SCHEDULING FOR COMPOSITE EVENT DETECTION IN WIRELESS SENSOR NETWORKS

by

Arny I. Ambrose

A Thesis Submitted to the Faculty of
The College of Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Master of Science

Florida Atlantic University
Boca Raton, Florida
December 2008

SCHEDULING FOR COMPOSITE EVENT DETECTION IN WIRELESS SENSOR NETWORKS

by

Arny I. Ambrose

This thesis was prepared under the direction of the candidate's thesis advisor, Dr. Mihaela Cardei, Department of Computer Science and Engineering, and has been approved by the members of her supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

SUPERVISORY COMMITTEE:

MCardei
Mihaela Cardei, Ph.D.
Thesis Advisor

Imad Mahgoub, Ph.D.

Ionut Cardei, Ph.D.

Borko Furht, Ph.D.
Chair, Department of Computer Science and Engineering

Karl K. Stevens
Karl K. Stevens, Ph.D., P.E.
Dean, College of Engineering and Computer Science

Barry T. Rosson
Barry T. Rosson, Ph.D.
Dean, Graduate College

Nov. 21, 2008
Date

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Mihaela Cardei for her guidance. She introduced me to Wireless Sensor Networks and then made it possible for me to complete a thesis in this field. She has helped me in every aspect of this work, this includes improving my writing skills, formatting and troubleshooting. She was a wealth of information and was always available to assist me.

I would also like to thank my friends and especially my family who are always available for information and moral support. My mother in particular for her constant support, motivation and grammar checking.

Lastly I would like to thank the other members of my committee, Dr. Imad Mahgoub and Dr. Ionut Cardei.

ABSTRACT

Author:	Arny I. Ambrose
Title:	Scheduling for Composite Event Detection in Wireless Sensor Networks
Institution:	Florida Atlantic University
Thesis Advisor:	Dr. Mihaela Cardei
Degree:	Master of Science
Year:	2008

Wireless sensor networks are used in areas that are inaccessible, inhospitable or for continuous monitoring. The main use of such networks is for event detection. Event detection is used to monitor a particular environment for an event such as fire or flooding. Composite event detection is used to break down the detection of the event into the specific conditions that need to be present for the event to occur. Using this method, each sensor node does not need to carry every sensing component necessary to detect the event. Since energy efficiency is important the sensor nodes need to be scheduled so that they comsume as little energy as possible to extend the network lifetime. In this thesis, a solution to the sensor Scheduling for Composite Event Detection (SCED) problem will

be presented as a way to improve the network lifetime when using composite event detection.

To my parents, Agatha and Everton Ambrose

TABLE OF CONTENTS

LIST OF FIGURES viii

1 Introduction 1

  1.1 Wireless Sensor Networks 1

    1.1.1 Wireless Sensor Networks Design Factors 3

    1.1.2 Technical Challenges 8

    1.1.3 Applications 10

  1.2 Contributions 14

    1.2.1 Problem Statement 15

2 Composite Event Detection in Wireless Sensor Networks 16

3 Coverage in Wireless Sensor Networks 22

  3.1 Worst and Best Coverage 23

    3.1.1 Worst Case Coverage – Maximal Breach Path 24

    3.1.2 Best Case Coverage – Maximal Support Path 25

  3.2 The k-UC Coverage Problem 26

  3.3 Area Coverage 30

    3.3.1 Energy Efficient Coverage 30

    3.3.2 Energy Efficient Connected Coverage 32

  3.4 Point Coverage 33

    3.4.1 Using Node Coverage as an Approximation 33

4 Contribution on Coverage for Composite Event Detection in Wireless Sensor Networks 34
4.1 Problem Definition 34
4.2 Algorithm 38
4.2.1 Distributed Approach to Decide Sensing Nodes 40
4.2.2 The Case when $R_c \geq r_1\sqrt{5}$ 44
4.2.3 The Case when $R_c < r_1\sqrt{5}$ 45
4.3 Simulation Study 50
4.3.1 Simulation Settings 50
4.3.2 Simulation Results 52
5 Conclusions 57
6 References 58

LIST OF FIGURES

Figure 1 The components of a sensor node [18] 2

Figure 2 The general format of a WSN [18] 2

Figure 3 Event query in a WSN [4] 20

Figure 4 The coverage problem where sensing ranges are unit disks [7] 27

Figure 5 Determining segment and perimeter coverage 28

Figure 6 Sensors deployed in an $l \times l$ square area 37

Figure 7 Main steps of a network round 39

Figure 8 Two sensors s_i and s_k are in the same $r_h \times r_h$ region 43

Figure 9 Maximum distance between sensors in neighboring regions 44

Figure 10 Connectivity mechanism 46

Figure 11 Network lifetime of distributed scheduling compared to no scheduling 52

Figure 12 Network lifetime with different sensing energy when R_c=30m 53

Figure 13 Average number of sensors active per round 54

Figure 14 Average number of sensors active per round with different sensing energy ... 55

Figure 15 Number of sensors still alive 56

1 Introduction

In this chapter we introduce the main features of Wireless Sensor Networks (WSNs) in section 1.1, followed by a discussion on the main contributions of the thesis in section 1.2.

1.1 Wireless Sensor Networks

A wireless sensor network (WSN) is a wireless network that consists of spatially distributed, autonomous devices called sensor nodes that make use of different sensors to cooperatively monitor physical or environmental conditions.

A sensor node is made up of four main components as shown in Figure 1[18]: a sensing unit, the processing unit, a transceiver unit and a power unit. More features maybe added to increase the functionality, such as global positioning systems (GPS) or directional antennas. Sensor may also contain an analog-to-digital converter (ADC) which will read the analog signals of the sensors and convert this information to digital signals which are sent to the processing unit. The processing unit manages all the schemes and procedures in which the sensor node works together with other nodes in order to carry out the allocated tasks. The transceiver unit connects the node to the network. The power unit may contain a battery or solar cell and these are used to power the sensor node.

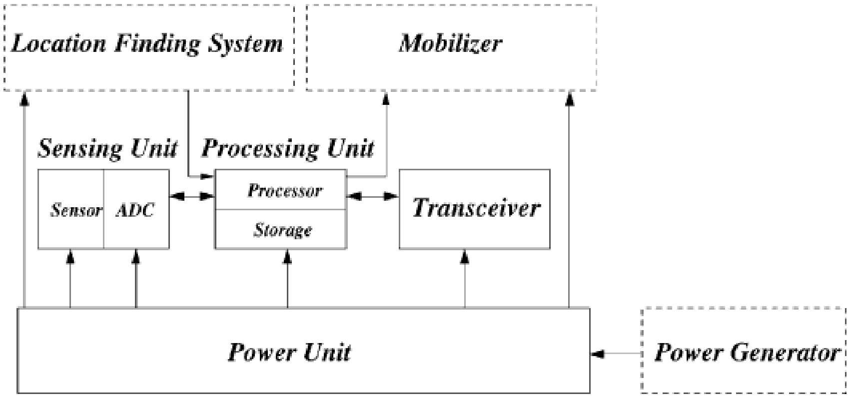


Figure 1 The components of a sensor node [18]

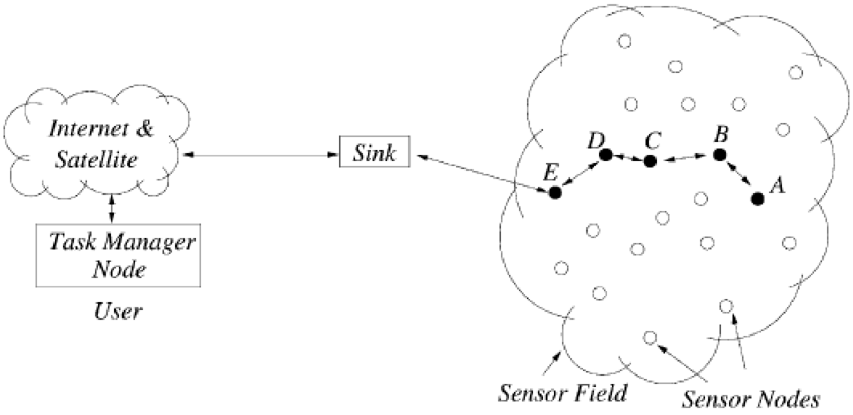


Figure 2 The general format of a WSN [18]

Sensors are usually distributed in a specific area as shown in Figure 2 above. The sensor nodes collect information about the environment they are placed in and relay this information in a multi-hop fashion to the sink or Base Station (BS). Specific nodes are designated as relay nodes and form a path so that information can be sent to the BS (Fig. 2). The BS then routes the information to the user by means of internet, satellite or other convenient architecture.

1.1.1 Wireless Sensor Networks Design Factors

There are many factors that need to be considered when designing a WSN. These include *fault tolerance, scalability, operating environment, network topology and power consumption* [18]. Since energy is also an important factor in WSNs energy efficiency such as computing the optimal number of sensor nodes needed to design a WSN that is sustainable for a given lifetime [25] is also important. This includes factors such as sensor active and sleep times and also the energy used by the sensor when in the active mode. In this thesis, we will discuss the most important factors that affect the design of a WSN.

1.1.1.1 Fault Tolerance

Fault tolerance is the ability to sustain function network functionalities without any interruption due to sensor node failure [27]. Sensor nodes fail due to environmental interference, physical damage or lack of energy. The level of fault tolerance necessary in a network depends on what the network is being used for. For example a network used to measure temperature around a building would require less fault tolerance than a network used to detect the concentration of contaminants in drinking water.

In [26], fault tolerance is modeled using the *Poisson distribution* to show the probability of not having a failure over a certain time period (0, t): $R_k(t) = \exp(-\lambda_k t)$ where λ_k is the failure rate of a sensor node *k* in the time interval *t*.

Due to the importance of these networks and the types of terrain that the sensor nodes would be placed in, it is necessary that the network can compensate for the failure of a few nodes so that the important information can still be monitored.

1.1.1.2 Scalability

The density of the sensor node distribution will usually be dictated by the application that the sensor network is being used for. A network that is being used in a home may only contain sensors in the order of the tens while a sensor network to be deployed to analyze an area in the forest may contain thousands of sensors. It is essential that the scheduling algorithm be able to adapt to the increased density of the sensor nodes and take advantage of this to increase the network life time and fault tolerance.

1.1.1.3 Operating Environment

WSNs are used to monitor phenomena and so are deployed near or inside the region where a certain phenomenon is occurring. Because of this the sensors are usually placed in remote geographic locations and have to operate unattended. Some environments that sensors are placed in are [18]:

- mountainous areas
- lakes, rivers and oceans
- biologically or chemically contaminated fields or waterways
- war zones in enemy territory
- attached to animals
- animal habitats
- fast moving vehicles

The list above illustrates the harsh conditions that sensor nodes are expected to perform under. The type of environment that the sensor nodes will be used in will determine the types of sensors that should be used.

1.1.1.4 Network Topology

It can be challenging to setup and maintain a sensor network topology when deploying thousands of sensors over an area. In [18], the deployment of the sensor nodes is separated into three distinct phases.

The pre-deployment and deployment phase. This is the phase where the sensor nodes are initially tossed randomly or placed in the sensor field. This can be done by dropping from a plane, delivered by a rocket or missile or placed by a human or robot.

Due to the large numbers of sensors that need to be placed the preferred method of deployment is an unattended type. What determines how the sensor nodes are deployed into an area, is the cost of deployment, the flexibility of the arrangement and the ability of the network to be fault tolerant and self organized once deployed.

Post-deployment phase. Once the sensor nodes are deployed, the network topology may change due to:

- the position of a node
- node failure
- reachability – which may decrease due to noise and other interference
- lack of energy

Sensor nodes are usually static but mobile sensor nodes can be used to change the topology of the network after deployment.

Redeployment of additional nodes phase. Additional nodes might be added to the network to replace malfunctioning nodes or to fill a void. This will cause a change in the topology of the network and will have to be reorganized.

1.1.1.5 Power Consumption

A wireless sensor node is a microelectronic device that is powered by a battery. Power is limited and in many applications there is no resource for replenishing the power supply so power conservation and management is essential.

The main functions of a sensor node are to detect events, perform small processing tasks and communicate with other sensor nodes. Power consumption therefore, must take place in these areas.

A sensor node expends most of its energy communicating with its neighbors. This includes the sending and reception of messages. Low power radios [28] and energy saving algorithms are used so that a node does not have to keep its radio in the listening mode when there are no messages to be transmitted.

Data processing uses considerably less energy than data communication [29]. Since less energy is expended if the transmitted messages are smaller, performing calculations within the sensor node reduces the amount of energy spent especially in a multi hop network.

1.1.2 Technical Challenges

WSNs face a multitude of technical challenges that may hinder the operation of the network. According to [24], they are:

- Ad Hoc Deployment
- Dynamic Environmental Conditions
- Unaided Operations
- Limited Energy Supply

Ad hoc deployment – WSNs are usually dispersed in a random manner. This means that for proper operation, the sensor nodes must be able to adapt to the resultant distribution and still be able to form connections among the nodes such that they are able to carry out their necessary monitoring.

Dynamic environmental conditions – factors such as weather, animal interference and node failure will cause a change in the network topology. The network should be able to adapt to these modifications.

Unaided operations – the network operates unattended and so any changes in the configuration that need to be made should be done automatically.

Limited energy supply – the main power source for sensor nodes is a battery. Any scheduling algorithm should have energy efficiency as an important criterion of the scheme. Examples of this include putting the sensor nodes in sleep mode when they are not necessary for the network to function at that time.

The authors in [24] propose some methods of addressing these challenges that face WSNs:

- Collaborative Signal Processing
- Exploiting Redundancy
- Hierarchical, Tiered Architecture

Collaborative signal processing – this may occur among nodes in the same area that are exposed to the same events. This will enhance the efficiency measured in the data transmitted by the nodes.

Exploiting redundancy – The deployment of the sensor nodes is random. Nodes should be distributed with high density so as to compensate for errors and failures that may occur in some nodes during deployment. Also, a high density distribution will prevent areas that cannot be covered due to the absence of sensor nodes.

Hierarchical, tiered architecture – this can help increase the lifetime and performance of the network. High level processes can be used for processing tasks to reduce the work load of the low level processes. These can then be used exclusively for the sensing tasks.

1.1.3 Applications

Sensor networks consist of large numbers of sensor nodes that are deployed densely in an area to monitor a particular phenomenon. Sensors nodes may be deployed at random which is advantageous for inaccessible areas, disaster areas or war zones. Since the sensor nodes are random, it is necessary that a scheduling algorithm is used to organize the nodes and their functions. The nodes possess a certain level of processing power and so more data manipulation is possible before the data reaches the BS.

The features of the sensor nodes make WSNs suitable to a myriad of applications. The authors in [18] describe that they can be used in health, security and military applications. An example of this is a doctor who can remotely monitor a patients' physiological condition. This may be a more convenient monitoring method for both the patient and the doctor. WSNs can also be used to monitor the environment for pollutants. Air or water can be continuously monitored so that the appropriate people may be alerted when the concentration becomes higher than what is considered safe.

Sensor nodes contain a variety of sensors such as temperature, visual and infrared, that enable them to sense ambient conditions. The authors in [19] list a variety of phenomenon that these sensors may be useful for:

- temperature
- humidity
- vehicular movement
- lightning condition

- pressure
- soil makeup
- noise levels
- the presence or absence of certain kinds of objects
- mechanical stress levels on attached objects
- current characteristics such as speed, direction, and size of an object

Due to the large variety of sensors that can be added to nodes, these networks can be used for many more applications than were previously described.

1.1.3.1 Military Applications

WSNs are an important part of military systems. A network can be used for *command, control, communications, computing, intelligence, surveillance, reconnaissance and targeting* (C4ISRT) [18]. Sensor networks are suitable for this application due to their ability to be rapidly deployed in any area including inhospitable ones such as war zones. Sensor nodes can be organized using scheduling algorithms and this aids the speed and efficiency of random deployment. Also, the sensor nodes are inexpensive and disposable which is important for information gathering in hostile areas.

Some other military applications of WSNs include [18]: monitoring friendly forces and ammunition, battlefield surveillance, reconnaissance of opposing forces and terrain, targeting, battle damage assessment and nuclear, biological and chemical attack detection and reconnaissance.

According to [20], VigilNET is a long-lived real-time wireless sensor network for military surveillance. The purpose of VigilNET is to notify control and command unit to specified occurrences in unfriendly territory. These occurrences may include vehicles and people with weapons. VigilNET is used to detect and then track the point of interest over the specific area. The information gathered from the sensors can then be sent to the BS for further processing.

VigilNET is a network that consists of over 200 sensor nodes. The sensor is a *tripwire* based scheme where a target entering the sensing field activates the monitoring sensors. This scheme provides sufficient energy conservation that the entire network is able to last 3-6 months. The tripwire function is also used to activate other sensors that are not part of the VigilNET network so that they continue to monitor the target when it is no longer in range. This improves the range of the network while increasing the network lifetime.

1.1.3.2 Environmental Applications

WSNs are widely used to monitor the conditions of the environment. This includes monitoring the movement and numbers of animals such as birds, mammals and insects. They may also be used to monitor the suitability of the environment for crops, livestock and other agricultural applications. These networks may be used for soil monitoring, fire and flood detection, global monitoring, meteorological and geophysical contexts.

The random nature of WSNs lends itself to detecting events such as forests fires. When a fire is detected an alarm is raised and the appropriate response can be made. WSNs distinguish such events faster than other methods such as a satellite based detection approach [21]. Flood detection is also important and the network system ALERT has been deployed in the United States [18]. The ALERT system contains rainfall, water level and weather sensors that supply information to a centralized database system that is used to monitor and detect possible flooding.

1.1.3.3 Health Applications

Sensor networks can be applied to health such as integrated patient monitoring, drug administration in hospitals and tracking and monitoring doctors and patients inside the hospital.

In drug administration, the sensor nodes are attached to the medication and the patients will also have a sensor node that corresponds with their condition and allergies so this will minimize possible errors in prescriptions. Keeping track of doctors and monitoring patients are also very important for hospitals. These sensors will monitor the patient, while another sensor tracks the movements of the doctors around the hospital so that they can be located in a short amount of time.

AlarmNet [22, 23] is a medical oriented sensor network system for large-scale assisted living facilities. Devices are used throughout the facility where some may be attached to the patient and others are inside the patients' living space. This is done to enable the healthcare provider to have current information on the status of that person. The data are collected and processed using a variety of sensors such as pressure sensors, pollution sensors and environmental sensors. This information may be connected to a database and can be used with existing patient information.

This network was design to be used in a facility with a large number of units over an extended period of time and so although there are many wireless sensor nodes, there are other nodes that are able to use the line power. This ensures that the patient is always continuously monitored.

1.2 Contributions

In this thesis we address the issue of energy management in WSNs. We propose algorithmic approaches to prolong WSN network lifetime. In general, there are two mechanisms used to conserve the energy spent on wireless communications. One method is to put sensors in sleep mode when they are not actively participating in sensing or data relaying. A second mechanism is to adjust transmission range to the minimum value needed to reach the set of intended receivers.

In this thesis, we use the first mechanism to save energy and thus prolong WSN lifetime. We design a scheduling mechanism that allows sensors to go to sleep as long as the coverage and connectivity requirements are met. We consider that sensors are equipped with multiple sensing components, thus allowing the WSN to perform composite event detections. To obtain a good sensing quality, the coverage requirement is imposed, where each sensing component must cover the whole deployment area. Another important requirement is network connectivity, needed to ensure that the data collected by the sensors can be collected by the sink or BS.

The rest of the thesis is organized as follows. In Section 2 we present related works on composite event detection in WSNs. We continue in Section 3 with a survey on coverage in WSNs. Section 4 introduces the formal definition of the problem that we propose in this thesis, the sensor Scheduling for Composite Event Detection in WSNs (SCED) problem, our solution, and simulation results. Section 5 concludes our thesis.

1.2.1 Problem Statement

The contribution for this thesis is based on the **sensor Scheduling for Composite Event Detection in WSNs (SCED problem)**:

Given a WSN deployed for watching a composite event $\{x_1, x_2, \ldots, x_M\}$*, design a sensor scheduling mechanism such that the set of active sensors ensure the coverage and connectivity conditions and WSN lifetime is maximized.*

2 Composite Event Detection in Wireless Sensor Networks

WSNs are commonly used for detecting states in the real world. Such states may be light, temperature or motion, which are detected using sensors that can detect these real world inputs. This is known as *event detection*. According to [5], an event is defined as a change of a real world state. In [6], an event is classed as a *simple event*, such as the detection of temperature > 60°C, or a *composite event* which would be characterized as the combination of two or more simple events. Composite events require the input of multiple sensors for their detection.

Events detected in a WSN are usually used for alarm purposes such as a fire or an explosion. During the event detection process, sensors must continuously monitor events so that when an event occurs it can be identified by one or more sensors and dealt with accordingly. For good quality event detection sensors need to remain active such that every area in the network is covered and events are not missed. Also, valuable energy is wasted if each sensor is responsible for sensing every component that would compose that particular event. For example, temperature and smoke are two sensing components of the composite event fire. It is usually the case that an event remains undetected due to a blind spot created by a sensor node that has run out of battery power, a failure on one of the sensors or collisions in the network.

The sensor nodes used in event detection are very small and so resources are limited. Energy efficiency is therefore very important. Longevity in the sensor network is usually obtained using a duty cycle where the sensors are put into a sleep mode such that their energy can be conserved. The problem that arises with duty cycling is that the entire sensor area needs to be covered regardless of the sleeping sensors. This poses a particular challenge and there have been many different approaches used to solve this problem. In most cases of composite event detection, the sensors need to sense the event and this information has to be transferred to the base station (BS) for processing. Usually for an event to be recognized reliably, it must be detected by a preset number of sensors, *k*, to prevent false positives. When this event does occur however, it is desirable that the time between sensing the event and receiving the event notification by the BS be reduced as much as possible.

It has been a general consensus that all sensors must contain all the sensing components necessary to detect every simple event that comprises a composite event. The authors in [6] indicate that this is not valid due to the fact that sensor nodes are manufactured in such a way that different sensors have different sensing capabilities. Therefore, it might be such that a sensor may not be manufactured with all the sensing components necessary to detect a particular composite event. Also, there is a distinct possibility that even though every sensor contains all the necessary components to detect each simple event of a composite event, one or more sensors may fail. For example, sensor node *x* contains a temperature, humidity and motion sensor but the humidity sensor fails leaving *x* with only two out of the three required sensors. Since energy is an important part of a WSN, a sensor node may choose to discontinue the use of one of its sensing components in an effort to conserve energy.

The authors in [5] developed an algorithm such that *k* sensors of each type are necessary to detect a particular event. For example, in a fire event there may be specific factors that indicate that there is a fire in a particular area. Such factors can be for example the temperature and smoke. In regular event detection, each sensor node would be required to have a sensor for each of the preceding components. This information would then be sent to the BS periodically for processing. In the case of an event, the BS would receive readings from multiple sensors that exceed the threshold. When such information has been received from *k* or more sensors, the message that the event has occurred will be sent and dealt with accordingly. In the case of a fire, an alarm might be activated. The algorithm takes advantage of the observations of [5] and does not assume that every sensor contains all the necessary components. [4] states that it would be more cost efficient to have multiple sensor nodes with only one or a combination of the necessary sensors. Some sensors may sense only smoke, some both temperature and smoke and so on. In this way, the entire area can be detected without the need for many sensors overlapping in the detection area.

The next important part of this type event detection is the determination of an event. When an event occurs, there must be a preprogrammed method of determining that this is indeed an event and not something else irrelevant to the measurements being taken. [1] explains the definition of a composite event using the example of a fire, For example, the event *fire* is a fusion of multiple sensed values of multiple different attributes, *i.e.*, the occurrence of fire should satisfy some conditions such as *temperature* > 100 °*C AND smoke* > 100*mg/L*, rather than a simple condition *temperature* > 100 °*C* or *smoke* > 100*mg/L* alone. Any change in either t*emperature* or *smoke density* that makes *temperature* > 100 °*C* or *smoke* > 100*mg/L* true is an *atomic event*. The event that is a combination of several atomic events is a *composite event*, *e.g.*, the composite event *fire* is represented as *temperature* > 300 °*C AND smoke* > 100*mg/L*. The advantage of this kind of interpretation of data is that it can be inherent in the settings of the sensor such that the BS does not have to form conclusions based on the data sent. Sensors can collaborate to decide if a composite event took place and then send this information to the BS. The BS then can react in a timely manner.

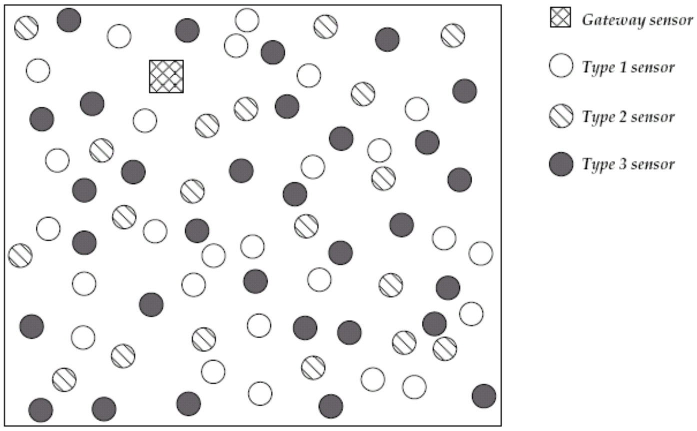


Figure 3 Event query in a WSN [4]

Figure 3 above illustrates the deployment of a WSN. One node will act as a gateway node for the other nodes in the area. This role will be rotated between all the other nodes in the cluster so that energy may be conserved. In this example, the types 1-3 are the sensing components light, temperature and smoke, respectively. An event E is defined by a compound propositional function:

$$E = (light > threshold_1) \wedge (temperature > threshold_2) \wedge (smoke > threshold_3) .$$

Each sensor already contains the threshold for its particular sensing component. When it is detected that the temperature, for example, has exceeded the threshold temperature, this information will be sent to the gateway node. It is important to note that information is only sent to the gateway node when the threshold of that sensor has been exceeded and in such cases there is no raw data sent. The gateway node only receives a confirmation that the threshold has been exceeded. When this occurs, the gateway node then sends warning to the BS.

In [4], the authors propose a greedy, centralized algorithm which tries to compute as many detection sets as possible. The sensors in each set must ensure *k* watching of the composite event. The detection sets are non-disjoint sets which are activated successively in time: where a set is active, all the other nodes go to the sleep mode.

The centralized algorithm is executed by the BS, which uses a Breadth First Search (BFS) [17], approach, to construct each detection set. In this way the notification time from active sensors to the BS is smaller, since BFS obtains the shortest paths.

An advantage of this method is that only small amounts of data are sent to the base station. Because of this, there is a reduction in the amount of energy required by the sensors for transmitting and this extends the life of the network. There is also less information being transmitted at any time so there will be a great reduction in the number of collisions and contentions. Since the information is sent to the base station by gateway nodes only, when an event occurs, the location of such an event will be immediately obvious based on the gateway node that transmitted the warning. Also, if there is a failure of one or more sensors, an event will still be detected in a timely manner as sensing components of many sensors are being employed in the detection process.

3 Coverage in Wireless Sensor Networks

Coverage is a major issue in WSNs. According to [7] coverage is defined as how well the sensor network is monitored or tracked by the sensors. Sensors are usually distributed throughout the target area in a random manner.

Since sensors are usually deployed very densely, it is important that there is some algorithm used to schedule the sensors. This must be done in an effort to prevent network inefficiency that would result if the entire dense population of sensors were to remain active at the same time. Sensors are likely to be placed in the same location, very close to each other. In such a case, it would be a waste of energy to have all sensors active at the same time. This would also cause multiple problems. Since sensors are sensing the same area, they will transmit identical information to the same relay nodes at the same times. The data are redundant and this increases collisions and contentions in the network as many sensors try to transmit their data at the same time.

Energy efficiency is also important. If all sensors are active at the same time this results in a shorter network lifetime. By taking advantage of a dense population of sensors and keeping active only minimum number of sensors necessary for the coverage requirement, the network lifetime will be dramatically increased as other sensors in the same or similar locations can go to sleep mode.

The idea of *coverage* is such that every point in the network area is observed by one or more sensors at any given time. The *Art Gallery Problem* [9] determines the number of observers necessary to cover an art gallery so that every point in the gallery is monitored by at least one observer. The gallery represents a WSN and the observers are sensor nodes. 2D model of such a problem does possess and optimal solution but when the same principal is applied to a 3D plane the problem becomes NP-hard; a different approach is necessary.

3.1 Worst and Best Coverage

Many sensor networks are used to detect events in the natural landscape that is inaccessible for normal monitoring. With this in mind it is important to acknowledge the weaknesses and strengths of the networks that have been deployed. In [8], the authors propose that an algorithm is required to test the coverage of the network after the sensors have been deployed.

In most cases of sensor deployment, it is hard to determine the exact placement of the sensors. The sensors are usually set out randomly. This can be done uniformly or based on one of the many different stochastic schemes such as Gaussian. The authors in [8] introduce two perspectives of sensor coverage: worst and best case coverage. Worst case coverage looks specifically for areas of low connectivity in which to detect a breach in the network. Similarly, the best case coverage looks at the areas of high connectivity from the sensors which would provide the best support regions.

3.1.1 Worst Case Coverage – Maximal Breach Path

The algorithm in [8] is used to find the maximal breach path in the network coverage. The problem is defined as: Identify P_B, the maximal breach path in sensor set S, starting with location I and ending with F.

P_B is a path through a field A with the property that for any point p on the path, the distance between p and the closest sensor is maximized. The starting and ending point are arbitrary.

These are the steps proposed in [8] to solve this problem.

1. Generate a Voronoi diagram for S
2. Apply graph theory abstraction.
3. Find P_B using binary search and Breadth First Search (BFS)

Since the edges of the Voronoi extend to infinity, the concern is the area in A, the diagram is cut back to the boundaries of A. Also, the edges along the boundaries of A create a valid path so these edges must be added to the diagram.

In this algorithm, the Voronoi diagram is generated and then a weighted, undirected graph is constructed by creating a node for each vertex and an edge that relates to each line on the Voronoi diagram. Weights are then assigned, determined by the minimum distance between the edge and the closest sensor node in S. Then binary search and BFS are used to define a path that is larger than the criteria *breach weight*.

This maximal path created by this algorithm is not unique as it is based on the breach weight that is calculated in the binary search portion of the algorithm. To improve the breach weight the sensors can be moved or new sensors can be added. This will decrease the breach weight and improve the worst case coverage.

3.1.2 Best Case Coverage – Maximal Support Path

This algorithm shows the maximal support path of a network which highlights the areas of greatest connectivity in the network. This is defined as: Identify P_s, the path of maximal support in S, starting in location I and ending in F.

P_s in this case is a path through a field A, with the property that for any point p on the path P_s, the distance from p to the closest sensor is minimized.

The algorithm for the maximal support path is similar to that of the maximum breach path with few exceptions.

1. The Voronoi diagram is replaced with Delaunay triangulation as the underlying geometric structure.
2. The weight is now support weight instead of the breach weight.

As in the last case, the maximal support path may not be unique. Also, the addition or rearranging of sensors will result in an improvement in the support path.

3.2 The k-UC Coverage Problem

In [7], the authors propose a general sensor coverage problem, given a set of sensor in a target area, the objective is to determine whether the set of sensors provides the *k* coverage such that every point in the target area is covered by at least *k* sensors. This is done by focusing on the perimeter of the sensors' sensing range. In this way the assumption is made that as long as the perimeter of a sensor's sensing range is covered then the entire area is covered as well. This reduces an NP-hard problem to a polynomial time algorithm. Figure 4 below shows a section of the network where the sensing ranges of the sensor nodes are assumed to be disks of equal size as in the coverage problem below. The numbers represent the number of sensor nodes that overlap the perimeter of another node.

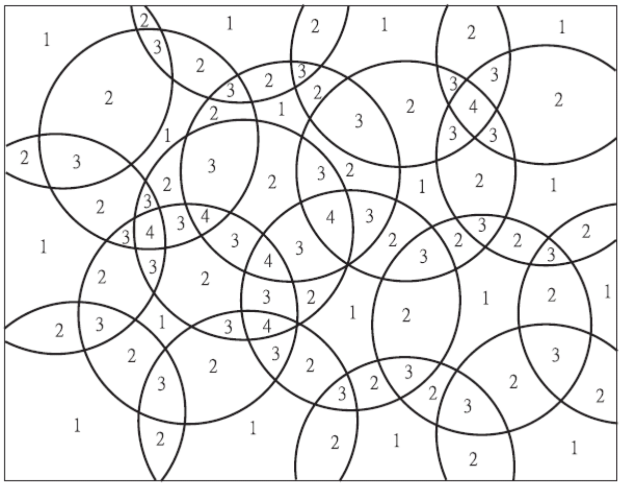


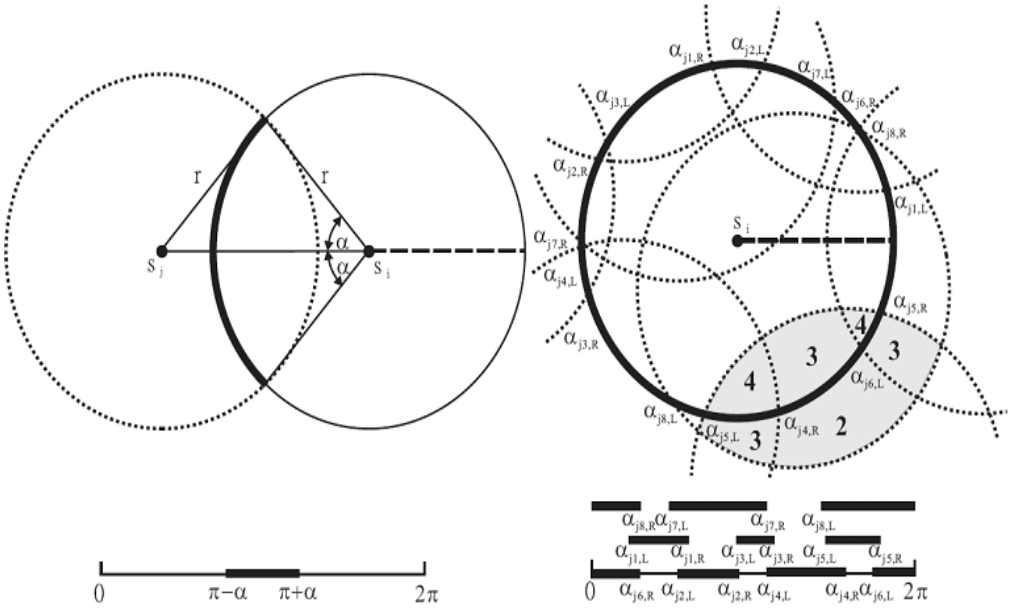
Figure 4 The coverage problem where sensing ranges are unit disks [7]

In most cases, *k=1* is sufficient for regular monitoring purposes. Where multiple sensors are required to sense a particular event, *k* may be greater than 1. The definition of the problem is as follows: *Given a natural number k, the k-Unit-disk Coverage (k-UC) Problem is a decision problem whose goal is to determine whether all points in a location, A, are k-covered or not subject to the constraint that* $r_1 = r_2 = \ldots = r_n$ *(where r is the sensing radius of the sensors).*

This algorithm focuses on the coverage of the perimeter of the sensing range instead of the entire region. It seeks to determine whether the perimeter of the sensor is adequately covered. Information about the network must be collected and therefore has been defined by the authors.

- Consider any two sensors s_i and s_j. A point on the perimeter of s_i is perimeter-covered by s_j if this point is within the sensing range of s_j.

- Consider any sensor s_i. s_i is k perimeter-covered if all points on the perimeter of s_i are perimeter-covered by at least k sensors other than s_i itself. Similarly, a segment of s_i's perimeter is k perimeter-covered if all points on the segment are perimeter-covered by at least k sensors other than s_i itself.



a. Segment of s_i's perimeter covered by s_j b. Perimeter-coverage of s_i's perimeter

Figure 5 Determining segment and perimeter coverage

The algorithm determines whether a sensor is k-perimeter-covered where d is the number of sensors which have intersection with that sensor. Consider sensors s_i and s_j located in positions (x_i, y_i) and (x_j, y_j). Denote by $d(s_i, s_j) = \sqrt{| x_i - x_j |^2 + | y_i - y_j |^2}$ the distance between s_i and s_j. If $d(s_i, s_j) > 2r$, then s_j does not contribute any coverage to

s_i's perimeter. Otherwise, the range of perimeter of s_i covered by s_j can be calculated as follows (as in Figure 5(a)). Without loss of generality, let s_j be resident on the west of s_i ($y_i = y_j$ and $x_i > x_j$). The angle $\alpha = \arccos(\frac{d(s_i, s_j)}{2r})$, so the arch of s_i falling in the angle $[\pi - \alpha, \pi + \alpha]$ is perimeter covered by s_j.

This is the algorithm as proposed by the authors in [7]:

1. For each sensor s_j such that $d(s_i, s_j) \le 2r$, determine the angle of $s_i^{/}s$ arch, denoted by $[\alpha_{j,L}, \alpha_{j,R}]$, that is perimeter-covered by s_j.
2. For all neighboring sensors s_j of s_i such that $d(s_i, s_j) < 2r$, place the points $\alpha_{j,L}$ and $\alpha_{j,R}$ on the line segment $[0, 2\pi]$ and sort all the points in an ascending order into a list *L*. Also, properly mark each point as a left or right boundary of a coverage range.
3. (Sketched) Traverse the line segment $[0, 2\pi]$ by visiting each element in the sorted list *L* from the left to right and determine the perimeter-coverage of s_i.

Let *d* be the maximum number of sensors that are neighboring to a sensor $(d \le n)$. Step 3 is sketched and can be implemented as follows. Whenever an element $\alpha_{j,L}$ is traversed, the level of perimeter-coverage should be increased by one. Whenever an element $\alpha_{j,R}$ is traversed, the level of perimeter-coverage should be decreased by one. Since the sorted list L will divide the line segment $[0,2\pi]$ into as many as $2d + 1$ segments. An example of this is shown in Figure 5(b).

The running time of this algorithm is $O(nd \log d)$ where d is the maximum number of sensors that may intersect the perimeter of another sensor.

This algorithm is a useful solution to the coverage problem. It successfully determines whether an area is fully covered based on the perimeter coverage and also is useful for k coverage. This is very important in composite event detection where more than one sensor is necessary to monitor the same region. This is a very complicated algorithm however, and in this thesis complete coverage can be maintained using multiple sensors without such complicated methods.

3.3 Area Coverage

The *area coverage* problem demonstrates how a network covers an area. The objective is to have every point in the area to be covered be monitored by at least one sensor. In [10], the algorithms surveyed fall under two distinct types: *energy efficient coverage* and *energy efficient connected coverage*.

3.3.1 Energy Efficient Coverage

The algorithms considered used a large number of randomly distributed sensor nodes to monitor a specific area. The number of sensor nodes actually deployed far exceeds the number of sensor nodes required to cover the specified area a scheduling algorithm is used. Only the sensors required to cover the area at anytime remain active while the others go into sleep mode. This is done by separating the sensor nodes into *disjoint sets* where each set can independently monitor the entire area. The remaining sets can enter sleep mode therefore saving energy. The important part of this approach is to create an algorithm that produces the maximum number of coverage sets so that the network lifetime is increased as much as possible.

In [11], the deployment area is separated into fields and the algorithm to compute the sets gives precedence to the sensor nodes that are in critical positions in the network. These are those that cover a large number of fields, do not redundantly cover any fields and cover a sparsely populated field. Another algorithm in [12] represents the sets as disjoint dominating sets where the sensor nodes form the vertex set and an edge connects any two vertices that are within the sensing range of the other sensor node. A graph-coloring mechanism is used to obtain the sets disjoint dominating sets. The nodes are colored. Non dominating sets are considered in an increasing color and converted to dominating sets if they contain the minimum number of higher color vertices. This method creates more disjoint sets than the former which would make it a more energy efficient algorithm.

The solution in [13] is based on probing and node scheduling. All the sensor nodes have the same sensing range and coverage is characterized by the ratio between the area being monitored and the entire network area. To determine offline eligibility, a sensor node sends a probing message over a specified range. If there is one or more response to this message the node enters sleep mode.

3.3.2 Energy Efficient Connected Coverage

Connectivity is also very important to consider when looking at coverage. A network is said to be connected, [10], if any active node in the network can communicate with any other active node. This may be done using intermediate nodes to serve as relays between them. The network must be connected so that information can be transferred from one node to another or from the nodes to the BS. To produce the most energy efficient connected coverage, it is important that the coverage includes the minimum number of active sensor nodes that allows the network to be fully covered and completely connected.

The authors of [14] prove that the optimal communication range for a sensor node should be at least twice the sensing range. Since sensor nodes are said to be neighbors if their sensing ranges overlap, then the importance of a large sensing range is self evident. It is unwise however, to make the communication range too large as this will cause unnecessary interference caused by nodes that are further away hearing or responding to messages. The algorithm proposed in [14] puts nodes into three different states: *on, off* or *undecided*. At the beginning of the algorithm, all nodes are *undecided*. The nodes keep a record of the states of their neighbors and when a node receives an *on* message it checks whether its sensing area is completely covered by its neighbors. If it is, the node will put itself into the *off* state. This serves to extend the lifetime of the network without sacrificing connectivity.

3.4 Point Coverage

The objective of *point coverage* is to cover a set of points with known locations. The problem is that are large number of sensors are dispersed close to the intended targets. Every target must be observed continuously by at least one sensor node.

In [10] the authors note that similar to the area coverage problem the sensor nodes can be partitioned into disjoint sets such that each set completely covers all the targets to be monitored. By scheduling the nodes such that only one set remains active while the others sleep and also determining the largest number of sets possible for the sensor node distribution, the network lifetime can be prolonged.

3.4.1 Using Node Coverage as an Approximation

The coverage of the sensor nodes can be used to approximate the area coverage when there is a large and dense population of sensor nodes. One method described in [10] that will ensure coverage and connectivity is the use of a *connected dominating set (CDS)*.This is constructed using the marking process proposed in [15]. A node becomes a *coverage node* if two of its neighbors are not within transmission range of each other and therefore not connected. Coverage nodes make up a CDS. If a pruning process is then applied such as in [16], the number of coverage nodes can be reduced while still keeping the properties of a CDS. This pruning rule can help the sensor nodes decide which node can go to sleep without affecting the entire CDS. This would lengthen the network lifetime.

4 Contribution on Coverage for Composite Event Detection in Wireless Sensor Networks

4.1 Problem Definition

In this thesis, we consider that each sensor is equipped with one or multiple sensing components from the set of sensing components $\{x_1, x_2, \ldots, x_M\}$. In general, each sensing components (e.g. presence detector sensor, camera, acoustic) might have a different coverage capability, which means that they have different sensing ranges.

Let us consider that each sensing component x_i has coverage range $r_i\sqrt{2}$. Then any sensor with sensing component x_i located in a $r_i \times r_i$ grid region will completely cover that region.

For simplicity, let us assume that $r_i = 2^t r_{i-1}$, where $t \geq 0$. If $t = 0$, then the two consecutive sensing components x_i and x_{i-1} have the same coverage.

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. Thus, in many papers the idle mode is not differentiated from the receiving state. A radio is in sleep mode when both the transmitter and the receiver are turned off.

According to [1], studies on several commercial radios (e.g. WaveLAN, Metricom) show that in sleep mode the power consumption ranged between 150-170mW, while in idle mode the power consumption went up by an order of magnitude. In most commercial radios, typically the energy consumed in transmitting is roughly twice the amount of energy consumed in receiving a packet.

According to [2], the power consumed by the Rockwell WINS seismic sensor nodes is as follows: transmit mode 0.38 ÷ 0.7W, receive mode 0.36W, idle mode 0.34W, and sleep mode 0.03W. The power consumed by the seismic sensor is 0.02W.

Work [3] presents LEACH (Low-Energy Adaptive Clustering Hierarchy), a data gathering protocol for WSNs. The first order radio model [3] assume a simple model where radio dissipates E_{elec} = 50 nJ/bit to run the transmitter and the receiver circuitry and

ε_{amp} = 100 pJ/bit/m^2 for the transmit amplifier. The energy used to transmit a *k*-bit message over a distance *d* is:

$$E_{Tx}(k, d) = E_{Tx\text{-}elec}(k) + E_{Tx\text{-}amp}(k, d) = E_{elec}*k + \varepsilon_{amp}*k*d^2 \qquad (1)$$

The energy consumed to receive a k-bit message is:

$$E_{Rx}(k) = E_{Tx\text{-}elec}(k) = E_{elec}*k \qquad (2)$$

There are two main mechanisms used to conserve the energy spent on wireless communication. One method is to put sensors to sleep mode when they are not actively participating in sensing or data relaying. A second mechanism is to adjust the transmission range to the minimum value needed to reach the set of intended receivers. Then the energy spent on data transmission is reduced, according to equation (1).

In this thesis we will use the first mechanism in order to conserve energy and thus to prolong wireless sensor network lifetime. We are concerned with designing a scheduling mechanism that will allow sensors to go to the 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_i, for $1 \le i \le M$.

The *connectivity requirement* requires that the set of active sensors to be connected. This condition is needed 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.

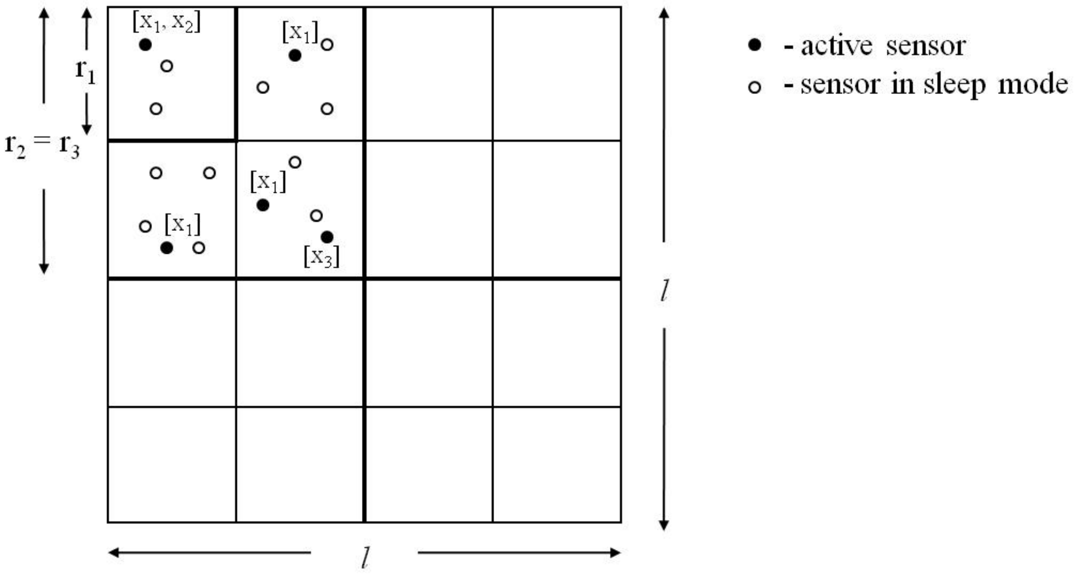


Figure 6 Sensors deployed in an $l \times l$ square area

Figure 6 shows an example with M = 3 sensing components $\{x_1, x_2, x_3\}$, where $r_3 = r_2 = 2r_1$. The figure illustrates sensors in a quarter of the deployment area, both active and in the sleep mode. The set of active sensors satisfy 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.

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

Given a WSN deployed for watching a composite event {x_1*,* x_2*, …,* 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.*

4.2 Algorithm

In this section, we propose a sensor scheduling algorithm for the SCED problem. The network lifetime is organized in rounds. Each round has two phases: *initialization* and *data collection*. In the *initialization* phase, the scheduling algorithm is run to decide which sensors remain active and which sensors go to sleep during the current round. In the *data collection* phase, active sensor nodes perform sensing and data relaying.

A 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 6 shows the two main phases of a network round.

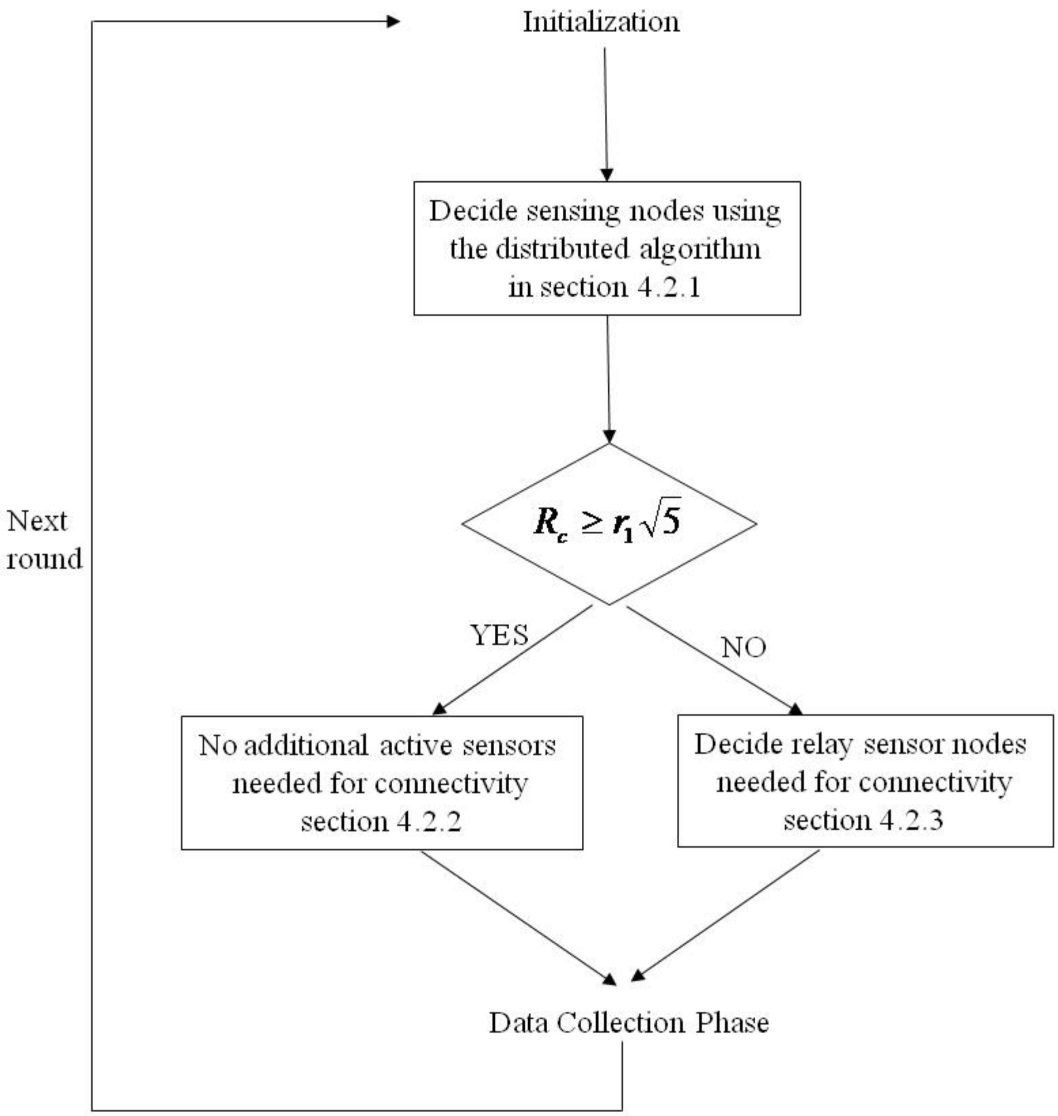


Figure 7 Main steps of a network round

We consider that the deployment area is a square divided into a grid of size $r_1 \times r_1$. Let us denote with l the size of the deployment area square, where $l = k \cdot r_M$ for some integer k. For example in Figure 6, M = 3, k = 2, and $l = 2 \cdot r_3$. The smallest grid granularity is $r_1 \times r_1$ and the largest grid granularity is $r_M \times r_M$.

4.2.1 Distributed Approach to Decide 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 assume that each sensor knows its location using GPS or another location computation mechanism.

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

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 located in the same $r_j \times r_j$ region becomes active.

Below we present the algorithm *DecideStatus()* run by each sensor s_i at the beginning of each time interval t_j.

```
Algorithm DecideStatus (sensor $s_i$, time interval $t_j$)
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 return
compute contribution function $\chi_i \leftarrow f(E_i, R_c) \bullet \frac{h_i}{sc_i}$
start timer with value $\tau_i$, which is inversely proportional to $\chi_i$
if message StatusActive($s_k$, $s_k$'s location, $s_k$'s sensing components $\geq j$) received
    for each $s_k$'s sensing component $h \geq j$
        if ($s_i$ is equipped with $x_h$) and ($g_h = 0$) and ($s_i$ and $s_k$ are in the same $r_h \times r_h$ region)
        then $g_h \leftarrow 1$
    if $g_j = 1$ then return
if timer fires up
    then sensor $s_i$ is set active and $g_j \leftarrow 1$
        broadcast StatusActive($s_i$, $s_i$'s location, $s_i$'s sensing components $\geq j$) in $s_i$'s region $r_w \times r_w$, where w is the index of the largest sensing component of $s_i$
return
```

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, 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 $\chi_i \leftarrow f(E_i, R_c) \bullet \frac{h_i}{sc_i}$.

The contribution function of a sensor s_i is computed similar to [4] where:

- $f(E_i, R_c)$ is a function to calculate s_i's lifetime depending on its current residual energy E_i and its current communication range d_i. This function can be computed as $f(E_i, R_c) = \frac{E_i}{Tx_i + Sx_i}$ where Tx_i is the energy consumed on transmission in one round and Sx_i is the energy consumed on sensing in one round
- h_i is the number of s_i's helpful sensing components. A sensing component x_u of s_i is called a helpful sensing component if $u > j$ and $g_u = 0$.
- sc_i is the number of all sensing components that s_i is equipped with.

A larger contribution function means s_i has larger priority in becoming active, since it has more residual energy and/or more sensing components to be examined in the future. 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, that means that s_k has become active recently. Then for each s_k's sensing component $h \geq j$, if s_i is equipped with x_h and if s_i and s_k are in the same $r_h \times r_h$ region, then set $g_h \leftarrow 1$. If g_j becomes 1, then the procedure returns without requiring s_i to become active.

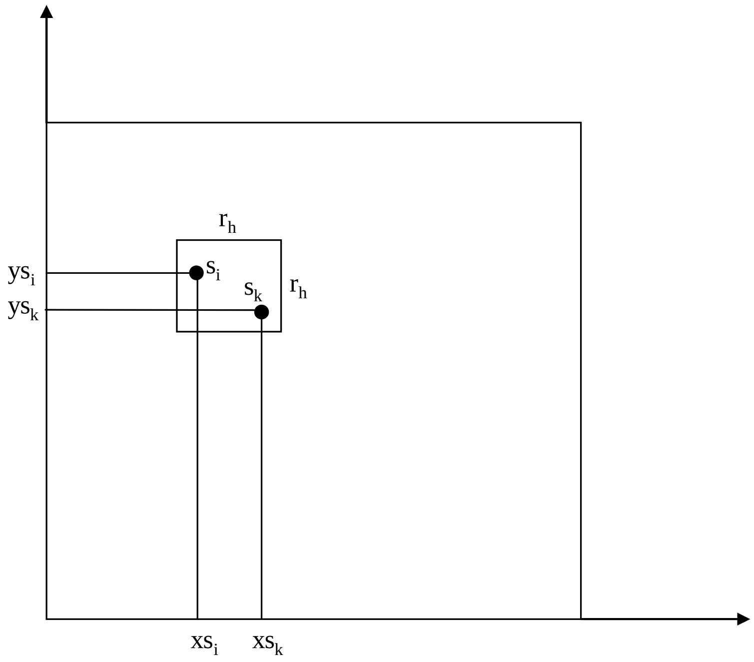


Figure 8 Two sensors s_i and s_k are in the same $r_h \times r_h$ region

Next, we address the issue of how to determine if s_i and s_k are in the same $r_h \times r_h$ region, see Figure 8. Let 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 $r_h \times r_h$ region if and only if $\left\lfloor \frac{xs_i}{r_h} \right\rfloor = \left\lfloor \frac{xs_k}{r_h} \right\rfloor$ and $\left\lfloor \frac{ys_i}{r_h} \right\rfloor = \left\lfloor \frac{ys_k}{r_h} \right\rfloor$. If g_j becomes 1 (line 8), then the procedure returns and skips to the next interval t_{i+1} without requiring s_i to become active at this time.

If the 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 ≥ j) in s_i's region $r_w \times r_w$, where w is the index of the largest sensing component of s_i. A sensor forwards the message only the first time it is received and only if it is located in the same $r_w \times r_w$ region as s_i. Any forwarding node sets up its g variables based on s_i's sensing components, similar to lines 4…7 in the pseudocode.

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

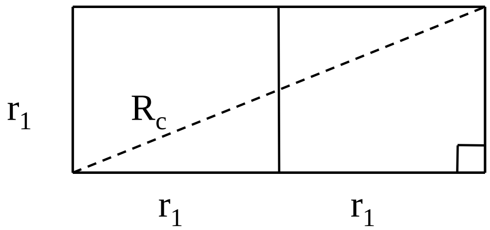


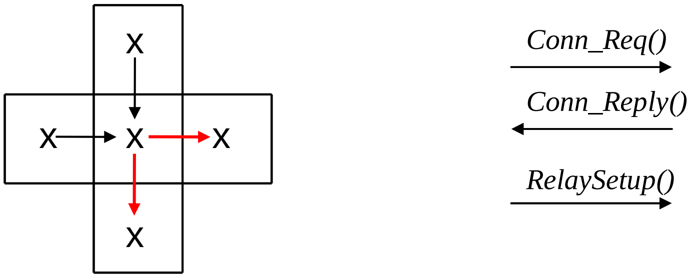
Figure 9 Maximum distance between sensors in neighboring regions

According to Figure 9, 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.

4.2.3 The Case when $R_c < r_1\sqrt{5}$

Connectivity becomes an issue when the communication range $R_c < r_1\sqrt{5}$. Then 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 connectivity purpose, it is sufficient to ensure that each square region is connected to its bottom and right neighbors.



a. Connection of a region with bottom and right regions

b. Connectivity: three-way message exchange mechanism

Figure 10 Connectivity mechanism

After satisfying the *sensing requirement*, there will be at least one active sensor in each $r_1 \times r_1$ square region. In this section 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 region is connected to its bottom and right neighbor regions, see Figure 10a. 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 then responsible to ensure connectivity with bottom and right regions. Let us take a CH s_i. If node s_i heard *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 10b: 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 square connections (bottom and/or right), sending node id, TTL, 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 (e.g. active according to section 4.2.1) in the destination bottom and/or right regions; otherwise 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. If s_j is not an active sensing node then s_j 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 now the case when no reply message was 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, that 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 thesis, 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 pseudocode below.

Algorithm Connectivity (CH sensor s_i)

```
1. if Hello messages received from sensors in bottom and right regions, then return TRUE
2. wait random time $t_i$
3. send Conn_Req(CH $s_i$, grid position of $s_i$, list of needed square connections (bottom and/or right), sending node id, TTL, number of relay hops)
4. 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
5. then for each needed square connection choose the Conn_Reply message with 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 $s_j$)
6. if all needed square connections are satisfies, then return TRUE
7. if $TTL < TTL_{max}$ then increase TTL and go to line 2
8. else return FALSE
```

4.3 Simulation Study

4.3.1 Simulation Settings

In this section, the performance of a distributed algorithm is evaluated. A custom simulator was developed using C++, according to the framework described in section 4.2.1. The distributed algorithm was implemented on a 320m × 320m area. The area was divided into three grids, $r_1 \times r_1$, $r_2 \times r_2$ and $r_3 \times r_3$, where r_1=20m, r_2=40m and r_3=40m. The sensor nodes were equipped with a random combination of three sensing components x_1, x_2 and x_3.

The algorithm was tested with one thousand to ten thousand (1000 – 10000) sensor nodes, increasing in one thousand sensor increments. The communication ranges and sensing energy of each sensor node was varied. This was done to test the influence of both communication range and sensing components on the *network lifetime*. Network lifetime is defined in this thesis as the number of rounds completed while the area is entirely covered by each sensing component x_i and all active sensors are connected.

The initial energy contained by each node, $E_{initial}$ = 50mJ. After each round, the energy for that round is subtracted from the total energy, according to the equation:

$$E_{round} = E_{Tx} + E_{sense} \qquad (3)$$

where:

- E_{Tx} is the energy required to transmit a message over a communication range d. This equation was used as in LEACH [3].

$$E_{Tx} = E_{elec}*k + \varepsilon_{amp}*k*d^2$$

- E_{sense} is the energy needed to sense an event [4].

$$E_{sense} = sc * E_{comp}$$

  sc is the number of sensing components and E_{comp} is the sensing energy consumed by a sensing component.

The energy required to receive a transmission, E_{Rx} [4], was neglected as no specific data collection tree is constructed. It is assumed that if the location of the sink is known then a data collection tree can be computed using the BFS algorithm.

4.3.2 Simulation Results

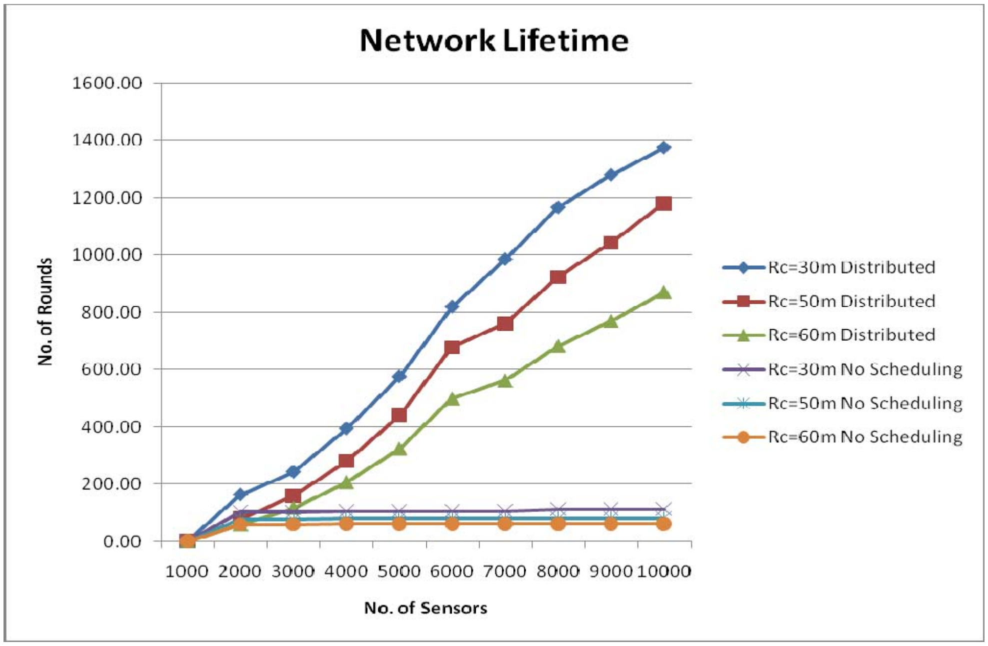


Figure 11 Network lifetime of distributed scheduling compared to no scheduling

The simulation showed for different communication ranges that the distributed algorithm creates a longer network lifetime than composite event detection with no scheduling. As the communication range decreases, there is an increase in the network lifetime because much of the energy consumed by the sensor nodes is used for communication and not sensing.

In the network with no scheduling it is evident that an increase in the number of sensor nodes does not affect the overall network lifetime. This is because the nodes are all active at the same time and so deplete their energy at the same rate regardless of the number of nodes present.

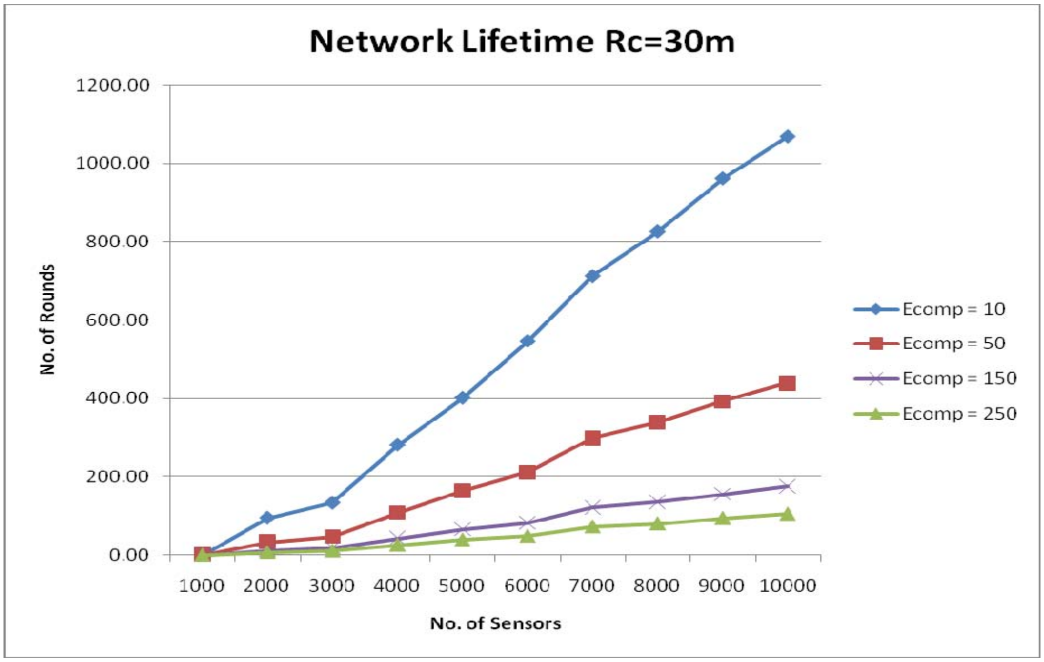


Figure 12 Network lifetime with different sensing energy when R_c=30m

The amount of energy consumed by the sensing component depends on the functions that it is designed to perform. Figure 12 above shows the change in network lifetime when the sensing energy consumed, E_{comp}, is varied and the communication range is constant. In a sensor network, the E_{comp} value can vary depending on the number of sensor readings that are performed per round. The graph shows that there was a significant decrease in the network lifetime as the sensing energy increased.

The schedule was presented to manage the number of active sensors such that there would be a minimal number of sensors active per round. The results in Figure 13 below show that for communication ranges above 44.72m ($R_c < r_1\sqrt{5}$, as described in section

4.2.2) the sensor nodes would require relay nodes to have full connectivity throughout the network. This can be seen in graph lines indicating 10m and 40m communication ranges. The smaller the communication range the larger the number of relay nodes required to connect the network.

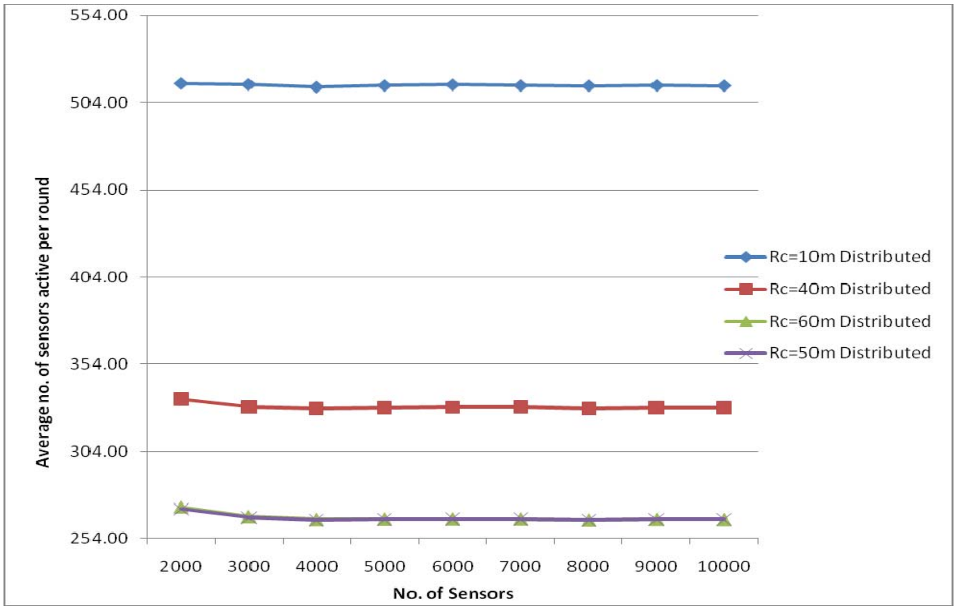


Figure 13 Average number of sensors active per round

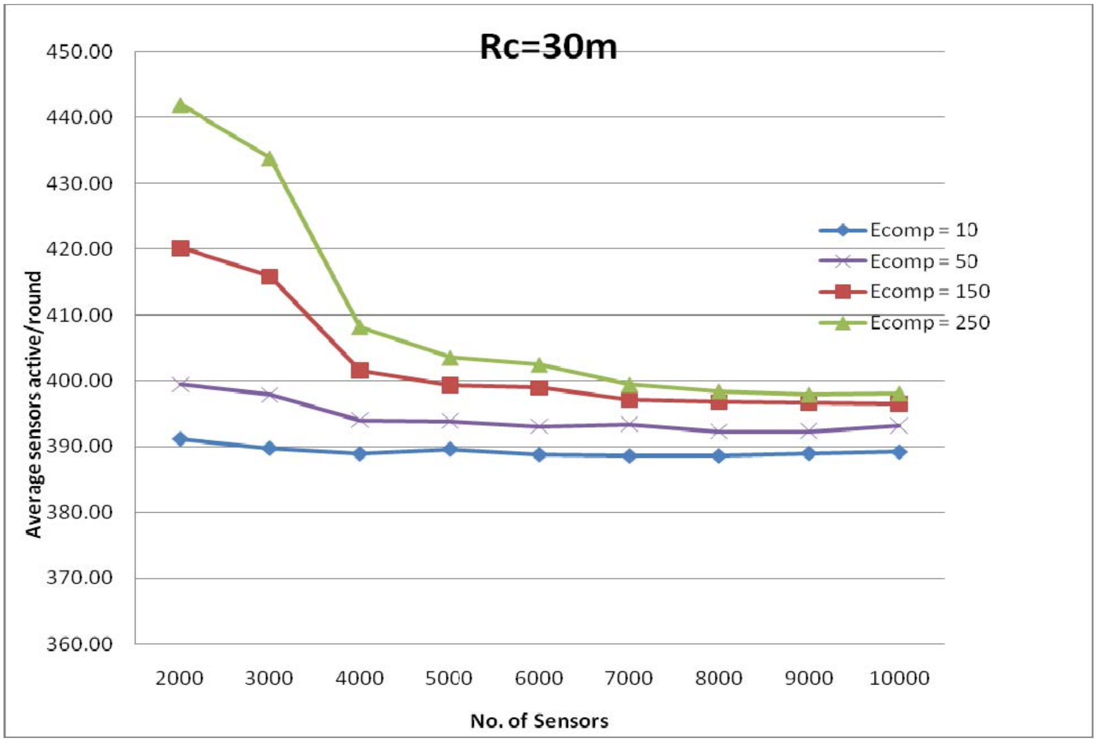


Figure 14 Average number of sensors active per round with different sensing energy

Figure 14 above shows the changes in active sensors per round when the sensing energy is varied. The number of sensor nodes remains fairly constant for values of 4000 and higher where the algorithm performs at its best. When the number of sensors is too low, the algorithm performs much like an unscheduled network due to the low density of sensor nodes. This is especially evident when connectivity is an issue as relay nodes are also required.

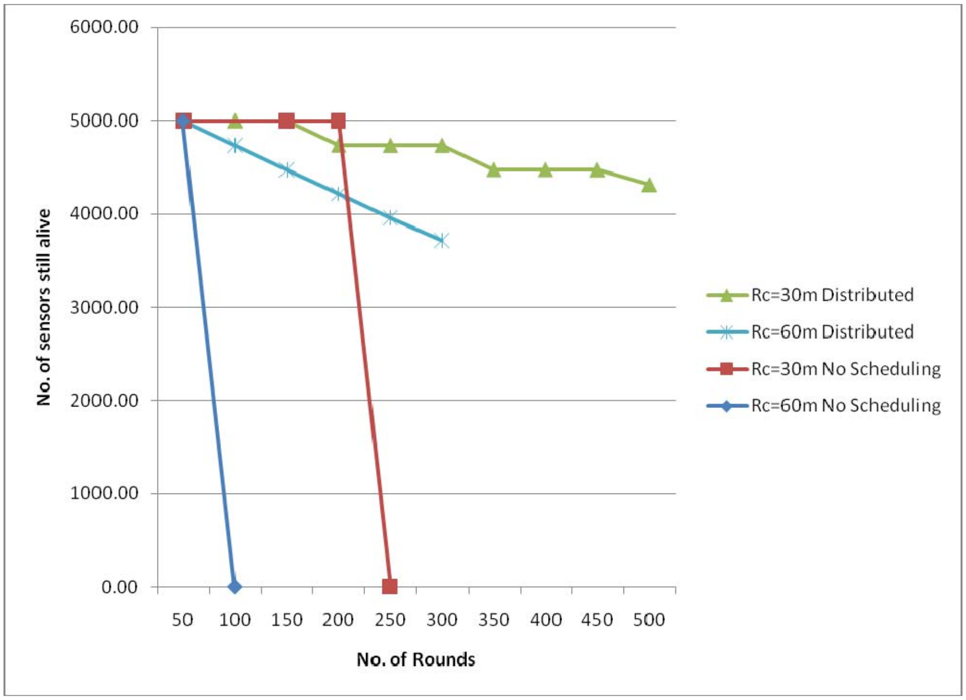


Figure 15 Number of sensors still alive

The number of sensors left alive was also investigated. It was found for the unscheduled algorithm that, as expected, all the sensors were depleted of their energy at the same rate and died at the same time. For the distributed algorithm, Figure 15 above shows that there is a gradual decrease in the number of sensors still alive after a number of rounds. At 30m range, the sensors die at a slower pace than when the range is 60m. This is also implied by the network lifetime graph, Figure 11, which also shows that the 30m network lasts longer than that of 60m.

5 Conclusions

In this thesis, we investigated the SCED problem in WSNs. A distributed algorithm was presented as a possible solution for this problem. The area was set up in a grid format so that it is easier to divide the area based on the sensing capabilities of each sensing component. The algorithm decides which sensor nodes would become active based on the energy remaining and the contribution that it can make to the network. All other nodes are put to sleep in an effort to conserve energy.

The simulation results show that for large areas with increasingly large numbers of sensors the distributed scheduling extended the network lifetime and decreased the number of active sensors per round. This was a marked improvement when compared to a network where there is no scheduling present to organize the nodes. A significant improvement was seen when the communication range remained as small as possible without falling below the connectivity limit and therefore making the use of relay nodes unnecessary.

As part of future work we plan to develop additional scheduling mechanisms for composite event detection, in which we can relax the requirement that each sensor knows its location.

6 References

[1] S. Singh, C. S. Raghavendra, and J. Stepanek, Power Efficient Broadcasting in Mobile Ad Hoc Networks, *Proceedings PIMRC'99*, Vol. 3, pp. 1438 – 1442, 1999.

[2] V. Raghunathan, C. Schurgers, S. Park, M. B. Srivastava, Energy-Aware Wireless Microsensor Networks, *IEEE Signal Processing Magazine*, Vol.19, No.2, pp. 40 – 50, March 2002.

[3] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-Efficient Communication Protocol for Wireless Microsensor Networks, *Hawaii International Conference on System Sciences* (HICSS), Vol. 2, pp. 10, 2000.

[4] C. V. Vu, R. A. Beyah, and Y. Li, Composite Event Detection in Wireless Sensor Networks, *26th IEEE IPCCC*, pp. 264 – 271, 2007.

[5] K. Römer and F. Mattern, Event-based Systems for Detecting Real-World States with Sensor Networks: A Critical Analysis, *Intelligent Sensors, Sensor Networks and Information Processing Conference*, pp. 389 – 395, 2004.

[6] D. Janakrim, A. V. U. P. Kumar, A. M. Reddy, Component Oriented Middleware for Distributed Collaboration Event Detection in Wireless Sensor Networks, *MPAC*, 2005.

[7] C. Huang and Y. Tseng, The Coverage Problem in a Wireless Sensor Network, *Mobile Networks and Applications*, Vol. 10, pp. 519 – 528, 2005.

[8] S. Meguerdichian, F. Koushanfar, M. Potkonjak and M B Srivastava, Coverage Problems in Wireless Ad-hoc Sensor Networks, *IEEE INFOCOM*, Vol. 3, pp. 1380 – 1387, 2001.

[9] J. O'Rourke, Art Gallery Theorems and Algorithms, Oxford University Press, Oxford, 1987.

[10] M. Cardei and J. Wu, Energy-efficient Coverage Problems in Wireless Ad-hoc Sensor Networks, *Computer Communications 29*, pp. 419 – 420, 2006.

[11] S. Slijepcevic and M. Potkonjak, Power Efficient Organization of Wireless Sensor Networks, *Proceedings of IEEE International Conference on Communications 2*, pp. 472 – 476, 2001.

[12] M. Cardei, D. MacCallum, X. Cheng, M. Min, X Jia, D. Li, D.-Z. Du, Wireless Sensor Networks with Energy efficient organization, *Journal of Interconnection Networks 3*, pp. 213 – 229, 2002.

[13] F. Ye, G. Zhong, S. Lu, L. Zhang, Energy Efficient Robust Sensing Coverage in Large Sensor Networks, *Technical Report UCLA*, 2002.

[14] H. Zhang and J.C. Hou, Maintaining Sensing Coverage and Connectivity in Large Sensor Networks, *Technical Report UIUC*, 2003.

[15] J. Wu and H. Li, On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks, *Proceedings of the Third International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pp. 7 – 14, 1999.

[16] F. Dai and J. Wu, Distributed Dominant Pruning in Ad Hoc Wireless Networks, *Proceedings of IEEE International Conference on Communications*, Vol. 1, pp. 353 – 357, 2003.

[17] T. Cormen, C. Leiserson, R. Rivest and C. Stein, Introduction to Algorithms, McGraw-Hill Book Company, North America, 2003.

[18] I. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, Wireless Sensor Networks: A Survey, *Computer Networks 38*, pp. 393 – 422, 2002.

[19] D. Estrin, R. Govindan, J. Heidemann, S. Kumar, Next Century Challenges: Scalable Coordination in Sensor Networks, *ACM MobiCom '99*, pp. 263 – 270, 1999.

[20] J. Stankovic, Wireless Sensor Networks, *ACM Journal of Wireless Networks*, 2006.

[21] L. Yu, N. Wang and X. Meng, Real Time Forest Fire Detection with Wireless Sensor Networks, *International Conference on Wireless Communications, Networking and Mobile Computing*, Vol. 2, pp.1214 – 1217, 2005.

[22] A.Wood, G. Virone, T. Doan, Q Cao, L Selavo, Y. Wu, L. Fang, Z. He, J. Stankovic, AlarmNet, *ACM SenSys*, April 2005.

[23] G. Virone, A.Wood, L Selavo, Q Cao, L. Fang, T. Doan, Z. He, R. Stoleru, S. Lin and J. Stankovic, An Assisted Living Oriented Information System Based on a Residential Wireless Sensor Network, *Proceedings D2H2*, 2006.

[24] D. Estrin, L. Girod, G. Pottie, G. Srivastava, Instrumenting the World with Wireless Sensor Networks, *IEEE Conference on Acoustics, Speech and Signal Processing*, Vol. 4, pp. 2033 – 2036, 2001.

[25] M. Halgamuge, Design Factors for Sustainable Wireless Sensor Networks, *Third International Conference on Information and Automation for Sustainability, ICIAFS07*, pp. 106 – 110, 2007.

[26] M. Abd-El-Barr, M. Youssef, M. Al-Otaibi, Wireless Sensor Networks: Part 1 Topology and Design Issues, *Canadian Conference on Electrical and Computer Engineering*, pp. 1165 – 1168, 2005.

[27] G. Hoblos, M. Starosweicki, A. Altouche, Optimal Design of Fault Tolerant Sensor Networks, *IEEE International Conference on Control Applications*, pp. 467 – 472, 2000.

[28] A. Porret, T. Melly, C. Enz, E. Vittoz, A Low-Power Low-Voltage Transceiver Architecture Suitable for Wireless Distributed Sensors Network, *IEEE International Symposium on Circuits and Systems*, Vol. 1, pp. 56 – 59, 2000.

[29] G. Pottie, W. Kaiser, Wireless Integrated Network Sensors, *Communications of the ACM*, Vol. 43, Issue 5, pp. 51 – 58, 2000.