WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2011; **11**:1168–1181
Published online 15 April 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.907

SPECIAL ISSUE PAPER

Coverage for composite event detection in wireless sensor networks

Yinying Yang, Arny Ambrose and Mihaela Cardei*

Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431, U.S.A.

ABSTRACT

A wireless sensor network can detect single (or atomic) events or composite events. Since sensors are battery powered and in general, it is hard to recharge them, energy management is always an important issue. In this paper, we study the SCED problem; given a wireless sensor network deployed for watching a composite event $x_1, x_2, \ldots, x_M$, design a sensor scheduling mechanism such that the set of active sensors ensure the coverage and connectivity conditions and WSN lifetime is maximized. Network lifetime is organized in rounds. Each round has two phases: initialization and data collection. In the initialization phase, the goal is to choose a set of active sensors as sensing nodes or relay nodes such that to achieve both coverage and connectivity requirements. Sensors which are not chosen to be active go to sleep to save energy. In the data collection phase, active sensors perform sensing and data relaying. When another round begins, a new set of active sensors are determined. Two solutions are proposed for the SCED problem, a grid-based distributed algorithm and a localized algorithm. We analyze their performance through simulations. Copyright © 2010 John Wiley & Sons, Ltd.

KEYWORDS

composite event detection; connectivity; coverage; wireless sensor networks

***Correspondence**

Mihaela Cardei, Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431, U.S.A.
E-mail: mihaela@cse.fau.edu

1. INTRODUCTION

Sensors are used to monitor and control the physical environment. A Wireless Sensor Network (WSN) is composed of a large number of sensor nodes that are densely deployed either inside the phenomenon or very close to it [1,2]. Sensor nodes measure various parameters of the environment and transmit data collected to one or more sinks. Once a sink receives sensed data, it processes and forwards it to the users.

A WSN can detect single (or *atomic*) events or *composite* events [3]. Taking the sensor productions of Crossbow Technology, Inc. as an example, if sensors are equipped with MTS400 multi sensor board [4], it can sense temperature, humidity, barometric pressure, and ambient light.

Let us consider a single sensing component, for example, the temperature. If the sensed temperature value exceeds a predefined threshold, we say that an *atomic* event occurred. A *composite* event is a combination of several *atomic* events. For example, the composite event fire may be defined as the combination of the temperature and light. The *composite* event fire occurs only when both the temperature and the light exceed some predefined thresholds.

Sensors are battery powered and in general, it is hard to recharge them. It will take a limited time before they deplete their energy and become un-functional. So energy management is an important issue in WSNs.

In this paper, we focus on sensor Scheduling for Composite Event Detection (SCED) problem. We assume that sensors are densely deployed and they are equipped with multiple sensing components for watching a composite event. Sensing components have the same or different sensing ranges. To prolong the network lifetime, one method is to put some 'redundant' sensors to sleep. In the paper, network lifetime is organized in rounds. Each round has two phases, initialization phase and data collection phase. In the initialization phase, our goal is to choose a set of active sensors as sensing nodes or relay nodes while achieving both coverage and connectivity requirements. Sensors which are not chosen to be active in this round go to sleep to save energy. In the data collection phase, active sensors perform sensing and data relaying. When another round begins, a new set of active sensors are determined. Those which have been active in the previous rounds have lower priorities to be chosen in the following rounds.

Two solutions are proposed for the SCED problem, a grid-based distributed algorithm and a localized algorithm. We analyze their performance through simulations.

The rest of the paper is organized as follows. In Section 2 we present related works. In Section 3, we define the SCED problem. We continue in Section 4 with our two solutions. Section 5 presents simulation results and Section 6 concludes our paper.

2. RELATED WORKS

Event detection and coverage problem is one of the most important research issue in wireless networks [5**?**–17]. There are recent research works focusing on the composite event detection, such as Reference [18,3]. Both of them focus on collaboration of sensors to detect composite events, while do not consider the area coverage problem.

In Reference [18], Kumar *et al.* study a framework for both single and composite event detection. The framework consists of a protocol which builds a tree using a scheme similar to the publish-subscribe communication model. An application subscribes to an interested event with a corresponding location, and then the protocol builds an event-based tree. The data will then be collected along the tree. For composite events, a counter is maintained for each atomic event part of the composite event. Counters keep track of the number of sensors which can sense atomic events. Sensors are added into the tree until counters exceed some predefined thresholds.

In Reference [3], Vu *et al.* propose an algorithm to construct a set of tree-structured detection sets to achieve energy efficient and reliable surveillance. To achieve reliable surveillance, a composite event must be k-watched by the sensor nodes. That means there must be k sensing components watching each atomic event part of the composite event. The algorithm forms a tree-structured detection set satisfying the k-watching requirement in a greedy manner. At each step, the sensor node with the greatest contribution is added into the tree. The algorithm is repeated to find as many detection sets as possible, based on the sensors' energy constraint. Different detection sets work alternatively to achieve energy efficiency and to maximize network lifetime.

References [19] and [20] focus on the coverage problem for single event detection. Reference [19] proposes a localized method to find the area dominating sets, which is the smallest subset of sensors that covers the whole monitored area. Each sensor computes the area covered by its neighbors with higher priorities, if its sensing area can be fully covered by those neighbors and the subgraph built by them is connected, it decides to go to sleep. Reference [20] considers the coverage problem without the connectivity requirement. The idea is that if a sensor's area is covered by its neighbors' sensing area, it can be turned off. The geometry calculation is given to compute the overlapping area when two sensors are located within their sensing range.

In References [21,22], research is based on grid-based network structure. In Reference [22], the network is divided into small grids, and all sensors in the same grid form a cluster. A cluster head is chosen in each grid, who takes care of the communication among cluster members within the grid and communication with neighbor clusters.

References [23–30] focus on construction of dominating set. References [23,25,28–31] propose localized methods. Reference [25] proposed an approach called Span, to select a set of special nodes called coordinators. A node v becomes a coordinator if it has two neighbors that are not directly connected, indirectly connected via one intermediate coordinator, or indirectly connected via two intermediate coordinators. Before a node changes its status from non-coordinator to coordinator, it waits for a backoff delay, which is viewed as a priority. Nodes that have a shorter backoff delay have a higher chance of becoming coordinators. Reference [31] proposes a localized algorithm for connected dominating set formation, which includes a marking process to form a connected dominating set, which is similar to Span and a pruning rule to reduce the number of nodes in the connected dominating set. Reference [28] proposes an iterative localized solution for connected dominating sets in ad hoc wireless networks. Their goal is to decrease the number of sensors in the dominating set iteratively. Based on the current dominating set, each node selects a new priority, exchanges information with its neighbors, and executes the basic dominating set algorithm again until no nodes can be removed from the dominating set.

Our solutions extend the solutions of single event detection for composite event detection. We assume that each sensing component might have different sensing ranges and we consider both coverage and connectivity requirements. We use an iterative mechanism in both solutions; however, we have a different goal than Reference [28], which is to add sensors to be active iteratively for coverage or connectivity purpose.

3. PROBLEM DEFINITION

In this paper, we consider that each sensor is equipped with one or multiple sensing components from the set of sensing components $\{x_1, x_2, \ldots, x_M\}$ due to the following reasons [18]: they might be manufactured with different sensing capabilities, a sensor node might be unable to use some of its sensing components due to the lack of memory for storing data, or some sensing components might fail over time. A sensor can be equipped with at most one sensing component of each type. Sensors may have different numbers and types of sensing components. For example, let $M = 3$ where x_1 is the temperature, x_2 is the humidity, and x_3 is the smoke. A sensor may be equipped with only $\{x_2\}$, while another sensor may be equipped with $\{x_1, x_3\}$. All of a sensor's sensing components turn on or off simultaneously. In general, each sensing component (e.g., temperature, light, humidity) might have a different coverage capability,

therefore we consider that the sensing components have different sensing ranges.

For simplicity, we assume that $Rs_1 \leq Rs_2 \leq \ldots \leq Rs_M$, where $Rs_1, Rs_2, \ldots, Rs_M$ are the sensing ranges for the sensing component $x_1, x_2, \ldots, x_M$. We can always achieve this by sorting the sensing components according to their sensing ranges.

An important issue in WSNs is energy management. Sensor nodes are battery powered and in general, they cannot be recharged. It will take a limited time before they deplete their energy and become un-functional. One of the major components that consume energy is the radio.

A radio can be in one of the following modes: transmit, receive, idle, and sleep. A radio is in idle mode when the host is not transmitting or receiving data, and usually the power consumption is as high as in the receive mode. A radio is in sleep mode when both the transmitter and the receiver are turned off.

According to Reference [32], studies on several commercial radios (e.g., WaveLAN, Metricom) show that in sleep mode the power consumption ranged between 150–170 mW, while in idle mode the power consumption went up by an order of magnitude.

In this paper, we put sensors to sleep mode when they are not actively participating in sensing or data relaying to conserve energy and to prolong the network lifetime. We are concerned with designing a scheduling mechanism that will allow sensors to go to sleep as long as the coverage and the connectivity requirements are met.

The *coverage requirement* requires that the deployment area to be continuously covered by each sensing component x_j, for $1 \leq j \leq M$.

The *connectivity requirement* requires that the set of active sensors to be connected. This condition is necessary in order to collect the sensed data. If the sink is connected to any of the active sensors, then a data collection tree rooted at the sink can be formed and data can be gathered by the sink.

We assume that sensors are densely deployed and the network itself is connected and the coverage requirement can be met.

Figure 1 shows an example with $M = 2$ sensing components $\{x_1, x_2\}$. The communication range of sensors, R_c, is $R_c = 2Rs_1$, where Rs_1 is the sensing range of sensing component x_1. The black nodes are active sensors and the white nodes are sensors in sleep mode. The dashed circles represent the sensing area of sensing component x_1, and the continuous line circles represent the sensing area of x_2. The set of active sensors meets the coverage requirement: the whole monitored area is covered by the sensing component x_1 and the sensing component x_2. The connectivity requirement is met as well: the active sensors are connected so that the sensed data can be transmitted to the sink.

Next, we present the problem definition for sensor Scheduling for Composite Event Detection in WSNs (SCED problem).

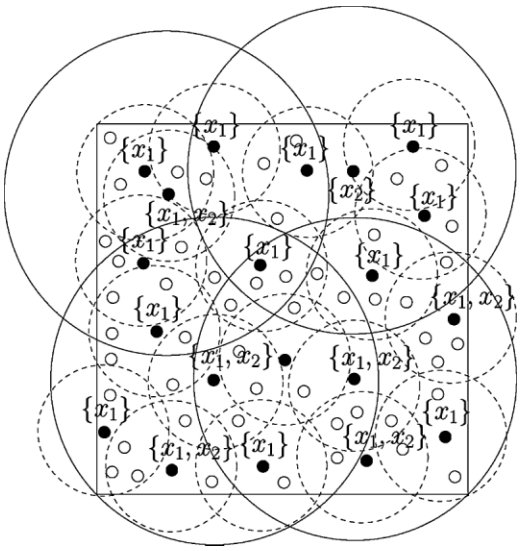


Figure 1. The monitored area is covered considering both sensing components x_1 and x_2.

Definition 1 (SCED problem)**.** *Given a WSN deployed for watching a composite event* $\{x_1, x_2, \ldots, x_M\}$*, design a sensor scheduling mechanism such that the set of active sensors ensures the coverage and connectivity requirements and the WSN lifetime is maximized.*

4. SOLUTIONS FOR THE SCED PROBLEM

In this section, we propose two sensor scheduling solutions for the SCED problem. The network lifetime is organized in rounds. Each round has two phases: initialization and data collection. In the initialization phase, a scheduling algorithm is run to decide which sensors remain active and which sensors can go to sleep during the current round. In the data collection phase, active sensor nodes perform sensing and data relaying. We assume that sensors know their location information using GPS [33] or other localization protocols [34,35].

4.1. Distributed approach to decide active nodes

In this distributed solution, the monitored area is divided into grids, see Figure 2. Let us consider that each sensing component x_j has coverage range $r_j\sqrt{2}$. Then any sensor with sensing component x_j located in a $r_j \times r_j$ grid region will completely cover that region.

For simplicity, let us assume that $r_j = 2^u \cdot r_{j-1}$, where $u \geq 0$. It is straight-forward to extend this approach to other cases. If $u = 0$, then the two consecutive sensing components x_j and x_{j-1} have the same sensing range.

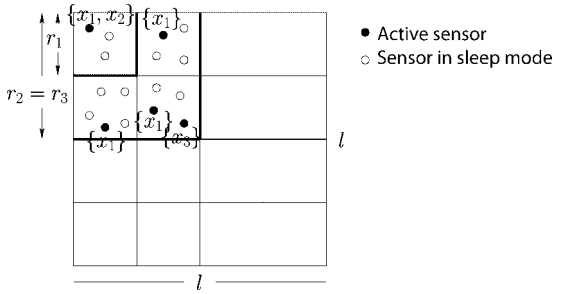


Figure 2. Sensors deployed in a $l \times l$ square area.

Figure 2 shows an example with $M = 3$ sensing components x_1, x_2, x_3, where $r_3 = r_2 = 2r_1$. Figure 2 illustrates sensors in a quarter of the deployment area, both active and in the sleep mode. The set of active sensors satisfies the coverage condition: each grid $r_1 \times r_1$ has an active sensor equipped with sensing component x_1, each grid region $r_2 \times r_2$ has an active sensor equipped with sensing component x_2, and each grid region $r_3 \times r_3$ has an active sensor equipped with sensing component x_3.

The scheduling mechanism consists of two algorithms: (1) one to decide the sensing nodes and (2) one to decide the relay nodes needed to ensure a connected network. All other nodes are in the sleep mode. Figure 3 shows the two main phases of a network round.

Algorithm 1 DecideStatus (sensor s_i, time interval t_j)

1: **if** (s_i has status active) OR (s_i is not equipped with x_j) OR (s_i is equipped with x_j and $g_j = 1$) **then**
2: return
3: **end if**
4: compute contribution function $\chi_i = f(e_i, R_c) \cdot help_i$
5: start timer with value τ_i, which is inversely proportional to χ_i
6: **if** message StatusActive(s_k, s_k's location, s_k's sensing components $\geq j$) received **then**
7: **for** each s_k's sensing component $v \geq j$ **do**
8: **if** (s_i is equipped with x_v) AND ($g_v = 0$) AND (s_i and s_k are in the same $r_v \times r_v$ region) **then**
9: $g_v \longleftarrow 1$
10: **end if**
11: **end for**
12: **if** (s_i receives the message for the first time) AND (s_i and s_k are in the same $r_w \times r_w$ region, where w is the largest index of the sensing components of s_i) **then**
13: broadcast the StatusActive message it receives
14: **end if**
15: **if** $g_j = 1$ **then**
16: return
17: **end if**
18: **end if**
19: **if** timer fires up **then**
20: sensor s_i is set active and $g_j \longleftarrow 1$
21: broadcast StatusActive(s_i, s_i's location, s_i's sensing components $\geq j$) in s_i's region $r_w \times r_w$
22: **end if**

Algorithm 2 Connectivity (CH sensor s_i)

1: **if** (*Hello* messages received from sensors in bottom and right regions) **then**
2: return TRUE
3: **end if**
4: wait a random time
5: send *Conn_Req*(CH s_i, grid position of s_i, list of needed region connections (bottom and/or right), sending node ID, TTL, number of relay hops)
6: **if** one or more *Conn_Reply*(s_j, grid position of s_j, CH s_i, grid position of s_i, number of relay hops) messages received **then**
7: **for** each needed region connection **do**
8: choose the *Conn_Reply* message with the minimum number of relay hops, let's say from node s_j, and send *RelaySetup*(CH s_i, grid position of s_i, s_j, grid position of s_j)
9: **end for**
10: **end if**
11: **if** all needed region connections are satisfied **then**
12: return TRUE
13: **end if**
14: **if** $TTL < TTL_{\max}$ **then**
15: increase TTL and go to line 4
16: **else**
17: return FALSE
18: **end if**

4.1.1. Decide the sensing nodes.

In this section, we present a distributed algorithm to decide which sensors will stay active in order to satisfy the coverage requirement.

We consider the initialization phase divided into a sequence of time intervals $t_1, t_2, \ldots, t_M$. The basic idea is to ensure that during the time interval t_j at least one sensing component x_j becomes active in each square region $r_j \times r_j$.

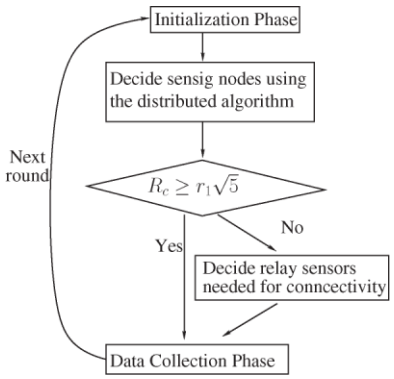


Figure 3. Main steps of the distributed approach.

Each sensor s_i keeps a Boolean variable g_j for each sensing component x_j that it is equipped with. Initially, all variables $g_j = 0$. For example, a sensor $s_i\{x_1, x_3, x_4\}$ initializes variables $g_1 = g_3 = g_4 = 0$.

Variable g_j becomes 1 if there is at least one active component x_j in the $r_j \times r_j$ region where s_i is located. This happens when s_i or another sensor containing sensing component x_j that is located in the same $r_j \times r_j$ region becomes active.

The DecideStatus() algorithm is run by each sensor s_i at the beginning of each time interval t_j. Based on the partitioning of the field, a sensor s_i belongs to exactly one $r_j \times r_j$ square region. At the beginning of each interval t_j, each sensor s_i containing x_j performs the following steps. In time interval t_j, at least one sensing component x_j becomes active in each square region $r_j \times r_j$, if such a component exists.

In line 1 of the DecideStatus() algorithm, if s_i is not equipped with x_j or if another x_j component has become active ($g_j = 1$), then s_i will not be set active during the time interval t_j and the procedure returns.

Otherwise, sensor s_i computes its contribution function $\chi_i = f(e_i, R_c) \cdot help_i$ similar to Reference [3], where $f(e_i, R_c) = \frac{e_i}{Tx_i + Sx_i}$ is a function to calculate s_i's lifetime depending on its current residual energy e_i and its communication range R_c, where Tx_i is the energy consumed on transmission in one round and Sx_i is the energy consumed on sensing in one round, $help_i$ is the number of s_i's helpful sensing components. A sensing component x_k of s_i is called a helpful sensing component if $k \geq i$ and $g_k = 0$. The condition $k \geq i$ is used to check the sensing components which have not been taken care of in the previous time intervals. The condition $g_k = 0$ means that in the previous time intervals, no sensor that is equipped with x_k has been activated in the corresponding region. In this way, it counts the number of sensing components that might be helpful in the following time intervals. The sensor that has more helpful sensing components has a higher priority to become active in the current time interval. If such a sensor is equipped with some sensing component, e.g., x_{i+1}, no other sensor needs to be active for x_{i+1} in the time interval t_{i+1}, in the corresponding region. This definition helps to reduce the number of sensors that have to be active in the network. A larger contribution function value means that s_i has a larger priority in becoming active. Then s_i starts a timer inversely proportional to χ_i.

If a sensor s_i receives a *StatusActive()* message from a sensor s_k, it means that s_k has become active recently. Then for each s_k's sensing component $v \geq j$, if s_i is equipped with x_v and if s_i and s_k are in the same $r_v \times r_v$ region, then set $g_v \longleftarrow 1$. If g_j becomes 1, then the procedure returns without requiring s_i to become active.

Next, we address the issue of how to determine if s_i and s_k are in the same $r_v \times r_v$ region, see Figure 4(a). Let us assume that the coordinates of the sensors are $s_i(xs_i, ys_i)$ and $s_k(xs_k, ys_k)$. Then s_i and s_k are in the same $r_v \times r_v$ region if and only if $\left\lfloor \frac{xs_i}{r_v} \right\rfloor = \left\lfloor \frac{xs_k}{r_v} \right\rfloor$ and $\left\lfloor \frac{ys_i}{r_v} \right\rfloor = \left\lfloor \frac{ys_k}{r_v} \right\rfloor$.

If timer fires up after time τ_i, then sensor s_i is set active and thus g_j becomes 1. As soon as a sensor becomes active, it broadcasts a message StatusActive(s_i, s_i's location, s_i's sensing components $\geq j$) in s_i's region $r_w \times r_w$, where w is the largest index of the sensing components of s_i. A sensor forwards the message only the first time it receives the message and only if it is located in the same $r_w \times r_w$ region. Every forwarding sensor sets up its g variables based on s_i's sensing components, similar to lines 7... 11 in the pseudocode.

4.1.2. Decide the relay nodes: the case when $R_c \geq r_1\sqrt{5}$.

In this section, we address the case when the sensor communication range $R_c \geq r_1\sqrt{5}$. In this case, the coverage requirement implies the connectivity requirement. According to Figure 4(b), the communication range (R_c) needed for direct communication of two sensors located at maximum distance on the diagonal ends is $R_c^2 = r_1^2 + 4r_1^2 = 5r_1^2$. Thus, if the communication range satisfies $R_c \geq r_1\sqrt{5}$, this will ensure direct communication of any two sensors located in adjacent regions.

Based on the coverage requirement, there will be at least one active sensor in each grid region $r_1 \times r_1$. Since $R_c \geq r_1\sqrt{5}$, it follows that any two active sensors in two adjacent regions can communicate directly.

4.1.3. Decide the relay nodes: the case when $R_c < r_1\sqrt{5}$.

Connectivity becomes an issue when $R_c < r_1\sqrt{5}$. Two active sensors in adjacent regions might not communicate directly. We consider that sensors in the same region $r_1 \times r_1$ can communicate directly, that is $R_c \geq r_1\sqrt{2}$. Length r_1 can always be chosen such that this condition is met.

For the connectivity purpose, it is sufficient to ensure that each square region $r_1 \times r_1$ is connected to its bottom and right neighbors. After satisfying the sensing requirement, there will be at least one active sensor in each $r_1 \times r_1$ square region. We propose a distributed algorithm which ensures that the active sensors form a connected topology. The basic idea of the algorithm is to ensure that each $r_1 \times r_1$ square

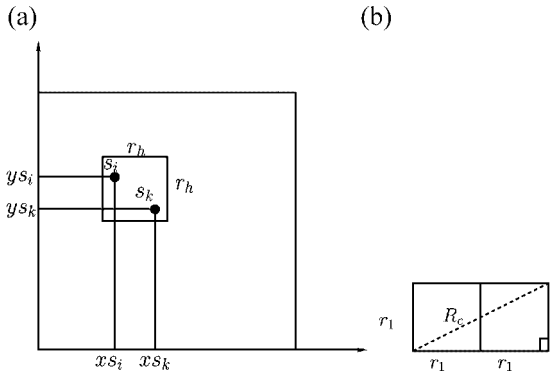


Figure 4. (a) Two sensors s_i and s_k are in the same $r_h \times r_h$ region. (b) Maximum distance between sensors in neighboring regions.

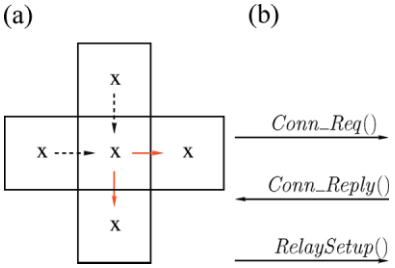


Figure 5. Connectivity mechanism (a) Connection of a region with bottom and right regions. (b) Connectivity: three-way message exchange mechanism.

region is connected to its bottom and right neighbor regions, see Figure 5(a). This algorithm is described below.

First, each $r_1 \times r_1$ region chooses a cluster head (CH), which is for example, the active sensing node with the largest remaining energy. This can be done by having each sensor s_i send a message *Hello*(s_i, s_i's region location, s_i's residual energy). Then the node with the largest residual energy acts as a CH. In order to avoid collisions, each sensor waits a random time before sending the *Hello* message.

The CH of each region is responsible to ensure connectivity with bottom and right regions. Let us take a CH s_i. If node s_i hears *Hello* messages from any nodes in the bottom and right regions, then nothing has to be done. Otherwise, node s_i starts the connection mechanism which is a three-way message exchange mechanism, see Figure 5(b): request using *Conn_Req*, reply using *Conn_Reply*, and relay set-up using *RelaySetup*.

Node s_i broadcasts a message *Conn_Req* (CH s_i, grid position of s_i, list of needed region connections (bottom and/or right), sending node ID, TTL, and number of relay hops).

The Time-To-Live (TTL) variable can be set up to a smaller value initially, and if no reply is received, then it can be increased using an incremental ring search approach. An intermediate node s_j receiving a *Conn_Req* message will forward the message if s_j is not an active sensing node in the destination bottom and/or right regions; otherwise it will reply with a *Conn_Reply* message.

If s_j is not an active sensing node in a destination region, then s_j will forward the *Conn_Req* message if this is the message with the minimum number of relay hops so far and increases the number of relay hops. In addition, s_j updates the following fields: sending node ID (which is now s_j), TTL = TTL-1, and the number of relay hops. The message will be forwarded only if TTL > 0.

Let us consider now the case when s_j is an active sensing node in one of the destination regions (in the bottom or right region). Then s_j will not forward the *Conn_Req* message, but it will reply with a message *Conn_Reply*(s_j, grid position of s_j, CH s_i, grid position of s_i, number of relay hops). The reply messages will be sent to s_i along the reverse pointers which were set-up temporarily when the requests were forwarded.

If CH node s_i receives one or more *Conn_Reply*() messages from a region of interest (bottom or right), then s_i chooses the node s_j from the *Conn_Reply*() message with the minimum number of relay hops, since this is the number of relay nodes that has to be activated in order to establish a path between s_i and s_j. Then s_i sends a message *RelaySetup*(CH s_i, grid position of s_i, s_j, grid position of s_j). This message will be sent along the forward pointers which were set-up temporarily when the replies were forwarded. Each node that forwards the *RelaySetup* message will become active and will serve as a relay node in order to ensure connectivity.

Let us address the case when no reply message is received from a destination region (bottom or right). In this case, according to the incremental ring search approach, TTL is increased and the process is repeated, which means s_i will broadcast a *Conn_Req*() message with the new TTL.

TTL can be increased up to some predefined maximum value $TTL_{\max}$. If no reply is received after a number of trials, then connectivity cannot be satisfied. One approach in this case is that s_i will send a message to the sink announcing this event. If sink location is not known, this can be accomplished by flooding a corresponding message in the whole network. In this paper, we define network lifetime as the number of rounds where both the sensing and the connectivity can be satisfied.

The connectivity steps are summarized in the Algorithm 2.

4.2. Localized approach to decide active nodes

In this section, we propose a localized algorithm for choosing active sensors to meet both coverage and connectivity requirements.

The initialization phase is divided into M time intervals $t_1, t_2, \ldots, t_M$. In each time interval $t_j (1 \leq j \leq M)$, sensors focus on the coverage and connectivity of sensing component x_j.

Sensors can have three possible states in this algorithm: active, pending, and sleep. Active and sleep are final states. Pending is an intermediate state. Initially, all sensors are in pending state. The sensors which are chosen to be the sensing nodes or relay nodes are in the active state. Sensors which are not chosen to be active in the current time interval might remain in the pending state and they might become active in the following time intervals. Sensors in the sleep state go to sleep to save energy.

Every sensor s_i has a priority P_i^j in each time interval t_j, which is a 4-tuple $P_i^j = (state, f(e_i, R_c), help_i, ID_i)$. In P_i^j, *state* is the most important factor. Sensors in the active state have higher priorities than those in the pending or sleep state, so we have $state = 1$ when the sensor is in the active state and $state = 0$ when it is in the pending or sleep state. $f(e_i, R_c)$ is the same as that defined in the previous distributed algorithm, and it is used to break tie for the sensors with the same *state*. Sensors having higher $f(e_i, R_c)$

have higher probabilities to be active in the following time intervals. $help_i$ is the number of s_i's helpful sensing components. A sensing component x_k in time interval t_j is helpful when $k \geq j$. $help_i$ is used to break tie for sensors with the same *state* and $f(e_i, R_c)$. ID_i is the node identification.

All sensors are initialized to be in the pending state. Each sensor exchanges *Hello* messages among their neighbors. The *Hello* message includes the state of the sensor, the list of the sensing components the sensor has and the priority of the sensor. The neighbors are defined as follows.

Definition 2 (Neighbor)**.** *The neighbor set of sensor s_i in time interval t_j is defined as* $N_j(s_i) = \{s_k \in \aleph | d(s_i, s_k) \leq Rs_j, s_k \neq s_i\}$.

$\aleph$ is the sensor set containing sensors deployed in the whole monitored area. $d(s_i, s_k)$ is the distance between two sensors and Rs_j is the sensing range in the current time interval t_j, which is the sensing range of the sensing component x_j.

The neighbors may be more than one hop away depending on the Rs_j and R_c. When sensors within Rs_j receive the *Hello* message, they broadcast it. If non-neighbor sensors receive the *Hello* message, they discard it and do not broadcast it.

In time interval t_j, the sensors in the active or sleep state in the previous time interval t_{j-1} remain the same state in t_j. Each sensor s_i that is in the pending state and is equipped with sensing component x_j considers the following three conditions. If all three conditions are met, it goes to the sleep state. Otherwise, if $Condition_1$ and $Condition_2$ are met, but $Condition_3$ is not met, it remains in the pending state. If at least one of $Condition_1$ and $Condition_2$ is not met, then the sensor s_i goes to the active state.

- $Condition_1$: s_i's sensing area can be covered by its neighbors that are in the active or pending state, contain sensing component x_j and have higher priorities. We discuss how this is done in detail in Section 4.2.1.
- $Condition_2$: For any pair of s_i's neighbors that are in the active or pending state, there exists an h-hop ($h \geq 1$) path connecting them. The path should only contain the sensors that are not in sleep state and have higher priorities than s_i, which implies that s_i cannot be in the path.
- $Condition_3$: The sensor does not contain any sensing component $x_k, k > j$, that is, x_{j+1}, or $x_{j+2}, \ldots,$ or x_M.

The $Condition_1$ regards the coverage requirement and ensures that only those sensors whose sensing area can be covered by their neighbors can go to sleep. The $Condition_2$ regards the connectivity requirement. If sensor s_i's neighbors can be connected without the help of s_i, then s_i is a candidate to go to sleep. Each sensor collects neighborhood information through Hello message exchange so that a sensor can know its neighbors' location, state, and priority information. These informations are sufficient for the sensor to confirm whether $Condition_2$ is satisfied. One method to determine if two neighbors are connected is to use the Breadth-First Search algorithm. In $Condition_3$, if s_i has a sensing component whose index is larger than j, then it means that it still has other sensing components that need to be checked in the following time intervals.

Note that since $Rs_{j+1} \geq Rs_j$, we only add more active sensors in every following time interval to meet the coverage and connectivity requirements for larger sensing range and we do not turn previously active sensors to the pending or sleep state.

If a sensor remains in the pending state at the end of the current time interval t_j, in the time interval t_{j+1}, it updates its priority by recomputing $help_i$. If a sensor goes to the active state at the end of the current time interval t_j, it updates its priority by setting $state = 1$. The similar process repeats using the updated priorities.

After M time intervals, the sensors either go to the sleep or active state. Thus, the set of active sensors is found. The sensing and data collection phase begins. The main steps are summarized in the Algorithm 3.

Theorem 1 (Coverage)**.** *After applying the localized approach, the coverage requirement is met, which means the monitored area is continuously covered by all active sensors considering all sensing components.*

Proof. We first prove that at the end of time interval t_1, the monitored area is covered by active sensors considering x_1. At the beginning of t_1, all sensors are in the pending state. Suppose the monitored area is not covered by active sensors considering x_1 at the end of t_1. We turn the sensors which are in the sleep or pending state to be active one by one in descending order of sensor's priority. We can find the first sensor s_v that covers the uncovered area. However, it is impossible, because if it is in the sleep or pending state, its sensing area should be covered by its active and pending neighbors that contain x_1 and have higher priorities according to $condition_1$. So the coverage is ensured by all active sensors after time interval t_1.

In the following time intervals, we only add more active sensors for coverage purpose. We can use the similar way to prove at the end of time interval $t_j (1 < j \leq M)$, active sensors ensure the coverage for $x_1, x_2, \ldots, x_j$. ■

Note that if $R_c \geq 2Rs_1$, then meeting the coverage requirement ensures the connectivity. In the general case, the connectivity is proved as follows.

Theorem 2 (Connectivity)**.** *If the original network G is connected, after applying the localized approach, the subnetwork G' containing all the sensors in the active state after M time intervals is also connected.*

Proof. We first prove that at the end of each time interval t_j ($1 \leq j \leq M - 1$), the subnetwork built by all sensors in the active or pending state is connected. Suppose it is not connected at the end of time interval t_j. We turn the sensors which are in the sleep state to active one by one in descending order of sensor's priority. We can find the first

sensor s_v that reconnects the subnetwork, that is it makes the subnetwork connected. However, it is impossible because if it is put to sleep, its active and pending neighbors must be connected according to $Condition_2$. So the subnetwork containing active and pending sensors is connected after $M-1$ time intervals.

At the end of time interval t_M, sensors in the pending state go to either the active state or the sleep state. We prove that at the end of the time interval t_M, the subnetwork containing all active sensors is connected in a similar way. ■

Algorithm 3 Localized algorithm to decide active sensors

```
1: for each sensor s_i do
2:    compute remaining energy and initialize priority
      P_i^1 = (state = 0, f(e_i, R_c), help_i, ID_i)
3:    set status to the pending state
4: end for
5: for k = 1 to M time intervals do
6:    for each sensor s_i do
7:       exchange Hello messages among its neighbors
         within range Rs_j
8:       if (s_i is equipped with sensing component x_j)
         AND (it is in the pending state) then
9:          if Condition_1 AND Condition_2 AND
            Condition_3 then
10:            go to the sleep state
11:         end if
12:         if Condition_1 AND Condition_2 AND
            !Condition_3 then
13:            remain in the pending state
14:            update its priority by recomputing help_i
15:         end if
16:         if !Condition_1 OR !Condition_2 then
17:            go to the active state
18:            update its priority: state = 1
19:         end if
20:      end if
21:   end for
22: end for
```

4.2.1. Compute the area coverage.

In this section, we discuss how a sensor determines if its sensing area is covered by its neighbors given their location information. A sensor's sensing area in time interval t_j is a circle centered at itself with radius Rs_j. In the Algorithm 3, because we choose active sensors for one sensing component in each time interval, we only need to consider the case when all sensors have the same sensing range.

We use similar notations as in Reference [20], which are shown in Figure 6. d is the distance between two sensors. The sensing area of s_2 intersects s_1's sensing area at points P_1 and P_2. The sector bounded by radius s_1P_1, s_1P_2 and inner arc P_1P_2 is called the sponsored sector by sensor s_2 to s_1. The central angle of the sponsored sector is denoted as $\theta = 2\arccos\frac{d}{2Rs_j}$. The direction of sensor s_2 referred to s_1 is denoted as ϕ, which is computed as follows.

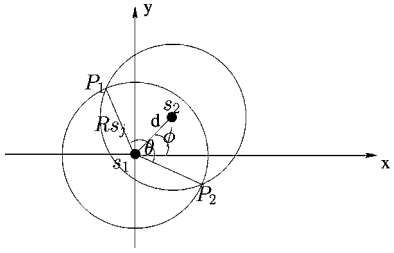


Figure 6. Notations for area coverage computation.

$$\phi = \begin{cases} \arctan\frac{y_2-y_1}{x_2-x_1}, & x_2 > x_1 \text{ and } y_2 > y_1 \\ \pi - \arctan\left|\frac{y_2-y_1}{x_2-x_1}\right|, & x_2 < x_1 \text{ and } y_2 > y_1 \\ \pi + \arctan\left|\frac{y_2-y_1}{x_2-x_1}\right|, & x_2 < x_1 \text{ and } y_2 < y_1 \\ 2\pi - \arctan\left|\frac{y_2-y_1}{x_2-x_1}\right|, & x_2 > x_1 \text{ and } y_2 < y_1 \\ \frac{\pi}{2}, & x_2 = x_1 \text{ and } y_2 > y_1 \\ \frac{3\pi}{2}, & x_2 = x_1 \text{ and } y_2 < y_1 \\ 0, & y_2 = y_1 \text{ and } x_2 > x_1 \\ \pi, & y_2 = y_1 \text{ and } x_2 < x_1 \end{cases}$$

The basic idea of the coverage calculation for a sensor s_i is to compute the sum of the sector areas covered by its neighbors within the sensing range. We do not consider the sensors that are outside the sensing range, because even they have some overlapped area, there are always some areas near s_i left uncovered. The central angle of the sponsored sector by s_k to s_i refereed to the x-axis is denoted as A_{s_k} called sector angle, which is computed from θ and ϕ. We compute the sum of A_{s_k} for each $s_k \in N_j(s_i)$ and compare it with 2π. If the sum is greater than or equal to 2π, then it means that s_i is covered by its neighbors.

Definition 3. *The sensor s_i's sensing area is covered when the following condition is met: $\bigcup_{s_k \in N_j(s_i)} A_{s_k} \geq 2\pi$, where A_{s_k} is computed according to the equations below.*

- When $0 \leq \phi < \frac{\theta}{2}$, as shown in Figure 7(a), the sector angle has two parts, which are $A^1 = \left[0, \frac{\theta}{2} + \phi\right]$, and $A^2 = \left[2\pi - \frac{\theta}{2} + \phi, 2\pi\right]$.
- When $\frac{\theta}{2} \leq \phi \leq 2\pi - \frac{\theta}{2}$, as shown in Figure 7(b), the sector angle is $A = \left[\phi - \frac{\theta}{2}, \phi + \frac{\theta}{2}\right]$.
- When $2\pi - \frac{\theta}{2} < \phi \leq 2\pi$, as shown in Figure 7(c), the sector angle has two parts, which are $A^1 = \left[0, \frac{\theta}{2} + \phi - 2\pi\right]$, and $A^2 = \left[\phi - \frac{\theta}{2}, 2\pi\right]$.

When computing the sum of sector angles, the sensor checks its neighbors one by one and records the current sum. Each time when it checks a neighbor, it merges the contribution from this neighbor with the existing sum. Figure 8 shows an example. For s_1, the sponsored sector by s_2

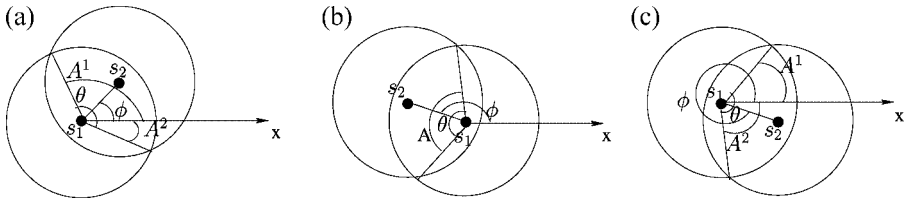


Figure 7. Three cases for the computation of the coverage angle: (a) $0 \leq \phi \leq \frac{\theta}{2}$, (b) $\frac{\theta}{2} < \phi \leq 2\pi - \frac{\theta}{2}$, and (c) $2\pi - \frac{\theta}{2} < \phi \leq 2\pi$.

has $A_{s_2} = \left[0, \frac{2}{3}\pi\right]$ and the sponsored sector by s_3 has $A_{s_3} = \left[\frac{1}{3}\pi, \pi\right]$. Then the sum is $\left[0, \frac{2}{3}\pi\right] \bigcup \left[\frac{1}{3}\pi, \pi\right] = [0, \pi]$.

5. SIMULATION

In this section, we present the simulation results of our solutions for the SCED problem. We study the network lifetime improvement, and the average number of active sensors under different conditions.

5.1. Simulation environment

Metrics in the simulations include the network lifetime and the average number of active sensors. The network lifetime shows how many sets of active sensors can be chosen while both coverage and connectivity requirements are met. In the simulations, in each round, we apply our solutions to choose one set of active sensors and then all active sensors report once to the sink. The network lifetime is computed as the number of rounds where both the coverage and connectivity requirements are met.

The average number of active sensors is computed as $\frac{numActiveSensor}{numRound}$, where $numActiveSensor$ is the total number of active sensors during network lifetime and $numRound$ is the network lifetime in terms of the number of rounds.

We use a similar energy model as that presented in LEACH (Low-Energy Adaptive Clustering Hierarchy) [36]. It considers a simple model where radio dissipates $E_{elec} = 50\,\text{nJ/bit}$ to run the transmitter and the receiver circuitry and $\varepsilon_{amp} = 100\,\text{pJ/bit/m}^2$ for the transmit amplifier. The energy used to transmit a k-bit message over a distance d is: $E_{Tx}(k, d) = E_{Tx-elec}(k) + E_{Tx-amp}(k, d) = E_{elec} * k + \varepsilon_{amp} * k * d^2$. The energy consumed to receive a k-bit message is: $E_{Rx}(k) = E_{Tx-elec}(k) = E_{elec} * k$. We assume that the packet size of the *Hello* messages is much smaller than the size of data messages.

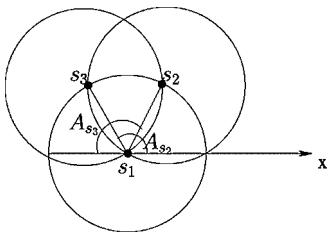


Figure 8. Example of the area coverage computation.

To form a data delivery tree, the sink broadcasts a *Hello*(*numberOfHops*) message, where *numberOfHops* is the number of hops the message is forwarded and the message is flooded in the network among active sensors. Each active sensor records the current minimum number of hops to the sink. When it receives the message, if $numberOfHops + 1$ is smaller than the minimum number of hops the sensor records, it updates its shortest path (minimum number of hops) and forwards it. It also records the sensor from whom it receives the message and uses it as the next hop to the sink in the data deliver tree.

We assume that each active sensor applies a data aggregation algorithm. The packet size that an active sensor reports is computed as $\alpha \cdot msgReceived$, where α is the aggregation factor and we choose $\alpha = 0.5$ in our simulations, $msgReceived$ is the total packet size it receives from its children in the data delivery tree.

In the simulations, we vary the scale of the monitored area. We consider the composite event is a combination of up to four sensing components, $\{x_1, x_2, x_3, x_4\}$, representing the temperature, light, pressure, and humidity, respectively. The number and types of sensing components a sensor is equipped are randomly chosen. We study the performance when the locations of the sink are different. We also vary the transmission ranges to study the performance for the cases when $R_c < r_1\sqrt{5}$ and when $R_c \geq r_1\sqrt{5}$. Large transmission ranges (larger than the sensing ranges) and small transmission ranges (smaller than the sensing ranges) are both considered according to References [37,38].

We conduct the simulations on a custom discrete event simulator, which generates a random initial sensor deployment. All the tests are repeated 50 times and the results are averaged.

5.2. Simulation results

Figures 9–11 study a small scale network. The network size is 100×100 units. $M = 3$, which means the composite event is a combination of three sensing components, $\{x_1, x_2, x_3\}$. The sensing range for x_1 is 36 units, and the sensing range for x_2 and x_3 is the same, which is 71 units. For the distributed solution, the grid size for x_1 is 25×25 units, and the grid size for x_2 and x_3 is the same, which is 50×50 units.

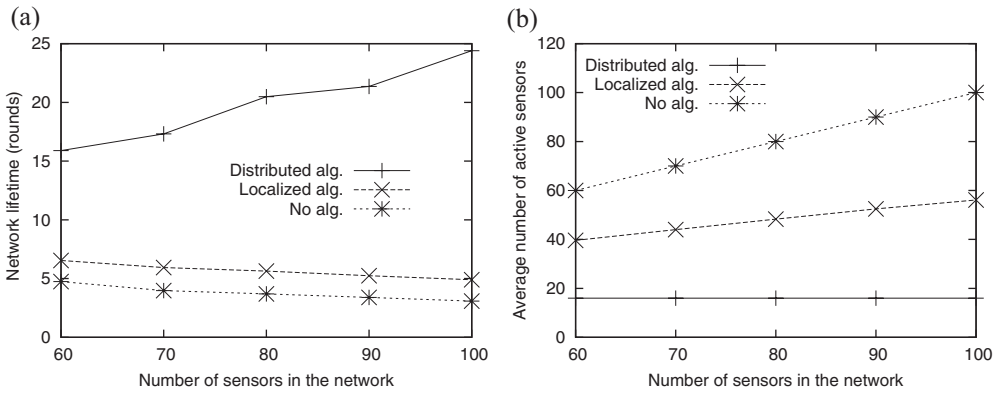


Figure 9. The transmission range is 60 units, the sink is located at the bottom left corner of the monitored area. (a) Network lifetime. (b) The average number of active sensors.

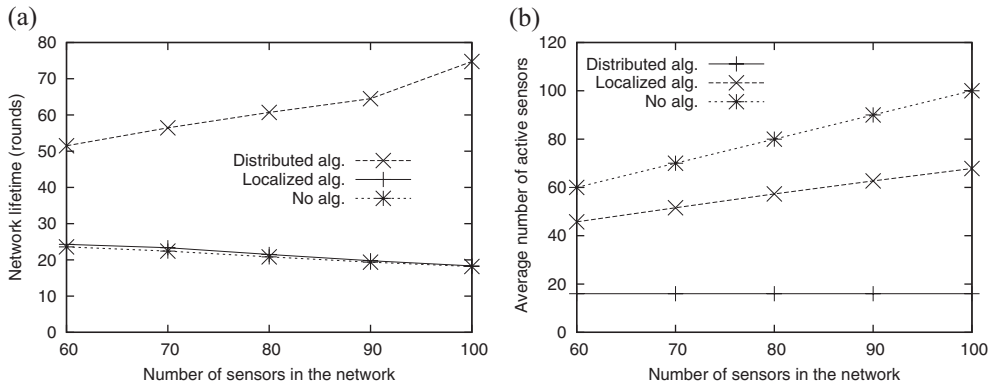


Figure 10. The transmission range is 60 units, the sink is located at the center of the monitored area. (a) Network lifetime. (b) The average number of active sensors.

In Figure 9, the transmission range is 60 units and the sink is located at the bottom left corner of the monitored area. In Figure 10, the sink is located at the center of the network. In Figure 11, the transmission range is 36 units and the sink is located at the bottom left corner of the monitored area. We study the network lifetime and the average number of active sensors among the distributed solution, the localized solution and the case when no scheduling algorithm is applied. In all three figures, the distributed solution and the localized approach have a longer network lifetime than the case when no scheduling algorithm is applied. The localized solution has more average active

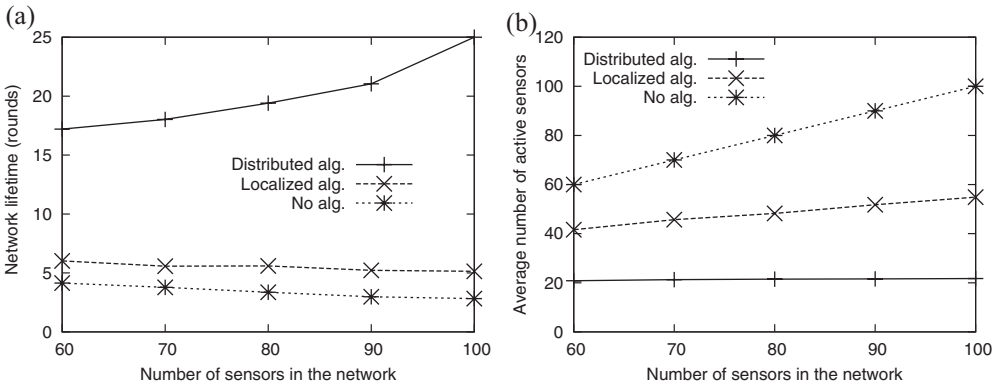


Figure 11. The transmission range is 36 units, the sink is located at the bottom left corner of the monitored area. (a) Network lifetime. (b) The average number of active sensors.

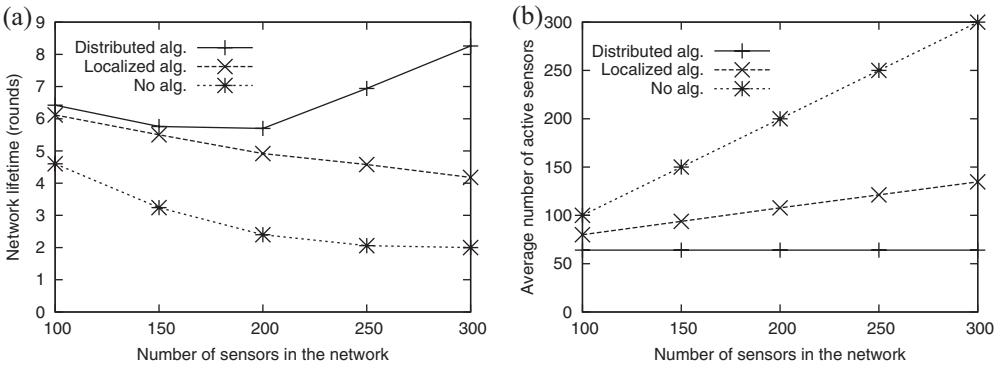


Figure 12. The transmission range is 200 units, the sink is located at the bottom left corner of the monitored area. (a) Network lifetime. (b) The average number of active sensors.

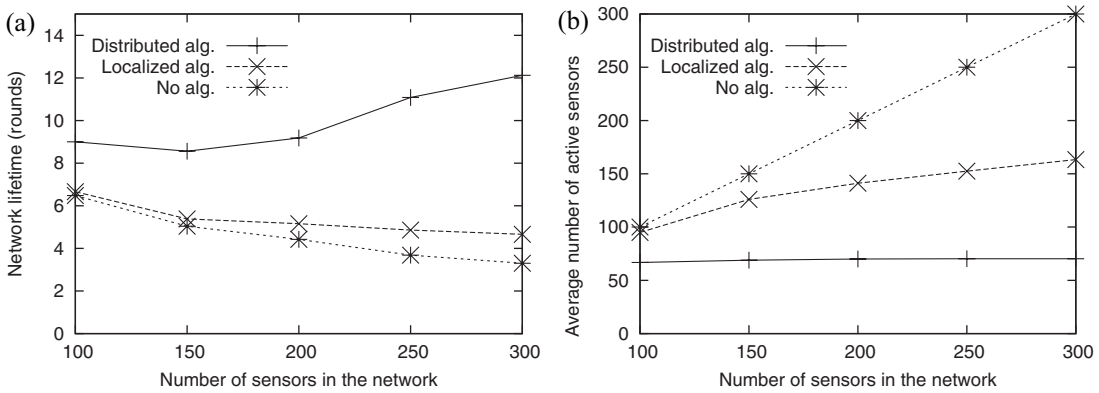


Figure 13. The transmission range is 150 units, the sink is located at the bottom left corner of the monitored area. (a) Network lifetime. (b) The average number of active sensors.

sensors and therefore, it has a shorter network lifetime compared with the distributed approach. The energy consumption in *Hello* message exchange is also taken into account in simulations. Compared with the case when there is no algorithm is applied, the localized algorithm involves more *Hello* message exchanges, however, it still performs better in terms of the network lifetime.

Compared with the case when the sink is located at the bottom left corner as shown in Figure 9(a), the network lifetime is longer when the sink is located at the center of the network, as shown in Figure 10(a), for all the distributed solution, the localized solution and the case when no algorithm is applied. That is because the data delivery path is longer in the case when the sink is located at the bottom left corner. In Figure 10, more active sensors are one hop away from the sink and they can directly send data to the sink. While in Figure 9, more active sensors are more than one hop away from the sink. They rely on the sensors which are 1-hop away from the sink to relay data. In this case, the sensors that are 1-hop away from the sink will consume more energy in forwarding data. They will die first and become the bottleneck.

Figures 12–16 study a larger scale network. The network size is 640×640 units and $M = 3$. The sensing range for x_1 is 114 units, the sensing range for x_2 is 227 units, and the sensing range for x_3 is 453 units. In the distributed solution,

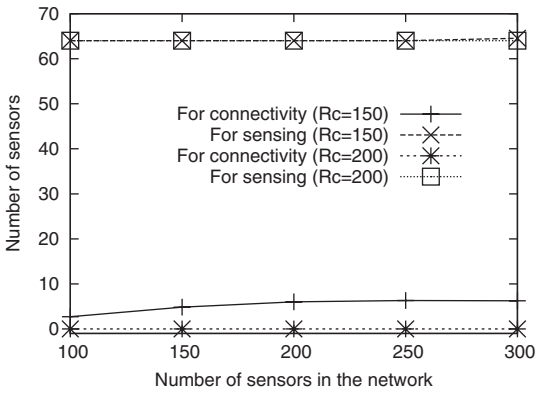


Figure 14. The number of sensors for the sensing or connectivity purpose.

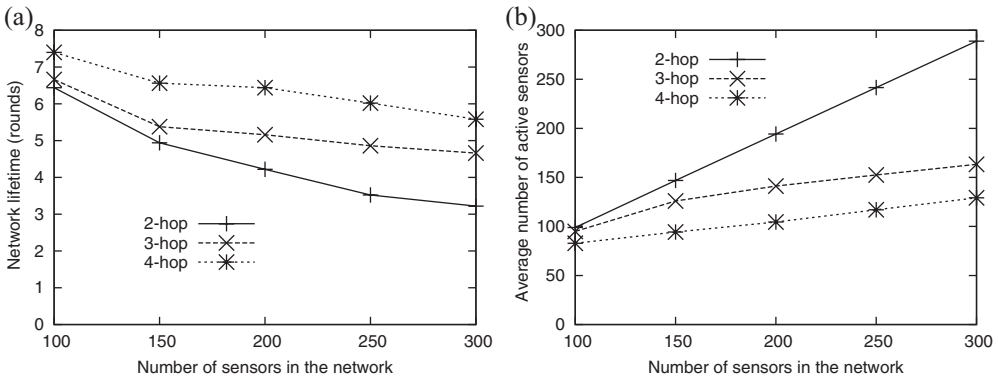


Figure 15. Different values of *h*. (a) Network lifetime. (b) The average number of active sensors.

the grid size for x_1 is 80×80 units, the grid size for x_2 is 160×160 units, and the grid size for x_3 is 320×320 units.

In Figure 12 and Figure 13, the sink is located at the bottom left corner of the monitored area and the transmission range of the sensors is 200 units and 150 units, respectively. Consistent with Figures 9–11, both the distributed solution and the localized approach perform better than the case when no scheduling algorithm is applied. The distributed solution provides the longest network lifetime among the three cases. The localized solution has more active sensors and therefore, it has a shorter network lifetime.

In the distributed approach, some sensors are chosen to be active for the sensing purpose and some sensors are chosen to meet the connectivity requirement. Figure 14 shows the number of active sensors for sensing the events and the number of active sensors for the connectivity purpose. The transmission range of the sensors is 150 or 200 units, the sink is located at the bottom left corner of the monitored area and $M = 3$. When the transmission range is 200 units, which is larger than $r_1\sqrt{5} = 179$, all the active sensors are for the sensing purpose and no sensors need to turn to active to meet the connectivity requirement. When the transmission range is 150 units, which is shorter than $r_1\sqrt{5}$, besides the sensors for the sensing purpose, a few sensors need to be active to meet the connectivity requirement.

In the localized solution, when a sensor considers the $Condition_2$, it needs to check whether each pair of two neighbors are connected by a h-hop ($h \geq 1$) path consisting of higher priority nodes in active or pending state. Figure 15 shows the effect of different values of h for the localized solution. The transmission range of the sensors is 150 units, the sink is located at the bottom left of the monitored area and $M = 3$. We can see that when h is 2, there is the largest average number of active sensors as shown in Figure 15(b) and consequently, it provides the worst network lifetime as shown in Figure 15(a). When h is 4, fewer sensors need to be active and the network lifetime is longer.

In Figure 16, we vary the number of sensing components and measure the network lifetime and the average number of active sensors. When $M = 2$, the composite event is a combination of $\{x_1, x_2\}$ and when $M = 4$, the composite event is a combination of $\{x_1, x_2, x_3, x_4\}$. The sensing range for x_1 is 114, for x_2 and x_3 is 227, and x_4 is 453. In the distributed approach, the grid size for x_1 is 80×80, for x_2 and x_3 is 160×160, and x_4 is 320×320. The transmission range of the sensors is 150 units, the sink is located at

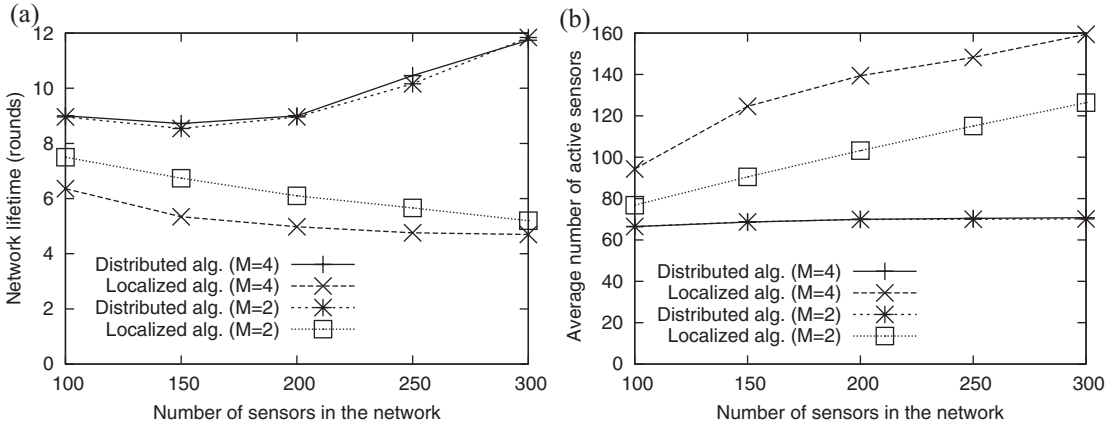


Figure 16. Different numbers of sensing components. (a) Network lifetime. (b) The average number of active sensors.

the bottom left of the monitored area. For the distributed algorithm, $M = 4$ and $M = 2$ get the similar results in terms of both the network lifetime and the average number of active sensors. While for the localized approach, the case when $M = 4$ has a larger number of average active sensors and consequently, it has a shorter network lifetime compared with the case when $M = 2$. That is because for the localized approach, when there are more atomic events to detect, the coverage requirement is harder to meet and more sensors are chosen to be active.

Simulation results show that both the distributed solution and the localized approach improve the network lifetime. The localized approach does not rely on the grid division, while the distributed solution performs better in terms of the network lifetime. The location of the sink and transmission range of sensors can affect the network lifetime as well.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we focus on the sensor Scheduling for Composite Event Detection in WSNs (SCED) problem. Given a WSN deployed for watching a composite event $x_1, x_2, \ldots, x_M$, our goal is to design a sensor scheduling mechanism such that the set of active sensors ensure the coverage and connectivity conditions and WSN lifetime is maximized. One grid-based distributed algorithm and one localized algorithm are proposed. Simulation results show that our methods are effective.

ACKNOWLEDGMENT

This work was supported in part by the NSF grant CCF 0545488.

REFERENCES

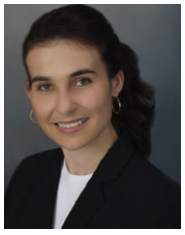
1. Akyildiz I, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. *Computer Networks* 2002; **38**(4): 393–422.
2. Culler D, Estrin D, Srivastava M. Overview of sensor networks. *IEEE Computer* 2004; **37**(8): 41–49.
3. Vu CV, Beyah RA, Li Y. Composite event detection in wireless sensor networks. *26th IEEE IPCCC*, 2007.
4. http://www.xbow.com/Products/Product_pdf_files/ Wireless_pdf/ MTS400-420_Datasheet.pdf
5. Akkaya K, Younis M. Coverage and latency aware actor placement mechanisms in WSANs. *International Journal of Sensor Networks* 2008; **3**(3): 152–164.
6. Beyer JC, Du DHC, Kusmierek E. Improved n 1-cover discovery using perimeter coverage information. *International Journal of Sensor Networks* 2008; **3**(3): 175–190.
7. Bhattacharyya M, Kumar A, Bayoumi M. Boundary coverage and coverage boundary problems in wireless sensor. *International Journal of Sensor Networks* 2007; **2**(3/4): 273–283.
8. Cardei I, Cardei M. Energy-efficient connected-coverage in wireless sensor networks. *International Journal of Sensor Networks* 2008; **3**(3): 201–210.
9. Ferrari G, Pagliari R, Martalo M. Decentralised binary detection with non-constant SNR profile at the sensors. *International Journal of Sensor Networks* 2008; **4**(1/2): 23–36.
10. Kumar S, Kambhatla KKR, Zan B, Hu F, Xiao Y. An energy-aware and intelligent cluster-based event detection scheme in wireless sensor networks. *International Journal of Sensor Networks (IJSNET)* 2008; **3**(2): 123–133.
11. Li JH, Yu M. Sensor coverage in wireless ad hoc sensor networks. *International Journal of Sensor Networks* 2007; **2**(3/4): 218–229.
12. Liu C, Wu K, Xiao Y, Sun B. Random coverage with guaranteed connectivity: joint scheduling for wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems* 2006; **17**(6): 562–575.
13. Iyengar R, Kar K, Banerjee S. Low-coordination wake-up algorithms for multiple connected-covered topologies in sensor nets. *International Journal of Sensor Networks* 2009; **5**(1): 33–47.
14. Iyer JV, Yu H, Kim H, Kim EJ, Yum KH, Mah P. Assuring K-coverage in the presence of mobility and wear-out failures in wireless sensor networks. *International Journal of Sensor Networks* 2009; **5**(1): 58–65.
15. Thai MT, Wang F, Hongwei Du D, Jia X. Coverage problems in wireless sensor networks: designs and analysis. *International Journal of Sensor Networks* 2008; **3**(3): 191–200.
16. Wang J, Zhong N. Minimum-cost sensor arrangement for achieving wanted coverage lifetime. *International Journal of Sensor Networks* 2008; **3**(3): 165–174.
17. Zhou S, Wu M, Shu W. Improving mobile target detection on randomly deployed sensor networks. *International Journal of Sensor Networks* 2009; **6**(2): 115–128.
18. Kumar AVUP, Reddy V AM, Janakiram D. Distributed collaboration for event detection in wireless sensor networks, *MPAC'05*, 2005.
19. Carle J, Simplot D. Energy-efficient area monitoring for sensor networks. *IEEE Computer* 2004; **37**(2): 40–46.
20. Tian D, Georganas ND. A coverage-preserving node scheduling scheme for large wireless sensor networks. *WSNA'02*, 28th September 2002.
21. Liu B, Towsley D. On the coverage and detectability of wireless sensor networks. *Proceedings of WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2003.
22. Yang S, Li M, Wu J. Scan-based movement-assisted sensor deployment methods in wireless sensor networks.

IEEE Transactions on Parallel and Distributed Systems 2007; **18**(8): 1108–1121.
23. Adjih C, Jacquet P, Viennot L. Computing connected dominated sets with multipoint relays. *Ad Hoc & Sensor Networks* 2005; **1**: 27–39.
24. Alzoubi KM, Wan P-J, Frieder O. Distributed heuristics for connected dominating sets in wireless ad hoc networks. *Journal of Communications and Networks* 2002; **4**(1): 22–29.
25. Chen B, Jamieson K, Balakrishnan H, Morris R. Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *ACM Wireless Networks Journal* 2002; **8**(5): 481–494.
26. Guha S, Khuller S. Approximation algorithms for connected dominating sets. *Algorithmica* 1998; **20**(4): 374–387.
27. Wan PJ, Alzoubi K, Frieder O. Distributed construction of connected dominating set in wireless ad hoc networks. *Proceedings of the IEEE INFOCOM 2002* 2002; **3**: 1597–1604.
28. Wu J, Yang S, Dai F. Iterative local solutions for connected dominating set in ad hoc wireless networks. *IEEE Transactions on Computers* 2008; **57**(5): 702–715.
29. Wu J. Extended dominating-set-based routing in ad hoc wireless networks with unidirectional links. *IEEE Transactions on Parallel and Distributed Systems* 2002; **9**(3): 189–200.
30. Wu J, Li H. On calculating connected dominating sets for efficient routing in ad hoc wireless networks,*Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1999; 7–14.
31. Dai F, Wu J. An Extended localized algorithm for connected dominating set formation in ad hoc wireless networks. *IEEE Transactions on Parallel and Distributed Systems* 2004; **15**(10): 908–920.
32. Singh S, Raghavendra CS, Stepanek J. Power efficient broadcasting in mobile ad hoc networks. *Proceedings of the PIMRC'99*, 1999.
33. McNeff JG. The global positioning system. *IEEE Transactions on Microwave Theory and Techniques* 2002; **50**: 645–652.
34. He T, Huang C, Blum BM, Stankovic JA, Abdelzaher T. Range-free localization schemes for large scale sensor networks. *ACM Mobicom*, 2003.
35. Hu L, Evans D. Localization for mobile sensor networks. *ACM Mobicom*, 2004.
36. Heinzelman W, Chandrakasan A, Balakrishnan H. Energy-efficient communication protocol for wireless microsensor networks, *Hawaii International Conference on System Sciences (HICSS)*, 2000.
37. Arora A, Ramnath R, Ertin E, *et al*. ExScal: elements of an extreme scale wireless sensor network. *RTCSA* 2005; 102–108.
38. Kulkarni P, Ganesan D, Shenoy PJ. The case for multi-tier camera sensor networks. *NOSSDAV* 2005; 141–146.

AUTHORS' BIOGRAPHIES

Yinying Yang is a Ph.D. candidate at the department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, Florida, USA. Yinying obtained her Master degree in Computer Science in 2005 from Southeast University, China. Her main research interests are in the areas of wireless sensor networks.

Arny Ambrose is a Ph.D. candidate at the department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, Florida, USA. She received her Master degree at Florida Atlantic University in 2008. Arny's main research interests are in wireless sensor networks.



Mihaela Cardei is an Associate Professor in the department of Computer Science and Engineering at Florida Atlantic University, Boca Raton, Florida, USA and Director of the NSF-funded Wireless and Sensor Network Laboratory. Dr Cardei received her Ph.D. and M.S. in Computer Science from the University of Minnesota, Twin Cities, in 2003 and 1999, respectively. Her research interests include wireless networking, wireless sensor networks, network protocol and algorithm design, and resource management in computer networks. Dr Cardei is a recipient of an NSF CAREER Award and a recipient of the 2007 Researcher of the Year Award at Florida Atlantic University. She is a member of IEEE and ACM.