collection of LVPs given by

$$\mathrm{LVor}(V) = \{LVor(s_i) : s_i \in V\} \quad (10)$$

is called the *localized Voronoi diagram* (LVD) generated by the node set V. The boundary of $LVor(s_i)$, i.e., $\partial LVor(s_i)$, may consist of line segments, half lines, or infinite lines, which are all called *local Voronoi edges*.

Lemma 1. *Properties of VPs, LVPs, and TLVPs:*

1. *$LVor(s_i)$, $TLVor(s_i)$, and $Vor(s_i)$ are convex sets,*
2. *$Vor(s_i) \subseteq LVor(s_i) \subset TLVor(s_i)$, and*
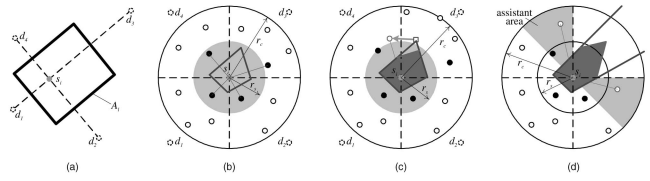3. *Plane $\mathbb{R}^2$ is completely covered by $\mathrm{LVor}(V)$.*

Fig. 2. Illustration of the LVP-based boundary node detection algorithm.

Proof.

1. Since a half plane is a convex set and the intersection of convex sets is a convex set, an LVP (or a TLVP) as well as a VP is a convex set.
2. From (7), (8), and (9), we have

$$Vor(s_i) = LVor(s_i) \bigcap \left(\bigcap_{s_j \in V, s_j \notin Neig(s_j)} Dom(s_i, s_j) \right),$$

$$LVor(s_i) = TLVor(s_i) \bigcap$$
$$\left(\bigcap_{s_j \in Neig, s_j \notin SubNeig(s_i)} Dom(s_i, s_j) \right),$$

which directly leads to Lemma 1 point 2.
3. It is well known in computational geometry that

$$\bigcup_{s_i \in V} Vor(s_i) = \mathbb{R}^2. \quad (11)$$

(cf., [25, Property V1, pp. 77]). Combining (11) with Lemma 1 point 2, we can directly obtain Lemma 1 point 3. □

Therefore, the set $LVor(V) \cap A$ can fully cover an arbitrary subset $A \subseteq \mathbb{R}^2$. Note that this result can be easily extended to any cluster $Clust(s_i)$ in V, that is,

$$\bigcup_{s_j \in Clust(s_i)} LVor(s_j) = \mathbb{R}^2. \quad (12)$$

3.3 LVP-Based Boundary Node Detection

In this section, we present our BOND scheme for each node to detect whether it is a boundary node based on its own LVP or TLVP by taking node s_i as an example.

3.3.1 Input

Our BOND is a distributed scheme in that we only need positions of node s_i's neighbors as the input of our algorithm. We temporally assume that there is no location error and will relax this assumption in Section 6.1. We need to consider two cases based on whether the information about the border of A_l, i.e., ∂A_l, is available. In the first case when ∂A_l is unavailable at node s_i, our detection scheme is based on the construction of $LVor(s_i)$ (or $TLVor(s_i)$); in the second case when ∂A_l is available, we need to exploit this information by calculating $LVor(s_i) \cap A_l$ (or $TLVor(s_i) \cap A_l$). It can be shown that $LVor(s_i) \cap A_l$ must be a finite convex polygon. Thus, the second case can be transformed into the first case by introducing dummy nodes into $Neig(s_i)$. See Fig. 2a, for example, in which four dummy nodes, d_1-d_4, are introduced

such that perpendicular bisectors between s_i and the dummy nodes generate the four border edges of ROI. Then we can calculate $LVor(s_i) \cap A_l$ by following the same procedure for calculating $LVor(s_i)$. Therefore, we will only discuss the first case in what follows.

We notice that dummy nodes cannot be directly applied to the generalized cases, i.e., the border of A_l consisting of curves. However, in these cases, this just means that the information of border of A_l's border cannot be efficiently exploited, and the correctness of our scheme is not affected. There also exist two easy ways to remedy our BOND here. First, in general, a curve can be approximated with straight-line segments, and thus, the BOND is still applicable. Second, instead of checking whether the vertices of $LVor(s_i) \cap A_l$ are covered by $Disk(s_i, r_s)$ when A_l is a polygon, we still can correctly detect boundary nodes by checking every point on $\partial(LVor(s_i) \cap A_l)$ when A_l is not a polygon.

3.3.2 Algorithm

Our goal is to construct the $LVor(s_i)$ (or $TLVor(s_i)$), which is sufficient for the boundary node detection with the minimal requirement on the information about s_i's neighbors. We first divide $Disk(s_i, r_c)$ into four[4] quadrants. Then we construct the TLVP of s_i by using the nearest neighbors (solid nodes in Fig. 2b) in each of the four quadrants. Without lose of generality, we denote these four nearest neighbors as s_1, s_2, s_3, and s_4. The first TLVP is calculated by

$$TLVor(s_i) \leftarrow \bigcap_{j=1}^{4} Dom(s_i, s_j).$$

If all vertices of the TLVP are covered by $Disk(s_i, r_s)$, the procedure stops and this TLVP is saved. Otherwise, we need to find new neighbors which are the nearest to the uncovered vertices of the TLVP (cf., Fig. 2c), add those neighbors to $SubNeig(s_i)$, and calculate the TLVP again:

$$TLVor(s_i) \leftarrow TLVor(s_i) \bigcap \left(\bigcap_{s_j \in SubNeig(s_i), j \neq 1,2,3,4} Dom(s_i, s_j) \right).$$

The new vertices of the new TLVP will be checked to see whether they are covered by $Disk(s_i, r_s)$. This procedure continues until all the vertices of the TLVP are covered by $Disk(s_i, r_s)$ or the LVP of s_i is calculated and saved.

Note that when ∂A_l is unavailable, $LVor(s_i)$ may be infinite, which means that it is possible that we cannot find any nodes in one or more quadrants in the first step. See Fig. 2d for an example. If a quadrant contains no neighbors, we define two sectors of angle 45 degree, which are directly adjacent to the quadrant as the assistant area, and add the nodes in this area to $SubNeig(s_i)$ first. If all the nodes in the assistant area cannot make TLVP finite, we can conclude that LVP must be infinite without need to do further calculation.

_____
4. Other values will also work well.

Fig. 3. Illustration of the proof of Theorem 1.

3.3.3 Output

If $LVor(s_i)$ is infinite, s_i must be a boundary node. If $LVor(s_i)$ (or the final $TLVor(s_i)$) is finite with all the vertices covered by s_i, then $s_i \in IN(s_i)$. Otherwise, $s_i \in BN(s_i)$.

3.4 Algorithm Validation

In the VD, the VPs of different nodes are mutually exclusive, but in the LVD, the LVPs of different nodes may overlap. This difference makes the validation of our algorithm totally different from that of existing VP-based ones.

Theorem 1. _If there is a point $v \in LVor(s_i)$ which is not covered by s_i, i.e., $v \notin Disk(s_i, r_s)$, there must exist a point $h \in LVor(s_i)$ that is not covered by any node, and s_i must be an area coverage boundary node._

Proof. Without loss of generality, we assume that the node nearest to s_i and outside $Disk(s_i, r_c)$ is s_m and $\|s_i - s_m\| = r_c + \delta$ for $\delta > 0$. Let s_m' be the point on $\overline{s_i v}$ satisfying $\|s_i s_m'\| = \|s_i s_m\|$, and h be another point on $\overline{s_i v}$ such that $\|s_i h\| = r_s + \delta/2$ (see Fig. 3). By the triangular inequality, we have $\|s_m h\| + \|s_i h\| \geq \|s_i s_m\| = \|s_i s_m'\| = \|s_i h\| + \|h s_m'\|$. Therefore, $\|s_m h\| \geq \|h s_m'\| = \|s_i s_m'\| - \|s_i h\| = r_s + \delta/2$, which means that s_m cannot cover h and nor does any other node in $Disk(s_i, r_c)^C$. The reason is that, since $\|s_i s_l\| > \|s_i s_m\|$ holds for any node $s_l \in Disk(s_i, r_s)^C$ and $s_l \neq s_m$, we have $\|s_l' h\| > \|s_m' h\|$, where point s_l' is on the line $\overline{s_i v}$ and $\|s_i s_l'\| = \|s_i s_l\|$. Therefore, $\|s_l h\| \geq \|s_l' h\| > \|s_m' h\| = r_s + \delta/2$.

Since $v \in LVor(s_i)$, based on the convexity of $LVor(s_i)$, we have $\overline{s_i v} \in LVor(s_i)$. Therefore, $h \in LVor(s_i)$, which implies for any node $s_j \in Disk(s_i, r_c)$ and $s_i \neq s_j$, we have $\|s_j h\| \geq \|s_i h\| > r_s$, i.e., no nodes in $Disk(s_i, r_c)$ can cover h. Consequently, we can conclude that no node in the plane can cover h because $Disk(s_i, r_c) \cup Disk(s_i, r_c)^C = \mathbb{R}^2$. Note that from the above proof process, we can see that h can be arbitrary close to v', the intersection of circle $\partial Disk(s_i, r_s)$ and $\overline{s_i v}$. Therefore, s_i is a boundary node.   ☐

Theorem 2. _If there is a point $v \in A_l$ not covered by any sensor node, for every cluster $Clust(s_i)$, there must exist at least one sensor $s_j \in Clust(s_i)$ whose $LVor(s_j)$ is not completely covered by $Disk(s_j, r_s)$._

Proof. According to Lemma 1 (point 3) or (12), we have

$$\bigcup_{s_j \in Clust(s_i)} \left(LVor(s_j) \cap A_l \right) = A_l. \qquad (13)$$

Therefore, for any $v \in A_l$, it must lie in at least one $LVor(s_j) \cap A_l$ for $s_j \in Clust(s_i)$. $\qquad \square$

Theorems 1 and 2 prove that the sufficient and necessary condition for $Clust(s_i)$ to completely cover A_l is that $LVor(s_j) \cap A_l$ is completely covered by s_j for all $s_j \in Clust(s_i)$. The following theorem shows that when $LVor(s_i)$ or $LVor(s_i) \cap A_l$ is finite, the coverage of vertices of $LVor(s_i)$ by s_i is equivalent to the coverage of the whole $LVor(s_i)$ by s_i, which guarantees the correctness of our LVP-based algorithm:

Theorem 3. *$LVor(s_i)$ is fully covered by s_i if and only if $LVor(s_i)$ is finite and all the vertices are covered by s_i.*

Proof. Let $Ve(s_i)$ be the set of vertices of $LVor(s_i)$. Obviously, when $LVor(s_i)$ is completely covered by s_i, i.e., $LVor(s_i) \subset Disk(s_i, r_s)$, we have $v \in Disk(s_i, r_s)$ for all $v \in Ve(s_i)$ and $LVor(s_i)$ is finite. Since

$$\max_{u \in LVor(s_i)} \{\|s_i - u\|\} \le \max_{v \in Ve(s_i)} \{\|s_i - v\|\},$$

when $v \in Disk(s_i, r_s)$ for all $v \in Ve(s_i)$, we have $u \in Disk(s_i, r_s)$ for all $u \in LVor(s_i)$. $\qquad \square$

3.5 Discussions on BOND

Low overhead. It has been shown in [25] that in general, VPs cannot be computed locally. Therefore, the traditional VP-based schemes [9], [11], [36] are not distributed and are very expensive in terms of communication overhead. Our BOND scheme is a truly localized polygon-based solution because computing $LVor(s_i)$ (or $TLVor(s_i)$) only needs one-hop information (this can be directly obtained from (8) and (9)). Assuming that the number of neighbors is k, each node can compute its own $LVor(s_i)$ with complexity smaller than $O(k)$. In addition, the computation of the $LVor(s_i)$ only involves some simple operations on polygons, which can be efficiently implemented (e.g., PolyBoolean library [22]). We further simplify the detection process by constructing TLVPs first. For a densely deployed WSN, we have $LVor(s_i)$ or $TLVor(s_i) \rightarrow Vor(s_i)$, and it is well known in computational geometry that under the homogeneous spatial Poisson point process, the average number of vertices of $Vor(s_i)$ is 6 [25]. Therefore, when the node density is high, BOND, on average, only needs 4-6 nearest neighbors' information to successfully detect the boundary nodes. Moreover, when a neighbor node dies, BOND needs do nothing unless the dead node is used to construct the final $TLVor(s_i)$ or $LVor(s_i)$ in the last turn of LVP or TLVP construction. This unique property will greatly simplify the update of detection results and save precious energy of each sensor node. All these advantages cannot be achieved by other localized boundary node detection schemes in the literature, such as the perimeter coverage checking approach [16] and the crossing coverage checking approach [13], [42]. We refer to Section 5.1 for a detailed comparison.

Other applications. We are aware of the following applications of BOND or its basic operations, which are by no means a complete list:

- *Topology control and routing.* It has been shown in [17], [23] that the dual of LVP, called LDT, can be used to design distributed topology control and

▲ Base Station ——— One-Hop Link ‒ ‒ ‒ Multi-Hop Link

(a)   (b)
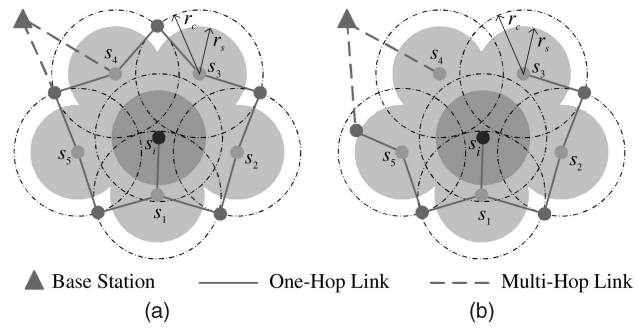
Fig. 4. Nonlocality of the boundary node detection when $r_c < 2r_s$.

routing protocols for wireless ad hoc networks with energy efficiency and the guarantee of the delivery. From the property of duality, we can directly obtain LVPs if LDTs are determined, or vise versa.

- *Spatial aggregation.* In distributed data processing for WSNs, to reduce the sampling errors in the aggregated spatial data, it is proposed in [31] to first calculate the VP of each sensor. As mentioned before, since the VP cannot be computed locally, the LVP can be used as a good approximation of the VP in spatial aggregation.

- *Coverage-preserving node sleeping scheduling.* Since sensor nodes are usually deployed with redundancy, it is possible to prolong the network lifetime while preserving the connected coverage by scheduling some nodes into the sleeping state [34]. Each node can locally decide whether its own LVP is covered by its neighbors. If this is the case, a node declares itself eligible for sleeping, announces this fact to its neighbors, and then goes to sleep. To avoid the forming of coverage hole caused by eligible nodes switching into sleep simultaneously, a randomly delayed announcement scheme using common random backoff approaches is proposed in [34].

3.6 Locality of Boundary Node Detection

In this section, we investigate further to show that it is impossible to find local algorithms for boundary node detection with arbitrary node distributions when $r_c/r_s < 2$. We first define what we mean by local algorithms. This definition is based on a model proposed in [27].

Definition 5 (Local Algorithm). *Assume that each computation step takes one unit of time and so does every message to get from one node to its directly connected neighbors. With this model, an algorithm is called **local** if its computation time is $O(1)$, in terms of the number of nodes n in the system.*

Our BOND shows that when $r_c = 2r_s$, sensors can locally determine if it is a boundary node. When $r_c > 2r_s$, since the node will have more information about other nodes around itself, it can still locally detect whether it is a boundary node. However, in the case of $r_c < 2r_s$, individual nodes can neither locally say "yes" nor "no" to the question of whether a given node is a boundary node. To see this, consider sensors deployed as in Fig. 4. Obviously, node s_i is an interior node. However, to confirm this, it needs to know the existence of nodes s_1-s_5 with the help of some relay nodes
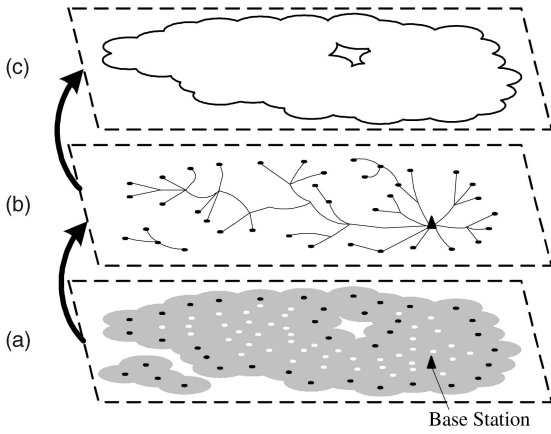
Fig. 5. Basic operations of the BOND-based CIP. (a) Each sensor executes BOND individually. (b) Boundary nodes (black dots in (a)) report themselves to the base station. (c) The BS reconstructs the coverage boundary. Note that the shaded area in (a) represents the coverage of sensors, and that the shaded area at the left bottom corner in (a) is lost in (c) because it is not the connected coverage.

(green nodes). In Fig. 4a, node s_4 is already 5 hops away from node s_i. In fact, the distance between these two nodes can be arbitrary long, which is shown in Fig. 4b. Therefore, for arbitrary node distributions, it is impossible to find a localized boundary node detection algorithm that always works. In [2], the authors considered general values of r_c/r_s with regular deployment patterns such as the hexagon, square grid, rhombus, and equilateral triangle. Unlike [2], in this paper, we prefer to make the strict assumption on the value of r_c/r_s rather than on the node distribution pattern. The reason is that even in some scenarios, sensor nodes are originally distributed with regular patterns (e.g., hexagon, square grid, etc.), due to node failures, node movements caused by animals or winds, etc., the topology of the WSN will become irregular sooner or later. In contrast, even in the case of $r_c/r_s < 2$, we can still assume a smaller value of r_s whereby to get a conservative inference of the coverage, which is desirable for some security-critical applications.

4 COVERAGE INFERENCE PROTOCOL

In this section, we describe how to use BOND to build a practical CIP.

Our design philosophy is that, since the minimum information required to describe the coverage is the positions of boundary nodes (cf., Section 3.1), we just need to detect boundary nodes. In other words, our scheme can ensure that, for the BS to reconstruct the "coverage image" without any distortion, the information transmitted from sensors to the BS is minimized. Also note that our BOND only involves local message exchanges. In a large-scale WSN, the overhead from local broadcast is very small as compared to that from the end-to-end communications from sensor nodes to the BS. Therefore, our approach can save the precious energy of sensor nodes. Fig. 5 illustrates the basic operations of our BOND-based CIP, which consist of the below three steps.

4.1 Neighborhood Monitoring and Self-Detection

After the deployment of the WSN, we assume that localization techniques are available for sensor nodes to

decide their positions [18]. Each node then collects the position information of its neighbors by broadcasting its own positions, and executes BOND to detect whether it is a boundary node. If so, it will report its position to the BS. We refer to the neighbors used to construct the LVP or TLVP in the last run of BOND as its *consulting neighbors*.

In our protocol, both interior and boundary nodes are required to broadcast an *Existence Updating Packet* (EUP) to their neighbors for a random period of time exponentially distributed with rate T_{EUP}. In addition, each interior node, say, s_i, maintains a timer $C_0(j)$ of expiry value much larger than T_{EUP} for each of its nonconsulting neighbors, say, s_j. If s_i does not overhear any packet (either an EUP or data packet) from s_j before $C_0(j)$ expires, it will treat s_j as a dead neighbor, which can become alive if s_i overhears any packet from it later. Node s_i also maintains two timers for each of its consulting neighbors, say, s_k: the neighbor monitoring timer $C_1(k)$ and the neighbor query timer $C_2(k)$. If s_i does not overhear any packet from s_k before $C_1(k)$ expires, it unicasts a *Neighbor Query Packet* (NQP) to s_k and starts $C_2(k)$. If still alive, s_k is required to send back an EUP immediately and wait for an ACK from s_i. If node s_i still does not overhear any packet from s_k before $C_2(k)$ expires, s_i will treat s_k as a dead neighbor and reexecute BOND with alive neighbors as input. In general, the expiry values of $C_0(j)$ and $C_1(k)$ should be in the same order of T_{EUP}, in order to guarantee that with high probability, each node will receive EUPs from all alive nodes in its neighborhood. The expiry values of $C_2(k)$ should be much smaller than T_{EUP} because we require that the node which receives the NQP needs to send back an EUP immediately.

Unlike previous neighbor monitoring schemes employing a single timer [30] or treating neighbors as the same [15], our scheme sets different timers for nonconsulting and consulting neighbors. The major reason for doing so is that data packets and EUP-like broadcast packets are subject to loss due to wireless transmission errors or collisions. As a result, a node may falsely identify an alive neighbor as a dead one. Obviously, for nonconsulting neighbors, we can decrease the false positive rate by setting a larger timer value. However, using a larger timer value for consulting neighbors will increase the response delay, i.e., the delay from when coverage holes emerge to when they are detected by boundary nodes. Therefore, we use two timers for consulting neighbors to ensure both a shorter response delay and a lower false positive rate: although the expiry value of $C_1(k)$ is small, we actively query the questionable neighbor before we treat it as the dead neighbor, which may significantly increase the accuracy of our scheme. As mentioned before, if the node distribution follows SPPP, each node only has 4-6 consulting neighbors, on average, which means the high feasibility of our two-timer scheme. Therefore, by adopting one-timer (C_0) scheme for nonconsulting neighbors and two-timer (C_1 and C_2) scheme for consulting neighbors, our design achieves a better balance among accuracy, delay, and communication overhead. Note that this benefit stems from the fact that our BOND divides each node's neighbors into two categories. For other boundary-node-based approaches in the literature (cf., Section 5.1) where there is no division in neighbors, either

one-timer or two-timer scheme should be adopted for all neighbors and the balance between communication overhead and the accuracy cannot be handled in this way.

Note that the communication overhead of our neighbor monitoring scheme can be further reduced. The EUP packets can piggyback onto regular local broadcast packets used to learn link conditions, maintain the routing information, and facilitate other network operations. In addition, in the presence of data traffic, any packet overheard by a node should be regarded as an EUP packet, and any data packet sent from a node can cancel the next EUP packet it should broadcast.

4.2 Self-Reporting of Boundary Nodes

Whenever identifying itself as a boundary node, a sensor node should send its position information to the BS, which can reconstruct the "image of the coverage" based on all the received position information of boundary nodes.

4.3 Explicit ACKs from the BS

Since the packet loss ratio due to collisions or noise is pretty high in the WSN [18], boundary nodes need some mechanisms to ensure that their reports have been received by the BS. Otherwise, they have to repeatedly resend their reports, which causes energy waste. The issue of reliable sensor-to-BS communication, thus far, has not been addressed thoroughly in WSN research community. The work on reliable communication in WSNs first appears in [35], and then in [32]. However, no guaranteed reliability semantics are provided in these works. In [29], the authors first propose the notion called "event-to-sink" reliability, and study the reliable communication from this perspective. Their solution is mainly based on adjusting the reporting frequency of source nodes, and is applicable for monitoring a continuously changing event or reporting a huge amount of data. For our case, each sensor node will be on the boundary or not for a relatively long period, and each boundary node only needs to report its location information to the BS. Therefore, the scheme proposed in [29] cannot be applied here. In [26], the authors study the "BS-to-sinks reliability," and argue that the requirement and implementation of reliability in a WSN is firmly dependent upon the specific application, and there is no one-for-all solution. For our problem seating, an intuitive solution is to require the BS to send individual ACK to each boundary node from where the report has been received at the BS. In what follows, we will show that this intuitive solution can be improved by introducing the Bloom filter [6]. For the sake of clarity, we start with the description of ACK format with the Bloom filter. Then we analyze its performance quantitatively.

The basic idea here is to use Bloom filters to design an energy-efficient approach for the BS to broadcast only one ACK to acknowledge multiple boundary nodes. Let $s_1, s_2, \ldots, s_a$ be the a boundary nodes which the BS wants to acknowledge explicitly. Let $h_1, h_2, \ldots, h_b$ be the b hash functions of the Bloom filter, each with range $\{1, 2, \ldots, l\}$. Let $ack(t) = (b_1, b_2, \ldots, b_l)$ be a bit vector of length l. Note that $ack(t)$ is the tth Bloom filter used to indicate boundary nodes the BS wants to acknowledge, and it is initially set to $(0, 0, \ldots, 0)$. For $\forall i$, $1 \leq i \leq b$ and $\forall j$, $1 \leq j \leq a$, the BS computes $h_{ij} = h_i(ID_{s_j})$, where ID_{s_j} indicates the unique ID of node s_j, and sets $b_{h_{ij}} = 1$. The BS can then use the Bloom filter $ack(t)$ as ACK and send it back using efficient broadcast or geographic multicast protocols for WSNs [18]. When a boundary node s_j receives the $ack(t)$, it performs a membership test: it computes $h_i(ID_{s_j})$ for $\forall i$, $1 \leq i \leq b$; if all of these positions are 1 in the $ack(t)$, then boundary node s_j knows that its report has been received and acknowledged by the BS.

Note that the Bloom filter may induce a small number of false positives, i.e., a few unacknowledged boundary nodes may pass the membership test, and therefore, believe that their reports have been received by the BS. On the other hand, Bloom filters ensure that there are no false negatives, i.e., all acknowledged nodes are guaranteed to pass the membership test. In practice, we can tune our b and l parameters to enable trade-off between communication and computational overhead and false positive rate.

Now we analyze the benefit of utilizing ACKs with Bloom filters. First of all, we need quantify the communication overhead of individual-ACK scheme. Suppose that there are n sensor nodes in the network, and the length of each node ID is $\log n$ bits. If there are a boundary nodes which the BS wants to acknowledge explicitly, the BS needs send out a ACK packets, and each packet with the size of $O(\log n)$ bits. This is because the header of the ACK packet should include the destination node ID with $\log n$ bits. Next, we consider the multihop communication scheme that can be used to support the BS-to-sensor communication. Note that each sensor node only has a finite buffer, and the buffer size is relatively small compared to n for a large-scale WSN. Therefore, it is almost impossible to provide unicast communication from the BS to individual sensor node. Note that to support the unicast communication from the individual sensor to the BS is still possible (cf., Section 5.2.2). This is caused by the asymmetry of traffic pattern in WSNs [18]. For the sensor-to-BS unicast communication, we can construct a routing tree rooted at the BS (see Fig. 11 for an example), and each node only needs to relay the packet from its child nodes to its parent node in this routing tree. To maintain this routing tree, each node only needs to add one entry into its routing table with the destination as the BS and the next hop as its one-hop parent node. For the BS-to-sensor unicast communication, the situation will be totally different. For every child node in the subtree rooted at a particular node, this node needs to add one entry in its routing table for that child node. For the node near the BS, it may need $O(n)$ entries in the routing table, which is too large for the buffer-constraint sensors. So, in practice, there are no efficient schemes to support unicast downstream communication (from the BS to sensors) [18]. Especially for the small size packet, to establish a tentative route by broadcast, a route discovery packet is invaluable. Therefore, even for the BS-to-sensor unicast communication, when the packet size is small, we still use global broadcast from the BS.

In individual-ACK scheme, the BS needs to perform a broadcasts, each with packet size $O(\log n)$ bits. For Bloom-filter-based scheme, the BS needs to perform one broadcast with packet size $O(l)$ bits. For the standard Bloom filter [7], in order to keep a low false positive rate, l is on the order of n. For example, in our simulation, $l = 9n$ bits, $b = 6$, and the false positive rate is 1.33 percent. Therefore, when $a = \Omega(n/\log n)$, it is beneficial to use Bloom-filter-based scheme. This is equivalent to the situation when the

percentage of boundary nodes is $\Omega(1/\log n)$. For a large-scale WSN, $1/\log n$ is a very small number, which means that the Bloom-filter-based ACk should be used in practice with high probability.

To sum up, in this section, we design some complementary components to BOND, to complete our BOND-based CIP protocol. We emphasize that these designs are intuitive and straightforward because simplicity, i.e., minimal communication and computational overhead, is our principal design objective. However, the advantages of adopting these simple designs stem from the BOND algorithm. Specifically, BOND needs to check only minimal number of consulting neighbors, which enables us to utilize two-timer scheme without introducing too much communication overhead. Moreover, BOND is able to identify boundary nodes, which facilitates us to employ ACK-based scheme to provide reliable sensor-to-BS communications.

5   COMPARISON AND SIMULATION

To the best of our knowledge, there is no other coverage inference protocol developed for WSNs, which is as complete as our CIP. We, however, note that many interesting ideas proposed by researchers for WSN coverage and self-monitoring may be adapted to infer WSN coverage. In this section, we exploit these possible alternatives and compare them with our CIP via both theoretical analysis and simulations.

The most important metric used in the comparison is the *energy consumption* incurred by different coverage inference protocols. In addition, we assume that packet transmissions are subject to noise or collision. Under this assumption, we further define two other performance metrics. The first one is the *false alarm probability*, defined as the probability that a nonboundary sensor node is falsely diagnosed as a boundary node. This may happen if consecutive EUPs from a sensor node get lost due to collision and/or noise. The other metric is the *response delay*, defined as the time from when a sensor node becomes a boundary node to when this event is locally detected. Apparently, the later two metrics are conflicting in essence for a time-out-based coverage inference protocol like our CIP. In particular, to achieve a smaller false alarm probability (i.e., more accurate boundary detection) desires a larger time-out value, which, in turn, would result in a longer response delay, and vice versa. Therefore, we would compare the false alarm probabilities of different coverage inference protocols for a given response delay, or equivalently compare their response delays for a given false alarm probability.

The evaluation philosophy and simulation settings are given in the Appendix. In order to mimic independent and random node failures, we simulate multiple sensor networks by varying the number k of neighbors for each node (or equivalently, the node density λ) through NS2 network simulator [24]. To facilitate the presentation, we also classify the possible coverage inference protocols into two categories, namely boundary-node-based approaches and aggregation-based approaches, based on how the coverage information is gathered and reported to the BS.

5.1   Boundary-Node-Based Approaches

Approaches in this category are to find boundary nodes first and then transmit such information to the BS in a way similar to what was described in Section 4. In what follows, we further classify these approaches according to the boundary node identification methods they adopted.

5.1.1   Polygon-Based Schemes

In [9], [11], [36], *Voronoi diagram* (VD) is used for boundary node detection. Briefly speaking, the VD of a node set V is the partition of the euclidean space into polygons, called VPs and denoted by $Vor(s_i)$ for $s_i \in V$ such that all the points in $Vor(s_i)$ are closer to s_i than to any other node in V. According to the closeness property of VPs, if some portion of a VP is not covered by nodes inside the VP, it will not be covered by any other node, which implies a coverage hole. Therefore, it is claimed in [9], [11], [36] that each node can locally check whether it is on the coverage boundary under the assumption that VPs can be derived locally. However, it has been shown that, in general, VPs cannot be locally computed [40]. Intuitively, our LVP-based BOND will have smaller communication overhead, or equivalently, smaller energy consumption than the VP-based schemes since LVPs can be locally computed. In what follows, we prove this intuition in a formal way.

Theorem 4. *If there exist boundary nodes, the costs of the NEP-based and LVP-based algorithms are always smaller than the cost of the VP-based ones.*

The proof of the theorem depends on the following lemma:

Lemma 2. *For any $s_i \in V$, the VP $Vor(s_i)$ can be locally computed if and only if $Clust(s_i)$ can completely cover the plane $\mathbb{R}^2$ (or A_l, when the information of ∂A_l is available), i.e., $Cover(s_i) = \mathbb{R}^2$ (or $Cover(s_i) \cap A_l = A_l$).*

Proof. Let $d = \max\|v - s_i\|$ for any $v \in Vor(s_i)$. Since $Vor(s_i)$ is a convex set, then $d = \infty$ if $Vor(s_i)$ is infinite, otherwise, d is the distance from a vertex of $Vor(s_i)$ to s_i. $Vor(s_i)$ can also be computed in a similar way as LVP with set V as input. Specifically, we first compute $LVor(s_i)$ as the tentative VP of s_i and then refine the tentative VP iteratively. In each iteration, we add one-more-hop information about node positions. We can determine that the construction of $Vor(s_i)$ is completed when all the nodes in $Disk(s_i, 2d)$ have been counted. Therefore, $Vor(s_i)$ can be locally computed, which implies that $2d \le r_c$ or $d \le r_s$, and thus, guarantees the complete coverage of $Vor(s_i)$. Since this holds for all $s_i \in V$, we can ensure the complete coverage of the plane.

From Theorems 1 and 2, a node set V can completely cover $\mathbb{R}^2$ if and only if $LVor(s_i)$ is fully covered by $Disk(s_i, r_s)$ for any $s_i \in V$. From Lemma 1, this implies that $Vor(s_i) = LVor(s_i)$ for any $s_i \in V$. Therefore, $Vor(s_i)$ can be locally computed by s_i just as $LVor(s_i)$.   □

Therefore, when there are boundary nodes, it is impossible to compute all $Vor(s_i)$s locally based on only one-hop information. Since multihop communications are unavoidable, the energy consumption of the VP-based approaches will be higher than our LVP-based BOND. Only when the node density is so high that the ROI is completely covered (not considering the ROI border), is the cost of the VP-based approaches equal to that of ours.
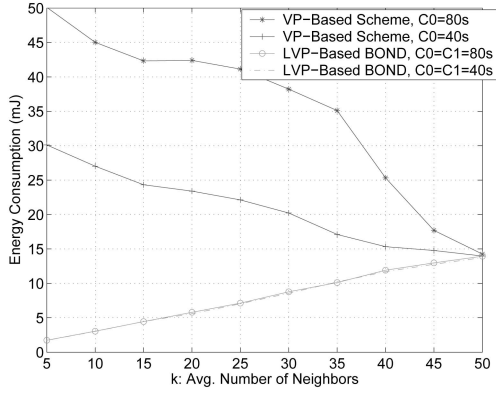
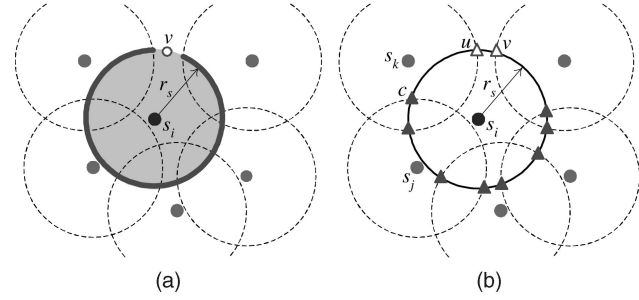Fig. 6. Energy consumption for the LVP- and VP-based schemes.

Fig. 7. Perimeter-based boundary node detection approaches. (a) Perimeter coverage checking approach proposed in [16]. The solid curve represents the portion of perimeter of sensing disk covered by neighbor nodes. (b) Crossing coverage checking approach proposed in [13], [42]. Solid and open triangles represent covered and uncovered crossings, respectively.

However, in this case, there is no need for coverage boundary detection at all. So Theorem 4 guarantees that when boundary detection algorithms are helpful, the cost of our BOND is definitely smaller than the VP-based ones.

We answer the question of how significant the cost savings are by simulations. To focus on comparing the energy consumption of two schemes, we set $T_{EUP} = 1s$, and $C_0 = C_1 = 40s$ and $80s$, and $C_2 = 0$ for our BOND. For a fair comparison, the VP-based scheme uses the same timer C_0 for one-round boundary node detection. We set a very large time-out value to make sure that false alarm probability is very small, and thus, can be neglected here. The packet size of EUP is 64 bytes (no NQP here), and any control packet transmitted for the VP-based scheme to reconstruct VPs is of 64 bytes as well.

Fig. 6 shows the average node energy consumption for the VP-based and the LVP-based schemes as a function of k. We can see that energy consumption of our LVP-based BOND only slightly increases as k becomes larger, as the reception energy consumption becomes larger with the increasing number of neighbors. In addition, the energy savings of our LVP-based BOND are quite significant compared to the VP-based scheme. We can also observe that the difference between the two schemes becomes smaller with the increase of k. The reason is that the number of hops needed to reach $2d$ for VP computing will become smaller with the increase of k. In particular, when $k = 50$, where the ROI is fully covered with probability 0.9999, VPs can be locally calculated as LVPs, so there will be no difference between these two schemes. Therefore, compared to the VP-based scheme, our BOND-based CIP can achieve remarkable energy savings when the functional nodes are sparsely deployed ($4.5 < k < 4 \log n + 4 \log \log n$). Also note that a small C_1 has no effect on our BOND-based CIP because $C_1 = 40s$ is still enough for one-hop information gathering (note that $k < 50$). However, a small C_0 has a great effect on the VP-based scheme because it significantly reduces the multihop communication, and thus, only allows a node to calculate a tentative VP. This observation shows that calculating the final VP is not necessary for most situations we are interested in, and LVP itself is enough for boundary node detection.

5.1.2 Perimeter-Based Schemes

The first realistic localized boundary node detection algorithm is proposed in [16], which is based on the information about the coverage of the perimeter of each node's sensing

disk. It can be shown that node s_i is a boundary node if and only if there exists at least one point $v \in \partial Disk(s_i, r_s)$ which is not covered by any $s_j \in Neig(s_i)$ (cf., Fig. 7a). Based on this criterion, an algorithm with the complexity O($k \log k$) is designed in [16] to locally check whether one node is a boundary node, where k is the number of neighbors. A crossing coverage checking approach proposed in [13], [42] simplifies the previous perimeter coverage checking approach by just checking some special points called crossings on the perimeter. A *crossing* is defined as an intersection point of two perimeters of sensing disks. A node s_i is a boundary node if and only if there exists at least one crossing $v \in \partial Disk(s_i, r_s) \cap \partial Disk(s_j, r_s)$, which is not covered by any other $s_k \in Neig(s_i) - \{s_j\}$. Fig. 7b shows an example where c is a crossing determined by two perimeters $\partial Disk(s_i, r_s)$ and $\partial Disk(s_j, r_s)$, which is covered by the third sensing disk of node s_k. For simplicity, below we will just compare our BOND-based CIP with the crossing coverage checking approach (denoted by CROSS).

Our BOND and CROSS can both provide truly localized boundary node detection with operational difference in the neighborhood monitoring phase. In particular, CROSS requires each node to check the positions and status of all it neighbors, which is quite inefficient when sensor nodes are densely deployed. This situation becomes worse every time when a node dies, as all its neighbors need recheck the coverage of their perimeters or crossings. In contrast, our BOND-based CIP only has consulting neighbors perform boundary node detection. When sensor nodes are densely deployed, from Lemma 1, we have $LVor(s_i)$ or $TLVor(s_i) \rightarrow Vor(s_i)$, and it is well known in computational geometry that under the homogeneous SPPP, the average number of vertices of $Vor(s_i)$ is 6 [25]. Therefore, when the node density is high, each node, on average, only has 4-6 consulting neighbors. Fig. 8 shows the average number of neighbors needed for the CROSS and our BOND to detect boundary nodes as a function of k. It is of no surprise to see that when the node density increases, the number of nodes needed remains constant for BOND-based CIP while increasing dramatically for CROSS. This means that, in contrast to our BOND-based CIP, CROSS will incur a significant overhead at the initial stage of WSNs, where sensor nodes are normally densely deployed to provide adequate redundancy and fault tolerance.

Now we compare BOND-based CIP with CROSS regarding their trade-offs among the updating time interval
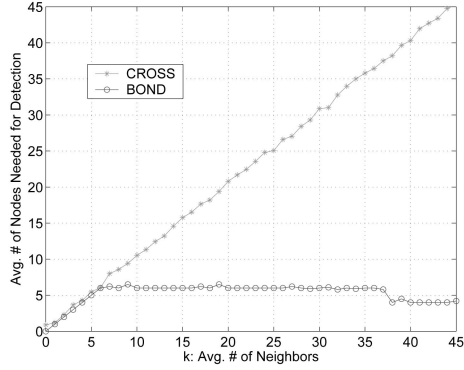
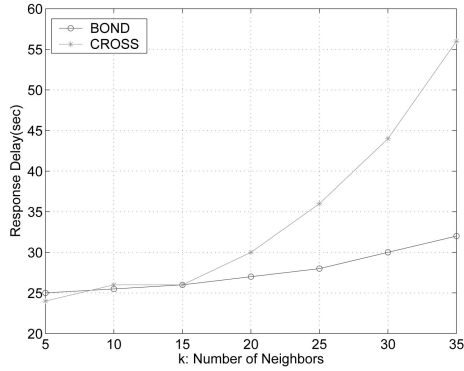Fig. 8. Average number of neighbor nodes needed for the crossing coverage checking approach and our BOND.

Fig. 9. Simulation results with $T_{EUP} = 10s$, $Pr[FA] \leq 0.01$.

T_{EUP}, response delay, and false alarm probability (denoted by $Pr[FA]$) via simulations. Since CROSS needs the information of all neighbors, it can only use the same timer C_0 for all neighbors. Therefore, the response delay of BOND-based CIP and CROSS can be measured by $C_1 + C_2$ and C_0, respectively. The packet sizes of EUP and NQP are both 64 bytes. Our objective here is to select optimal system parameters to meet a certain performance requirement on either the response delay or $Pr[FA]$.

We first consider that the smallest response delay can be achieved for different schemes when $T_{EUP} = 10s$ and $Pr[FA] \leq 0.01$. Simulation results in Fig. 9 show that, when the average number of neighbors (k) increases, the response delay for CROSS dramatically increases due to more packet collisions in the shared wireless channel, but only slightly increases in our BOND-based CIP.

Fig. 10 shows the power consumption for different schemes when $Pr[FA] \leq 0.01$ and the response delay ($C_1 + C_2$ or C_0) is bounded by $40s$. It can be seen that, in order to keep $Pr[FA] \leq 0.01$, CROSS has to adopt a small value of T_{EUP}, which would dramatically increase the power consumption. The energy consumption of our BOND-based CIP slightly increases with k because the number of consulting neighbors will not increase with k and the related NQPs will be used with very small probability.

In Figs. 9 and 10, we can also observe that when functional nodes are sparsely deployed ($k \leq 15$), CROSS almost has the same performance as our BOND-based CIP. To sum up, the VP-based approaches only perform well when functional nodes are densely deployed; the perimeter-based approaches only work well when functional nodes
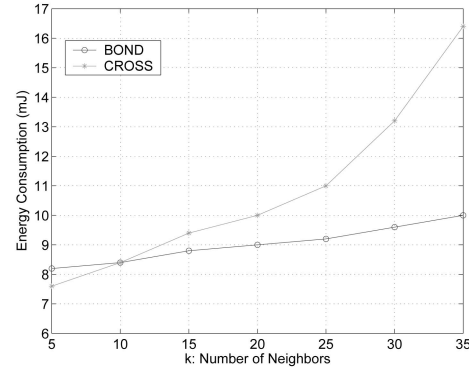
Fig. 10. Simulation results with response delay $\leq 40s$, $Pr[FA] \leq 0.01$.

are sparsely deployed; and only our BOND works equally well in both cases.

5.2 Aggregation-Based Approaches

Aggregation-based approaches are quite different from our BOND in that each node actively communicates with the BS no matter whether it is a boundary node or not. Therefore, we need to compare those schemes to our BOND-based CIP as a whole in this section. In what follows, we classify these approaches into naive ones and spatial ones.

5.2.1 Naive Schemes

The most straightforward scheme requires that each node periodically sends EUPs ("keep-alive" messages) to inform the BS about its existence so that the BS can always learn the connected coverage of the WSN. If the BS does not receive the update information from a particular node in a predetermined time interval, it can infer that this node is dead. This approach obviously suffers from significant communication overhead, and thus, energy consumption.

An alternative to the naive scheme is to let each node report the positions of its dead neighbors in which nodes can locally cooperate to ensure that each dead node is reported once. However, this scheme is unlikely to be the optimal one because the death of some nodes does not necessarily imply the change of connected coverage especially when the WSN is densely deployed. In addition, there might be redundant information transmitted to the BS, which means that this scheme is also not energy efficient.

5.2.2 Spatial Aggregation-Based Schemes (SAB)

Since the coverage information is highly spatially correlated, a natural way to improve the energy efficiency of the above naive scheme or its alternative is to perform spatial aggregation at intermediate nodes to reduce redundant transmissions.

Almost all techniques for spatial aggregation require the construction of a routing tree for propagating data from source nodes to the BS [19]. Once the routing tree is established, each node utilizes the routing tree to find a path to the BS. A simple method for constructing the routing tree is as follows: The BS broadcasts an initialization message into the network, which contains hop count specifying the distance from the BS. All nodes that hear the initialization message will select the BS as their parent, increase the hop count by 1, and then rebroadcast the message. The message will propagate down the network until every node has
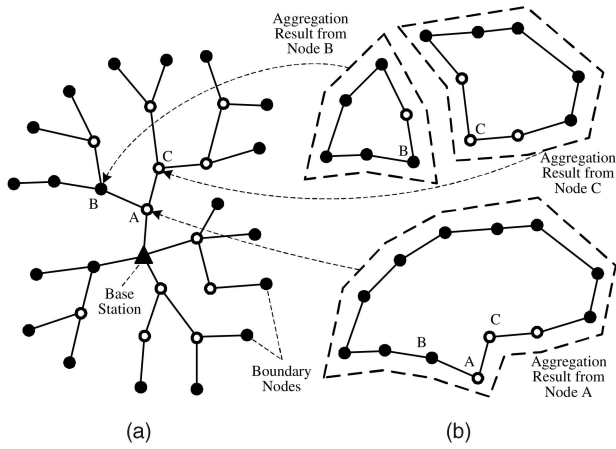
Fig. 11. Illustration of SAB. (a) The routing tree for the spatial aggregation is constructed with the root as the BS. (b) Node A aggregates coverage information by combining polygons from its children B and C. Note that only the solid dots are real boundary nodes. If the open dot appears in the aggregation results, it represents redundant information and leads to energy waste.

Fig. 12. Energy cost ratio of SAB and CIP to naive scheme.

established a sender as its parent node leading toward the BS (see Fig. 11a, for example). The coverage information in each node is represented by a polygon. Let V_{child} denote itself, its children, its children's children, ... This polygon covers all nodes in V_{child}, and its vertices are the boundary nodes of V_{child}. To obtain this polygon, each node only needs to receive the polygons of its children and aggregates them into a new one which is sent along the routing tree toward the BS. Some boundary nodes reported from the children may be deleted in this step because the parent nodes have more information about the coverage, and thus, may find that those nodes are not real boundary nodes. Refer to Fig. 11 for an illustration of this process.

This kind of approach is used by *e-Scan* [43], which focuses on monitoring the residual energy of sensor nodes, and *isobars* [14], which focuses on advanced aggregation techniques for highly spatially correlated sensing data. Basically speaking, the energy efficiency of SAB depends on the time interval between two successive active reports of individual nodes. In practice, it is difficult to adaptively tune this parameter to achieve the best trade-off between information freshness and energy efficiency.

It is possible to adapt SAB to a more passive scheme. In particular, only when a node finds that its aggregation polygon changed, does it update the polygon to its parent. However, there is still room to further improve the energy efficiency. Note that, to reconstruct the connected coverage image at the BS, the only useful information is the positions of boundary nodes. As shown in Fig. 11b, in this layered aggregation scheme, aggregation results from the $(i+1)$-hop node (e.g., B or C) will contain more redundant information (nonboundary nodes) than those of the i-hop node (e.g., A), and only the polygon aggregated at the BS is redundant-free. In contrast, our CIP directly gets the real boundary nodes from the first step instead of removing the redundancy layer by layer.

We compare the naive scheme in which periodic "keep-alive" messages need be sent, SAB, and CIP under the homogeneous SPPP model through simulations. We set $T_{EUP} = 1s$ and $C_0 = C_1 = 40s$ or $80s$, and $C_2 = 0$ for our BOND-based CIP. Since C_1 is much greater than T_{EUP}, the
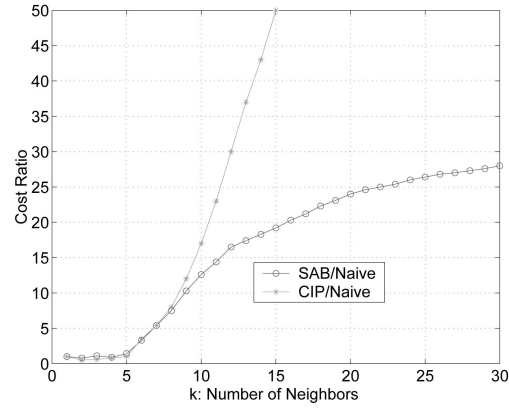
false alarm probability is very small and can be neglected here. We also assume that the packet size of EUP is 64 bytes (no NQP here), and that it requires 32 bits to represent a position. Therefore, in SAB, when a child sends a packet about its polygon to his parent node, and this polygon has m vertices, the packet size will be $64 + 4(m-1)$ bytes. For a fair comparison, all the schemes use the same routing tree as shown in Fig. 11. Let R denote the *energy cost ratio* which is defined as the ratio of the energy consumption of SAB or BOND-based CIP to that of the naive scheme. Fig. 12 plots R versus different node density $\lambda = k/\pi r_c^2$, where k is the average number of neighbors per node. As we can see, when $k \leq 4$, almost every node connected to the BS is a boundary node, and the naive scheme is the best ($R < 1$ for both BOND-based CIP and SAB). However, this is the situation the WSN tries to avoid under which the WSN has already collapsed. In addition, when $k \geq 5$, the R of BOND-based CIP grows exponentially, which shows the significant energy savings of our scheme over both the naive scheme and SAB. Also note that when the ROI is almost fully covered, the cost of local broadcasts is negligible as compared to end-to-end communications from sensor nodes to the BS. Therefore, it is of no surprise to see that $R \to \infty$ for BOND-based CIP when $k \geq 25$. Recall that although BOND-based CIP has better performance compared to SAB, it is based on the assumption that $r_c \geq 2r_s$. If this assumption does not hold, boundary nodes cannot be locally identified (as we proved in Section 3.6), and all advantages will be lost. However, for SAB, it can be applied to all values of r_c/r_s, which gives another motivation for us to present SAB here.

6 EXTENSIONS TO CIP

In this section, we extend our BOND-based CIP by considering location errors and energy depletion, which shows the flexibility of BOND and its ability to deal with some practical considerations.

6.1 Location-Error-Tolerant CIP

So far we have assumed that each node knows its accurate location. Our CIP can also be extended to tolerate bounded location errors. In this section, we assume that the location error (defined as the distance between the actual location of a node and its estimated location) is upper bounded by δ. We then have the following theorem:

Theorem 5. *If the location error is upper bounded by δ, and a given node, e.g., s_i, is an interior node when all nodes are at their estimated locations and each node uses a sensing range $r_s - \delta$, node s_i must be an interior node when all nodes are at their actual locations and each node uses the actual sensing range r_s.*

Proof. We prove by contradiction. Let a and a' represent the actual and estimated locations of point a, respectively. Suppose that node s_i is a boundary node with the actual sensing range r_s when there is no location error. There must exist a point p in $LVor(s_i)$ that is not covered by $Disk(s_i, r_s)$. On the other hand, p must be covered by $Disk(s_i', r_s - \delta)$ when p is still in $LVor(s_i')$ or p is in $LVor(s_j')$ for $s_j' \in Neig(s_i')$. Note that here $LVor(s_i')$ and $LVor(s_j')$ are calculated with estimated locations and the sensing range $r_s - \delta$. We therefore have $\|s_i' - p\| \le r_s - \delta$ or $\|s_j' - p\| < \|s_i' - p\|$ for $s_j' \in Neig(s_i')$. Since $\|s_i - s_i'\| \le \delta$ and $\|s_j - s_j'\| \le \delta$, from the triangle inequality, we have $\|s_i - p\| \le \|s_i' - p\| + \|s_i - s_i'\| \le r_s$ or $\|s_j - p\| \le \|s_j' - p\| + \|s_j - s_j'\| < \|s_i - p\|$. Hence, p is covered by $Disk(s_i, r_s)$ or p is not in $LVor(s_i)$, which contradicts our assumption that p is in $LVor(s_i)$ and not covered by $Disk(s_i, r_s)$. ☐

From Theorem 5, we can design our location-error-tolerant CIP as follows: When the location error is upper bounded by δ, our CIP can assume the sensing range of $r_s' = r_s - \delta$. Since $\delta > 0$, the condition that $r_c/r_s' \ge 2$ still holds, and from the discussion in Section 3.6, we know that our BOND can correctly detect boundary nodes with r_s' and estimated locations. Based on Theorem 5, all the boundary nodes with the actual sensing range r_s and locations will be detected by BOND. It is worth noting that it is possible that some interior nodes in terms of their actual locations will be mislabeled as boundary nodes. Therefore, our location-error-tolerant CIP gives a conservative inference on the connected coverage, which is desirable for many WSN applications such as security-critical ones.

6.2 Prediction-Based CIP

Since node death caused by energy depletion is predictable, it is possible to design prediction-based CIP by exploiting the residual energy information of sensor nodes. The challenge here is that the death of some nodes does not necessarily indicate the change of coverage especially when the WSN is densely deployed. Therefore, in order to minimize the information relayed to the BS, we need to detect the nodes whose death caused by energy depletion will affect the coverage.

We first define *healthy* nodes as those with residual energy more than a threshold. Whenever a node finds that its own residual energy is less than the threshold and it is not a boundary node, it needs locally broadcast this information. Let $HNeig(s_i)$ be the set of healthy neighbors of node s_i. When node s_i finds that its residual energy is less than the threshold, it first calculates the VP for each of its healthy neighbors, e.g., s_j, as follows:

$$Vor(s_j) = \bigcap_{s_k \in HNeig(s_i) - \{s_j\}} Dom(s_j, s_k).$$

Node s_i then checks all the vertices of $Vor(s_j)$ ($s_j \in HNeig(s_i)$) in $Disk(s_i, r_s)$. If at least one of them is only covered by $Disk(s_i, r_s)$, the death of node s_i will cause the
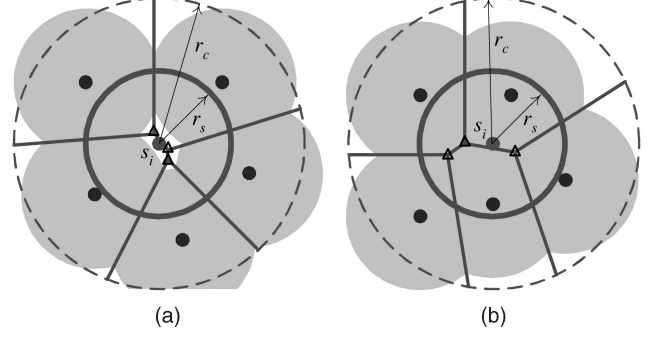

Fig. 13. Voronoi-diagram-based coverage hole prediction.

coverage hole (see Fig. 13a, for example), and node s_i will report the event that its residual energy is smaller than the threshold to the BS. Otherwise, node s_i can conclude that its death will not effect the coverage (see Fig. 13b, for example), and thus, does not need to report itself to the BS. Based on the collected residue energy information, the BS can predict where the coverage hole will emerge with high probability. Note that, when the number of healthy neighbors of node s_i is k, there are elegant algorithms in computational geometry [25] to calculate the Voronoi diagram of $HNeig(s_i)$ with complexity $O(k log k)$, and there are at most $O(k)$ vertices need to be checked. The polygon operations in BOND can also be reused here. Since only local broadcasts are involved in our scheme, the computational and communication overhead introduced is rather small.

7 CONCLUSION

In this paper, we propose a CIP that allows the BS to get an accurate and in-time measurement of the current connected coverage of the WSN. The CIP is built upon a novel lightweight distributed BOND scheme. Detailed theoretical analysis and simulation studies show that both BOND and CIP are highly effective and efficient. As the future research, we plan to evaluate the performance of BOND and CIP in real sensor platforms. We also intend to further investigate other potential applications of BOND in WSNs such as load balancing, topology control, distributed storage, and network health monitoring.

APPENDIX

NETWORK TOPOLOGIES USED IN PERFORMANCE EVALUATIONS

Consider the situation where sensor nodes are independently and randomly placed in the ROI. Such a random initial deployment is required when individual sensor placement is infeasible and is desirable when a priori knowledge of the ROI and the monitored target is limited or not available. In this case, it is widely accepted in the literature [12] that the locations of sensors can be modeled by a 2D homogeneous SPPP with density λ.

Definition 6 (homogeneous SPPP). *A homogeneous SPPP with density λ can be defined by the following two properties: First, for any measurable subset of A_l with area B,*
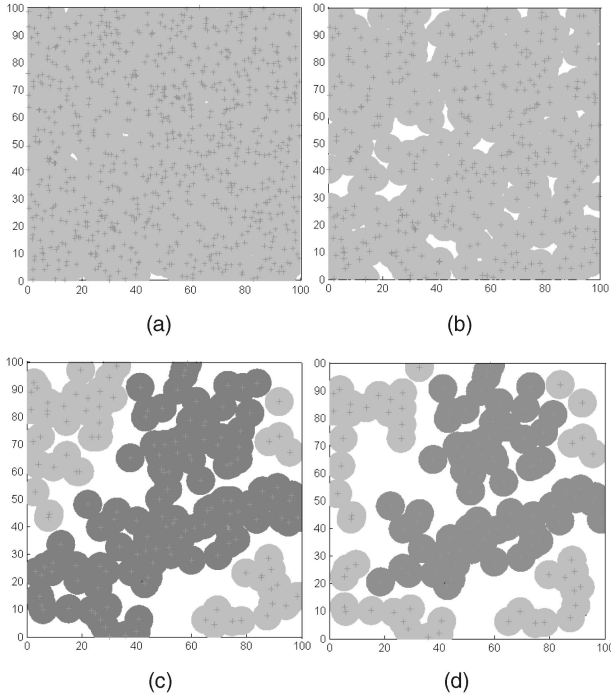
Fig. 14. Some network topologies used in our performance evaluation. (a) $k = 40$. (b) $k = 15$. (c) $k = 5$. (d) $k = 4$. In all situations, the position of the BS is $(50, 50)$. Shaded area represents the coverage of sensors. Note that in (c) and (d), only darkly shaded area represents connected coverage needed to be measured.

$$\Pr\{\text{finding } i \text{ nodes in the region of area } B\} = \frac{(\lambda B)^i e^{-\lambda B}}{i!}.$$

Second, the number of nodes in disjoint (nonoverlapping) area are independent random variables.

Each node is expected to have $k = \pi r_c^2 \lambda$ neighbors, on average, and the expected number of nodes in A_l is given by $n = \lambda \cdot A_l$. When each node fails independently and uniformly with probability p. It has been shown that functional nodes still form a homogeneous SPPP with density $\lambda' = (1 - p)\lambda$ [33]. In this case, the network can be uniquely identified by the current node density λ (or equivalently k).

Note that by tuning parameter k (or λ), we can get different network topologies corresponding to different coverage patterns. Four situations get great interests in the literature (see Fig. 14 for an illumination) [28]:

1. When $k > 4 \log n + 4 \log \log n$, the ROI is almost fully covered. There are no boundary nodes when border information is available (cf., Fig. 14a).
2. When $k > \log n$, the whole network is connected. The number of interior nodes is larger than that of boundary nodes (cf., Fig. 14b).
3. When $k > 4.5$, the network is percolated. There only one big cluster, and the number of interior nodes is smaller than that of boundary nodes (cf., Fig. 14c).
4. When $k < 4.5$, the network is subcritical (collapsed) and consists of many small clusters (cf., Fig. 14d).

Therefore, in our evaluation, we focus on the cases when

$$4.5 < k \leq 4 \log(n) + 4 \log \log(n). \qquad (14)$$

It can be found that, although we only use homogeneous SPPPs to generate the network topologies, when the node density is smaller than the critical value for the connectivity, the network topologies will become very irregular due to the disconnectedness.

In our simulations, we use NS-2 and assume $r_s = 5$ units, the data size for position representation is 32 bits, the energy consumed to transmit and receive 1 bit is $0.8~\mu\text{J/bit}$ and $0.6~\mu\text{J/bit}$, respectively. Sensor nodes are distributed in a square ROI with side $l = 100$ units. In the network initial deployment phase, in order to ensure coverage, the total number of nodes deployed in the ROI is 1,000 (corresponding to $k = 40$). The MAC protocol used in our simulation is 802.11.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks,* vol. 38, no. 4, pp. 393-422, Mar. 2002.

[2] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. Lai, "Deploying Wireless Sensors to Achieve Both Coverage and Connectivity," *Proc. ACM MobiHoc,* May 2006.

[3] X. Bai, C. Zhang, D. Xuan, J. Teng, and W. Jia, "Full-Coverage and K-Connectivity (k=14,6) Three Dimensional Networks," *Proc. IEEE INFOCOM,* Apr. 2009.

[4] X. Bai, C. Zhang, D. Xuan, J. Teng, and W. Jia, "Low-Connectivity and Full-Coverage Three Dimensional Networks," *Proc. ACM MobiHoc,* May 2009.

[5] Y. Bejerano, "Simple and Efficient K-Coverage Verification without Location Information," *Proc. IEEE INFOCOM,* Apr. 2008.

[6] B. Bloom, "Spaxe/Time Trade-Offs in Hash Coding with Allowable Errors," *Comm. ACM,* vol. 13, no. 7, pp. 422-426, June 1970.

[7] A. Broder and M. Mitzenmacher, "Network Applications of Bloom Filters: A Survey," *Internet Math.,* vol. 1, no. 4, pp. 485-509, 2005.

[8] M. Cagalj, S. Capkun, and J. Hubaux, "Wormhole-Based Anti-Jamming Techniques in Sensor Networks," *IEEE Trans. Mobile Computing,* vol. 6, no. 1, pp. 100-114, Jan. 2006.

[9] Q. Fang, J. Gao, and L. Guibas, "Locating and Bypassing Routing Holes in Sensor Networks," *Proc. IEEE INFOCOM,* Mar. 2004.

[10] M. Franceschetti, L. Booth, M. Cook, R. Meester, and J. Bruck, "Percolation of Multi-Hop Wireless Networks," Technical Report UCB/ERL M03/18, Electrical Engineering and Computer Science Dept., Univ. of California, Berkeley, 2003.

[11] A. Ghosh, "Estimating Coverage Holes and Enhancing Coverage in Mixed Sensor Networks," *Proc. Ann. IEEE Int'l Conf. Local Computer Networks (LCN '04),* Nov. 2004.

[12] P. Gupta and P.R. Kumar, "Critical Power for Asymptotic Connectivity in Wireless Networks," *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming,* Birkhauser, 1998.

[13] P. Hall, *Introduction to the Theory of Coverage Processes.* John Wiley Sons, Inc., 1988.

[14] J.M. Hellerstein, W. Hong, S. Madden, and K. Stanek, "Beyond Average: Towards Sophisticated Sensing with Queries," *Proc. Second Int'l Workshop Information Processing in Sensor Networks (IPSN '03),* Mar. 2003.

[15] C. Hsin and M. Liu, "Self-Monitoring of Wireless Sensor Networks," *J. Computer Comm.,* special issue on sensor networks, vol. 29, no. 4, pp. 462-476, Feb. 2006.

[16] C.F. Huang and Y.C. Tseng, "The Coverage Problem in a Wireless Sensor Network," *Proc. Second ACM Int'l Workshop Wireless Sensor Networks and Applications (WSNA '03),* Sept. 2003.

[17] S. Kapoor and X. Li, "Proximity Structures for Geometric Graphs," *Algorithms and Data Structures,* Springer, July 2003.

[18] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks.* John Wiley Sons, Inc., 2005.

[19] R. Kumar, M. Wolenetz, B. Agarwalla, J. Shin, P. Hutto, A. Paul, and U. Ramachandran, "Fuse: A Framework for Distributed Data Fusion," *Proc. First ACM Conf. Embedded Networked Sensor Systems (Sensys '03),* Nov. 2003.

[20] S. Kumar, T.H. Lai, and A. Arora, "Barrier Coverage with Wireless Sensors," *Proc. ACM MobiCom,* Aug. 2005.

[21] S. Kumar, T.H. Lai, and J. Balogh, "On K-Coverage in a Mostly Sleeping Sensor Network," *Proc. ACM MobiCom,* Oct. 2004.

[22] M. Leonov, "Polyboolean Library (2004)," http://www.complex-a5.ru/polyboolean, 2010.

[23] X. Li, G. Calinescu, P. Wan, and Y. Wang, "Localized Delaunay Triangulation with Applications in Wireless Ad Hoc Networks," *IEEE Trans. Parallel and Distributed Systems,* vol. 14, no. 10, pp. 1035-1047, Oct. 2003.

[24] S. McCanne and S. Floyd, "Network Simulator Version 2," http://www.isi.edu/nsnam/ns, 2010.

[25] A. Okabe, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams.* Wiley, 2000.

[26] S.-J. Park, R. Vedantham, R. Sivakumar, and I.F. Akyildiz, "A Scalable Approach for Reliable Downstream Data Delivery in Wireless Sensor Networks," *Proc. ACM MobiHoc,* May 2004.

[27] D. Peleg, *Distributed Computing: A Locality-Sensitive Approach.* Soc. Industrial and Applied Math. (SIAM), 2000.

[28] M. Penrose, *Random Geometric Graphs.* Oxford Univ. Press, 2003.

[29] Y. Sankarasubramaniam, O.B. Akan, and I.F. Akyildiz, "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks," *Proc. ACM MobiHoc,* June 2003.

[30] S. Chessa and P. Santi, "Comparison Based System-Level Fault Diagnosis in Ad-Hoc Networks," *Proc. IEEE 20th Symp. Reliable Distributed Systems (SRDS),* Oct. 2001.

[31] M. Sharifzadeh and C. Shahabi, "Utilizing Voronoi Cells of Location Data Streams for Accurate Computation of Aggregate Functions in Sensor Networks," *GeoInformatica,* vol. 10, no. 1, pp. 9-36, Mar. 2006.

[32] F. Stann and J. Heidemann, "Reliable Data Transport in Sensor Networks," *Proc. Int'l Workshop Sensor Net Protocols and Applications (SNPA),* Apr. 2003.

[33] D. Stoyan, W. Kendall, and J. Mecke, *Stochastic Geometry and Its Applications,* second ed. Wiley, 1995.

[34] D. Tian and N.D. Georganas, "A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks," *Proc. First ACM Int'l Workshop Wireless Sensor Networks and Applications (WSNA '02),* Sept. 2002.

[35] C.-Y. Wan, A. Campbell, and L. Krishnamurthy, "PSFQ: A Reliable Transport Protocol for Wireless Sensor Networks," *Proc. ACM Int'l Workshop Sensor Networks and Architectures,* Sept. 2002.

[36] G. Wang, G. Cao, and T.L. Porta, "Movement-Assisted Sensor Deployment," *Proc. IEEE INFOCOM,* Mar. 2004.

[37] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks," *Proc. ACM Int'l Conf. Embedded Networked Sensor Systems (SenSys '03),* Nov. 2003.

[38] C. Zhang, Y. Zhang, and Y. Fang, "Detecting Boundary Nodes in Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Networking, Sensing and Control,* Apr. 2006.

[39] C. Zhang, Y. Zhang, and Y. Fang, "Energy-Efficient Coverage Measurement for Wireless Sensor Networks," *Proc. IEEE GLOBECOM,* Nov. 2006.

[40] C. Zhang, Y. Zhang, and Y. Fang, "Localized Coverage Boundary Detection for Wireless Sensor Networks," *Proc. Third Int'l Conf. Quality of Service in Heterogeneous Wired/Wireless Networks (QShine),* Aug. 2006.

[41] C. Zhang, Y. Zhang, and Y. Fang, "Localized Algorithms for Coverage Boundary Detection in Wireless Sensor Networks," *Wireless Networks,* vol. 15, no. 1, pp. 3-20, Jan. 2009.

[42] H. Zhang and J. Hou, "Maintaining Sensing Coverage and Connectivity in Large Sensor Networks," *Wireless Ad Hoc and Sensor Network,* vol. 1, nos. 1/2, pp. 89-123, Jan. 2005.

[43] J. Zhao, R. Govindan, and D. Estrin, "Computing Aggregates for Monitoring Wireless Sensor Networks," *Proc. First IEEE Int'l Workshop Sensor Network Protocols and Applications (SNPA '03),* May 2003.

Chi Zhang received the BE and ME degrees in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in July 1999 and January 2002, respectively. He is currently working toward the PhD degree in the Department of Electrical and Computer Engineering at the University of Florida, Gainesville. His research interests include network and distributed system security, wireless networking, and mobile computing, with an emphasis on mobile ad hoc networks, wireless sensor networks, wireless mesh networks, and heterogeneous wired/wireless networks. He is a student member of the IEEE.

Yanchao Zhang received the BE degree in computer communications from the Nanjing University of Posts and Telecom, China, in 1999, the ME degree in computer applications from Beijing University of Posts and Telecom, China, in 2002, and the PhD degree in electrical and computer engineering from the University of Florida in July 2006. He then joined NJIT as an assistant professor of the Electrical and Ccomputer Engineering Department. His primary research interests include network and distributed system security, wireless networking, and mobile computing. He is currently an associate editor of the *IEEE Transactions on Vehicular Technology*, a feature editor of the *IEEE Wireless Communications*, and the TPC cochair of the Communication and Information System Security Symposium, IEEE GLOBECOM 2010. He has also been serving on the program committees of many conferences including INFOCOM, MobiHoc, ICDCS, and PerCom. He was a winner of the US National Science Foundation CAREER Award in 2009. He is a member of the IEEE.

Yuguang Fang received the PhD degrees in systems engineering and electrical engineering, respectively, from Case Western Reserve University in January 1994 and Boston University in May 1997. He was an assistant professor in the Department of Electrical and Computer Engineering at the New Jersey Institute of Technology from July 1998 to May 2000. He then joined the Department of Electrical and Computer Engineering at the University of Florida in May 2000 as an assistant professor, got an early promotion to an associate professor with tenure in August 2003, and to a full professor in August 2005. He held a University of Florida Research Foundation (UFRF) Professorship from 2006 to 2009, a Changjiang Scholar Chair Professorship with Xidian University, Xi'an, China, from 2008 to 2011, and a Guest Chair Professorship with Tsinghua University, China, from 2009 to 2012. He has published more than 250 papers in refereed professional journals and conferences. He received the US National Science Foundation Faculty Early Career Award in 2001 and the US Office of Naval Research Young Investigator Award in 2002, and was the recipient of the Best Paper Award at the IEEE International Conference on Network Protocols (ICNP) in 2006 and the recipient of the IEEE TCGN Best Paper Award in the IEEE High-Speed Networks Symposium, IEEE Globecom in 2002. He is also active in professional activities. He is currently serving as the editor-in-chief for the *IEEE Wireless Communications* and serves/served on several editorial boards of technical journals including the *IEEE Transactions on Communications*, the *IEEE Transactions on Wireless Communications*, the *IEEE Wireless Communications Magazine*, and the *ACM Wireless Networks*. He was an editor for the *IEEE Transactions on Mobile Computing* and currently serves on its steering committee. He has been actively participating in professional conference organizations such as serving as the steering committee cochair for QShine from 2004 to 2008, the technical program vice chair for IEEE INFOCOM 2005, technical program symposium cochair for IEEE Globecom 2004, and a member of the technical program committee for IEEE INFOCOM (1998, 2000, 2003-2010). He is a fellow of the IEEE and a member of the ACM.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.