RESEARCH ARTICLE

A pragmatic approach to area coverage in hybrid wireless sensor networks†

Nadeem Ahmed*, Salil S. Kanhere and Sanjay Jha

School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia

ABSTRACT

Success of Wireless Sensor Networks (WSN) largely depends on whether the deployed network can provide desired area coverage with acceptable network lifetime. This paper seeks to address the problem of determining the current coverage achieved by the non-deterministic deployment of static sensor nodes and subsequently enhancing the coverage using mobile sensors. We identify three key elements that are critical for ensuring effective area coverage in Hybrid WSN: (i) determining the boundary of the target region and evaluating the area coverage (ii) locating coverage holes and maneuvering mobile nodes to fill these voids, and (iii) maintaining the desired coverage over the entire operational lifetime of the network. We propose a comprehensive solution that addresses all of the aforementioned aspects of the area coverage, called MAPC (mobility assisted probabilistic coverage). MAPC is a distributed protocol that operates in three distinct phases. The first phase identifies the boundary nodes using the geometric right-hand rule. Next, the static nodes calculate the area coverage and identify coverage holes using a novel probabilistic coverage algorithm (PCA). PCA incorporates realistic sensing coverage model for range-based sensors. The second phase of MAPC is responsible for navigating the mobile nodes to plug the coverage holes. We propose a set of coverage and energy-aware variants of the basic virtual force algorithm (VFA). Finally, the third phase addresses the problem of coverage loss due to faulty and energy depleted nodes. We formulate this problem as an Integer Linear Program (ILP) and propose practical heuristic solutions that achieve similar performance as that of the optimal ILP solution. A guiding principle in our design process has been to ensure that the MAPC can be readily implemented in real-world applications. We implemented the boundary detection and PCA algorithm (i.e., Phase I) of the MAPC protocol on off-the-shelf sensor nodes and results show that the MAPC can successfully identify boundary nodes and accurately determine the area coverage in the presence of real radio irregularities observed during the experiments. Extensive simulations were carried out to evaluate the complete MAPC protocol and the results demonstrate that MAPC can enhance and maintain the area coverage, while reducing the total energy consumption by up to 70% as compared with the basic VFA. Copyright © 2010 John Wiley & Sons, Ltd.

KEYWORDS

coverage estimation; coverage maintenance; hybrid wireless sensor network; mobility

*Correspondence

Nadeem Ahmed, School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia.
E-mail: nahmed@cse.unsw.edu.au

1. INTRODUCTION

Providing adequate sensing coverage of the target area is of paramount importance for proper functioning of a WSN. In theory, this can be achieved by placing sensor nodes at predetermined locations in the target area. However, in hostile, and harsh environments such as enemy territory in battlefields, and vast forest-lands, the sensor nodes cannot be carefully deployed according to a pre-determined regular topology. Random (possibly aerial) deployment of sensor nodes is a solution in such scenarios. But in this case, it is very difficult to ensure adequate coverage of the target area. This may happen due to the presence of obstacles, sloping grounds like hills, presence of strong winds or dense forestation during aerial deployment etc. As a result, it is highly likely that certain regions in the target area remains uncovered, leading to coverage holes. Mobility capable sensors are a potential solution to enhance the existing

†The results in this paper have been presented in part at the Workshop on Wireless Local Networks (WLN, 2005), Mobile Ad-hoc and Sensor Systems (MASS, 2006), and Mobile Ad-hoc and Sensor Networks (MSN, 2007).
coverage by plugging these coverage holes. As mobile sensors are costlier than their static counterparts, an all mobile network is economically not feasible. A hybrid sensor network offers a pragmatic alternative, wherein, randomly deployed static sensors are supplemented by a limited number of mobile sensor nodes to reach an adequate coverage level. The mobile nodes in a hybrid network can also exercise self-maintenance of the WSN. They can replace any faulty, damaged, or energy depleted nodes to ensure that the network continues to operate at the desired coverage level.

The area coverage in hybrid sensor network is defined as follows. Given a set of static and mobile sensors and a target area, the area is adequately covered if every point in that target area is covered by at least one sensor. There are several aspects of the area coverage problem that need careful consideration:

- How to identify nodes lying on the outer boundary of a randomly deployed network in a distributed manner?
- How to evaluate area coverage provided by a deployment and to identify coverage holes?
- How to plug the coverage holes by maneuvering mobile sensor nodes?
- How to maintain the coverage at the desired level over the operational lifetime of the WSN by replacing energy-depleted and faulty nodes?

This paper proposes a comprehensive mobility assisted probabilistic coverage (MAPC) protocol that addresses all of the aforementioned issues.

### 1.1. Protocol overview

Figure 1 illustrates the building blocks of the MAPC protocol. The protocol works in three distinct phases. Phase I and II are executed during the initial deployment of a WSN, while phase III is executed over the entire operational lifetime of the network. Phase I consists of two sub-tasks; boundary estimation and coverage estimation. Detection of nodes lying on the outer boundary of a network is essential for self-organization of a sensor network. From a coverage perspective, boundary nodes circumscribe the target area that needs to be adequately covered. We propose a distributed boundary estimation algorithm based on the geometric right-hand rule. Boundary estimation results in the identification of virtual boundary formed by the static nodes lying on the perimeter of the deployed network. After the target region has been defined, all the static nodes in the network estimate the area coverage. The widely used binary detection model assumes that the sensing coverage of a sensor node is uniform in all directions, often represented by a unit disk. However, in reality, the sensing capabilities of a sensor are often affected by environmental factors. In particular, for range-based sensor modalities such as acoustics, radio, etc, the signal strength of the triggering signal decays as a function of distance. In an effort to employ more realistic models in the computation of area coverage, we propose the probabilistic coverage algorithm (PCA), that takes into account the probabilistic sensing behavior of sensor nodes.

Mobile nodes are introduced in the topology in phase II. They are assumed to be initially concentrated at one or more points on the boundary of the target area. Phase II of MAPC is responsible for managing the iterative relocation of the mobile sensors to meet two objectives. The primary goal is to fill the coverage holes identified by the static nodes in Phase I. The secondary goal is to uniformly spread the additional mobile sensors in the target area, so that these nodes can replace any faulty or dead sensor nodes during the operational lifetime of the network. Further, given that the mobile nodes expend significant energy in maneuvering, the energy expenditure during the relocation process should be minimized. We propose a set of coverage and energy aware variants of virtual force algorithm (VFA) to meet these objectives.

The goal of the third phase of MAPC is to ensure that the area coverage is maintained during the operational lifetime of WSN. This pre-empts any possible loss of coverage due to nodes depleting their batteries (proactive maintenance) or due to dead sensors, e.g., nodes with hardware failure or physical damage (reactive maintenance). Phase II seeks to first identify such coverage holes and then mobilize the mobile nodes to fill the same. In this paper, we have modeled the coverage maintenance problem as an integer linear program (ILP), and have proposed heuristic solutions that perform close to the optimal.

### 1.2. Contributions

The main contribution of this work is the design and evaluation of MAPC, a distributed and comprehensive three phase protocol, that can effectively estimate, enhance, and maintain the area coverage in a hybrid sensor network. To the best of our knowledge, no such comprehensive solution has been presented in the literature.

This paper makes the following four specific contributions. First, we propose a practical boundary estimation scheme that identifies the nodes lying on the outer boundary of a deployed network topology. Our approach differs from the existing schemes in that we assume that sensors cannot detect the physical boundary of the region in outdoor
environments. The scheme employs the geometric right-hand rule for identifying the boundary nodes. We have conducted extensive simulations and experimentation on real sensor hardware to validate the boundary estimation scheme.

Second, we propose a novel coverage estimation algorithm, PCA, that evaluates the area coverage in WSN. PCA takes into account the realistic variations in sensing behavior associated with range-based sensors. This is in contrast to the simplistic binary unit disk sensing model, where an event occurring within a circular disk is always assumed detected with probability of 1. PCA takes advantage of the sensing contributions from neighbors for accurate estimation of a node’s area coverage. PCA has also been evaluated through simulations and experiments on real mote hardware.

The third contribution falls in the area of mobile nodes deployment. We propose a set of integrated deployment and coverage enhancement algorithms that are based on the VFA. The algorithms use movement triggering thresholds that are based on real radio characteristics. These algorithms not only deploy mobile nodes from an initial concentrated position but also increase coverage by plugging the coverage holes in the topology. For different types of initial deployments, simulations show that the proposed movement algorithms consume only 30–40% of the energy consumed by the basic VFA.

Finally, the fourth contribution addresses the problem of coverage maintenance in WSNs by addressing the loss in coverage due to damaged and energy depleted nodes. Simulation results show that the proposed coverage maintenance scheme is a good heuristic approximation of a centralized optimal but impractical ILP solution.

The remainder of this paper is organized as follows. We discuss related research in Section 2. Section 3 elaborates the boundary estimation scheme while Section 4 details the probabilistic coverage estimation. Phase II of the MAPC protocol detailing the deployment of mobile nodes is covered in Section 5 while coverage maintenance (Phase III) is described in Section 6. Section 7 concludes the paper.

2. RELATED WORK

2.1. Boundary estimation in WSN

Boundary estimation refers to the detection of nodes lying on the boundary of the target region. Algorithms for boundary estimation are presented in References [1–5]. Jarvis [1] suggested a centralized approach to find the convex hull of a set of points. The algorithm is simple and effective when the number of points on the convex hull is relatively small compared to the number of points in the points set. Authors in Reference [2] proposed the BoundHole algorithm using the right-hand rule to identify nodes on the boundary of geometric holes. Stajmenovic et al. [3] and more recently Liu et al. [4] proposed the quorum based location service. They proposed the use of the right-hand rule on edges of Gabriel graph to locate the boundary nodes that act as location servers for the deployed ad hoc network. More recently, Kroller et al. [5] proposed an overall framework for self organization by using a combination of topology and geometry to determine the boundary nodes and the topology of the whole network.

Our proposed boundary estimation algorithm is distributed and lightweight and hence more suited to the energy constrained wireless sensor networks (WSN). It does not require flooding for gathering the topology information as is the case with Jarvis walk [1]. Authors in References [3] and [4] applied the right-hand rule on the edges of a planar Gabriel graph to locate the boundary nodes. It was shown by Kim et al. [6] that planar graphs such as the Gabriel graph that are derived from unit disk graphs may result in crossing edges due to radio irregularity. Moreover, the maintenance of planar graph at each node introduces additional overhead. Thus, in order to have a practical and efficient approach, we do not assume a planar graph. Rather, we introduce explicit checks to detect crossing edges among the neighbors.

2.2. Probabilistic coverage estimation

Probabilistic coverage for sensor networks has been explored in References [7–11], but in different context to our work. Kuo et al. [7] proposed a signal strength based approach with an error model targeting a location estimation application assuming probabilistic coverage for sensors. Ren et al. [8] used temporal probabilistic coverage, where any point in a sensing field is sensed with a certain probability at any time, for tracking a moving object in the sensor field. In Reference [9], Zou et al. proposed a grid-based clustered approach to evaluate the target localization. The scheme is centralized whereby a cluster head is responsible for calculating the likely position of a target by employing a probabilistic scoring-based localization algorithm. Xing et al. [10] proposed a probabilistic event detection model based on data fusion while Kumar et al. in [11] studied randomized independent sleeping (RIS) where each sensor is active with a probability independently from other sensors.

In this work, we propose a computational geometry based approach that assumes probabilistic coverage characteristics for the deployed sensor nodes. The coverage is calculated at the perimeter of each node’s sensing range instead of generalized grid points. This gives us a more accurate coverage calculation for each node. Our approach extends the notion of the perimeter coverage proposed by Huang et al. [12] and is completely distributed.

2.3. Coverage enhancement

Use of mobility capable sensors to guarantee coverage during deployment has been proposed in prior work [13–16]. In Reference [13], the authors proposed three different
deployment protocols that spread out the mobile sensors once coverage holes are detected using Voronoi diagrams. Reference [14] is a potential field based approach assuming compact initial concentration of the mobile nodes. A centralized incremental deployment scheme is proposed in Reference [15] that deploy sensors one by one while requiring line of sight among the nodes. Similarly a centralized VFA for the movement of mobile sensor nodes is proposed in Reference [9]. In Reference [16], the coverage problem is solved by a moving robot that is guided by the already deployed nodes for exploring poorly covered areas.

In this work, we propose a set of localized and distributed variants of the VFA for spreading out the mobile sensors as opposed to the centralized VFA employed in Reference [9]. We also propose the simulated movement (SM) approach for energy efficient movement of mobile sensor nodes. Finally, for mobile sensor relocation, we propose metrics to decide the movement schedule based on coverage and energy considerations.

2.4. Coverage repair and maintenance

Use of mobile sensors for network repair is proposed in References [17–22]. A bidding protocol is proposed in Reference [17] where static nodes bid for mobile nodes while mobile nodes also consider the loss of coverage at its existing location due to the movement. In another work by the same authors [18], a cascaded movement approach is used to move redundant mobile sensors to respond to node failure. In Reference [19], the authors proposed a scheme called Co-Fi that moves mobile nodes to replace low energy nodes. The protocol does not work if the coverage loss is due to physical damage to the nodes. In Reference [20], authors proposed dynamic coverage maintenance algorithms to move the neighbors to compensate the loss of coverage in an all mobile network. Mei et al. [21] proposed the use of mobile robots that are capable of dropping functional sensor nodes at specific locations to repair coverage. Li and Santoro proposed an integrated framework that includes deployment and coverage maintenance [22] assuming an all mobile network. Our work addresses the problem of coverage loss due to both energy depletion and physical damage to the nodes.

3. Phase I—Topological Boundary Estimation

Phase I of the MAPC protocol is initiated with the deployment of static sensor nodes in the target region. When nodes are deployed randomly, there is a need to determine which nodes are located at the boundary of the network. We propose a distributed boundary estimation scheme, whereby static nodes estimate the boundary of the unknown region by using the right-hand rule. Boundary estimation results in identification of B-nodes i.e., nodes lying on the outer boundary of a deployed topology.

3.1. Boundary estimation by static nodes

A naive approach for boundary detection is to collect the topology information at the sink/base station by flooding probing packets over the whole network. The topology information once gathered can help in identification of boundary nodes by geometric computation of convex hull [1]. This approach is centralized, involves high communication and computation cost, and does not scale with the size of the network. A localized and distributed boundary node selection algorithm is desired.

3.1.1. Boundary estimation algorithm.

The proposed B-node selection algorithm is based on the simple geometric right-hand rule. It consists of two parts: application of right-hand rule to compute the next B-node, and sending of a special B-message to the next selected B-node. After neighbor information has been exchanged through local broadcast of Hello messages, a node that receives a B-message marks itself as a B-node. It then sweeps clock-wise from the direction of the sending node and selects the first node that intersects the sweeping line from its neighbor list (by simple geometric calculations as location information of neighbors is available). In the case when multiple neighbors are found at the same minimum sweeping angle from the previous B-node (e.g., prospective nodes are lying in a straight line), the nearest such neighbor is selected as the forward B-node. The node then unicasts a B-message to the next selected node (referred as forward B-node) and stores the information about previous B-node (node from which the B-message was received) and the forward B-node. The node that initially originated the B-message is referred as OB-node (originating B-node) and each B-message contains the OB-node ID. Each B-node also stores the OB-node ID from the received B-message. The B-message processing continues through the boundary nodes of the deployed topology until the B-message reaches back to the OB-node.

3.1.2. Selection of the originating B-node.

Without loss of generality, we assume that a sink node is available (outside the WSN convex region) and that it can communicate with a set of deployed nodes. After learning the location information of the neighboring static nodes that are within its communication range, the sink selects the nearest neighbor (with the least Euclidean distance) as the OB-node. The sink unicasts a B-message to the selected node to trigger the B-nodes selection algorithm. The OB-node applies the right-hand rule by sweeping clock-wise from the direction of the sink (see Figure 2) to select the forward B-node. Note that the sink is only used for starting

‡ We note that neighbor and connectivity information can also be obtained by other low cost alternatives such as link layer feedback etc.
the B-node selection process and it does not participate any further in the selection process. Also the OB-node does not mark the sink as its previous B-node, the correct previous B-node will connect with the OB-node when the B-node selection terminates.

The algorithm functions correctly even when multiple sinks are assumed located outside the WSN region. Each sink, in this case, selects its own OB-node and generates B-messages with their respective OB-node ID’s. A B-node compares the OB-node ID for each additional B-message that it receives. Normal processing of the B-message continues only if the B-message has a different OB-node ID.

3.1.3. Local minima.

Consider Figure 3. Node X receives the B-message from node Y. X applies the right-hand rule and selects node A as the forward B-node. As node A has only one neighbor, node X, it cannot select any new B-node except node X. We call this special case the local minima for the algorithm. Recall that each node maintains the previous and forward B-node information. Node A sends the B-message, with a local minima flag, back to previous B-node X without marking itself as a B-node. Node X removes node A as the forward B-node, continues with the right-hand rule and selects node Z as the new forward B-node now. In case a node receiving the B-message with local minima flag has no other neighbor to select than its previous B-node, it again sends a local minima B-message to the previous B-node (see Figure 3(b)). The algorithm can thus recover from a local minima. Note that some of the nodes lying on the convex hull (e.g., Node A in Figure 3(a)) are not selected as B-nodes due to the local minima cases.

3.1.4. Algorithm termination.

The algorithm terminates when the B-message (without the local minima flag) is received at the initiating OB-node. Recall that each OB-node is selected to be on the boundary and that each B-message contains the OB-node ID. This results in correct termination of the algorithm even when multiple OB-nodes are selected to initiate the boundary estimation process.

3.2. Simulations

The boundary estimation algorithm has been implemented in NS2. Simulations were run on different random topologies with 50 nodes. Each node transmits at 0 dBm and the communication range is set to be 16 m (based on Tmote-sky motes calibration tests discussed in Section 3.3). Sensor nodes use 802.11 MAC with the RTS/CTS mechanism turned off and data rate curtailed to 250 kbps. Radio propagation model used is shadowing with a path loss exponent value of 2.0.

Figure 4 shows the simulation results by running the boundary estimation algorithm. Filled circles represent the selected B-nodes and the dashed line between the B-nodes indicate the virtual boundary. OB-node has been marked with solid perimeter. The simulation results show that the boundary estimation algorithm successfully identifies the virtual boundary of the deployed network.

3.3. Experiments

To validate our simulations results, we ran experiments with Tmote-sky devices using the TinyOS operating system. We started by conducting preliminary calibration experiments to ascertain the effect of radio antenna orientation and the height above ground on radio range of Tmote-sky.
motes. The aim is to find a realistic value of the Tmote’s transmission range in the given outdoor environment.

### 3.3.1. Calibration experiment setup.

Our experimental setup is shown in Figure 5. There is a single sender with 20 receivers arranged in orthogonal directions. The sender’s internal antenna is pointing towards right. The motes are placed at 5 m distance increments in four directions (arranged as to give 0, 90, 180, and 270° antenna orientation with respect to the sender antenna). The sender transmits at 0 dBm transmit power with a packet rate of 10 packets per second for 25 min. The motes were horizontally mounted at 0.1 and 1 m above ground in separate set of experiments. Each receiver logs the packet sequence number, RSSI (received signal strength indicator), and LQI (link quality indication) values for each packet that it receives. All these experiments were conducted in an open unobstructed field with a new set of batteries.

### 3.3.2. Calibration experiment results.

Figures 6–8 show our calibration experiment results.

#### 3.3.2.1. Effect of height

Figure 6 shows the percentage packet loss, RSSI and LQI values versus distance plots for both 0.1 and 1 m height above ground experiments. The plotted values are the average of the values measured in four directions. The error bars show the maximum and minimum values received in any direction while the range of the error bars shows the variation due to receiver orientation in different directions.

Results show that for 0.1 m mounting, average of about 52% of the transmitted packets are lost in all directions at 15 m distance from the sender while for 1 m mounting case, low packet loss rate (less than 10%) is observed for all receivers. Also much better RSSI and LQI values are received when both the sender and the receiver are placed 1 m above the ground as compared to 0.1 m. The reason for lower RSSI at 0.1 m mounting is that more transmitted energy is absorbed by the ground at 0.1 m than at 1 m mounting. For 1 m mounting, RSSI and LQI values show irregularity with distance as higher values are received at 20–25 m than 15 m receiver distance. Low variation in RSSI values for correctly received packets is observed in different directions as compared to the LQI values. These results are consistent with those reported in Reference [23] where the LQI values show a higher range of errors. It is clear from these calibration results that height above ground has a significant effect on the performance of the Tmote radio, affecting the transmission range and the link quality.

#### 3.3.2.2. Effect of orientation

Figures 7 and 8 show the packet loss and RSSI values received in different directions for 0.1 m mounting at a distance of 15 m from the sender. Results indicate that the packet loss rate not only varies spatially, different in different directions, but also has a temporally varying behavior. Moreover, low packet loss rate with more fluctuations is observed when the sender and receiver antennas are aligned with each other. Figure 8 shows that a maximum difference of about 8 dBm in RSSI
values is observed in different orientations. As the data sheet for Tmote sky motes indicates the accuracy of the ChipCon radio RSSI values as ±6 dBm, we can conclude that the observed variation in RSSI values in different directions is mainly due to the inherent hardware characteristics.

As 0.1 m mounting above ground is a more realistic deployment option, we analyzed the calibration results with 0.1 m height above ground for selecting suitable values of maximum and reliable transmission ranges. We selected 6 m as the value of reliable transmission range for Tmote sky devices as this distance value gives an average packet loss rate of less than 10% (Figure 6). Similarly, 16 m is selected as the maximum transmission range giving us an average RSSI value of about -90 dBm a value close to the lowest receiver sensitivity of Tmotes sky motes, and an average packet loss rate of around 55%.

Once the values of maximum and reliable transmission ranges are available, we use the neighbors in nominal range area (NINRA) [24] to find out the numbers of nodes \( N \), with transmission range \( T_r \), required for a given deployment area \( A \).

\[
\text{NINRA} = \frac{N}{A} \pi \cdot (T_r)^2
\]

(1)

For our experiments, we chose to deploy 50 nodes in a 30 m \( \times \) 30 m area giving us a NINRA value of about 6. The value of \( T_r \) has been set equal to the reliable transmission range (6 m).

### 3.3.3. Algorithm evolution based on experiments.

We implemented the boundary estimation algorithm on Tmotes and carefully laid out the exact random topologies used in the simulations in a flat field. Note that we did not use any localization algorithm rather the motes were hard-coded with their location information at the initialization stage. All communications between deployed motes were logged in a mote directly attached to a laptop for debugging and analysis. The initial run of experiments with the implementation of basic boundary estimation algorithm indicated that sometime the algorithm failed to correctly identify the B-nodes. Debugging the results identified the following anomalies:

- Nodes appear in the neighbor list due to the reception of Hello messages but some of these nodes cannot be sent control messages (e.g., a B-Message) due to link asymmetry.
- Some nodes can have bi-directional communication at longer distances while other nodes cannot communicate with nodes at much shorter distances. This results in the presence of unexpected nodes in the neighbor list.

These anomalies results in non-deterministic behavior of the algorithm. Based on the initial experimental results, we modified the boundary estimation algorithm to make it robust to the observed radio behavior and suitable for field deployment.

#### 3.3.3.1. Neighbor discovery

To cater for asymmetric links, for robust neighbor discovery, we employed \( m \) out of \( k \) strategy. The nodes broadcast \( k \) Hello messages and the neighbors only add a node to the neighbor list if \( m \) Hello messages from a neighbor has been received. Note that \( m \) and \( k \) are tunable parameters that reflect the extent of radio irregularity. Larger the ratio of \( m/k \), more stable is the selected neighbor link. For our deployment scenario, based on the calibration tests, we selected 6 and 10 as values for \( m \) and \( k \).

#### 3.3.3.2. Reliable communication

To cater for temporal radio interference, we introduced a special Probe message in the protocol to check the link symmetry before transmitting B-messages. The B-message is sent to the forward B-node only if it receives an Ack for the Probe message from the forward B-node. In case of no Ack, the Probe message is re-sent two more times before the neighbor is removed from the neighbor list and the B-node selects another potential B-node neighbor for probing. A B-message is also re-sent a total of three times in case no Ack is received, before the neighbor is removed from the neighbor list.

#### 3.3.3.3. Constraint on node selection

A constraint was introduced that the forward B-node must be further away from the previous B-node than the node making the decision i.e., distance between current node and forward B-node should be less than that between previous B-node and forward B-node.

### 3.3.4. Experiment results.

Fifty of the Tmotes were programmed with the modified boundary estimation algorithm. Figure 9 shows our experimental results. Comparing these results with those from the simulations in Figure 4, we can observe that some of the boundary nodes that get selected as B-nodes are different in simulations than in experiments. This is because of...
Tmotes showing different communication ranges in different direction, a behavior that is not modeled in simulations. Also note that the mote in the lower left corner does not get selected as B-node. The neighbor announcements from this node fails the 6 out of 10 check.

The modified boundary estimation algorithm can thus reliably identify B-nodes with stable symmetric links in the presence of realistic radio irregularities. These identified B-nodes form the virtual boundary of the region.

4. PHASE I: COVERAGE ESTIMATION

Once the boundary nodes have been identified, static nodes evaluate the existing coverage of the target area (Figure 1). To capture the real world sensing characteristics of sensor nodes, we assume that the signal propagation from a target to a sensor node follows a probabilistic model. In particular, for range-based sensor modalities such as acoustics, radio, etc, the signal strength of the triggering signal decays as a function of distance. This implies that the detection capabilities of these sensors would exhibit similar distance dependant characteristics as opposed to a uniform sensing range. In fact, the unit disk sensing model (based on binary detection model, everything inside the circle always detected with probability 1) over-estimates the coverage achieved by the sensor network and would possibly lead to false negatives i.e., in situations where certain events are not detected, the unit disk model indicates that the location of these events are covered.

This work is based on the path loss log normal shadowing model [25] although it can be extended to incorporate different signal decay models e.g. acoustic signal model (where signal roughly decays at inverse square of distance). We propose the PCA, a novel coverage estimation algorithm. The PCA extends the concept of perimeter coverage proposed by Huang et al. [12], to evaluate the effective coverage that can be provided to the application utilizing the sensor network.

4.1. Technical preliminaries

Using the log-normal shadowing model, the path loss $PL$ (in dB) at a distance $d$ is given by Equation (2).

$$PL(d) = PL(d_0) + 10 \cdot n \cdot \log\left(\frac{d}{d_0}\right) + X_\sigma$$

where $d_0$ is the reference distance, $n$ is the path loss component, indicating the rate at which the path loss increases with distance, $X_\sigma$ is the zero-mean Gaussian distributed random variable (in dB) with $\sigma$-variance (shadowing, also in dB), and $PL(d_0)$ is the mean path loss at $d_0$.

$X_\sigma$ in Equation (2) captures various environmental factors resulting in different received signal values at different locations although the distance between the target and sensor is the same. $PL(d_0)$ can be measured experimentally for given event and sensor characteristics or can be calculated using free space path loss model [25].

Each sensor has a receive threshold value $\gamma$ that describes the minimum signal strength that can be correctly decoded at the sensor. The probability $P_{\text{rec}}$ that the received signal level, $P_{\text{rec}}$, at a sensor $S_i$ will be above this receive threshold, $\gamma$, is given by Equation (5), requiring $Q$-function to compute probability involving the Gaussian process. The $Q$-function is defined as

$$Q(z) = \frac{1}{\sqrt{2\pi}} \int_{z}^{\infty} \exp(-\frac{x^2}{2})dx \quad (3)$$

where

$$Q(z) = 1 - Q(-z)$$

$$\Pr[P_{\text{rec}}(d) > \gamma] = Q\left[\frac{\gamma - P_{\text{rec}}(d)}{\sigma}\right]$$

For a given transmit power and receive threshold value, we can calculate the probability of receiving a signal above the receive threshold value. $\gamma$, at a given distance using Equations (3) and (5).

Figure 10 shows the decrease in detection probability for a sensor based on the shadowing model. The continuous change in detection probabilities with distance can be approximated by discrete values at constant distance increments around the sensor location (shown by dashed lines in Figure 10). Assuming this rate of change in the detection probability to be constant in every direction, we can draw concentric circles around a sensor location to represent its discrete detection probabilities. Each circle thus represents the probability of correctly receiving a signal with strength above receiving threshold at distance equal to radius of the circle.

![Random Topology](image)
For a deployed sensor network, a point in the target region can be covered by more than a single sensor. The cumulative detection probability at a point in the region, $Pr$, is given by Equation (6).

$$Pr = 1 - \prod_{i=1}^{N} (1 - Pr_i)$$

where $N$ is the number of sensor nodes covering a particular point and $Pr_i$ is the detection probability of a point for a sensor $i$.

4.2. Probabilistic coverage algorithm

Assuming that the signal transmit power, $P_t$ (characteristic of the event) and receive threshold for sensor, $\gamma$, is known through experiments and sensor calibration, a probability table, PT (see Table I) can be pre-computed (using Equations (2)–(5)) that provides the discrete detection probabilities at various distances from the sensor.

We define the effective coverage range, $R_{effec}$, of a sensor $S_i$ as distance of the target from the sensor beyond which the detection probability is negligible. For this work, $R_{effec}$ is taken as the distance at which the probability of detection falls below 0.1 (Note that adding an additional sensor with a detection probability of 0.1 in Equation (6) results in increase of between 5 and 1% if the existing detection probability is between 50–90%). Two sensors $S_i$ and $S_j$ are thus considered effective neighbors in a region $A$, contributing to coverage of each other, only if the Euclidean distance between them, $d_{ij}$, is less than twice the effective coverage range $R_{effec}$.

Figure 11 shows the individual as well as the cumulative detection probabilities, for two effective neighbors in a region. The cumulative detection probability is always higher than the individual detection probabilities with the minimal value at the midpoint between the two sensors.

The detection probability at any location is thus increased by contributions from all the sensors covering that point. A location in region $A$ is said to be sufficiently covered if its cumulative detection probability, due to sensors located within the effective coverage range $R_{effec}$ of this location, is equal to or greater than the detection probability desired by the application.

4.3. The algorithm

We adopt a computational geometry based approach and propose PCA to check whether the current deployment supports the required coverage probability or not. We make the following assumptions for this work:

- Location information is available to each sensor node by using some GPS-less sensor network localization scheme [26].
- Communication range of sensors is at least twice $R_{effec}$ to ensure that effective neighbors are able to communicate with each other.
- Transmit power of target ($P_t$) and receive threshold ($\gamma$) for a sensor are known and $\gamma$ is the same for all the sensors.

In the initialization phase of the algorithm, a sensor $S_i$ receives Hello messages that contain the location information from all of its one-hop communication neighbors, including the static nodes, and the B-nodes. It calculates the distances to all such neighbors and keeps them in a list sorted on distances. $S_i$ has two sensing circles with radius $d_{reqd}$ and $d_{eval}$. $d_{reqd}$ is the distance from the sensor providing $\rho_{reqd}$ while $d_{eval}$ is the next distance increment that is greater than $d_{reqd}$ providing a lower detection probability than $\rho_{reqd}$. Both $d_{reqd}$ and $d_{eval}$ are taken from the PT.

Sensor $S_j$ now calculates the virtual boundary formed by the neighboring B-nodes known through Hello messages. If

<table>
<thead>
<tr>
<th>Table I. A sample probability table (PT).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance (m)</td>
</tr>
<tr>
<td>Detection probability</td>
</tr>
</tbody>
</table>

DOI: 10.1002/wcm
the virtual region boundary intersects the circle of \( S_i \) at \( d_{\text{eval}} \), the sensor marks points on the perimeter (e.g., segment of \( S_j \) between \( b_1 \) and \( b_2 \) in Figure 12) that lie outside the virtual boundary of region.

In the next step, neighbor’s contribution towards detection probability is calculated. A sensor \( S_i \) that is a neighbor of \( S_j \) has several concentric circles representing regions of different detection probabilities (Figure 13). Sensor \( S_i \) calculates the cumulative detection probability at intersection of its circle at \( d_{\text{eval}} \) with various circles of neighbor \( S_j \).

Let’s look at an example shown in Figure 12. \( S_j \) calculates the cumulative detection probability using Equation (6) at the point \( x \), the intersection of its circle with radius \( d_{\text{eval}} \) with its neighbor \( S_j \) circle \( c_j \). The segment on perimeter that is covered by the circle \( c_j \) is calculated using the cosines rule.

\[
\cos \alpha = \frac{(d_{\text{eval}}^2 + d_{ij}^2 - c_j^2)}{2 \cdot d_{\text{eval}} \cdot d_{ij}} \tag{7}
\]

where \( \alpha \) is the angle subtended by the segment \( xy \) on perimeter of \( S_i \). Coverage on segment \( yz \) is similar to that in segment \( xy \) as total angle subtended by segment \( xz \) is \( 2\alpha \). This calculation is repeated for all the circles of the neighbor \( S_j \) that are intersecting \( S_i \) circle at \( d_{\text{eval}} \) (see Figure 13). \( C(r, p) \) in Figure 13 represent the circle around \( S_j \) with radius \( r \) providing probability of detection \( p \).

The cumulative detection probability is then placed on a line segment \([0, 2\pi]\) representing the perimeter of \( S_i \) at \( d_{\text{eval}} \) (see lower part of Figure 13). This is repeated for each neighbor until whole perimeter is found covered by probability \( \rho \geq \) required probability. If this happens, PCA can declare that region around the sensor \( S_i \) bounded by the circle with radius \( d_{\text{eval}} \) is sufficiently covered, otherwise, only the region bounded by circle with radius \( d_{\text{eval}} \) is sufficiently covered.

If all the sensors report sufficiently covered perimeters at \( C(d_{\text{eval}}) \), the whole region is sufficiently covered. If a sensor finds that its perimeter is not sufficiently covered, it has identified a coverage hole in the region, an area that is not covered to the required detection probability.

After executing the PCA, if a node finds that its evaluated perimeter is uncovered, the node can calculate a deployment location where a redundant helper node, \( S_h \), can be placed such that the perimeter coverage constraint for the node is satisfied. If \( \rho_{\text{exist}} \) is the existing detection probability in the uncovered segment (\( \rho_{\text{reqd}} < \rho_{\text{exist}} \)), the probability, \( \rho_{\text{help}} \), that can enhance \( \rho_{\text{exist}} \) to at least \( \rho_{\text{reqd}} \) can be calculated by Equation (8) based on the product of individual detection probabilities.

\[
\rho_{\text{help}} = 1 - (1 - \rho_{\text{reqd}})(1 - \rho_{\text{exist}}) \tag{8}
\]

\( \rho_{\text{help}} \) is then used to index the PT, to select \( \rho_{\text{select}} \geq \rho_{\text{help}} \). Coordinates for deployment (location of \( S_h \) that would cover the uncovered segment of perimeter) can be calculated using \( \rho_{\text{select}} \) and \( d_{\text{eval}} \) using simple geometric calculations.

### 4.4. Request aggregation

Each sensor runs the PCA algorithm in a distributed manner. Several sensors facing a common coverage hole can thus discover uncovered perimeters and each can send a request for additional deployment. However, deployment of a single additional sensor may benefit other neighbors facing the same coverage hole. In such a case, we can aggregate requests to reduce the communication overhead of the greedy approach when each sensor is independently sending its own request for deployment.

A node that discovers an uncovered perimeter starts a random wait timer. Upon expiry of this timer, the node broadcasts a Check message announcing an uncovered perimeter and the requested deployment point. Neighboring nodes with uncovered perimeters, that are still in their random wait period, cancel their timers and ascertain whether the requested deployment point fulfills their coverage requirement or not. A node checks if the requested deployment point is within twice \( d_{\text{eval}} \) from its own location and if so, it calculates the potential contribution of a node at the requested deployment point on its uncovered perimeter. If the perimeter coverage become adequate, the node is done. Otherwise, it recalculates its own deployment point and restarts the random wait timer.

This wait and check mechanism ensures that deployment requests are aggregated. This aggregation of information not only reduces the overhead associated with additional communication but also results in deployment of fewer nodes than that required by the greedy approach.
4.5. Simulation study

The value of distance increment used in the PCA controls the computational complexity. Intuitively, lower distance increments for the PT will result in higher precision in estimating the area coverage. If \( n \) represent the number of neighbors for a node, and \( m \) represent the number of discrete distances in the PT, PCA has \( n \times m \) complexity. For a given number of neighbors, choosing a lower value of distance increment for circles will result in a higher value of \( m \) with corresponding increase in computational complexity. However, there is a tradeoff between the granularity/accuracy and the computational time of the algorithm. We ran different simulations (NS2) to study the effect of this approximation on coverage estimation using different values of \( m \). Distance increments used are 3, 1, and 0.5 m for each \( d_{\text{eval}} \) values of 7, 8, and 9 m. Value of \( d_{\text{reqd}} \) is taken as 6 m with 0.9 as the required probability of detection. Different random topologies with 60, 80, 100, 120, and 140 static nodes are generated in 100 m × 100 m region. Values of \( n \) and \( \sigma \) are taken as 2 (free space) and 4 (dBm), respectively.

Figures 14 and 15 show a subset of the simulation results. For 60 nodes, on average PCA reports perimeter coverage at 9 m circle for 20 nodes while binary detection (unit disk) model has only one node with whole perimeter covered with required probability of 0.9. At higher node density of 140, the corresponding average values are 102 for PCA and 39 for binary detection model. It is clear that the binary detection model (with radius = \( d_{\text{reqd}} \)) underestimates the total coverage. For both 1 and 3 m distance increments, the lower the value of \( d_{\text{eval}} \), the higher the number of sufficiently covered nodes. Lower distance increments result in more nodes being identified as sufficiently covered due to precise probability calculations. For \( d_{\text{eval}} \) values closer to \( d_{\text{reqd}} \) (e.g., 7 m), increase in precision has little effect on the overall result.

Note that more number of nodes with a smaller radius reporting sufficient perimeter coverage does not necessarily mean that larger area is covered. This is illustrated in Figure 15 for 60 nodes topology in a 100 m × 100 m region. Unit disk model with 6 m as the sensing range reports 47.2% of the target area as covered. Coverage provided by 7 m (for both 1 and 3 m increments) is about 52% while maximum coverage is estimated as 58.8% in case of \( d_{\text{eval}} \) value of 9 m and distance increment of 1 m. For calculating the area coverage, nodes with insufficient coverage use \( d_{\text{reqd}} \) as the coverage radius.

4.6. Implementation of PCA on Tmote Sky

An integrated implementation of PCA and the boundary estimation algorithm has been done on Tmote sky Mote platform and small scale experiments with 15 Tmotes were conducted. Each Mote applies the PCA on the list of neighbors and reports to the base station whether its perimeter at \( d_{\text{eval}} \) is sufficiently covered or not. The collected data at base station was then analyzed for the coverage. The results with several randomly generated topologies were in accordance with the theoretical simulation results.

Figure 16 show results of running the experiment with 15 Motes in uniform random topology. Coverage with both unit disk model (dark circles) and by applying the PCA (light gray areas) is shown. The coverage is increased from about 70% with unit disk model to about 99% by applying the PCA.

5. Phase II: Coverage Enhancement

Static sensors nodes execute phase-I to estimate the area coverage, discover the presence of coverage holes, and request deployment of mobile nodes to cover these identified coverage holes. In phase-II, some of the mobile nodes are moved to plug these discovered coverage holes, and the rest of the mobile nodes are spread evenly in the target area. It is pertinent to note that there is no loss in existing coverage that is associated with this movement as the mobile nodes do not participate in the probabilistic coverage calculation (Section 4).
Random aerial deployment is extremely challenging for mobile nodes as it may result in physical damage to their locomotive parts. In realistic deployments, the mobile nodes are normally accumulated at one or more points near the target area. We therefore, consider two different initial deployment methodologies namely Normal and Island distribution. In normal distribution, mobile sensors form a single cluster at the boundary, while in island distribution they form disconnected clusters at different locations on the boundary. Note that managing the mobile node’s relocation for these initial deployments is more challenging than assuming randomly distributed mobile nodes. For spreading of the mobile sensor nodes, we propose two variants of VFA: coverage and energy aware VFA (CEA-VFA) and CEA-VFA with SM. These algorithms works in rounds. In each round, the algorithms first moves the mobile nodes to plug in the coverage holes identified by neighboring static nodes (hence called coverage aware). Once this is done, remaining mobile nodes are spread out in the region to enable the discovery of more uncovered areas in subsequent rounds.

We make the following assumptions:

- Location information is available using any existing GPS-less WSN localization scheme for the mobile nodes such as Reference [27].
- The target area is an unknown obstacle-free environment.
- Mobile nodes have significantly more initial energy than the static nodes. For example, the initial energy of a Robomote [28] is 4528J (3.7V Lithium battery), while that of a Mica2 mote is about 3000J (2 AA batteries).

5.1. Coverage and energy aware virtual force algorithm (CEA-VFA)

This paper presents a coverage and energy aware variant of basic VFA proposed in References [9,14]. Basic VFA attempts to iteratively spread the mobile sensors in the target area by using a combination of attractive and repulsive forces. Two mobile sensors will exert virtual forces on each other if the Euclidean distance, \( d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \), between them is not between a given range of thresholds, \( T_{th_{push}} \) and \( T_{th_{pull}} \) (discussed in detail later in this section). This virtual force, \( F_{ij} \), is a pull or attractive force if the distance between the two mobile nodes is greater than the pull threshold, \( T_{th_{pull}} \), while if the distance is less than the push threshold, \( T_{th_{push}} \), a push or repulsive force is exerted. Equation (9) shows the model used for decision making.

\[
\tilde{F}_{ij} = \begin{cases} 
F_{push}, & \text{if } d_{ij} < T_{th_{push}} \\
0, & \text{if } T_{th_{push}} \leq d_{ij} \leq T_{th_{pull}} \\
F_{pull}, & \text{if } d_{ij} > T_{th_{pull}}
\end{cases}
\tag{9}
\]

where \( \tilde{F}_{ij} \) is the force exerted on mobile node \( S_i \) by neighbor \( S_j \). Taking the midpoint of range between \( T_{th_{push}} \) and \( T_{th_{pull}} \) as the desired distance between the mobile nodes, Equations (10) and (11) represent the push/pull virtual forces.

\[
F_{push} = \left( \frac{T_{th_{pull}} + T_{th_{push}}}{2} \right) - d_{ij} \tag{10}
\]

\[
F_{pull} = d_{ij} - \left( \frac{T_{th_{pull}} + T_{th_{push}}}{2} \right) \tag{11}
\]

Representing magnitude of the force in terms of distance, the share of the virtual force for a node is given by Equation (12). We represent \( \tilde{d}_{mj} \) as the force absorbed by node \( i \) due to \( j \), the magnitude of which is denoted by \( |\tilde{d}_{mj}| \).

\[
\tilde{d}_{mj} = \frac{F_{ij}}{2} \tag{12}
\]

where \( |\tilde{d}_{mj}| = |(F_{ij}/2)| \).

We can express the total force exerted on a mobile sensor \( S_i \) by its \( n \) mobile neighbors, denoted by \( \tilde{F}_i \), as

\[
\tilde{F}_i = \sum_{j=1,j\neq i}^{n} \frac{\tilde{F}_{ij}}{2} = \sum_{j=1,j\neq i}^{n} \tilde{d}_{mj} \tag{13}
\]

Note that \( \tilde{F}_i \) is the vector sum of all the forces acting on mobile sensor node \( S_i \), the magnitude and orientation of which can be easily calculated, e.g., Robomote [28] uses an on-board compass combined with localization information for navigation purposes. These virtual force calculations are performed in each round of the VFA and mobile nodes are iteratively moved to attain a more uniform distribution in the region. We now detail the changes incorporated in the basic VFA in the following sections.

5.1.1. Coverage awareness.

In CEA-VFA, a mobile node first checks whether it is in the vicinity of a coverage hole and if so, it reacts to plug in the discovered coverage hole before participating in the virtual force calculations. This coverage check is performed by each mobile node in each round of the CEA-VFA. The coverage check starts with mobile nodes (and B-nodes) exchanging the location information using Hello messages. A Hello message contains the sender node ID, current location coordinates, and the remaining energy. A \( TM_{wait} \) timer is started (and reset) by mobile nodes each time a Hello message is received. Static nodes with uncovered perimeter starts a \( TS_{wait} \) timer on receiving a Hello message from the neighbor mobile nodes. This timer is reset each time a Hello message is received. On expiry of the \( TS_{wait} \) timer, a static node with uncovered perimeter selects the nearest mobile node from its mobile node neighbor list and sends a Help message to the selected mobile node. The Help message contains sender node ID, location, requested deployment point, and the expected gain in coverage (difference between the required detection probability \( p_{req} \) and the existing probability at uncovered perimeter \( p_{exis} \)).
A mobile node may receive multiple Help messages from different static nodes with uncovered perimeters. The mobile node selects the requested deployment point that involves the highest gain in coverage, broadcasts a Move message, and starts moving toward the requested deployment point. As the Move message is a broadcast, it is received by both mobile and static sensor nodes.

Neighboring mobile nodes that receive the Move message, removes the sender node from the list of neighbor mobile nodes (as the mobile node has left its previously announced position). The TMwait is reset when a new Move message is received.

Static nodes with uncovered perimeters also receive the Move message. These nodes check whether the Move message is in response to their Help message. If not, these static nodes now send a Help message to the next candidate mobile node from the neighbor list.

Figure 17 shows this contention resolution using Help and Move messages. Note that all these messages are exchanged with one-hop neighbors only. The mobile nodes after expiry of the TMwait timer use the CEA-VFA to spread out in the topology to discover more coverage holes.

5.1.2. Energy awareness.

To ensure that the mobile nodes do not exhaust all of their energy during the deployment phase, we introduce an adaptive policy based on the residual energy of the nodes. The virtual forces are made proportional to the residual energy of nodes $S_i$ and $S_j$ respectively. We have $e_{ij} = (E_{ci} - E_{cj})/E_i$, where $e_{ij}$ is the proportional energy coefficient. Representing magnitude of the force in terms of distance, the share of the virtual force for a node is given by Equation (14).

$$\tilde{d}m_{ij} = \frac{\tilde{F}_{ij}}{2}(1 + e_{ij})$$

(14)

where $|\tilde{d}m_{ij}| = |(F_{ij}/2)(1 + e_{ij})|.$

We can now express the total force exerted on a mobile sensor $S_i$ by its $n$ mobile neighbors, denoted by $\tilde{F}_i$, as

$$\tilde{F}_i = \sum_{j=1,j\neq i}^{n} \frac{\tilde{F}_{ij}}{2}(1 + e_{ij}) = \sum_{j=1,j\neq i}^{n} \tilde{d}m_{ij}$$

(15)

5.1.3. Choice of thresholds, $Th_{push}$ and $Th_{pull}$.

We want to ensure that the spreading of the mobile nodes results in a connected topology after termination of the VFA. We also want to ensure that mobile nodes are able to communicate with neighbors (mobile nodes) during round by round operation of the algorithm. For this purpose, rather than using static values of movement triggering thresholds, $Th_{push}$ and $Th_{pull}$, these thresholds are made dependant on the link quality. For better quality communication links, sensor nodes can be placed further apart (higher value of $Th_{pull}$ can be used). Although other complex models can also be used, we use a simple radio propagation model based on log-normal shadowing (discussed in Section 4.1) to characterize the communication link between two sensors.

Following the terminology in Reference [29], there are three distinct reception regions in a wireless link: connected, transitional, and disconnected. The transitional region has highly unreliable links and its region bounds can be found either by analytical or empirical methods [29]. Figure 18 shows the probability of successful transmission versus the distance plot using Equations (2)–(5). Let $d_{tr}$ and $d_{dis}$ represent the points where the transitional and disconnected regions begin respectively. We define $Th_{push}$ and $Th_{pull}$ as $(1 - \alpha)d_{tr}$ and $(1 + \alpha)d_{tr}$ respectively, where $\alpha$ denotes the error tolerance coefficient. Note that the values of $Th_{push}$ and $Th_{pull}$ are bounded by $d_{dis}$, $\alpha$ is dictated by the application and it reflects the tolerance to the errors in localization and odometry during navigation of the mobile nodes. As long as the position after movement is within this range, the deviation from the ideal trajectory during movement can be tolerated by our movement algorithm.

5.1.4. Boundary considerations.

To ensure that mobile sensors are confined within the boundary of the region while moving in the unknown environment, B-nodes (identified in phase I) exert virtual

![Figure 17. Contention resolution mechanism.](image)

![Figure 18. Change in communication probability with distance.](image)
forces ($\vec{F}_{ib}$) on mobile sensor nodes. A repulsive virtual force $\vec{F}_{ib}$, based on the mobile node’s distance from a B-node, is included in the resultant vector sum of all virtual forces. The new total force is then expressed as

$$\vec{F}_i = \sum_{j=1, j\neq i}^{n} \vec{F}_{ij}(1 + e_{ij}) + \sum_{b=1}^{k} \vec{F}_{ib}$$  \hspace{1cm} (16)$$

where $\vec{F}_{ib}$ is the repulsive force exerted on the mobile node $S_i$ by its $k$ neighbor B-nodes and is modeled by Equation (17).

$$\vec{F}_{ib} = \begin{cases} \vec{F}_b, & \text{if } d_{ib} < \frac{(Th_{push} + Th_{pull})}{2} \\ 0, & \text{otherwise} \end{cases}$$  \hspace{1cm} (17)$$

The repulsive force $\vec{F}_b$ is given by Equation (18) where $d_{ib}$ is the distance between the node and the B-node.

$$\vec{F}_b = \frac{(Th_{push} + Th_{pull})}{2} - d_{ib}$$  \hspace{1cm} (18)$$

As B-nodes are all static nodes, mobile nodes absorb all of the virtual force resulting from the B-nodes. As a final check, mobile nodes should not cross the virtual boundary formed by the known B-nodes.

5.1.5. Per round synchronization.

As the mobile nodes move variable distances in each round of the CEA-VFA, they require some kind of signaling mechanism to indicate when to start the next round. We achieved this per round synchronization by restricting the maximum distance a mobile node can travel in each round of CEA-VFA. Linking this maximum distance, $d_{max}$, with the energy considerations, a mobile node should not move a per round distance that consumes more than $K_1 \times E_i$ energy, where $K_1$ is a tunable parameter that controls the maximum energy allowed to be consumed in each round of movement and $E_i$ is the initial energy. The maximum time, $T_{round}$, a mobile node may take to complete movement in each round can be calculated as we know the speed, $v_{max}$, of a mobile node, $T_{round} = \frac{d_{max}}{v_{max}}$. Each mobile node waits for time $T_{round}$ before broadcasting its new position in Hello messages to start the new round of CEA-VFA.

5.2. Virtual force algorithm with simulated movement

Mobile nodes move after each round of CEA-VFA due to the virtual force exerted by its neighbors before settling down to their final position in the topology. If we could calculate the final position of a mobile node and move directly to that final position, we can save energy by moving much lesser distances, i.e., through the cut-through paths instead of the zig-zag paths that result from the round-by-round operation of CEA-VFA.

We propose CEA-VFA with SM approach that attempts to use the cut-through paths for movement. Nodes go through the CEA-VFA iterations and calculate new position after each round. The difference is that nodes do not physically move after each iteration, rather, they stay at their original position and simply assume the new calculated virtual position. The Hello messages at the start of the next round contain the new virtual positions enabling the recipients to use this updated position information for the upcoming round of CEA-VFA. Nodes only move once they have calculated their final positions.

This simplistic approach has a few disadvantages. B-nodes, static nodes with uncovered perimeters, and new mobile neighbors are discovered by the per round movements in CEA-VFA. If a fully simulated run of CEA-VFA is used, it is highly likely that the mobile nodes may not detect the presence of the other mobile nodes and B-nodes in the region. Similarly, it is not possible to locate all of the coverage holes.

One way to offset this disadvantage is to use intermittent SM instead of fully SM. In the CEA-VFA intermittent simulated movement (ISM) approach, nodes simulate the movement for $x$ number of rounds. Actual physical movement only takes place after every $x$ number of rounds e.g., in ISM2 nodes physically move after every second round. The SM approach reduces energy consumption due to movement by moving the mobile sensors in fewer number of rounds. This also results in quicker deployment due to reduction in the time spent in the per round zigzag movements of the CEA-VFA.

5.3. Performance evaluation

In this section, we discuss the simulation results that describe the performance of the ISM algorithms as compared to CEA-VFA.

5.3.1. A centralized optimization.

Assignment of mobile nodes to coverage holes identified by static sensor nodes and uniform distribution of the remaining mobile nodes in the topology can be formulated as a matching problem. A centralized optimization can be performed, if we assume that the information of all coverage holes and the total number of mobile nodes is available. Note that this centralized optimization can provide a base line for comparison of the performance of the distributed (and iterative) virtual force-based algorithms.

The assignment problem can be formulated as following: Let $S_m$ represent the set of $p$ mobile nodes, $S_m = \{S_{m1}, S_{m2}, \ldots, S_{mq}\}$ and set $S_p$ represent the location of $q$ coverage holes in the topology, $S_p = \{S_{p1}, S_{p2}, \ldots, S_{pq}\}$. After assigning $q$ out of $p$ mobile nodes, we have $r$ remaining mobile sensors, ($r = p - q$), that needs to be uniformly distributed in the coverage area. The set $S_d$ represent the deployment locations of these $r$ mobile nodes, $S_d = \{S_{d1}, S_{d2}, \ldots, S_{dr}\}$. Let $S_d = \{S_{d1}, S_{d2}, \ldots, S_{dr}\}$ represent
the combined deployment locations with $S_d = S_b \cup S_s$ and $t = q + r$. We want to optimally assign the available mobile sensors, firstly, to locations requested by static nodes and secondly, to locations in the topology so as to form a uniform distribution. This is a classical Assignment optimization problem, also referred to as bipartite weighted matching problem in graph theory. If a mobile node $i$ is assigned to a location, $j$, there is a cost (energy consumption due to movement) of $c_{ij}$. We can minimize the total value of assignment (minimize energy consumption).

The objective function can be defined as

“What is an assignment schedule in order to minimize energy consumption?”

The assignment problem can be formulated as an ILP as follows:

Minimize \( \sum_{j=1}^{t} \sum_{i=1}^{p} c_{ij}x_{ij} \)

With additional constraints that each mobile node can be assigned to exactly one location and each location must have one assigned mobile node.

- \( x_{ij} \leq 1; i = 1, \ldots, p; j = 1, \ldots, t \)
- \( \sum_{j=1}^{t} x_{ij} \leq 1; j = 1, \ldots, t \)
- \( \sum_{i=1}^{p} x_{ij} \leq 1; t = 1, \ldots, p \)

### 5.4. Simulations results

We implemented the B-node selection algorithm, PCA, basic VFA, CEA-VFA, and simulated variants in NS2. Values of $Th_{push}$ and $Th_{pull}$ were chosen as 25 and 33 m respectively. Nodes have initial energy of 4528 J (3.7 V, 345 mAh) while energy consumed in movement is 8.274 J/m (Robomote [28]). Maximum number of rounds was set to 12.

We conducted detailed simulations for basic VFA, CEA-VFA and different variants of ISM (ISM2, ISM3, ISM4, and ISM6) and compared the results with an optimized assignment. In order to perform a fair comparison, basic VFA uses the same link quality-based thresholds and uses B-nodes for boundary forces as for CEA-VFA. The centralized optimization discussed in Section 5.3.1 is performed in MATLAB. Comparing the virtual force-based iterative algorithms with this optimized assignment gives us a measure of two aspects. Are all the coverage holes detected and plugged by the virtual force-based algorithm? How well are the remaining mobile nodes spread as compared to a uniform grid deployment?

Results (Figure 19) show that CEA-VFA performs better than the basic VFA for both normal and island initial deployment, for different numbers of mobile nodes. On average, CEA-VFA causes the mobile nodes to move about 63 and 57% of the total distance moved in case of basic VFA for normal and island deployments respectively. The results are primarily due to the integration of coverage awareness in the VFA that enables CEA-VFA to discover coverage holes and react to plug them in each round of the VFA.

Among the ISM variants, mobile nodes using ISM6 consistently move the least distance for all types of deployment. For 140 static-40 mobile nodes topology, mobile nodes using ISM6 move about 45% of the distance moved by CEA-VFA, and about 37% of the distance moved by the basic VFA. This is because in ISM6, the mobile sensors only move in 2 of the 12 rounds while performing simulations in 10 rounds.

Figure 20 illustrates the empirical error cumulative distribution function (CDF) for both normal and island distributions for 100 static-30 mobile nodes topologies. Errors are calculated as the difference between the desired deployment points and the final topology position achieved by different virtual force-based algorithms. A zero error means that either a coverage hole has been plugged or a perfect grid point deployment has been achieved. For island distribution, the error CDF of ISM3 matches closely with that of CEA-VFA while ISM3 moves about 55% of the total distance moved by CEA-VFA. For ISM6, lesser number of nodes (about 27–33%) report zero deployment error than ISM3 and CEA-VFA (about 43–47%). Also the spread in error CDF is more for ISM6 than either ISM3 and CEA-VFA. Comparing the same movement algorithm for different initial deployments, we observe that island deployment always results in lesser movement.
Simulation results show that ISM variants save a considerable amount of energy by moving the mobile nodes lesser distances than the CEA-VFA for different types of initial deployment. However, this saving is at the cost of slight non-uniformity in the node distribution. Performance of ISM3 is comparable to ISM4 and ISM6 in terms of energy consumption and yet it achieves topology distribution closer to that of CEA-VFA with similar error CDF. To summarize, the simulation results show that ISM3 is a good compromise with significant savings in energy consumption.

Finally, Figure 21 shows the initial and final percentage of area with sufficient coverage (shown by bars) for CEA-VFA and ISM3 with different topologies. Static nodes are randomly deployed in a 100 m by 100 m region. Static nodes estimate their coverage using PCA with required coverage probability ($\rho_{reqd}$) of 0.9 and request assistance from mobile nodes if they find uncovered perimeters. Mobile nodes spread out from an initial island distribution.

For 100-30, 120-35, and 140-40 static-mobile node cases, more than 99% of the area is covered after relocation of mobile nodes for both CEA-VFA and ISM3. For 80-25 configuration, coverage is enhanced from 72 to 94% after execution of 12 rounds of the ISM3 algorithm while the corresponding coverage in CEA-VFA is 96.7%. Figure 21 also shows that for 140-40 topology, ISM3 only consume about 10% of their total initial energy in the deployment phase as compared to close to 19% for CEA-VFA (shown by lines). Also note that for 140 static nodes in 100 m $\times$ 100 m area, on average the initial area coverage is about 90% showing that coverage holes exist even in dense random deployments.

6. PHASE III: COVERAGE MAINTENANCE

Coverage maintenance can be classified in two distinct categories; proactive and reactive maintenance. Proactive coverage maintenance is performed before the actual loss in coverage takes place. This is typically undertaken to replace nodes that have become energy constrained before other nodes in the topology. This non-uniform resource depletion may be due to the edge effect, where nodes form favorable routes for increased data flow owning to reoccurring events in some locations. Similarly, nodes closer to a base station/sink consume more energy than nodes lying far away from it as they have to carry more transient traffic. Reactive coverage maintenance is performed once a coverage loss has already occurred in the deployed topology. This coverage loss can be due to dead or faulty nodes that have become useless without giving a warning. In reactive maintenance some other node has to manage the coverage recovery process.

Given that manual replacement of either the batteries or the node itself is not an option for harsh inaccessible environments or large scale WSN, redundant mobile nodes in a hybrid network are best utilized for coverage repair. However, the energy consumption for movement itself is a costly operation. Relocation thus results in lower total available energy in the network but with a more useful distribution. Our aim is to minimize this relocation cost to improve the system utility.

6.1. Proactive maintenance: replacing low energy nodes

Proactive maintenance aims at replacing low energy nodes before they deplete all of their available energy resource.
6.1.1. Problem formulation.

A replacement schedule should be able to deal with both iterative single node replacement and cumulative replacement of a set of low energy nodes. Thus the replacement problem can be formulated as following:

Let $S_r$ represent the set of $p$ redundant mobile nodes, $S_r = \{S_{r1}, S_{r2}, \ldots, S_{rp}\}$ and set $E_r$ represent their remaining energies, $E_r = \{E_{r1}, E_{r2}, \ldots, E_{rp}\}$. There are $q$ low energy sensors in set $S_l (q \leq p)$, $S_l = \{S_{l1}, S_{l2}, \ldots, S_{lq}\}$. We want to optimally assign the available redundant mobile sensors to replace the low energy nodes. If a mobile node $i$ is assigned to replace a low energy node, $j$, there is a cost (energy consumption due to movement) of $c_{ij}$ and benefit (energy introduced in the network i.e., remaining energies of mobile nodes after the movement) of $e_{ij}$. We can either minimize the total value of assignment (minimize energy consumption) or maximize the total value (maximize energy introduced in the network).

The objective function can either be defined as:

What is a replacement/assignment schedule in order to minimize energy consumption?

or

What is a replacement/assignment schedule in order to maximize energy introduced in the network?

There are additional constraints that each mobile node can be assigned to replace exactly one low energy node and each low energy node must have one assigned mobile node. The assignment problem can be formulated as an ILP as follows:

Minimize $\sum_{j=1}^{q} \sum_{i=1}^{p} c_{ij}x_{ij}$ or Maximize $\sum_{j=1}^{q} \sum_{i=1}^{p} e_{ij}x_{ij}$

Constraints:

- $x_{ij} \leq 1; i = 1, \ldots, p; j = 1, \ldots, q$
- $\sum_{i=1}^{p} x_{ij} \leq 1; j = 1, \ldots, q$
- $\sum_{j=1}^{q} x_{ij} \leq 1; i = 1, \ldots, p$

Note that these optimized solutions using ILP are centralized and serve as benchmarks to compare the performance of our proposed distributed heuristic algorithms.

6.1.2. Heuristic solutions.

The coverage maintenance process consists of two distinct tasks. (i) How to decide whether a mobile node relocation is required or not? (ii) If relocation is required, how to locate a suitable redundant mobile node for relocation? Task 1 basically evaluates whether the death of a low energy node has any effect on the overall coverage of the area. Once it is ascertained that the loss of a low power node will result in reduced coverage, task 2 aims to locate redundant mobile nodes and manage the relocation. Having this overall picture in place, we discuss distributed heuristic solutions to the coverage maintenance problem.

Virtual force-based algorithms described in Section 5 uniformly deploy the redundant mobile nodes in the topology. Once the deployment phase is over, mobile nodes announce their final positions in the topology to register with their static neighbors. Each static node thus maintains a mobile node neighbor list (referred as MNN list).

Random deployment of static nodes may result in redundant nodes that can be turned off without affecting the area coverage e.g., nodes in a cluster. In such a case, no replacement is required. A static node whose remaining energy falls below a pre-defined threshold, $E_{req}$, broadcasts a Low energy message. The neighbors on receiving this Low energy message recalculate their perimeter coverage by excluding the sender of the Low energy message. A neighbor that finds its perimeter coverage is insufficient after removing the contributions of the low energy node, broadcast a Replace message. Other neighbors with insufficient perimeter coverage suppress their broadcast of Replace message on receiving any Replace message. A low energy node that does not receive any Replace message in response to a Low energy message, eventually times-out. This establishes that the node is a redundant node and its death would not affect the area coverage.

A low energy node that has received a Replace message from one of its neighbors, initiates the mobile node discovery process. There are three possible scenarios that we discuss here:

6.1.2.1. A low energy node has one-hop mobile nodes in the MNN list If a low energy node has one-hop mobile nodes in its MNN list, the low energy node broadcasts a Help message with location of the low energy node as the deployment coordinates. This Help message is received by one or more mobile nodes.

If a mobile node receives Help message from only a single static node, the relocation decision is trivial. Mobile node sends an Avail message to the originator of the Help message, showing its willingness to move to the desired location. This Avail message contains mobile node ID, location and its remaining energy $E_r$. In general, a mobile node can receive Help messages from $n$ static sensor nodes and a static node can receive Avail messages from $m$ willing mobile sensor nodes. The task is to plan energy efficient movement of the mobile nodes by resolving contention. For multiple Help messages, the mobile node calculates the distance from its current location to the requested deployment points. It then selects the nearest requested deployment point that involves the least movement. The mobile node then sends an Avail message to the static node that has originated the corresponding Help message.

Willingness from multiple mobile nodes is then resolved at the static node that generated the Help message. For this purpose, we propose two distributed, heuristic solutions namely Heuristic-Minimize-Energy consumption (Hr-Min-E) and Heuristic-Maximize-Energy-Introduced (Hr-Max-E-I). These solutions differ in how a low power node selects a willing mobile node for replacement. For Hr-Min-E, the node compares the distances ($d_i$), from the
requested deployment point to each of the willing mobile nodes, and selects the nearest located mobile node (to minimize the energy consumption associated with movement). For Hr-Max-E-I, it uses the Er values in the received Avail messages to calculate the remaining energies after the movement (energy consumed to travel a certain distance can be easily calculated) and selects the mobile node with highest remaining energy after the movement. It then unicasts a Move message to the selected mobile node.

The mobile node on receiving the Move message broadcasts an Update message and starts moving towards the deployment point. The Update message contains the senders list of neighboring mobile nodes. Mobile nodes that receive this Update message removes the sender mobile node from their neighbor list. Static nodes update their MNN list by removing the sender and adding any unknown mobile node (and marking these as multi-hop neighbors) from the Update message.

Mobile nodes that do not receive a Move message in response to their Avail messages time out and select the next candidate static node from the Help message list to send an Avail message. This is repeated for each entry in the list until a response in the form of a Move message is received. If no Move message is received, the mobile node waits for the arrival of new Help messages.

A low energy static node that has initiated a Help message should receive either one or more Avail messages, or Update messages from all of its one-hop mobile node neighbors. For each Update message that is received, the low energy node continues to build its MNN list with information about multi-hop mobile nodes. If Update messages have been received from all one-hop mobile node neighbors, the low energy node now probes the multi-hop mobile node neighbors in the MNN list. Figure 22 illustrates the mobile node discovery phase of the coverage maintenance process.

6.1.2.2. A low energy node has only multi-hop mobile nodes in the MNN list

The low energy node unicasts the Help message to each known multi-hop mobile node and starts a wait timer. This message is delivered, over multi-hop, to the mobile nodes. If an Avail message is received from one of these mobile nodes, the wait timer is reset. On expiry of the wait timer, if Avail messages have been received, the low energy node applies the heuristic solutions (Hr-Min-E or Hr-Max-E-I) to invite a mobile node for replacement. Note that Update messages are only single-hop broadcasts and thus a low energy node probing a multi-hop mobile node will not receive the Update messages. The wait timer is necessary to safeguard against existence of stale information in the MNN list, whereby the multi-hop mobile nodes have already relocated.

6.1.2.3. A low energy node has an empty MNN list

The low energy node broadcasts a Locate message (with scope = 1, for one-hop neighbors only). Neighbors with populated MNN list reply to the Locate message by sending their MNN list to the low power node. The low power node builds its own MNN list based on the replies and then unicasts the Help message to the discovered mobile nodes. If no replies are received, the Locate message is sent again with an incremented scope value. Maximum value for the scope counter can be set based on the required reaction time/extent of the area.

The mobile node discovery mechanism described is similar to the well known expanding ring search but is more efficient in terms of number of broadcasts. Specifically, the update messages enables the discovery of mobile nodes located up to two hops away from the static nodes that are being probed. Let \( M \) denotes the number of mobile nodes and \( N \) represents total number of static nodes in the topology \( M << N \). Also suppose that \( l \) represents the maximum scope value that will cover all the static nodes in the topology. The additional overhead of \( M \) node broadcasting Update messages on movement, results in saving of \( n_l \) node broadcasts, where \( n_l \) is the number of static nodes between scope value \( l - 1 \) and \( l \).

6.2. Reactive maintenance: replacing dead/faulty nodes

The replacement mechanism described in Section 6.1 does not work for coverage loss due to dead sensors because it requires active participation from the node being replaced. Failure of a sensor node can be detected by its neighbors (loss of protocol messages/link layer Acks etc). These neighbors initiate the coverage calculation phase (PCA, Section 4.2) to check whether the dead neighbor sensor has resulted in any decreased coverage. In case the node failure does not result in any coverage loss, no further action is required.

Neighbors find that loss of the node has resulted in decreased coverage start a random delay timer. Upon expiry of this random timer, a node broadcasts a Proxy message to announce that the sender will act as proxy to manage the coverage maintenance process. Other nodes still in their random wait period, cancel their timers. The node acting as the proxy, now follows the processing described in Section 6.1 to manage the mobile node relocation. The Help messages in this case contain the location coordinates of the failed node as the requested deployment point.
6.3. Simulation results

We perform complete simulation of all the three phases of the MAPC protocol with 100 static-30 mobile nodes topology. Mobile nodes use ISM3 from an initial island deployment to enhance the coverage and after the deployment stage, 16 redundant mobile nodes are available at different locations in the topology. For proactive maintenance, we have 10 low energy static nodes (shown as small filled circles in Figures 23 and 24) lying along high activity data paths to sink. These low energy nodes go through coverage loss check and then initiate the Help messages.

Figures 23 and 24 show the movement schedules for the heuristic and the centralized optimized solutions while Figure 25 shows the quantitative performance comparison. The optimized assignment with minimize energy consumption (referred as Opt-Min-E) resulted in the least consumption of energy (1322 J) due to movements while
Hr-Min-E consumed 1433 J of energy for the replacement schedule. Optimized assignment with maximize energy introduced (referred as Opt-Max-E-I) resulted in introduction of highest total energy (41270 J) with distributed Hr-Max-E-I ranked second best by giving a replacement schedule that introduces 41160 J of energy in the network. The energy consumed for movement in Opt-Max-E-I and Hr-Max-E-I is almost similar.

For reactive maintenance, we assume that all nodes in a circle of radius 12.5 m are destroyed resulting in the creation of a void shown in Figures 27 and 28. Note that any redundant mobile node located inside the circle is also assumed to be destroyed. In the shown topology, eight static nodes (and one mobile node) get destroyed. Neighbors of these dead nodes detect this loss and initiate the node replacement process.
Figure 26 compares the performance of the algorithms in terms of energy consumption and total energy introduced in the network while Figures 27 and 28 show the movement schedules for the heuristic and the centralized optimized solutions. Similar to the simulation results for proactive maintenance, the Opt-Min-E resulted in the least consumption of energy for the replacement schedule. Opt-Max-E-I resulted in provision of highest total energy (32593 J with introduction of eight mobile nodes) while distributed Hr-Max-E-I gave a replacement schedule that introduces 32360 J of energy in the network. Energy consumption for node replacement by using Hr-Max-E-I is the highest among all the four replacement algorithms.

The results show that performance of these distributed, heuristic solutions is comparable with the centralized optimal solutions. The small performance degradation is due to the localized and distributed nature of the heuristic solutions.

7. CONCLUSION

This paper has proposed a comprehensive three phase protocol, MAPC, for ensuring area coverage employing a hybrid sensor network. MAPC utilizes mobile sensor nodes to ensure that adequate area coverage is maintained from deployment through the lifetime of a WSN. To our knowledge, no such comprehensive solution has been presented in the literature that addresses the deployment, coverage enhancement, and coverage maintenance issues for a hybrid sensor network in its entirety.

The primary contribution of this work is a pragmatic approach to sensor coverage and maintenance that we hope would lower the technical barriers to its field deployment. A distinguishing feature is that most of the assumptions made in the MAPC protocol are realistic e.g., practical boundary estimation, coverage calculations based on realistic sensing model, and use of movement triggering thresholds based on real radio characteristics.

First, we proposed a practical boundary estimation scheme with the objective of identifying the nodes lying on the outer boundary of a deployed network. The scheme is distributed and is executed only by a subset of all deployed nodes. The boundary information is utilized by static nodes during coverage estimation and mobile nodes during the coverage enhancement phase of the MAPC protocol. The performance of the boundary estimation scheme has been studied both through discrete event simulations and conducting experiments on real motes.

Second, we proposed a novel coverage estimation algorithm, PCA, that evaluates the area coverage in WSN. PCA takes into account the realistic variations in sensing behavior associated with range based sensors and evaluates probability of successful sensing at discrete intervals around a sensor location. The algorithm takes advantage of the sensing contributions from neighbors for accurate estimation of a node’s area coverage. The performance of the PCA has been investigated by simulation and experiments. Results show that PCA can accurately estimate the area coverage in a region.

Next, a set of integrated deployment and coverage enhancement algorithms have been proposed that are based on VFA. The algorithms use movement triggering thresholds that are based on real radio characteristics. Our evaluations show that, for different types of initial deployments, the proposed movement algorithms result in substantial energy savings, 60–70% as compared to basic virtual force-based algorithms, a major challenge for the success of a WSN.

Finally, this work addressed the problem of coverage maintenance in WSN by considering coverage loss due to damaged and energy depleted nodes. The coverage maintenance problem has been classified into two distinct categories: proactive and reactive maintenance. The coverage maintenance problem has been formulated as an ILP and implementable heuristics are developed that perform close to optimal solutions.

The evaluation results demonstrate that MAPC can enhance and maintain the area coverage by efficiently moving mobile sensor nodes to strategic positions in the uncovered area.

REFERENCES

8. Ren S, Li Q, Wang H, Chen X, Zhung X. A study on object tracking quality under probabilistic coverage in


**AUTHORS’ BIOGRAPHIES**

**Nadeem Ahmed** received the B.E. degree from the University of Engineering and Technology, Lahore, Pakistan and the M.S. and Ph.D. degrees in computer sciences from University of New South Wales (UNSW), Sydney, Australia in 2000 and 2007, respectively. He is currently a research officer with the School of Computer Science and Engineering at UNSW. His research interests include wireless sensor networks and mobile ad hoc networks.

**Salil S. Kanhere** received the B.E. degree in electrical engineering from the University of Bombay, Bombay, India in 1998 and the M.S. and Ph.D. degrees in electrical engineering from Drexel University, Philadelphia, U.S.A. in 2001 and 2003, respectively. He is currently a senior lecturer with the School of Computer Science and Engineering at the University of New South Wales, Sydney, Australia. His current research interests include wireless sensor networks, vehicular communication, mobile com-
puting, and network security. He is a member of the IEEE and the ACM.

Sanjay K. Jha is a Professor and Head of the Network Group at the School of Computer Science and Engineering at the University of New South Wales. He holds a Ph.D. degree from the University of Technology, Sydney, Australia. His research activities cover a wide range of topics in networking including wireless sensor networks, ad hoc/community wireless networks, resilience/quality of service (QoS) in IP Networks, and active/programmable network. He has published over 100 articles in high quality journals and conferences. He is the principal author of the book Engineering Internet QoS and a co-editor of the book Wireless Sensor Networks: A Systems Perspective. He is an associate editor of the IEEE Transactions on Mobile Computing. He was a Member-at-Large, Technical Committee on Computer Communications (TCCC), IEEE Computer Society for a number of years. He has served on program committees of several conferences. He was the Technical Program Committee chair of IEEE Local Computer Networks LCN2004 and ATNAC04 Conferences, and co-chair and general chair of the Emnets-1 and Emnets-II Workshop respectively. He was also the General Chair of ACM Sensys 2007 Symposium.