

# Location-Unaware Sensing Range Assignment in Sensor Networks

Ossama Younis, Srinivasan Ramasubramanian, and Marwan Krunz

Department of Electrical & Computer Engineering  
University of Arizona, Tucson, AZ 85721  
{younis, srini, krunz}@ece.arizona.edu \*

**Abstract.** We study field-monitoring applications in which sensors are deployed in large numbers and the sensing process is expensive. In such applications, nodes should use the minimum possible sensing ranges to prolong the “coverage time” of the network. We investigate how to determine such minimum ranges in a distributed fashion when the nodes are *location-unaware*. We develop a distributed protocol (SRAP) that assigns shorter ranges to nodes with less remaining batteries. To handle location-unawareness, we develop a novel algorithm (VICON) for determining the virtual coordinates of the neighbors of each sensor. VICON relies on approximate neighbor distances and 2-hop neighborhood information. Our simulations indicate that SRAP results in significant coverage time improvement even under inaccurate distance estimation.

## 1 Introduction

Sensor monitoring applications require node collaboration to maximize the network “coverage time,” defined as the time during which a specified fraction of the area is continuously monitored. In this work, we focus on applications in which the sensing process is the dominant source of energy consumption and sensing ranges are adjustable. Examples of such applications are those requiring sensors to send continuous long-range pulses for object detection (e.g., RADAR systems). In these applications, sensing is a continuous active process, while communication and processing are only invoked whenever an *object of interest* is detected. Other example applications are those requiring each sensor to analyze the collected data (e.g., environmental traces or images) before reporting it. Reducing the sensing range in such applications results in significant reduction in the data set to be analyzed, thus conserving energy. Currently, some commercially available sensors are capable of adjusting their sensing levels to control the cost associated with the sensing process (e.g., the Osiris photoelectric sensor [6]).

We study how to assign the minimum possible sensing range to every sensor without degrading field coverage. Selecting the optimal sensing ranges for all the sensors is an NP-hard problem [10] (the simplified version of this problem in which each sensor is either ON or OFF is also NP-hard [2]). In previous research that considered nodes with

---

\* This work was supported in part by the National Science Foundation under grants CNS-0627118, CNS-0313234, 0325979, and 0435490.

adjustable ranges, greedy techniques were proposed for target monitoring [1] or constructing connected covers [10]. However, the problem is more challenging in location-unaware networks in which a sensor is not capable of determining its location or the directions of the incoming signals. This occurs when the sensors can not perform network-wide localization based on location-aware anchor nodes (e.g., in forests or outer space).

## 1.1 Contributions

We develop a distributed sensing-range assignment protocol (SRAP) for location-unaware sensor networks, assuming that every node can tune its sensing range to one of an available set of ranges. In such networks, nodes are not aware of the field “boundary,” and therefore the objective of every sensor is to cover its own maximum sensing region. SRAP employs a novel localized algorithm (VICON) for determining the virtual coordinates of the neighbors of every node prior to range selection<sup>1</sup>. At a node  $v$ , VICON exploits the 2-hop neighborhood information and the estimated distances between  $v$  and its neighbors. VICON employs conservative heuristics to place as many neighbors of  $v$  as possible when the estimated distances are inaccurate or the graph of  $v$ 's neighbors is disconnected. To prolong the lifetime of every sensor, SRAP assigns sensing ranges based on the remaining sensor batteries. SRAP is also superior to previous work in eliminating redundancy.

## 1.2 Related Work

Under fixed sensing ranges, a node can be either ON or OFF. All previously proposed protocols for this model assumed that node locations or directions of neighbors can be estimated (refer to [8] for a list of these protocols). More recent proposals assumed variable sensing ranges. Cardei et al. [1] proposed centralized and distributed heuristics for maximizing the number of set covers (AR-SC) under this model. Their approach assumes synchronized nodes, base station intervention, and knowledge of node positions. We do not assume any of these capabilities and study a more general model. However, we use the greedy approach in [1] as a baseline for comparison. Zhou et al. [10] proposed another greedy algorithm for selecting a connected cover to optimize query execution under variable sensing and communication ranges. They focused on maintaining both network connectivity and field coverage. Our approach can be integrated with the one in [10] to maintain connected covers in location-unaware networks.

The rest of the paper is organized as follows. Section 2 provides the problem formulation. Section 3 introduces the VICON (VIRtual COordinates of Neighbors) algorithm. Section 4 provides details of the SRAP protocol and its properties. Section 5 evaluates the performance of SRAP. Finally, Section 6 gives concluding remarks.

## 2 Problem Statement

**Assumptions:** Let the maximum transmission range of each node be  $R_t$ . We refer to a node within distance  $\leq R_t$  as a “neighbor.” We assume the following: (1) nodes

---

<sup>1</sup> Note that SRAP is independent of VICON.

are stationary; (2) each node has a set of  $k$  usable sensing levels, which correspond to sensing ranges  $R_1, \dots, R_k$ , where  $R_k$  is the maximum sensing range. Turning off the sensing component corresponds to  $R_0 = 0$ ; (3) energy depletion is proportional to  $R_i^m$ , where  $1 \leq i \leq k$  and  $m$  is a constant  $\geq 1$ ; (4) a node can sense an event within a circular “sensing region” around it; (5) the sensing component in each node is continuously active and the sensing process is energy-intensive. The radio component, however, employs a low duty cycle; (6) neighbor locations and directions of received signals cannot be estimated; and (7) a node can estimate the distance between itself and a neighbor based on well-known techniques such as the time of arrival, received signal strength, etc [9]. For simplicity, we assume that  $R_t \geq R_k$ . We use a conservative approach to estimate neighbor distances in which  $R_t$  is divided into a discrete set of  $n_d$  distances and every range of signal strengths maps to one of these distances. Every node broadcasts the estimated distances to its neighbors so that every node is aware of its 2-hop neighborhood. We account for the inaccuracy in distance estimation in our algorithm presented in Section 3 and evaluate its effect in Section 5.

**Objectives:** Given a set of  $N$  deployed sensors, it is required to assign every sensor  $i$ ,  $1 \leq i \leq N$ , the minimum sensing range  $R_j$ , where  $0 \leq j \leq k$ , such that  $i$ 's sensing region is covered. Because the field boundary is unknown to individual sensors, the objective of every sensor is to ensure that its maximum sensing region is covered.

### 3 The VICON Algorithm

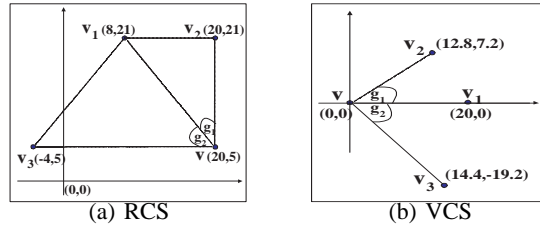
In VICON, a node computes “virtual” coordinates of its neighbors. A virtual coordinate space (VCS) of node  $v$ 's neighbors is one that keeps the *connectivity profile* of the real coordinate space (RCS). That is, the distances and angles between the neighbors of  $v$  are preserved. However, the VCS can have the neighbors rotated, which does not affect the coverage properties.

The problem of assigning neighbor coordinates is a special instance of the “graph embedding” problem, which was studied extensively in the literatures of graph theory and computational geometry [3]. Computing virtual coordinates was also studied in the networking literature, e.g., [5, 7]. In these studies, the objective was to assign coordinates to all the nodes in the network. Such approaches are not suitable for our work for three reasons. First, we do not have anchor nodes in the network since the entire network is location-unaware. Second, basic triangulation techniques do not handle disconnected graphs and fail when distances are inaccurate. Finally, we only require each node to compute the virtual coordinates of its neighbors, and do not need to compute network-wide coordinates. Our approach is a lightweight algorithm that can be easily employed in dynamic networks where new nodes are deployed at any time. We first describe VICON assuming accurate estimates of distances. Then, we extend it to mitigate the negative effects of inaccurate distance estimation.

#### 3.1 Details of VICON

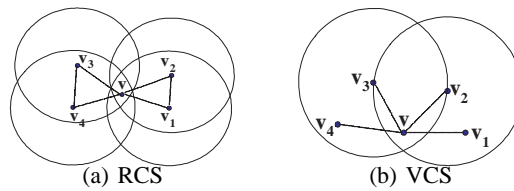
Prior to executing VICON, each node is aware of its 2-hop connectivity information (reachability and distances) through neighbor broadcasts. A node  $v$  executing VICON

proceeds as follows. Assume that  $v$  has three neighbors  $v_1$ ,  $v_2$ , and  $v_3$ , as depicted in Fig. 1(a). Node  $v$  assumes that it is positioned at the origin and places its first neighbor ( $v_1$ ) at  $(d_1, 0)$ , where  $d_1$  is the distance between  $v$  and  $v_1$  (see Fig. 1(b)). Using  $\|v_1, v_2\|$ ,  $\|v, v_1\|$ , and  $\|v, v_2\|$ ,  $v$  can compute the angle  $g_1$  shown in Fig. 1(a). To determine the virtual coordinates of  $v_2$ ,  $v_2$  is rotated by an angle  $g_1$  from the origin in the counter-clockwise direction. Similarly,  $v_3$  is rotated in the counter-clockwise direction with an angle  $g_2$  and assigned a tentative coordinate. The validity of this coordinate is then tested against all the already-placed sensors to determine whether the original connectivity is preserved. In this example, rotating  $v_3$  in the counter-clockwise direction causes it to be a neighbor of  $v_2$ , which contradicts with the RCS. Therefore,  $v_3$  is rotated by an angle  $g_2$  in the clockwise direction. Figure 1(b) illustrates that  $v$  is still covered by three nodes that are within  $g_1 + g_2$  total angle, and are all on one side of  $v$ .



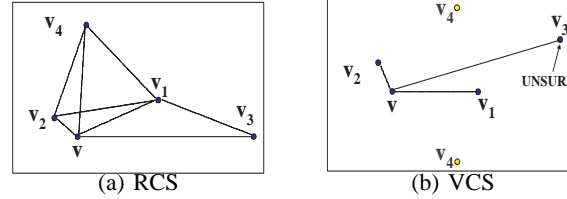
**Fig. 1.** Executing VICON at a node  $v$  to determine the virtual coordinates of  $v$ 's neighbors.

Two problems have to be considered. The first problem is depicted in Figure 2(a), where  $v$ 's neighbors form more than one connected component. This results in having a subset of the neighbors unable to find reference nodes that are already placed in the VCS. VICON handles this problem as follows. First,  $v$ 's neighbors are divided into groups, where each group represents a connected component (e.g., Fig. 2(a) shows two groups:  $\{v_1, v_2\}$  and  $\{v_3, v_4\}$ ). Second, the coordinates of the neighbors in each group are computed independently from the other groups. Finally, each group, other than the first one, is rotated to preserve the RCS connectivity. This is depicted in Fig. 2(b), where the two groups are placed closest to each other while preserving their disjointness.



**Fig. 2.** Assigning coordinates to disjoint neighbor groups.

The second problem is that a node may satisfy the connectivity requirements with the already-placed neighbors in both the clockwise or counter-clockwise direction. The problem is demonstrated in Fig. 3(a), where node  $v_3$  is a neighbor of node  $v_1$  but not of  $v_2$  or  $v_4$ . In the VCS,  $v_1$  and  $v_2$  are placed first. Node  $v_3$  is placed in the counter-clockwise direction from  $v_1$  (as shown in Fig. 3(b)) and it also satisfies the connectivity of the RCS when placed in the clockwise direction. As a result,  $v$  fails to determine a virtual coordinate for  $v_4$ .



**Fig. 3.** Failure to compute the virtual coordinates of  $v_4$  due to incorrect placement of  $v_3$ .

The above problem can be addressed using the following recursive approach. Assume that node  $v$  has a list of  $N_{nbr}$  neighbors. Node  $v$  processes these neighbors in sequence and pushes the IDs of the successfully placed neighbors in a stack named *FinishedNbr*. A neighbor that can be successfully placed in two positions is marked “UNSURE” in *FinishedNbr*, while a neighbor that can only be placed in one position is marked “SURE.” If  $v$  fails to compute coordinates for a neighbor  $i$  ( $2 < i \leq N_{nbr}$ ), then it pops neighbor IDs from *FinishedNbr* until it finds one referring to an UNSURE neighbor. This neighbor is then placed in the alternative direction, marked SURE, and pushed back in *FinishedNbr*. VICON then attempts to re-process the pushed-out neighbors. This approach ensures that incorrectly selected coordinates are corrected as more neighbors are placed. In our example depicted in Fig. 3(b), node  $v_3$  is marked UNSURE when placed. When  $v$  fails to place  $v_4$ , it pops  $v_3$  from *FinishedNbr*, places it in clockwise direction relative to  $v_1$ , then successfully places  $v_4$ .

VICON does not preserve the directions of neighbors, which is not a problem since the objective of every node is to determine “how much” area is uncovered, and not “which” area. Pseudo-code and proof of correctness of VICON can be found in [8].

### 3.2 VICON Under Inaccurate Distance Estimation

Inaccurate distance estimation may cause failures in node placement due to either magnifying or shrinking the angles between a node and its neighbors. We conducted numerical experiments under different settings to study the reasons behind this failure. These experiments revealed two important observations: (1) placement inaccuracy within a maximum inaccuracy  $I = R_t/n_d$  can be tolerated without sacrificing coverage, and (2) our distance estimation is overconservative for some distances, and less conservative for others. Based on these observations, we extend the basic VICON algorithm as follows.

- Assume that the distance  $d$  between node  $v$  and one of its neighbors  $u$  corresponds to the discrete distance  $\hat{d}$  ( $\hat{d} \geq d$ ). We set  $u$ 's distance from  $v$  to be  $\hat{d}-I/2$  to achieve average uncertainty in distance estimation instead of maximum uncertainty.
- The computed virtual coordinate of a neighbor  $u$  is acceptable if it preserves neighborhood within a distance  $\leq I$  of all  $u$ 's neighbors.

Note that under high densities, the shift in the angles can add up and result in failure to place some neighbors. Thus, the above measures do not ensure that all the neighbors will be eventually placed. In [8], we show the effect of node density on successful neighbor placement.

## 4 SRAP Protocol

**Protocol Design:** SRAP assigns longer ranges to nodes with higher “weights,” where a dynamic parameter is used to represent the weight of a node (e.g., remaining energy). In addition, SRAP is re-triggered at fixed intervals of time, referred to as the *cover update interval*  $t_{cu}$ , to efficiently balance the load among sensors.

The SRAP protocol is executed at every node in the network, typically via timer expiration<sup>2</sup>. Since sensor clocks are typically unsynchronized, the node with the fastest clock in its 1-hop neighborhood sends a message to its neighbors to trigger the execution of SRAP. Consequently, every node that receives this message sends a similar triggering message prior to executing SRAP. We assume that a node can be in one of two states: DECIDED or UNDECIDED and all the nodes start in the UNDECIDED state. SRAP has three phases: Phase I is for initialization, Phase II is the core operation of SRAP in which a node  $v$  decides on a sensing range  $R$ , and Phase III is for the optimization of  $R$ . A summary of the three phases is shown in Fig. 4.

**Phase I.** In the first phase of SRAP,  $v$  computes a real-valued weight  $wgt(v)$  as:  $wgt(v) = E(v)/E_{max}$ , where  $E(v)$  is the remaining energy in  $v$ 's battery and  $E_{max}$  is the maximum battery capacity. A neighbor discovery process is then initiated in which  $v$  broadcasts  $wgt(v)$ . Based on the replies that  $v$  receives, it broadcasts its neighborhood table (which includes the estimated neighbor distances). In the second step of this phase,  $v$  executes VICON to compute the virtual coordinates of its neighbors (this step is independent of SRAP). The final step is to check whether  $v$  *has to use* its maximum sensing range  $R_k$  or not. This is done by having  $v$  assume that all its neighbors are using  $R_k$ , and check if any part of its sensing region is not covered. If  $v$  passes this test, it quits SRAP and uses  $R_k$ . Otherwise,  $v$  executes Phase II.

**Phase II.** Node  $v$  computes its sensing range  $R$  based on its weight and the weights of its neighbors. Node  $v$  does not make a decision on  $R$  unless it has the highest weight among all of its undecided neighbors. This gives a chance for nodes with higher weights to decide first and choose longer ranges. At the same time,  $v$  sets a timer  $T_1$  (similar for all nodes) for this phase. If  $T_1$  expires before a decision is made,  $v$  computes its range assuming that all undecided neighbors use  $R_0$ .

<sup>2</sup> SRAP can be triggered asynchronously when detecting events such as node failures or when new nodes are deployed.

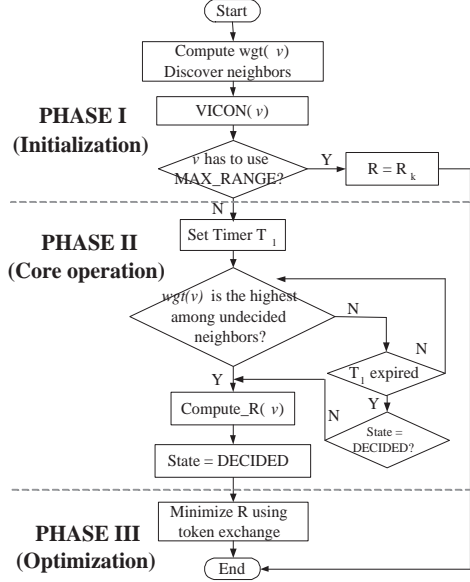


Fig. 4. The SRAP protocol executed at node  $v$ .

The function  $Compute\_R(v)$  proceeds as follows. Node  $v$  first sets its range  $R$  to  $R_{k-1}$  and sets the range of every undecided neighbor  $u$  to the largest  $R_j$  smaller than  $\lceil (wgt(u)/wgt(v))^{1/m} \times R \rceil$ , where  $j \leq k-1$  and  $m$  is a constant. Note that  $wgt(u)/wgt(v) < 1$ , which means that  $v$ 's undecided neighbors are assumed to use sensing ranges  $< R_{k-1}$ . Decided neighbors are set to the ranges that they have decided on. If this assignment results in covering the sensing region of  $v$ ,  $v$  sets its range  $R$  to  $R_{k-2}$  and the same process is repeated. If range  $R_i$ ,  $0 \leq i < k$ , fails to ensure complete coverage of  $v$ 's region, then  $v$  uses  $R = R_{i+1}$ , changes its state to DECIDED, and advertises  $R$  to its neighbors<sup>3</sup>.

**Phase III.** After Phase II,  $v$  can terminate SRAP and use its selected  $R$ . However, redundancies may have been introduced due to the order of the decision-making process in Phase II. Therefore, we propose an iterative approach for removing redundancies. When the node with the least weight in its neighborhood selects its sensing range, it sends a token to its neighbors, allowing them to proceed with Phase III. A node  $v$  starts a timer  $T_2$  (of a few seconds granularity) when it receives the first token from one of its neighbors. It waits to receive tokens from all the neighbors with less weights than its own. Once these tokens are available,  $v$  computes its final sensing range based on the advertised ranges of its neighbors, and releases a token that advertises the new range of  $v$ . If  $T_2$  expires before  $v$  has received enough tokens, it keeps its range as computed in Phase II and releases its token.

<sup>3</sup> Note that loss of messages in Phase II may only result in more conservative estimation of  $R$ . However, termination is not affected.

**Analysis of SRAP:** We analyze the SRAP protocol in terms of its correctness, computational complexity, and message overhead.

**Lemma 1.** *When SRAP terminates, the sensing region of every node with non-depleted battery is completely covered (Coverage property).*

**Proof.** When SRAP is executed at node  $v$ , two operations affect the final coverage of  $v$ 's sensing region:

**1. Selection of  $v$ 's sensing range in Phase I and II.** In Phase I, if  $v$  determines that its sensing region can not be completely covered by its neighbors, it sets its sensing range to  $R_k$  and terminates SRAP. If  $v$  goes through Phase II, it selects its sensing range based on both *advertised* ranges of its decided neighbors and *hypothetical* ranges of its undecided neighbors. An undecided neighbor will not be able to select a sensing range less than the largest hypothetical range made by any of its neighbors, unless its region is covered. This ensures that  $v$ 's sensing region is completely covered.

**2. Reduction of sensing ranges in Phase III.** A "hole" in field coverage may occur when two neighboring nodes (e.g.,  $v_1$  and  $v_2$ ) are allowed to reduce their sensing ranges simultaneously. Such scenario implies that both  $v_1$  and  $v_2$  had all the tokens they need from their neighbors with less weights to start Phase III simultaneously. This is not possible since we assume that the weight is a real number and either  $v_1$  or  $v_2$  will have less weight than the other.  $\square$

**Lemma 2.** *Every node  $v$  selects the minimum sensing range that satisfies coverage of its sensing region if accurate distances are used and the neighbors of  $v$  do not form multiple disjoint components (Minimality property).*

**Proof.** Let us first assume that the estimated neighbor distances are accurate; i.e., VICON computes virtual coordinates for all the neighbors of  $v$ . Also assume that  $v$  has selected  $R = R_i$  although  $R = R_j$  ( $j < i$ ) was sufficient to have  $v$ 's region covered. This may occur in Phase II depending on the order of SRAP execution among neighboring nodes of  $v$ . However, when Phase III is executed,  $v$  will be able to compute the minimum range  $R$  based on its neighbors final decisions. Since nodes getting tokens after  $v$  are only allowed to *reduce* their sensing ranges, the selected  $R$  is minimal (this applies to all nodes). Along with selecting covers from higher-weight nodes and refreshing covers, this result has a significant impact on the perceived coverage time.

If accurate distances are used for computing virtual coordinates, minimality can only be violated if the neighbors of  $v$  form multiple disjoint components. This is unlikely to occur, however, in dense networks. On the other hand, if inaccurate distances are used for obtaining virtual coordinates, minimality can be violated. The redundancy introduced in this case depends on node density and distribution in the field.

**Message overhead.** Four types of message exchange are required: (1) neighbor discovery, which requires  $O(1)$  messages, (2) advertisement of node's weight, which requires only one message whenever SRAP is re-triggered, (3) advertisement of the selected range, which requires  $O(1)$  messages, and (4) token exchange, which requires one message. Therefore, the total message overhead of SRAP is  $O(1)$  per node.

Note that if in future applications more parameters are added to the node weight computation (e.g., mobility, remaining uncovered area, etc.), then a node  $v$ 's weight has

to be advertised whenever one of  $v$ 's neighbors decides its range. This raises the message overhead to  $O(N_{nbr})$ , where  $N_{nbr}$  is the average number of neighbors per node (to ensure connectivity,  $N_{nbr}$  must be  $= O(\log N)$  in randomly deployed networks [4]). Note that Phase III may have to be modified in this case since it relies on a parameter that is assumed to be fixed during the range assignment process.

**Time complexity.** The time complexity of SRAP has two components: (1) convergence speed of any node in the network (ignoring the timers in the protocol), and (2) processing complexity. Phase I has  $O(1)$  convergence speed. The average-case convergence speed is proportional to the average number of neighbors of any node. The worst-case convergence speed for phases II and III can be proportional to the number of nodes (under very pessimistic distribution of nodes' remaining battery levels). (In our experiments, we found that the convergence speed of SRAP is significantly less than the worst case.) This justifies the use of timers  $T_1$  and  $T_2$  to limit the convergence speed and avoid indefinite waits in case of failures.

The main processing complexity in SRAP is in testing whether the node's sensing region is covered. This test is performed once in phases I and III, and every time a neighbor selects its range in Phase II. If we discretize the sensing region into a number of  $P$  points, then the complexity of the test is  $O(PN_{nbr})$ , where  $N_{nbr}$  is the average number of neighbors. The other source of complexity is the VICON algorithm, which is executed only once in static networks. VICON has a complexity of  $O(\alpha N_{nbr}^2)$  where  $\alpha = 2^5$  in the worst case since there can exist at most five neighbors of a node that are pairwise non-neighbors. Each of these neighbors can be assigned to at most two positions. Therefore, VICON does not introduce significant computational complexity.

## 5 Performance Evaluation

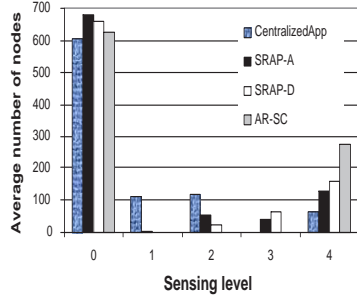
We study an operational scenario in which a number of sensors send their reports to a base station via multi-hop communication. We assume that nodes that are randomly distributed in a field from (0,0) to (50,50). A base station is placed at (25,25). All the nodes start with full batteries and the network is considered dead when the base station is disconnected. The simulation parameters used in our experiments are as follows:  $N = 900$  nodes,  $R_t = 5$  meters, number of discretized distances  $n_d = 5$ ,  $k = 4$ ,  $R_k = 5$  meters, battery capacity = 1.0 Joule, communication energy  $E_{comm} = 10^{-6}$  Watt, energy consumption parameter  $m = 2$ , and cover update interval  $t_{cu} = 2000$  seconds. For radio communications, we assume that a fixed amount of power is consumed from every active node during its operation. We set the energy consumed in communication to correspond to the energy consumed at  $R_1$ .

We developed a discrete event-driven simulator that is scalable and efficient for large-scale networks. We compare SRAP to AR-SC [1]. AR-SC is a distributed protocol proposed for target coverage. However, we extend it to area coverage by discretizing the field into a large number of points. AR-SC gives priority in decision-making (range assignment) to nodes seeing more uncovered targets. This is similar to typical set cover algorithms that aim at reducing the size of the selected set (e.g., [2]). We assume ideal conditions for the operation of AR-SC, which include full node synchronization, op-

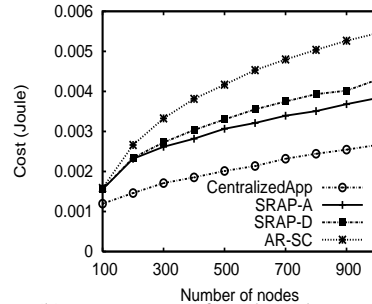
timal sequence of decision-making according to node priorities, and knowledge of the exact node coordinates.

We also compare SRAP to a generic centralized greedy algorithm (which we refer to as “CentralizedApp”). In CentralizedApp, a centralized entity that is aware of the locations of all the nodes in the network is responsible for range assignment. The network operation is divided into phases of equal duration. Given the energy spent by each sensor at the end of phase  $i$ , the minimal cover for phase  $i + 1$  is chosen such that the maximum energy spent by a sensor at the end of phase  $i + 1$  is minimized. The algorithm selects a minimal cover as follows. All the sensors are assumed to employ the maximum sensing range for phase  $i + 1$ . The sensors are arranged in the descending order of the expected energy spent at the end of phase  $i + 1$ . The algorithm selects the sensor with the highest value (say  $v$ ). If reducing  $v$ 's range by one step (i.e., from  $R_j$  to  $R_{j-1}$ ,  $0 < j \leq k$ ) violates coverage, then  $v$ 's sensing range is kept at  $R_j$ . Otherwise,  $v$ 's sensing range is reduced to  $R_{j-1}$ . The expected energy spent at the end of phase  $i + 1$  is updated, as well as the ordered set of sensors. The procedure is repeated until a minimal cover is obtained. Although, the algorithm is described here as a centralized manner, it may be distributed in a distributed manner using only one-hop neighborhood information. As the sensors reduce their range one step at a time, the worst-case running time of the algorithm is  $O(Nk)$ .

We study the operation of SRAP under accurate distance measurements (SRAP-A) and under discretized (inaccurate) distances (SRAP-D). For a fair comparison, we assume that the network boundary is not known by the application employing any of the compared techniques. We assume no packets are lost at the MAC layer. Packet losses may only add some redundancies to field coverage but have no impact on the operation of SRAP. The results provided below are the average of 10 experiments.



(a) Average number of nodes at each sensing level



(b) Energy cost of a selected cover.

**Fig. 5.** Properties of SRAP.

**Properties of SRAP and VICON.** We first focus on the selection of one cover by any of the compared algorithms. All the nodes are assumed to be alive. We compute the number of nodes selected at the  $k$  sensing levels (in addition to  $R_0 = 0$ ), the cost of the

selected cover (energy consumed in the network during its operation), and the ratio of successfully placed neighbors per node for SRAP-D.

Figure 5(a) demonstrates the number of nodes at each sensing level for different node densities. SRAP shows more collaborative behavior under both accurate and discretized distances than AR-SC. CentralizedApp shows the best collaborative behavior because it can reduce the ranges iteratively and not at one step per node as in SRAP and AR-SC. Figure 5(b) shows the cover cost for all algorithms. With maximum sensing ranges, the energy consumed in the network can be computed as  $(R_4 \times R_4 + 1) \times E_{comm}$  (which is 0.0234 Joule for  $N = 900$ ). As expected, CentralizedApp gives the smallest cover cost. SRAP significantly reduces the cover cost over AR-SC, especially when distances are accurately estimated. SRAP-D and AR-SC show, respectively, about 10-20% and 30-70% increase in cover cost over SRAP-A.

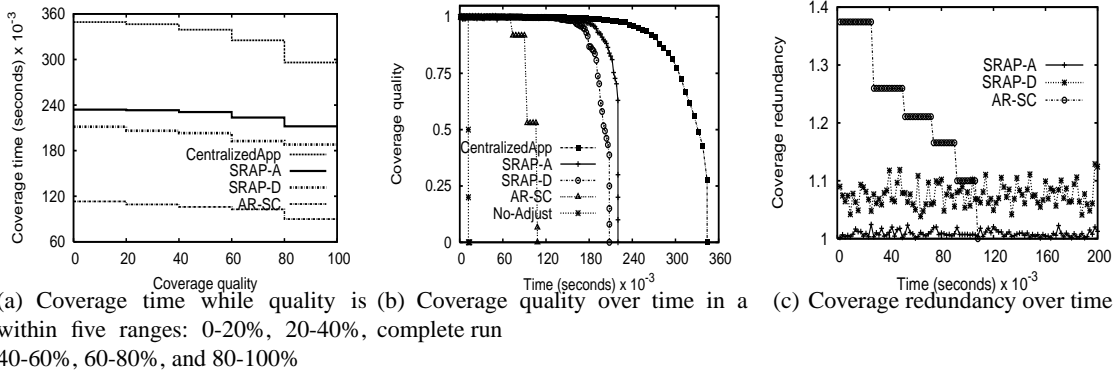


Fig. 6. Performance of SRAP in contrast to AR-SC and CentralizedApp.

**Network Operation.** We now evaluate the network operation when sensing range assignment is employed. We focus on three metrics. The first metric is the duration while the coverage quality of the field is within a specified range. The second metric is the coverage quality over time as the network operates. Coverage quality is the fraction of field coverage at specific instances of time. The third metric is coverage redundancy, which is the number of sensors covering the least covered point within a sensor's region. A coverage redundancy of 1 means that there is at least one point within any sensor's region that is covered by only one sensor. This metric indicates how *minimal* the selected cover and ranges are.

Figure 6(a) shows the coverage time during which the percentage of field coverage is within a specific range of coverage quality. CentralizedApp shows about 50% coverage time improvement over SRAP. SRAP-A and SRAP-D significantly improve coverage time over AR-SC, especially at the higher coverage quality ranges (60-80% and 80-100%). This is a desirable effect for applications that try to maximize field coverage for the longest possible time. Figure 6(b) demonstrate the coverage quality of the field over time as the network is operating. We include results of the application when

operated without sensing range adjustment (referred to as “No-Adjust”), i.e., all the sensors use their maximum sensing ranges ( $R_k$ ). The figure shows that under CentralizedApp and SRAP nodes die smoothly over time because of periodically refreshing the selected sensing ranges based on a dynamic parameter (battery level). We also study the redundancy in the selected covers under SRAP and AR-SC. CentralizedApp guarantees no redundancy in the selected cover and thus is not included in this experiment. Results reported in Fig. 6(c) indicate that for SRAP-A, the redundancy does not exceed 1 by more than 2-3%. We closely examined these redundancies and found that they occur at sensors which are assigned range  $R_0$ . For SRAP-D, redundancy may reach 9-10% due to the failure of VICON to place some neighbors for each node.

## 6 Conclusion

We studied the problem of sensing range assignment in location-unaware networks. To handle location-unawareness, we proposed a novel localized algorithm (VICON) that each node uses to compute virtual coordinates of its neighbors. We then proposed a distributed protocol (SRAP) which periodically assigns sensing ranges to nodes based on their remaining battery powers. SRAP has negligible message overhead and computational complexity. Our simulation results indicate that SRAP significantly improves coverage time, even under inaccurate distance estimation. To extend the functionality of SRAP for different applications, we plan to study how to incorporate other parameters in the node weights, such as mobility, node degree, or potential coverage.

## References

- [1] M. Cardei, J. Wu, M. Lu, and M. Pervaiz. Maximum network lifetime in wireless sensor networks with adjustable sensing ranges. In *Proc. of the IEEE Intl. Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, August 2005.
- [2] U. Feige. A threshold of  $\ln n$  for approximating set cover. *Journal of the ACM*, 45(4):634–652, July 1998.
- [3] J. L. Gross and T. Tucker. *Topological Graph Theory*. John Wiley and Sons, 2001.
- [4] P. Gupta and P. R. Kumar. Critical power for asymptotic connectivity in wireless networks. *Stochastic Analysis, Control, Optimizations, and Applications: A Volume in Honor of W.H. Fleming, W.M. McEneaney, G. Yin, and Q. Zhang (Eds.)*, Birkhauser, 1998.
- [5] T. Moscibroda, R. O’Dell, M. Wattenhofer, and R. Wattenhofer. Virtual coordinates for ad hoc and sensor networks. In *Proc. of DIALM-POMC*, October 2004.
- [6] Osiris Photoelectric Sensors, <http://schneider-electric.ca/www/en/products/sensors2000/html/osiris.htm>, 2007.
- [7] A. Rao, C. Papadimitriou, S. Ratnasamy, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proc. of the ACM MobiCom Conference*, September 2003.
- [8] O. Younis, M. Krunz, and S. Ramasubramanian. Sensing range assignment in location-unaware networks. Technical report, University of Arizona, November 2006.
- [9] M. Youssef and A. Agrawala. The Horus WLAN location determination system. In *Proc. of the ACM International Conference on Mobile Systems, Applications, and Services (ACM MobiSys)*, June 2005.
- [10] Z. Zhou, S. Das, and H. Gupta. Variable radii connected sensor cover in sensor networks. In *Proc. of the IEEE Communications Society Conference on Sensors and Ad Hoc Comm. and Networks (SECON)*, September 2004.