pCover: Partial Coverage for Long-Lived Surveillance Sensor Networks[1]

Limin Wang Sandeep S. Kulkarni

Software Engineering and Network Systems Laboratory

Department of Computer Science and Engineering

Michigan State University

East Lansing MI 48824 USA

Abstract

We present a protocol for partial (but high) coverage in sensor networks. We demonstrate that it is feasible to maintain a high coverage (>90%) while significantly increasing coverage duration when compared with protocols that provide full coverage. In particular, we show that our protocol maintains 94% coverage for a duration that is 2.3-7 times the duration for which existing protocols maintain full coverage. Through simulations, we show that our protocol provides load balancing and that the desired level of coverage is maintained (almost) until the point where all sensors deplete their batteries.

Keywords: sensor networks, node scheduling, partial coverage, lifetime, power conservation

Technical Area: Sensor Networks and Ubiquitous Computing

1 Introduction

Sensor networks have been extensively used in surveillance applications, such as intrusion detection [1, 2], environmental monitoring [3], and forest fire detection systems. In these applications, sensor nodes are deployed in remote environments and must self-organize to establish a wireless communication network. Nodes in the network generate and report data to the remote base station when an event of interest occurs. Examples of interesting events include intruding personnel or vehicles, movement of wildlife, change in temperature, etc.

A surveillance sensor network needs to operate unattended for a long time, usually from several weeks to several months.

[1]Email: `{wanglim1,sandeep}@cse.msu.edu`. Web: `http://www.cse.msu.edu/~{wanglim1,sandeep}`.

This work was partially sponsored by NSF CAREER CCR-0092724, DARPA Grant OSURS01-C-1901, ONR Grant N00014-01-1-0744, NSF equipment grant EIA-0130724, and a grant from Michigan State University.

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **2005**	2. REPORT TYPE	3. DATES COVERED **00-00-2005 to 00-00-2005**
4. TITLE AND SUBTITLE **pCover: Partial Coverage for Long-Lived Surveillance Sensor Networks**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Michigan State University,Department of Computer Science and Engineering,East Lansing,MI,48824**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES **The original document contains color images.**		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**25**	

However, sensor nodes are usually battery-powered, and, hence, can only operate continuously for a few days. Also, it is often very difficult to change batteries after deployment due to the large number and the embedded nature of sensor nodes. Therefore, energy conservation operations are critical for extending network lifetime. In this paper, we consider the approach that sensor nodes are over-deployed in a given area, only a subset of the nodes are in active mode to maintain a certain degree of sensing coverage (based on the desired system functionality), and the remaining ones are put in sleeping mode.

The work on sensing coverage can be broadly classified in terms of those that provide full coverage (single coverage and multiple coverage) and those that provide partial coverage. In full coverage, every point in the network is covered by at least one sensor. While such coverage is desirable in sensitive environments such as military surveillance, it requires a large number of sensors to be awake. In partial coverage, by contrast, only a subset of points in the sensor network are covered and, hence, the number of sensors that need to be awake is reduced. Thus, one of the desirable properties for an algorithm for partial coverage is tradeoff between the degree of coverage and the lifetime of the network. By degree of coverage, we mean the percentage of target area that is covered by working sensor nodes. In particular, an algorithm for partial coverage is especially desirable when it can provide a high degree of coverage ($>$90%) while significantly increasing (more than doubling) network lifetime compared to the algorithms that provide full coverage.

Another desirable feature of an algorithm that provides partial coverage is that the algorithm should be local. This has several implications. First, the algorithm should not be dependent on global properties such as time synchronization. Second, the decision of a node whether to stay awake or go to sleep should be made locally by individual nodes, based on its knowledge of neighbors' status. Third, the degree of coverage should be controllable locally. This is needed when network characteristics change. For example, sensing range changes due to environmental changes (grass, temperature, rain, etc.) or reduced battery level. In such case, we should be able to dynamically tune the degree of coverage by simply sending a parameter to all the sensor nodes in the network.

At any time, a subset of sensor nodes are actively monitoring the target area. For load balancing purpose, sensor nodes rotate their working load so that different sets of active nodes are selected at different time. A protocol that provides partial coverage should ensure that the degree of coverage is maintained during the process of node switching. Moreover, in a network with uniform deployment, if the energy levels of sensor nodes are well balanced, most sensor nodes will run out of power around the same time. The desired degree of coverage should be maintained (almost) to the point where all sensor nodes fail. As discussed in Section 6, along with a local algorithm, this feature can also be used to provide gradually decreasing coverage.

Contributions of the paper.

1. Based on the above motivation, we propose a protocol, *pCover*, which maintains a certain degree of partial sensing

coverage through sleep-awake scheduling of sensor nodes. *pCover* has the following properties:

- Each node makes local decisions as to work or sleep based on its knowledge of its neighbors, thus the algorithm is fully distributed and scalable to large networks.
- Our algorithm does not need time synchronization service, which is required for several existing node scheduling protocols [4, 5].
- It provides a tradeoff between the degree of sensing coverage and the lifetime of the network.
- Stable coverage is maintained as nodes switch between on/off stage.
- The policy of maintaining sensing coverage is independent of node density, as long as the node density is higher than a certain minimum.
- Energy consumption is balanced among sensor nodes. In a network where sensor nodes are uniformly deployed, sensor nodes run out of power essentially around the same time. And the degree of sensing coverage is maintained at desired level (almost) until the point where all sensor nodes die.

2. We conduct simulations to evaluate the protocol, and show the above mentioned properties of the protocol in the evaluation section (Section 4).

3. Compared to protocols that provide full coverage (e.g., [4, 5]), our protocol provides a significantly increased lifetime while providing high level of coverage. In particular, our protocol maintains 94% coverage for a duration that is 2.3 times (respectively, 7 times) the duration for which the protocol in [4] (respectively, [5]) maintains full coverage. Increasing or decreasing the desired degree of coverage shortens or prolongs the network lifetime accordingly.

Organization of the paper. In Section 2, we identify the system model and assumptions of the partial coverage problem. In Section 3, we present the protocol *pCover*. In Section 4, we present the evaluation results and study the tradeoff between the degree of coverage and the network lifetime. We compare related work in Section 5, and conclude in Section 6.

2 System Model and Assumptions

We consider a network with stationary sensor nodes. Each node knows its location (through some localization service) and its sensing range. The sensing ranges of sensor nodes do not necessarily to be the same. The different sensing ranges can be caused by the hardware (antenna, radio unit, etc.), the remaining power levels (which can change during nodes' lifetime), or the territory conditions.

We assume that a node i is able to detect an event within a distance of r_i, i.e., the sensing range of a node is a circle with radius r_i centered at the location of the node. We define the neighbor set of a node i as

$$N(i) = \{j \in R | d(i,j) \leq (r_i + r_j), j \neq i\}$$

where R is the entire set of sensor nodes in the target region, $d(i,j)$ is the distance between node i and node j.

A simple case is where all nodes have the same sensing range r. In this case, the neighbor set of a node i is

$$N(i) = \{j \in R | d(i,j) \leq 2r, j \neq i\}$$

From the definition, we can see that the nodes in the neighbor set of a sensor node i are the only ones that intersect with node i. We assume that the radio communication range is at least the distance to the farthest neighbor (i.e., no less than $2r$, in the case of equal sensing range). This assumption holds for most existing sensor products, such as [6–8]. Furthermore, some existing operating systems for sensor network, such as TinyOS, allow a sensor node to control the transmission power it uses in radio communication. In such a case, to reduce energy utilization and network congestion, a sensor node should use the lowest power level that reaches all its neighbors.

The partial coverage problem we address in this paper can be described as follows. The sensor nodes are deployed with redundant density. Sensor nodes self-organize to achieve a desired degree of coverage. This degree of coverage is maintained until some of the nodes have drained out power such that there are not enough nodes left to satisfy the coverage requirement.

3 *pCover*: Protocol Description

In this section, we present our partial coverage protocol, *pCover*. In *pCover*, a sensor node is either in working mode or sleeping mode. In Section 3.1, we present the local rule that a sensor node follows to determine whether it should go to sleep or stay awake. In Section 3.2, we describe state transition and how this local rule is used.

3.1 Sleep Eligibility Rule

We first define several terms we are going to use when we describe the local rule and the protocol.

Global coverage is the percentage of the target area that is covered by the working nodes. This is also called the *degree of coverage*.

Local coverage of a node is the percentage of the node's sensing area that is covered by its awake neighbors.

Contribution of a node is the percentage of the node's sensing area that is *not* covered by its awake neighbors.

From the above definitions, we can easily get the following equation for any node i:

$$Contribution = 1 - LocalCoverage$$

The basic idea of the sleep eligibility rule is that: a sensor node should turn itself off if and only if its *local coverage* (respectively, *contribution*) is higher (respectively, lower) than a certain threshold, $SleepThreshold$ (respectively, $1 - SleepThreshold$).

To facilitate calculation, we imagine that the target area is covered by a virtual grid (Figure 1). Let us consider node A as an example. When A calculates its eligibility to sleep, it counts the number of grid points that are within its sensing range, called *TotalGrid*. For each grid point x, it checks all its neighbors that are *awake*, to see if any of them can cover x. The number of grid points that are covered by one or more *awake* neighbors is recorded as *CoveredGrid*. By counting the number of grid points, we are able to compute the node's local coverage ($CoveredGrid/TotalGrid$) and contribution, and decide if the node should go to sleep or stay awake. The process of calculating sleep eligibility is shown in Figure 2. Note that when we make judgment on whether a grid x is within a sensor's sensing range or not, we count the grid's center point. If the center point of a grid x is within a sensor node A's sensing range, then we say x is covered by node A. Otherwise, if the center point of x is outside of A's sensing range, then x is not covered by A. Since the size of the grid is substantially smaller than the sensing area, this assumption is reasonable. Moreover, even if we change the definition to require that the entire grid be included for it to be counted, the results are similar.

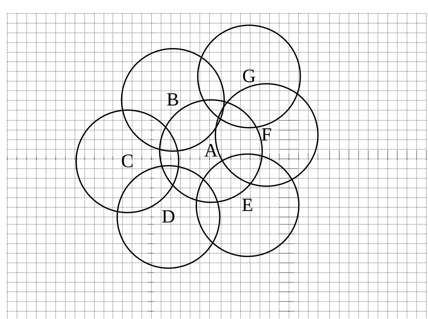


Figure 1. Virtual grid

Remark. Since our definitions of global coverage, local coverage, and contribution can be easily extended to the case where sensing area is not a circle, *pCover* can be easily extended to such a case as well.

```
for-each grid point x that is within my sensing range:
    if x can be covered by one of my awake neighbors
        my.CoveredGrid++
    endif
    my.TotalGrid++
end for-each

my.LocalCoverage = my.CoveredGrid / my.TotalGrid
my.Contribution = 1 − my.LocalCoverage
if my.Contribution < SleepThreshold
    Go to sleep
else
    Stay awake
endif
```

Figure 2. Sleep eligibility rule

3.2 State Transition

In this section, we describe the state transition in *pCover* (cf. Figure 3). Each node is in one of 4 states: *probing*, *awake*, *readyoff*, and *sleep*. Nodes switch among these states until they run out of power. Each node keeps a neighbor table, which maintains one record (ID, location, sensing range, status) for each neighbor.

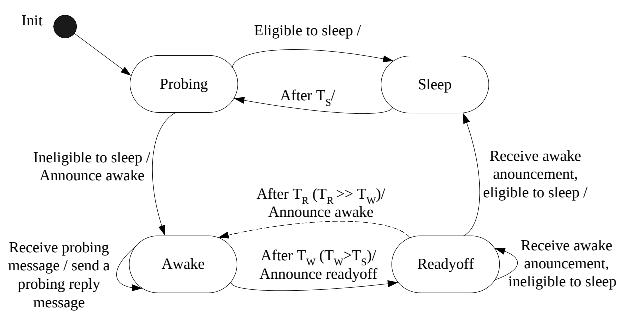


Figure 3. State transition in *pCover*

A node in *probing* state probes the environment, evaluates the sleep eligibility rule, and decides if it should start working or go to sleep. A node in *awake* state is a working node. It monitors the area within its sensing range, and contributes to global coverage. A node in *readyoff* state is one that wants to sleep but cannot do that until its local coverage is higher than a certain threshold. It is also a working node, and contributes to global sensing coverage. It announces its intention to sleep so that its neighbors do not count it when they calculate their sleep eligibility, and, hence, are more likely to stay awake. A node in *sleep* state only keeps a timer on, and turns off all the sensing, communication, and computation devices, in order to save energy. Now we describe the actions a sensor node performs in these states.

Actions in Probing State. In the initial stage, all the nodes are in probing state. Each node waits for a random back-off time, and then broadcasts a probing message. The probing message contains the node's ID, location and sensing range. It is used to collect *awake* neighbors' information. Whenever a neighbor node receives this probing message, if it is in *awake* state, it sends back a probing reply message, which also contains its ID, location and sensing range. Nodes update their neighbor tables based on the neighbor information they have collected, compute the sleep eligibility rule, and decide if they should go to sleep or stay awake.

When a node in probing state computes the sleep eligibility rule, it applies a *SleepThreshold*, which we call **on-threshold**. A probing node starts working if it finds that its local coverage is lower than on-threshold. Therefore a high on-threshold causes more nodes to turn on.

The random back-off time is used for two purposes. First, the probing messages are sent at different time, so that they are unlikely to collide. Second, nodes should decide their states in sequential order. For example, consider node A and node B in Figure 1, if A and B try to decide their states at the same time, since neither of them is in awake state at the moment (both are in probing state), they are not counted for each other's local coverage. As a result, the redundancy increases.

If a node decides to stay awake, it broadcasts an "AnnounceAwake" message, and switches to *awake* state. On the other hand, if a node decides to go to sleep, it turns off its radio and sensing units, and goes to *sleep* state.

Actions in Awake State. A node in *awake* state actively monitors the area within its sensing range. It remains in *awake* state for T_w period of time (T_w could be chosen deterministically or randomly), then changes its state to *readyoff*, and broadcasts a "ReadyToOff" message. By sending a "ReadyToOff" message, the node indicates a desire to turn off.

Actions in Readyoff State. Intuitively, *readyoff* state is a reception only state, i.e., a node in readyoff state does not respond to any probing messages. Therefore, a readyoff node appears to its neighbors just like a sleep node. However, a readyoff node is in fact a working node, and provides sensing coverage. The purpose of using readyoff state is for maintaining a stable global sensing coverage. To maintain the global coverage at a desired level, a node in readyoff state does not turn itself off until enough of its neighbors are awake to take over its sensing area. However, since it does not reply to probing messages, its neighbors do not count it when they compute their sleep eligibility. Also, a readyoff node watches for "AnnounceAwake" and "ReadyToOff" messages from its neighbors, and recomputes its sleep eligibility whenever it notices an increase in its local coverage. The *SleepThreshold* we apply here is off-threshold. If a readyoff node finds that its local coverage is higher than off-threshold, it will go to sleep state.

We also include an optional transition from readyoff state to awake state. A node that is in readyoff state for a long duration can switch to awake state and send an *AnnounceAwake* message. This transition allows one to deal with the case where a lot of nodes are in readyoff state although none of them can go to sleep state. Through simulations, we however found that a transition from readyoff state to awake state occurs only when there are no other alive nodes that can enable the node in readyoff state to sleep. In other words, in such a case, a node will continue to switch between awake state and readyoff state, thereby continually contributing to sensing coverage. However, in a real network, if the state of a node is corrupted for some reason, this transition will allow one to recover from the state where all nodes (large number of nodes) are erroneously in readyoff state. Since we do not model such faults in this paper, we do not include this transition while simulating *pCover*.

Actions in Sleep State. A node in *sleep* state wakes up every T_s period of time. When it wakes up, it performs two tasks. First, it senses its surrounding area (within its sensing range) to see if there is any interesting event. This guarantees that any event that lasts longer than T_s will always be detected as long as there are alive sensor nodes in the area where the event occurs. Second, it changes its state to probing (and proceeds to execute actions in probing state).

We note that a node refreshes its neighbor table periodically. Therefore it provides a certain level of fault tolerance. We assume that when a node fails, it simply stops working and does not send or receive any messages. If a node that was in awake state fails, its neighbors will be able to detect it within T_s, at the time when they wake up, probe the neighborhood and get the updated neighbor information.

4. Evaluation Results

In this section, we evaluate the performance of our protocol using simulation. The sensor nodes are deployed in a 160m$\times$160m field (called deployment area). The target area is the 140m$\times$140m square in the center of the field. Moreover, when we calculate the number of nodes, we only consider those in the central 100m$\times$100m area (called central area). The purpose of this setting is to eliminate the edge effect (where nodes on the edge are always awake) identified in [5]. Although our algorithm allows different sensing ranges, for simplicity, in these simulations, we set all the sensor nodes' sensing range r to be 10m.

In all the experiments, we set T_s to be 2 minutes, and T_w to be 10 minutes. We assume the lifetime of a sensor node is 1000 minutes, i.e., it is able to continuously work for 1000 minutes. We consider a network as "dead" when the global coverage of the network is less than a certain threshold even if all the alive nodes are working. We define the lifetime of a sensor network as the duration from the beginning of deployment until the network is dead. We use 50% as the threshold in our experiments. However, we found that the result is almost the same if a different threshold is used, as all the nodes ran out of power within a short period of time at the end of the simulation.

We study the performance of our protocol using two distribution patterns: deterministic distribution and random distribution. In the first case, sensors nodes are deployed in a grid topology. In the second case, nodes are deployed in a random manner. When using the grid topology, we are able to eliminate the non-uniform energy consumption and node lifetime caused by the random deployment of sensor nodes. We then perform simulations on a network with randomly deployed sensor nodes, and compare the difference.

In Section 4.1, we present the simulation results for networks with deterministic distribution. In Section 4.2, we present the results for networks with random distribution. In these two sections, we do not consider communication overhead when we compute the lifetime of the networks. This is because the dominant energy cost of a sensor node is the amount of time it spends in idle mode. Compared to that, the communication cost is negligible, as shown in previous studies [3, 9–11]. We also validate this claim in Section 4.3 using an example.

In our simulation, we also assume that there is no message collision or loss. (Note that these two assumptions, perfect wireless communication environment and no communication overhead, are made in almost all existing coverage preserving protocols (e.g., [4, 5, 12]).) In *pCover*, we use a random back-off scheme to avoid message collisions. However, message collision is still possible. In case that the probing or probing reply messages collide, we can extend our protocol by allowing a sensor node to send a probing message twice. In the second probing message, the node includes the IDs of the neighbors it has already heard from, so that these neighbors will not send reply messages repeatedly.

4.1 Networks with Deterministic Distribution

In this section, we study the performance of *pCover* in networks of grid topology. We study how the choices of *on-threshold*, *off-threshold*, and node density affect the global coverage and network lifetime.

4.1.1 Varying Off-threshold

We set the distance between two neighbor nodes to be 7m, thus 196 sensor nodes (in a 14×14 grid) fall in the central $100m\times100m$ area (node density is 1.96 nodes/r*r). We fix the *on-threshold* at 0.9, and vary the *off-threshold* from 0.3 to 0.9. The samples of global coverage are obtained periodically. When computing the global coverage, we count the percentage of grid points in the target area covered by working (*awake* or *readyoff*) nodes. In Figure 4, we show the global coverage over time, under different off-thresholds.

From Figure 4, we can see that, when the on-threshold is set to 0.9, global coverage is maintained at around 98%. Varying off-threshold does not have much impact on the global coverage. As we can see from Figure 4, the curves representing

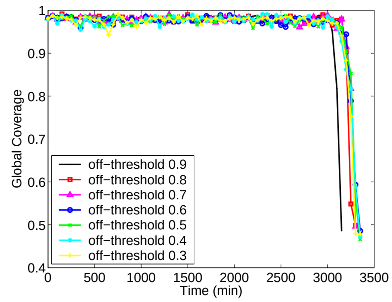


Figure 4. Grid topology: global coverage over time, when on-threshold is 0.9, and off-threshold varies from 0.3 to 0.9. Node density: 1.96 nodes/r*r.

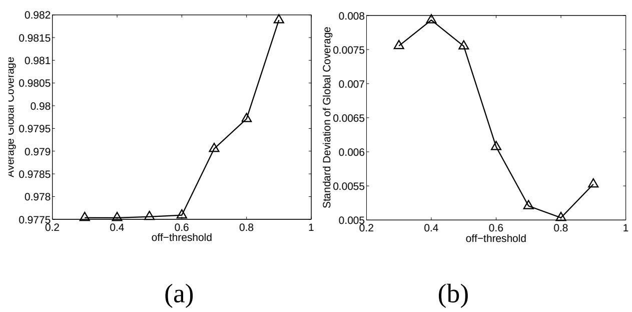


Figure 5. Grid topology: global coverage vs. off-threshold: (a) Average and (b) Standard Deviation, when on-threshold is 0.9. Node density: 1.96 nodes/r*r.

the global coverage when different off-thresholds are used, overlap each other and appear indistinguishable. To study the relationship between global coverage and off-threshold in further detail, we show the average value and standard deviation of the global coverage in Figure 5. We can see that the average global coverage slightly increases when off-threshold increases from 0.3 to 0.9, and the standard deviation of global coverage slightly decreases (with a few exceptions at the ends) with the increment of off-threshold. This is consistent with our intuition, i.e., high sleep threshold leads to high and stable global coverage.

In Figure 6, we show the lifetime of the network under different off-thresholds. We can see that, when the off-threshold is high, the global coverage is also high (Figure 5 (a)), while the network lifetime is short (Figure 6). However, the range within which global coverage or lifetime changes is short. When off-threshold varies from 0.3 to 0.9, the global coverage varies from 0.9775 to 0.982, and the lifetime of the network varies from 3329 to 3128 minutes. As we have already pointed out in the above paragraph, varying off-thresholds does not significantly affect network lifetime or global coverage.

From Figure 4, we also note that under all off-thresholds, the global coverage is well maintained until the point of 3050

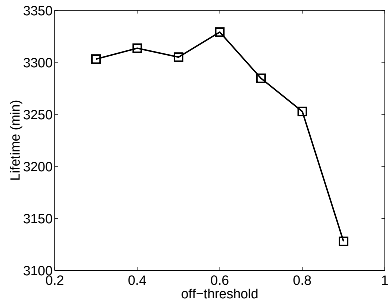


Figure 6. Grid topology: lifetime of the network vs. off-threshold, when on-threshold is 0.9. Node density: 1.96 nodes/r*r.

minutes, after which, the global coverage drops suddenly, and the network dies in a short period. To further illustrate the situation, we show the number of alive nodes over time, under different off-thresholds, in Figure 7. We found that the curves shown in Figure 7 are similar to those in Figure 4. All the nodes (196 in total) are alive until the point of 3000 minutes, then within a short period, all of them run out of power. This indicates that our algorithm maintains a balanced energy consumption.

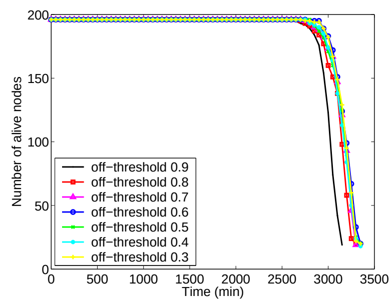


Figure 7. Grid topology: the number of alive nodes over time, when on-threshold is 0.9, and off-threshold varies from 0.3 to 0.9. Node density: 1.96 nodes/r*r.

We also evaluate our algorithm when on-threshold is changed to a lower value, 0.6, and other settings remain the same. We still vary off-threshold from 0.3 to 0.9. We show the global coverage over time in Figure 8, the average value and the standard deviation of global coverage in Figure 9, and the network lifetime in Figure 10. Compared to the situation when we use a higher on-threshold 0.9 (Figure 4, 5, 6), we find that, when off-threshold increases from 0.3 to 0.8, global coverage changes from 0.90 to 0.94 (Figure 9 (a)), and the network lifetime changes from 5189 to 4495 minutes (Figure 10). Although the trend is the same, i.e., higher off-threshold leads to higher global coverage and shorter network lifetime, the change in off-threshold has a little more impact on the global coverage and network lifetime, compared to the case when on-threshold is higher (0.9). Moreover, the global coverage fluctuates in a larger range (as shown in Figure 9 (b), the standard deviations of global coverage are slightly larger than those shown in Figure 5 (b)).

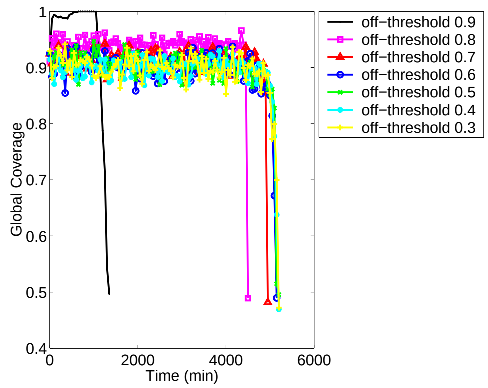
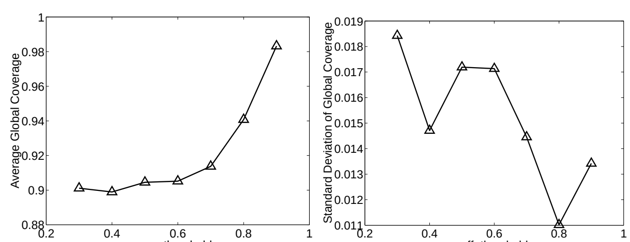


Figure 8. Grid topology: global coverage over time, when on-threshold is 0.6, and off-threshold varies from 0.3 to 0.9. Node density: 1.96 nodes/r*r.

We also find that when the off-threshold is 0.9, the global coverage is very close to 1, while the lifetime of the network is only a little longer than 1000 minutes, which is the lifetime of a single node. The reason for this phenomenon is that, when off-threshold/on-threshold ratio is high, sensor nodes switch to working mode easily, once they are on, it is difficult for them to turn off. As a result, almost all the sensor nodes are working at all time, and the network lifetime is close to the lifetime of a single node (1000 minutes). As expected, the node scheduling protocol is not effective in this situation.

In the next experiment, we increase the inter-node distance to 10m, i.e., the node density is reduced to 1 node/r*r, and on-threshold is kept at 0.9. We show global coverage over time in Figure 11, the average value and standard deviation of global coverage in Figure 12, and the network lifetime under different off-thresholds in Figure 13. We get the similar results as those from the previous experiment (where we set off-threshold to 0.6 and node density to 1.96 nodes/r*r). When the off-threshold



(a) (b)

Figure 9. Grid topology: global coverage vs. off-threshold: (a) Average and (b) Standard Deviation, when on-threshold is 0.6. Node density: 1.96 nodes/r*r.

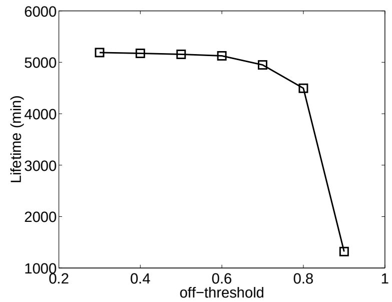


Figure 10. Grid topology: lifetime of the network vs. off-threshold, when on-threshold is 0.6. Node density: 1.96 nodes/r*r.

changes from 0.3 to 0.6, the average global coverage increases gradually from 0.92 to 0.94, and the network lifetime reduces from 2258 to 2101 minutes. However, when off-threshold is greater than or equal to 0.7, all the nodes are working at all time, and the lifetime of the network is close to a single node's lifetime (1000 minutes). It shows that when the network is sparse, a high off-threshold could cause difficulty in allowing sensor nodes to turn off, thus shortening the network lifetime.

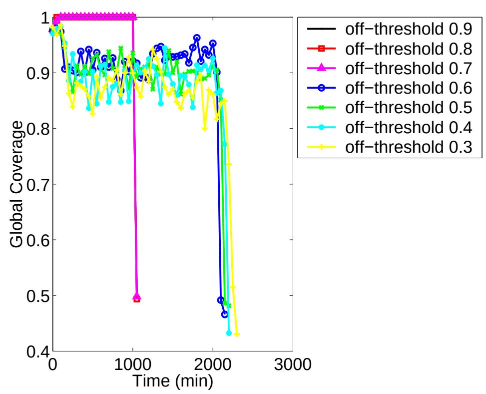
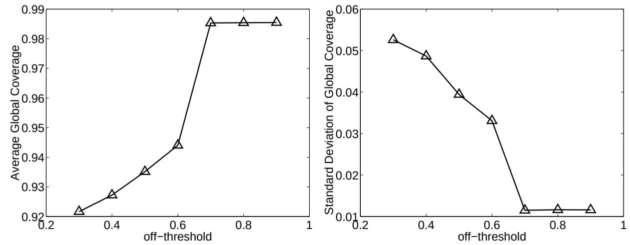


Figure 11. Grid topology: global coverage over time, when on-threshold is 0.9, and off-threshold varies from 0.3 to 0.9. Node density: 1 node/r*r.

4.1.2 Varying On-threshold

Based on the observations from Section 4.1.1, when the network density is greater than or equal to 1 node/r*r, off-threshold should be less than 0.7. Hence, in this section, we fix off-threshold at 0.6, and evaluate the performance when on-threshold varies from 0.3 to 0.9. We set the inter-node distance to 7m. This makes the node density to be 1.96 nodes/r*r. In Figure 14, we show global coverage over time. We observe that, with the increment of on-threshold, the global coverage strictly increases, and the lifetime of the network strictly decreases. This trend is also shown in Figure 15 (a) and Figure 16. We



(a) (b)

Figure 12. Grid topology: global coverage vs. off-threshold: (a) Average and (b) Standard Deviation, when on-threshold is 0.9. Node density: 1 node/r*r.

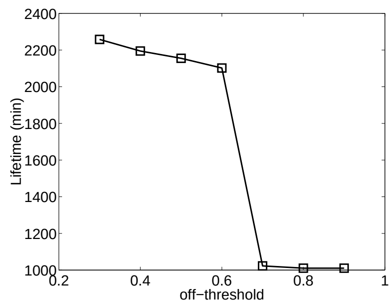


Figure 13. Grid topology: lifetime of the network vs. off-threshold, when on-threshold is 0.9. Node density: 1 node/r*r.

find that global coverage (respectively, network lifetime) increases (respectively, decreases) linearly with the increment of on-threshold.

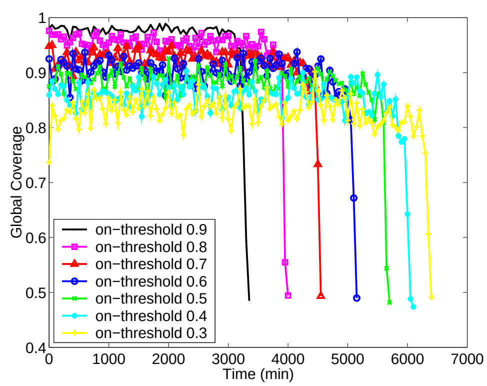
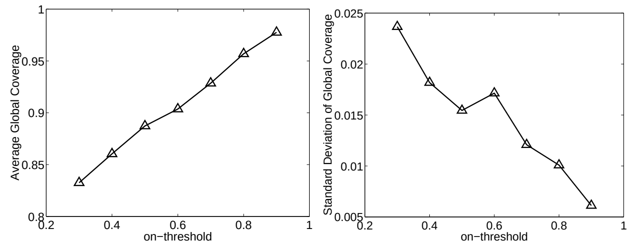


Figure 14. Grid topology: global coverage over time, when off-threshold is 0.6, and on-threshold varies from 0.3 to 0.9. Node density: 1.96 nodes/r*r.



(a) (b)

Figure 15. Grid topology: global coverage vs. on-threshold: (a) Average and (b) Standard Deviation, when off-threshold is 0.6. Node density: 1.96 nodes/r*r.

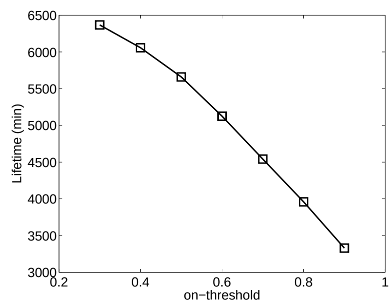


Figure 16. Grid topology: lifetime of the network vs. on-threshold, when off-threshold is 0.6. Node density: 1.96 nodes/r*r.

We repeat the same experiment with different node densities. In Figure 17, we show the relationship between global coverage and on-threshold. We can see that, global coverage linearly increases with the increment of on-threshold. Moreover, the curves that represent global coverage under different node densities are almost overlapping, with the exception of the node density being 1 node/r*r. In the case when the node density is as low as 1 node/r*r, the network is too sparse to meet the coverage requirement. It shows that global coverage is largely decided by on-threshold (and slightly decided by off-threshold, as shown in Section 4.1.1). Furthermore, when node density is reasonably high (e.g., not lower than 3 nodes/r*r), global coverage is independent of node density.

In Figure 18, we show that network lifetime decreases linearly when on-threshold increases. The space between the lines is even, which means that if we fix the on-threshold, the lifetime of the network is proportional to the node density, as we will show in the next section.

In Figure 19, we show the tradeoff between global coverage and network lifetime, under different node densities. As we have already noticed from Figure 17, when node density is as low as 1 node/r*r, the highest global coverage of a network that can be achieved is around 90%. When node density is higher, the limitation on the highest achievable global coverage no

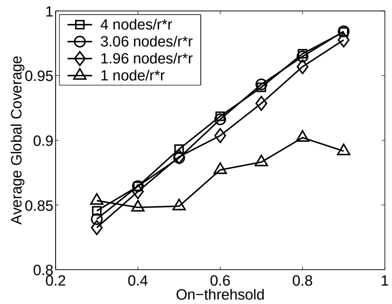


Figure 17. Grid topology: average global coverage vs. on-threshold, when off-threshold is 0.6.

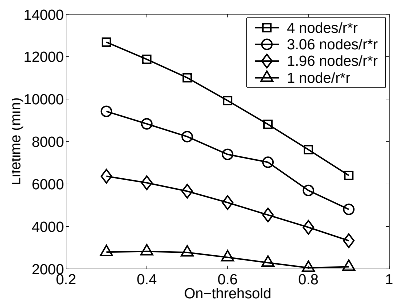


Figure 18. Grid topology: network lifetime vs. on-threshold, when off-threshold is 0.6.

longer exists. From Figure 19, we can see the tradeoff between global coverage and network lifetime, when the node density is higher than or equal to 1.96 nodes/r*r. Depending on the node density, decreasing the global coverage by 1% increases the network lifetime by 210 minutes (at 1.96 nodes/r*r) to 450 minutes (at 4 nodes/r*r).

4.1.3 Varying Node Density

We now set on-threshold to 0.7, off-threshold to 0.6, and vary the density of the network from 1 node/r*r to 4 nodes/r*r. We show the change of global coverage over time for different node densities in Figure 20. We find that global coverage is

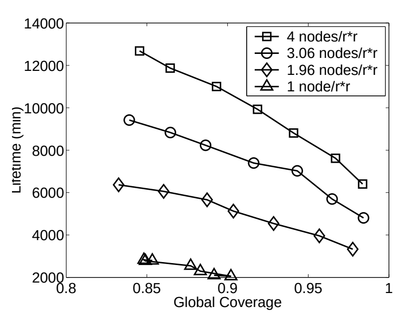


Figure 19. Grid topology: lifetime vs. global Coverage on grid topology, when off-threshold is 0.6.

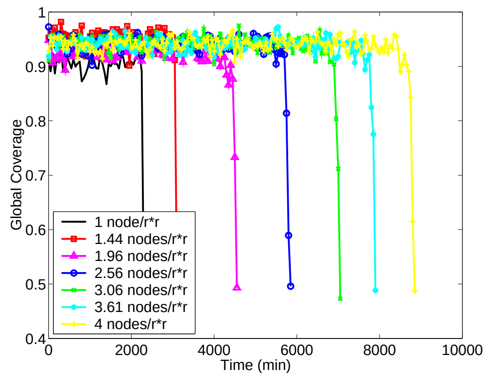


Figure 20. Grid topology: global coverage over time under different node densities, when on-threshold is 0.7, off-threshold is 0.6.

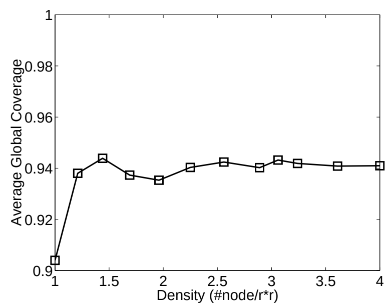


Figure 21. Grid topology: average global coverage vs. node density, when on-threshold is 0.7, off-threshold is 0.6.

maintained at the same level (about 94%) under all densities, except the case when node density is 1 node/r*r (also shown in Figure 21). As we mentioned in the Section 4.1.2, when node density is 1 node/r*r, the network is too sparse to achieve a high (>90%) sensing coverage. The lifetime of the network increases as the node density increases (Figure 22).

In Figure 21, we show that the global coverage is decided by the sleep thresholds (mainly on-threshold), independent of the density of the network. This is desirable, since it enables sensor nodes to control the global coverage locally by tuning the

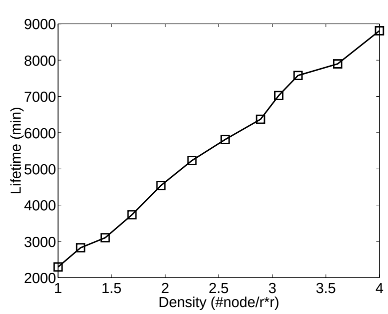


Figure 22. Grid topology: network lifetime vs. node density, when on-threshold is 0.7, off-threshold is 0.6.

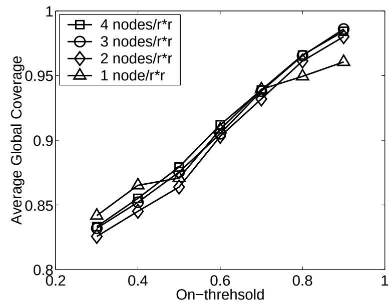


Figure 23. Random topology: global coverage vs. on-threshold, when off-threshold is 0.6.

sleep threshold (on-threshold) without knowing the density of the network. In Figure 22, we show that the lifetime of the network is linearly increasing with the increment of node density. This indicates that if we want to double the lifetime of the network without changing the global coverage, we could simply double node density in the target area.

4.2 Networks with Random Distribution

In this section, we evaluate the performance of *pCover* in a random topology. The node density is calculated based on the density in the deployment area. Since our experiments in Section 4.1.1 show that varying off-threshold does not have much impact on the global coverage and network lifetime, in this section, we fix the off-threshold to 0.6, and vary the on-threshold (Section 4.2.1) and node density (Section 4.2.2).

4.2.1 Varying On-threshold

Under a given node density, we vary the on-threshold from 0.3 to 0.9, and observe the average global coverage and network lifetime. In Figure 23, we show that, there is a fixed relationship between global coverage and on-threshold, independent of node density. Thus, given a desired global coverage (e.g., 0.98), we can easily decide the on-threshold that should be used (e.g., 0.9). Also, in Figure 24, we show that the network lifetime almost linearly decreases with the increment of on-threshold. And, in Figure 25, we show the relationship between network lifetime and coverage. Based on these results, we can see that, the relationships between global coverage, network lifetime, and on-threshold on a random topology (shown in Figures 23, 24, 25) are almost the same as that on a grid topology (shown in Figures 17, 18, 19).

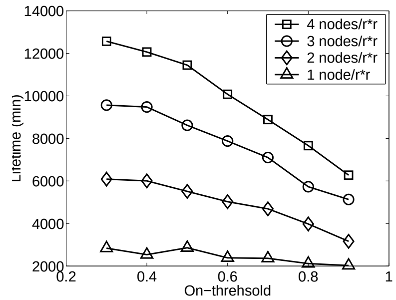


Figure 24. Random topology: network lifetime vs. on-threshold, when off-threshold is 0.6.

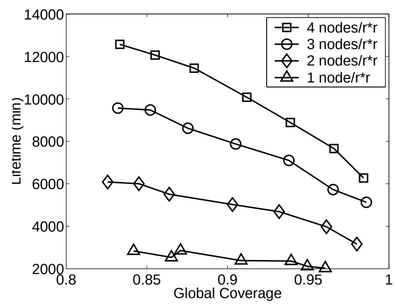


Figure 25. Random topology: lifetime vs. global Coverage on grid topology, when off-threshold is 0.6.

4.2.2 Varying Node Density

We perform the same experiments as we did on grid topology in Section 4.1.3. We fix on-threshold at 0.7, off-threshold at 0.6, and vary node density from 1 to 4 nodes/r*r. We show the change of global coverage over time under different node densities in Figure 26, the average global coverage at different node densities in Figure 27, and the network lifetime at different node densities in Figure 28.

Comparing Figure 20 and Figure 26, we find that, the amount of time for sensor nodes to die out is slightly higher in random topology than in grid topology. This is caused by the unevenness of random deployment. In random topology, some areas are covered by more number of sensors, and some areas are covered by fewer number of sensors. The sensor nodes that are deployed in “sparse” areas stay in working mode for a longer period of time than those deployed in “dense” areas. Therefore, the energy consumption is not as well balanced as that in grid topology. However, in random topology, the energy consumption among sensor nodes is still reasonably balanced. For example, when node density is 3.5 nodes/r*r, global coverage is maintained at the desired level (around 94%) for about 7000 minutes. It then drops to less than 50% after another 1000 minutes.

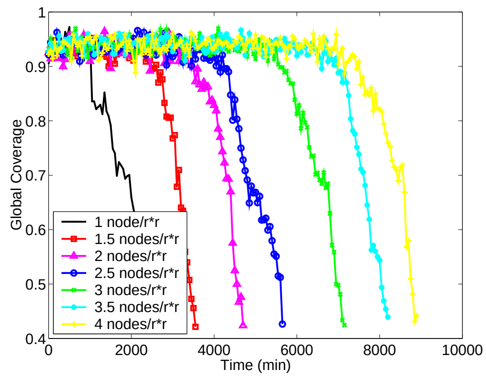


Figure 26. Random topology: global coverage over time under different node densities, when on-threshold is 0.7, off-threshold is 0.6.

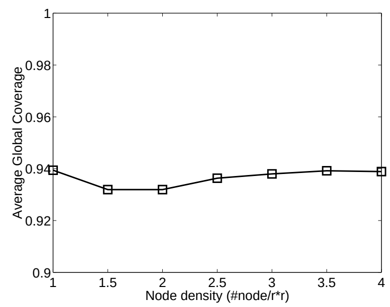


Figure 27. Random topology: global coverage vs. node density, when on-threshold is 0.7, off-threshold is 0.6.

4.3 Communication Overhead

In Sections 4.1 and 4.2, we did not consider communication overhead, as the energy consumed in communication is much less than the energy consumed in idle listening. To illustrate this, consider the case where sensor nodes are deployed in a grid topology. Let the node density be 1.96 nodes/r*r, on-threshold be 0.7, and off-threshold be 0.6. For this setting, the simulation

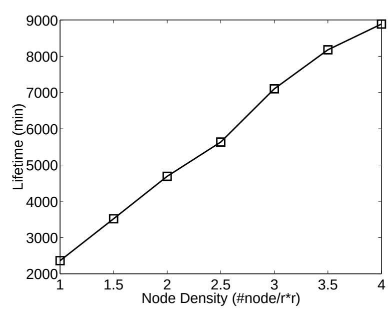


Figure 28. Random topology: network lifetime vs. node density, when on-threshold is 0.7, off-threshold is 0.6.

runs for 4541 minutes before the network dies. It turns out that the average number of transmissions per node is 10193, and the average number of receptions per node is 18202. Now we use the data from [3] to compute the communication cost on Mica motes. (Similar results can be found for other devices considered in [9, 10].) According to [3], the charge required by a Mica mote to transmit a packet is 20 nAh, to receive a packet is 8 nAh, and to idly listen for 1 millisecond is 1.25 nAh. Based on this data, the total communication cost (including transmissions and receptions) in the above simulation is equivalent to the cost that a node stays in idle mode for 4.66 minutes, which is only 0.1% of the network lifetime (4541 minutes).

5. Related Work

In recent years, several node scheduling approaches [4, 5, 12–14] have been proposed for surveillance sensor networks to minimize energy consumption while maintaining sensing coverage at the desired level. These approaches try to maintain a subset of sensor nodes in working mode and put the remaining nodes to sleep. In [5], a node scheduling scheme is proposed to preserve full sensing coverage by using an off-duty eligibility rule. Based on the rule, a sensor node is allowed to turn off if and only if its neighbor nodes are able to completely cover its sensing area. When a node computes its neighbors' contribution to its sensing coverage, it simplifies the calculation by using "sponsored sectors". Although it guarantees 100% coverage, it underestimates the area neighbor nodes can cover, and, hence, leads to redundancy and energy waste, as pointed out in [4]. Moreover, to balance energy consumption, the node scheduling operation is divided into rounds, and nodes perform rescheduling at the beginning of each round. This per-round based operation requires global synchronization service.

In [4], each node decides its working-sleeping schedule at the initialization phase. The set of working nodes are able to provide full coverage of the target area at any time. Moreover, a differentiated sensing coverage can be achieved by proportionally extending (k-coverage, $k > 1$) or shrinking (partial coverage) the work periods. Although this provides a solution on partial coverage as well, the study in [4] was only conducted on 1-coverage or k-coverage (where $k > 1$). Also like [5], the algorithm proposed in [4] requires time synchronization.

Both [5] and [4] guarantee 100% coverage. According to the simulation results in [4], when the node density is 4 nodes/r*r and the lifetime of a single node is 1000 minutes, the algorithm proposed in [4] is able to maintain full coverage for about 3000 minutes, and global coverage drops below 95% at around 4000 minutes, and drops below 90% at around 4500 minutes; the "sponsored coverage" scheme proposed in [5] is able to maintain full coverage for about 1000 minutes, and the global coverage drops below 90% at around 1500 minutes. We compare these two schemes with our algorithm in the case of random topology, under the same experiment settings. We find that using our algorithm, if the desired global coverage is 94%, the effective coverage is maintained for 7000 minutes, 2.3 times (respectively, 7 times) the duration for which the protocol in [4] (respectively, [5]) maintains full coverage. In the case of grid topology, our protocol maintains 94% coverage for 8500 minutes, which is 21.4% longer than that in the case of random topology (7000 minutes). Moreover, our algorithm enables a tradeoff between global coverage and network lifetime. For example, if we increase the coverage to 98%, the algorithm is able to maintain the effective coverage for about 5000 minutes. On the other hand, if we reduce the coverage to 90%, the duration that the coverage is maintained at above 90% is extended to about 8100 minutes.

Also, the coverage duration in our protocol is close to the theoretical estimate identified in [15]. In particular, if the coverage is 95% and the node density is 4 nodes/r*r then the *theoretical upper bound* for coverage duration is 9.8 times the lifetime of a single sensor node. In comparison, our protocol maintains 94% coverage for a duration that is 8.9 times the lifetime of a single sensor node.

In [12], Wang *et.al.* prove that an area is k-covered if all the intersection points in the same area are k-covered, where $k \geq 1$. Based on this, they propose the Coverage Configuration Protocol (CCP), in which a sensor node is eligible to turn off if all the intersection points inside its sensing circle are at least k-covered. CCP is designed to provide full coverage or higher degrees of coverage. According to the simulation results from [12], the number of active nodes required by CCP to provide 1-coverage in a unit area (r*r) is about 0.88-0.92, which is similar to that required by [4], and is more than twice the number of active nodes required by our protocol, in order to provide 94% coverage. Therefore, it provides significantly shorter network lifetime than our protocol.

PEAS [13] is a density control protocol. In PEAS, a sleeping node wakes up periodically and broadcasts a probing message to its neighbors within a certain range. Any node that is in working mode will respond to the probing message by transmitting a reply message. If a probing node receives a reply message before a timeout, it will go back to sleep. Otherwise, it starts working. By controlling the probing range and wakeup rate, PEAS can control the density of working nodes, thus the degree of coverage. In PEAS, a working node keeps awake continuously until its physical failure or depletion of power. This is undesirable since it causes unbalanced energy consumption. To address this problem, in [14], the authors propose PECAS as an extension to PEAS. PECAS lets a node go to sleep after it has been working for a given period of time. When a working node sends a probing reply message, it includes the remaining time it will continue working before going to sleep. Thus, the neighbor nodes will wake up in time to take over the sensing area. PEAS and PECAS do not need location information nor time synchronization. They allow a node to be off-duty as long as there is an awake neighbor within its probing range. However, the estimate is inaccurate, and, thus, they are not able to provide guarantees on degree of coverage.

Other node scheduling protocols include Randomized Independent Sleeping (RIS) ([17]) and MESH ([14]). Since these algorithms do not involve neighbor cooperation, they are not able to self-configure when the environment changes (e.g., node failure, sensing range changes). In a related area, several node scheduling protocols have been proposed for topology control, such as LEACH [18], GAF [19] and SPAN [20] for ad hoc networks, and ASCENT [21] for sensor networks. Although they perform in a similar way as to put the redundant nodes to sleep in order to save energy, these topology control protocols only consider communication connectivity, and try to establish a routing backbone. By contrast, the coverage problem addresses the issue of selecting active sensor nodes to cover every physical point in the target area. Finally, other results on partial coverage include [14, 22–26]. This work is orthogonal to our work in that in [14, 22–26], the authors have focused on the *quality* of partial coverage.

6. Conclusion

Maintaining a high degree of sensing coverage and prolonging the system lifetime are two conflicting goals for sensor networks. In this paper, we focused on the effect of providing a partial (but high) coverage in sensor networks on network lifetime. We demonstrated that it is possible to obtain a high level of sensing coverage (>90%) while substantially increasing lifetime when compared with protocols that provide a full (100%) sensing coverage. To illustrate this, we presented a protocol, *pCover*, that depends only on local information and does not depend on services such as time synchronization.

To illustrate the effectiveness of *pCover*, we simulated *pCover* on networks of grid topology and random topology. The simulation results show that under random topology, our protocol maintains 94% global coverage for a duration that is 2.3-7 times the duration for which existing protocols ([4, 5]) maintain full coverage. Under grid topology, our protocol maintains the same level of coverage for a duration that is 21.4% longer than that under random topology. Also, as discussed in Section 5, our protocol achieves a lifetime that is about 90% of the *theoretical upper bound* on lifetime identified in [15]. Moreover, we show that our protocol provides load balancing. In a network with uniform distribution, all the sensor nodes die within a short period. The desired degree of global coverage is maintained (almost) until the point where all sensor nodes die.

Based on the simulation results, we found that the relationship between global coverage and on-threshold is fixed, and is independent of node density (as long as the density is high enough to provide the desired coverage). Thus, we can locally control the global coverage by simply tuning the local parameter on-threshold. For a given on-threshold (or global coverage), we can predict the lifetime of the network under a certain node density, or compute the required node density in order to achieve a desired lifetime. We show that network lifetime is proportional to node density, under a certain on-threshold (or global coverage). Our protocol is also useful in dynamic reconfiguration. To illustrate this, consider the scenario when the sensor nodes have been deployed, we find that the network is required to last longer than originally planned. In this case, we can reduce the value of on-threshold in order to reduce the degree of coverage and extend network lifetime. The new value of on-threshold can be distributed to all the sensor nodes in the network through a reprogramming service (e.g., [27]).

Also, in *pCover*, the coverage is maintained (almost) until the point where energy of all sensors is depleted. *pCover* can be easily modified to achieve a gradually decreasing coverage. To achieve this, we need to reduce the on-threshold when the battery level of nodes drops below a certain point. (Alternatively, nodes can change their on-threshold when a certain amount of time has passed since deployment.) Since the nodes' battery is consumed at approximately the same rate, this would cause all nodes to change their on-threshold approximately at the same time. This would gradually reduce the level of coverage and result in increased lifetime.

References

[1] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y-R. Choi, T. Herman, S. S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita. A line in the sand: A wireless sensor network for target detection, classification, and tracking. *Computer Networks (Elsevier)*, 46(5):605–634, December 2004.

[2] The Ohio State University NEST Team. ExScal: Extreme scaling in sensor networks for target detection, classification, tracking, 2004. DARPA, `http://www.cse.ohio-state.edu/exscal`.

[3] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, Atlanta, GA, September 2002.

[4] T. Yan, T. He, and J A. Stankovic. Differentiated surveillance for sensor networks. *In Proceedings of the First International Conference on Embedded Networked Sensor Systems (SenSys)*, pages 51–62, November 2003.

[5] D. Tian and N. D. Georganas. A node scheduling schedule for energy conservation in large wireless sensor networks. *Wireless Communications and Mobile Computing Journal*, May 2003.

[6] Crossbow Technology, Inc. *MSP410 Datasheet*. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MSP410_Data%sheet.pdf`.

[7] Crossbow Technology, Inc. *MICA2 Datasheet*. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datas%heet.pdf`.

[8] Crossbow Technology, Inc. *TELOS-B Datasheet*. `http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB_Data%sheet.pdf`.

[9] V. Shnayder, M. Hempstead, B. Chen, G. Allen, and M. Welsh. Simulating the power consumption of large-scale sensor network applications. *In Proceedings of ACM International Conference on Embedded Networked Sensor Systems (SenSys)*, November 2004.

[10] M. Stemm and R. Katz. Measuring and reducing energy consumption of network interfaces in hand-held devices. *IEICE Transactions on Communications*, E80-B(8):1125–1131, August 1997.

[11] Oliver Kasten. Energy consumption. `http://http://www.inf.ethz.ch/personal/kasten/research/bathtub/energy_c%onsumption.html`.

[12] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill. Integrated coverage and connectivity configuration in wireless sensor networks. *In Proceedings of ACM International Conference on Embedded Networked Sensor Systems (SenSys)*, November 2003.

[13] F. Ye, G. Zhong, J. Cheng, S. W. Lu, and L. X. Zhang. PEAS: A robust energy conserving protocol for long-lived sensor networks. *In Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS)*, May 2003.

[14] C. Gui and P. Mohapatra. Power conservation and quality of surveillance in target tracking sensor networks. *In Proceedings of the Tenth Annual International Conference on Mobile Computing and Networking (ACM MobiCom)*, October 2004.

[15] H. Zhang and J. Hou. On deriving the upper bound of α-lifetime for large sensor networks. *In Proceedings of the Fifth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, May 2004.

[16] S. Kumar, T. H. Lai, and A. Arora. Barrier coverage with wireless sensors. *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2005.

[17] S. Kumar, T. H. Lai, and J. Balogh. On k-coverage in a mostly sleeping sensor network. *In Proceedings of the Tenth Annual International Conference on Mobile Computing and Networking (ACM MobiCom)*, pages 144–158, October 2004.

[18] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. *In Proceedings of the 33rd Hawaii International Conference on System Sciences*, Janauary 2000.

[19] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. *In Proceedings of the 6th ACM MOBICOM Conference*, July 2001.

[20] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *In Proceedings of the 6th ACM MOBICOM Conference*, July 2001.

[21] A. Cerpa and D. Estrin. ASCENT: Adaptive self-configuring sensor networks topologies. *IEEE Transactions on Mobile Computing*, 3(3):272–285, 2004.

[22] S. Ren, Q. Li, H. Wang, X. Chen, and X. Zhang. Analyzing object detection quality under probabilistic coverage in sensor networks. *In Proceedings of the 13th International Workshop on Quality of Service (IWQoS)*, June 2005.

[23] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak. Exposure in wireless ad hoc sensor networks. *In Proceedings of 7th ACM International Conference on Mobile Computing and Networking (Mobicom)*, pages 139–150, July 2001.

[24] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava. Coverage problems in wireless ad-hoc sensor networks. *In Proceedings of IEEE INFOCOM*, 3:1380–1387, April 2001.

[25] G. Veltri, Q. Huang, G. Qu, and M. Potkonjak. Minimal and maximal exposure path algorithms for wireless embedded sensor networks. *In Proceedings of ACM International Conference on Embedded Networked Sensor Systems (SenSys)*, November 2003.

[26] C. Huang and Y. Tseng. The coverage problem in a wireless sensor network. *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, pages 115–121, 2003.

[27] S. S. Kulkarni and L. Wang. MNP: Multihop network reprogramming service for sensor networks. *In Proceedings of the 25th International Conference on Distributed Computing Systems (ICDCS)*, pages 7–16, June 2005.