ACTIVE NODE DETERMINATION FOR CORRELATED DATA GATHERING IN WIRELESS SENSOR NETWORKS

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Efe Karasabun

July, 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. İbrahim Körpeoğlu(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Cevdet Aykanat(Co-supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Uğur Güdükbay

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Ali Aydın Selçuk

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Ezhan Karaşan

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet B. Baray
Director of the Institute

ABSTRACT

ACTIVE NODE DETERMINATION FOR CORRELATED DATA GATHERING IN WIRELESS SENSOR NETWORKS

Efe Karasabun
M.S. in Computer Engineering
Supervisors: Asst. Prof. Dr. İbrahim Körpeoğlu and
Prof. Dr. Cevdet Aykanat
July, 2009

In wireless sensor network applications where data gathered by different sensor nodes is correlated, not all sensor nodes need to be active for the wireless sensor network to be functional. However, the sensor nodes that are selected as active should form a connected wireless network in order to transmit the collected correlated data to the data gathering node. The problem of determining a set of active sensor nodes in a correlated data environment for a fully operational wireless sensor network can be formulated as an instance of the connected correlation-dominating set problem. In this work, our contribution is twofold; we propose an effective and runtime efficient iterative improvement heuristic to solve the active sensor node determination problem and a benefit function that aims to minimize the number of active sensor nodes while maximizing the residual energy levels of the selected active sensor nodes. Extensive simulations we performed show that the proposed approach can achieve a good performance in terms of both network lifetime and runtime efficiency.

Keywords: wireless sensor networks, correlated data gathering, active sensor node determination.

ÖZET

KABLOSUZ SENSÖR AĞLARINDA İLİNTİLİ VERİ TOPLAMA AMAÇLI AKTİF SENSÖR BELİRLENMESİ

Efe Karasabun
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticileri: Asst. Prof. Dr. İbrahim Körpeoğlu ve
Prof. Dr. Cevdet Aykanat
Ağustos, 2009

Bazı kablosuz sensör ağları uygulamalarında sensör aygıtlarının algıladıkları veriler ilintilidir. Bu gibi kablosuz sensör ağı uygulamalarının tamamen çalışır durumda olması için bütün sensör aygıtlarının aktif (çalışıyor durumda) olmalarına gerek yoktur. Buna karşılık, aktif olarak seçilen sensör aygıtlarının kendi aralarıda haberleşmelerini sağlayacak kablosuz bir ağ kurarak topladıkları ilintili verileri sorumlu merkeze göndermeleri gerekmektedir. Sensörler arasında ilintili veri bulunan kablosuz sensör ağları uygulamalarında hangi sensör aygıtlarının aktif durumda olacağının belirlenmesi, haberleşebilen ilinti-bazlı küme (connected correlation-dominating set) problemi olarak ifade edilebilir. Bu tez çalışmasının katkısı çift yönlüdür: İlk olarak aktif sensör aygıtlarının belirlenebilmesi için etkin ve hızlı çalışan tekrarlamalı iyileştirme gerçekleştiren buluşşal bir algoritma (iterative improvement heuristic) önerilmektedir. İkinci olarak ise aktif sensör aygıtı kümesine seçilen sensör aygıtı sayısı azaltılırken, bu kümeye seçilen sensör aygıtlarının yüksek enerjiye sahip olabilmelerine imkan veren bir yarar fonksiyonu önerilmektedir. Detaylı simülasyonlarla ileri sürdüğümüz bu yaklaşımın hem kablosuz sensör ağının işleme süresi bakımından, hem de algoritma çalışma zamanı bakımından iyi sonuçlar ortaya koyduğu görülmektedir.

Anahtar sözcükler: kablosuz sensör ağları, ilintili veri toplama, aktif sensör belirleme.

Acknowledgement

I would like to express my gratitude to my advisors Prof. Dr. Cevdet Aykanat and Asst. Prof. Dr. İbrahim Körpeoğlu for their expert guidance and valuable contributions to this thesis.

I would like to thank the jury members Assoc. Prof. Dr. Uğur Güdükbay, Asst. Prof. Dr. Ali Aydın Selçuk and Assoc. Prof. Dr. Ezhan Karaşan for reviewing and evaluating this thesis.

Finally, I would like to express my thanks and gratefulness to my mother and my father for supporting me throughout my life.

Contents

List of Figures

Chapter 1

Introduction

Wireless sensor networks (WSNs) are composed of a large number of spatially distributed sensor nodes which are limited in power. These sensor nodes are equipped with three main components to cooperatively collect information about a monitored region. These three main components of a sensor node are a processing unit with limited capability, environment sensor(s) and a short-range wireless transceiver. By the use of these components, sensor nodes can form a multi-hop wireless network and transmit the sensed data about the monitored environment to a data gathering node. Sensors are able to obtain various information about the monitored environment such as temperature, humidity, pressure, sound, motion, etc. Some WSN applications include environment and habitat monitoring, healthcare assisance, home automation, industrial process monitoring and control, and battlefield and border surveillance.

Limited energy available in sensor nodes makes network lifetime an important issue in WSN applications. To extend the network lifetime, energy efficient wireless sensor network protocols and algorithms have been devised in the literature. Node clustering, in-network data processing, data fusion and network coding are some of the measures taken to reduce the amount of data that is processed, sensed or transmitted. Minimization of energy spent in processing, sensing and transmitting of data allows sensor nodes to save energy. Such energy savings help to extend the lifetime of WSN applications.

In some WSN applications, not all sensor nodes are required to be active (turned on, thus spending energy) in order for the WSN application to be fully functional. In these types of applications, exploiting the inherent data correlations among the sensor devices may extensively help to prolong the network lifetime. The data correlations between the sensor devices may exist due to the characteristics of a sensor region and sensor node deployment such as the proximity of the sensor nodes. The data correlations among sensor nodes can be modelled as a set of two-tuples, where each tuple contains a source set of nodes which infers a sensor node. When a source set is selected into the active sensor node set, the sensor node inferred by that source set may stay inactive. In these types of WSN applications, since the data of some sensor nodes can be inferred from the data of some other nodes, it is crucial to determine the set of active sensor nodes that can be sufficient to infer the data of inactive sensor nodes. Only the active sensor nodes need to sense, process and transmit data. The inactive nodes will be turned off and therefore they will not spend any energy.

In this work, we aim to find effective and runtime efficient centralized active sensor node selection heuristics for correlated data gathering in WSNs to prolong the sensor network lifetime. For this purpose, we model the active node determination problem as an instance of the connected correlation dominating-set problem [11]. In connected correlation dominating-set problem, given a network and correlation information about which nodes infers which other nodes, we are interested in finding a set of (dominating) nodes that can infer the (correlated) data of the rest of the nodes. The authors of [11] propose a sophisticated but time-consuming constructive L-hop centralized heuristic. The objective of the L-hop centralized heuristic is to construct a connected correlation-dominating set with minimum number of sensor nodes by the use of a benefit function that they define. Our contribution in this work is twofold: We propose iterative active sensor node determination (IAND) heuristic, which is both effective and runtime efficient. The IAND heuristic is composed of a greedy constructive heuristic and an iterative improvement heuristic to find an effective and runtime efficient correlation-dominating set for WSNs. Furthermore, we define an energy-aware

benefit function that is used by both the greedy constructive heuristic and the iterative improvement heuristic while constructing and then improving the quality of the correlation-dominating set.

The purpose of the greedy constructive heuristic is to construct a correlation-dominating set with a given large correlation data set as the input in a runtime efficient manner. The iterative improvement heuristic is executed after the greedy constructive heuristic to improve the energy quality of the active sensor nodes selected by the greedy constructive heuristic. The basic operation in the iterative improvement heuristic is the swap of an already selected sensor node in the current correlation-dominating set with a set of unselected source sets. The objective in a swap operation is to find a set of unselected source sets which achieves the maximum amount of improvement in the energy quality of the WSN under the constraint of preserving the correlation-dominating set property. We formulate the problem of finding a good set of unselected source set for swapping a given sensor node as a subproblem of the original correlation-dominating set problem. The iterative improvement heuristic uses the 0-hop centralized heuristic of [11] to construct a solution to this swap subproblem. Although the 0-hop centralized heuristic is slow with large correlation data set as the input, it generates a better selection of active sensor nodes, in terms of sensor network lifetime, compared to that of the greedy constructive heuristic with small-scale correlation data as the input in the swap subproblem.

A correlation-dominating set constructed by the IAND heuristic does not necessarily have to result in a connected wireless network. To achieve wireless connectivity among active sensor nodes, we use the minimum Steiner tree construction heuristic [17]. The objective of the minimum Steiner tree is to construct a connected wireless network by adding the minimum number of additional nodes into the active sensor nodes set. Thus, the minimum Steiner tree forms the connected correlation-dominating set from the correlation-dominating set constructed by the IAND heuristic.

We performed extensive simulations to observe the performance of the IAND heuristics in Section 5. Furthermore, we compared our results with a recent

and state-of-the-art solution to the active sensor node determination problem proposed in [11]. We evaluate the heuristics in terms of sensor network lifetime and runtime efficiency and show that we are able to achieve considerable better results than the existing solution to the problem.

The rest of the paper is organized as follows: In Section 2, we give background information about WSNs, in Section 3, we discuss the related work and in Section 4.1 we give a formal definition of the problem. In Section 4, we describe our solution approach and detail our (IAND) heuristic. In Section 5, we provide the results of our simulation experiments done to evaluate the performance of our IAND approach. Finally, in Section 6 we conclude our work.

Chapter 2

Background Information

In this chapter, first, wireless sensor networks (WSNs) are introduced. Second, a clasification of WSN applications and the challanges in developing WSN applications are explained. Third, construction of Steiner trees is explained. The information in this section is compiled from [17] [14] [13].

2.1 Wireless Sensor Networks

Recent advancements in the area of embedded systems and wireless networking has made it possible for the emergence of a new research and application area referred to as wireless sensor networks (WSNs). The purpose of WSNs is to cooperatively sense and gather various information about the monitored region to a centralized processing center refered to as the sink or data gathering node. For that reason, WSNs are composed of a large number of spatially distributed sensor nodes (devices). The sensor nodes that constitute a WSN have very unique characteristics and capabilities. Firstly, sensor nodes are limited in power and since a WSN is composed of a large number of sensor nodes that are usually distributed in a large geographical area, it is not possible to recharge or replace sensor nodes whose power is depleted. Secondly, these sensor nodes are equipped with three

main components to cooperatively sense and gather information about the monitored region. The three main components of the sensor nodes are the processing unit with limited capability, environment sensors and short-range wireless transmitters. By the use of its components, sensor nodes form a wireless network and transmit the sensed data about the monitored environment to the data gathering node. Figure 2.1 shows an example to a MicaZ sensor node that is used in WSN applications, Figure 2.2 show the basic architecture of a sensor node and Figure 2.3 shows an example to a small WSN that is composed of multiple sensor nodes.

Figure 2.1: MicaZ sensor node

2.2 Classification of WSN Applications

WSN applications can be categorized based on the application objectives, traffic characteristics and data delivery requirements. Most of the current WSN applications fall into one of the following broad classes.

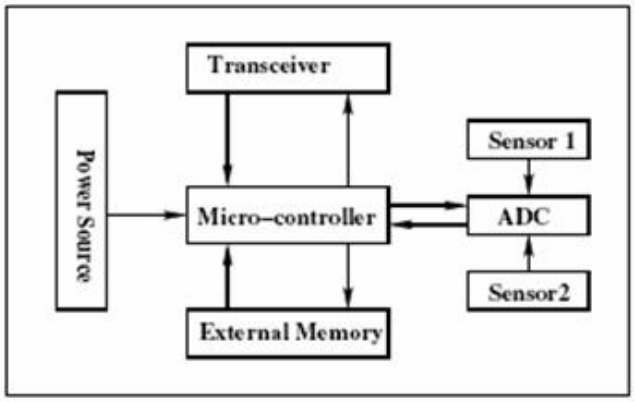


Figure 2.2: Basic architecture of a sensor node

2.2.1 Event Detection and Reporting

Military WSN applications such as intruder detection, and other civilian WSN applications such as forest fire detection and detecting anomalities in a manufacturing process are examples to WSN applications in this category. These WSN applications operate only once the event is detected. They generate report(s) about the detected event and send it to the data gathering node as soon as possible. Therefore, it is very important to organize the collaboration of the sensor nodes in such applications to generate more accurate report(s) about the detected event. This collaboration among sensor nodes also helps to reduce the number of false alarms generated in the WSN. Most of the time, the sensor nodes in these WSN application stay inactive. Therefore, the wireless network connectivity of the sensor nodes in these types of WSN applications should be organized in a way to send the generated report(s) as soon as possible to the data gathering node as most of the time the generated report(s) are time critical.

2.2.2 Data Gathering and Periodic Reporting

Applications in this category are monitoring the environmental conditions affecting crops or livestock, monitoring temperature, humidity and lighting in office

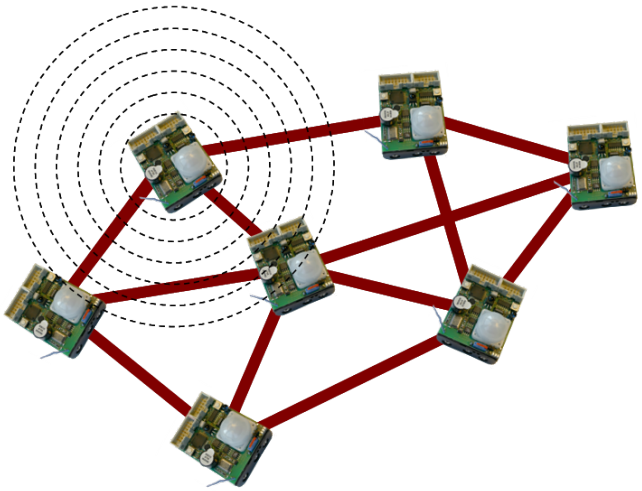


Figure 2.3: An example to a WSN where the links among sensor nodes indicate the wireless connectivity and the dashed region represent the sensing range of the sensor node

buildings, etc... In these types of WSN applications, periodic information about the monitored region is sent to the data gathering node. Usually the data gathering node is interested in the distribution of the gathered data as these applications are not time critical. Therefore data aggregation schemes such as node clustering and in-network processing can be applied in such scenarios. These data aggregation schemes will reduce the amount of data that is to be sent to the data gathering node. Since the amount data that is sent is reduced, this will lead to a longer network lifetime and smaller delays in the network.

2.2.3 Sink-initiated Querying

Applications in this category are similar to the applications in data gathering and periodic reporting section. However the difference is that rather than generation of periodic reports about the monitored region, the data gathering node queries the WSN or a subsection of the WSN according to the requirements of the WSN application. In these types of applications, the necessary data communication paths and routing mechanisms should be established between the data gathering

node and the sensor nodes in both directions.

2.2.4 Track-based Applications

A WSNs application that is based on tracking is border surveillance where it is important to accurately track the movements of a suspicious objects. Similarly, environmental applications include tracking the movements and patterns of insects, birds or small animals. Furthermore, transportation systems are often interested in wide-area tracking of vehicles. WSN applications for tracking combine some characteristics of the above three WSN application categories. For example, once the target is detected and the data gathering node is notified and it may need to query the WSN to receive location estimates of the tracked objects.

2.3 Challanges for WSNs

In this section the challanges that are needed to be solved while developing WSN applications is explained.

2.3.1 Characteristic Requirements

- **Quality of Service -** Quality of service requirements that are used in traditional computer networks such as bounded delay or minimum bandwidth do not apply to WSN applications. WSNs have their own characteristics such as being delay tolerant and having small available bandwidth. Therefore when applying QoS to WSN applications appropriate QoS metrics should be identified and used.
- **Fault Tolerance -** Sensor nodes in WSNs cannot be replaced when their energies are depleted. Therefore when sensor nodes die due to depleted energy or other environment factors, the WSN should be able to continue operating successfully. For this reason, deploying redundant nodes should

be done in WSN applications. Furthermore, the necessary mechanisms should be developed for the WSN to operate with these redundant nodes.

- **Network Lifetime -** Network lifetime is a very important issue in WSN applications. The network lifetime of the WSN determines the amount of time the application will be able to operate successfully. Therefore necessary mechanisms to perform energy savings must be considered. The required network lifetime for a WSN application depends on the requirements of that WSN application, however, longer the WSN operates the better it is.

- **Scalability -** Scalability is another import issue in WSNs. It is important for the WSN applications to support more nodes to cover larger geographical areas. Therefore, WSN applications should be designed with considering the scalability requirements of that WSN application.

- **Density -** Some WSN applications might require a very dense deployment of sensor nodes in the monitored region. The developed WSN application must be able to support operations, such as building a communication backbone, to operate successfully in such environments. Furthermore the sensor node density may also be heterogenous. Therefore WSNs should be designed considering the density requirements.

- **Programmability -** Sensor nodes need to process information and also be able to react flexibly on changes in their tasks. Therefore, sensor nodes should be programmable and should support updating the software they run when necessary.

- **Maintainability -** Both the WSN environment and the WSN itself may change due to depleted batteries, failing nodes and new tasks. The WSN should be able to monitor its status and adapt to the new conditions. The WSN should also be able to change operational parameters or choose different trade-offs. Therefore the WSN has to maintain itself.

2.3.2 Required Mechanisms

- **Multihop Wireless Communication -** Wireless communication over a long distance is a very energy consuming operation for sensor nodes with limited power. However, communication over small distances through the use of other sensor nodes is a relatively less energy consuming operation. Using multi-hop communication, energy that is consumed by the transmitting the data is divided among the forwarder sensor nodes.

- **Energy Efficient Operation -** Supporting energy efficient operations is an important technique for having long network lifetime in WSNs. Therefore, any operation that is being performed on the WSN, it should be performed in the most energy efficient way possible, according to the requirements of the WSN application.

- **Auto Configuration -** Rather than using fixed operational parameters, WSN applications should be able to configure their operational parameters according to the current state of the WSN application. For example, sensor nodes should be able to determine their geographical locations by communicating with other nodes in the WSN or they should be able to automatically synchronize their internal clocks with by communicating with each other.

- **Collaboration and in-network processing -** In some WSN applications, one sensor node might not be able to fully detect an event. For this purpose, collaboration of sensor nodes is an important way to better monitor the sensor region. Furthermore, in come cases besides collaboration to fully sense the necessary data, in-network processing can be applied to further analyze and extract more important information from the sensed data. Therefore, these techniques are very important for WSN applications to provide better results to data gathering node. Collaboration and in-network processing also may help to reduce the total amount of data that is sent to the data gathering node. Therefore, in that sense, they also help to achieve a longer network lifetime.

- **Data centric -** In traditional communication networks, data is transfered

between two specific devices, each equipped with (at least) one network address. The operation of such networks is based on an address-centric approach. In WSN applications, where nodes are deployed redundantly to protect against node failures due environmental factors or energy depletion or to compensate for the insufficiency of one sensor node's actual sensing equipment, the identity of the particular sensor node supplying data becomes unimportant. In that sense, the important issue is being able to correctly gather the required data. It doesn't matter which set of nodes provide the data. Therefore using a data centric approach may be more suitable for some WSN applications.

- **Locality -** Locality is very important especially for scalable WSNs. As a WSN becomes large, maintaining global information about the whole WSN becomes an infeasible task. Therefore, sensor nodes should communicate with close sensor nodes to achieve the given tasks.

- **Exploit trade-offs -** WSNs will have to exploit various inherent trade-offs between mutually contradictory goals, both during system/protocol design and at runtime according to the specifications of the WSN application. For example, the trade-off between having higher energy expenditure allows higher result accuracy. Likewise the trade of between network lifetime against the lifetime of individual nodes. According to the specifications of the WSN application, necessary trade-offs should be considered and appropriate action should be taken.

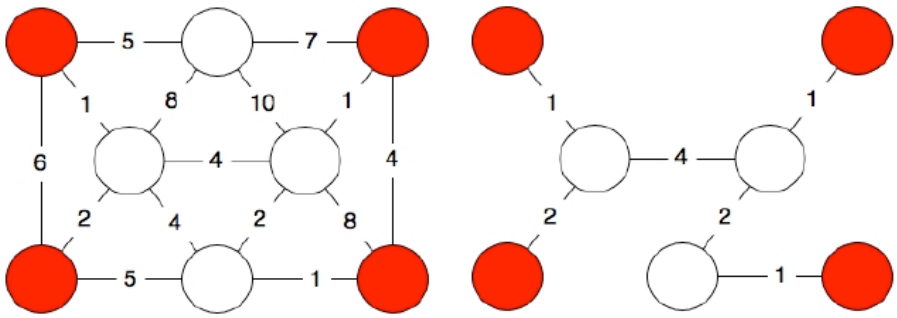
2.4 Steiner Trees

Consider a graph $G = (V, E)$ where each edge is associated with a weight, and $S \subseteq V$. A Steiner Tree T is a subgraph of G with minimal-weight that connects all the vertices of S. To construct T, additional vertices, referred to as Steiner vertices, that are in $V - S$ can be used. Consider the graph in Figure 2.11(a). Red vertices constitute the set of vertices that need to be connected. A minimal steiner construction of the given graph is constructed in Figure 2.11(b).

Finding a Minimal Steiner tree is an NP-Complete problem [8]. Therefore, heuristics [19] [17] have been deviced to solve this problem. Below we provide the algorithm of 2-approximation Steiner tree construction [17].

1. Construct the complete distance graph G_1 in which the distance from each vertex to every other vertex is computed.
2. Find a minimum spanning tree G_2 of G_1.
3. Construct a subgraph of G_3 of G by replacing each edge in G_2 with corresponding shortest path in G.
4. Find a minimum spanning tree G_4 of G_3.
5. Construct G_5 by deleting edges in G_4 so that no leaves in G_5 are Steiner vertices.

It should be noted here that the complete distance graph can be implemented using Dijsktra's shortest path algorithm and the minimum spanning tree can be implemented using Prim's minimum spanning tree algorithm. Figures 2.12–2.18 gives an example minimum steiner tree construct using the algorithm outlined above.



(a) Complete graph G in which red vertices S are needed to be connected

(b) Minimal Steiner tree of S with additional vertices

Figure 2.4: Steiner Tree Construction

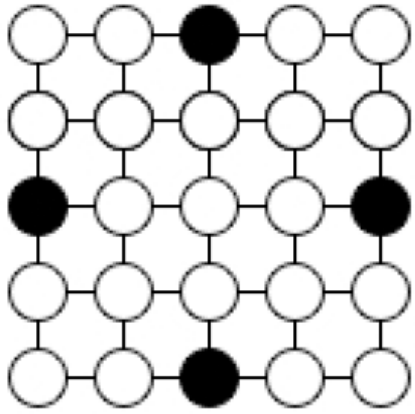


Figure 2.5: Given graph G in which black vertices represents vertices to be connected

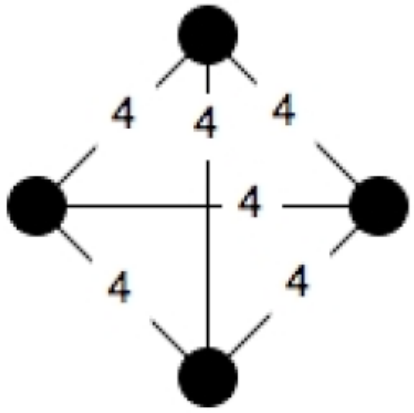


Figure 2.6: Complete distance graph G_1 of G

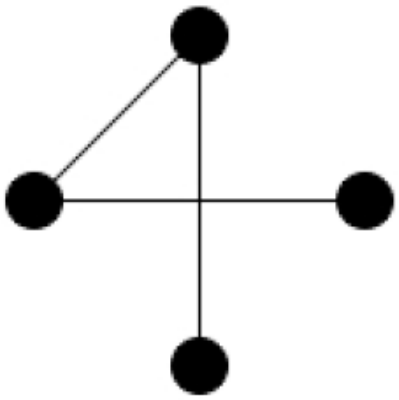


Figure 2.7: Minimum spanning tree G_2 of G_1

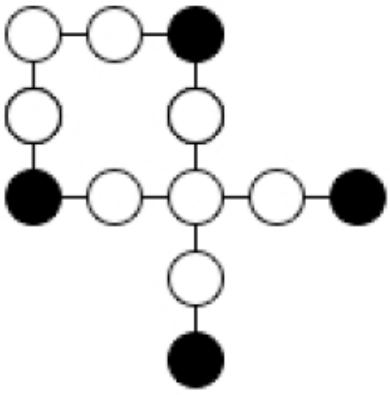


Figure 2.8: Computation of G_3 in which each edge in G_2 is replace with shortest path in G

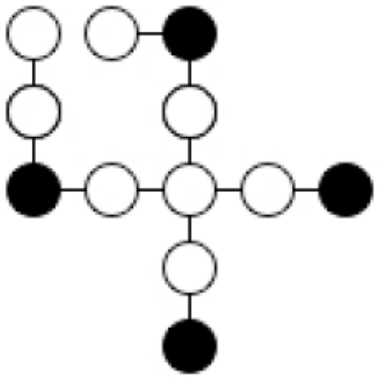


Figure 2.9: Minimum spanning tree G_4 of G_3

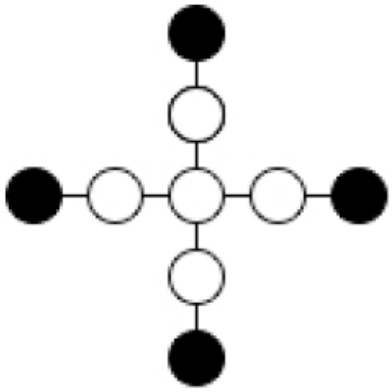


Figure 2.10: Removing of leaf Steiner vertices from G_4

Chapter 3

Related Work

In WSNs having data correlations between sensor nodes, reducing the total number of bits transmitted to the data gathering node is a common approach to avoid spending redundant energy and prolonging the network lifetime. Some approaches to achieve a longer network lifetime in a correlated data environment include using clusters for data aggregation, constructing data aggregation trees, utilizing network coding and constructing correlation-dominating sets.

Clustering in WSNs is a rather well studied topic [1]. On one hand, there are generic clustering algorithms for WSNs such as HEED [24] and LEACH [12] that do not consider data correlations between sensor nodes. On the other hand, [15] studies the effect of partially correlated data on the performance of clustering algorithms. It uses random geometry methodologies [20] to analyze the energy consumption for forwarding data in a multi-hop sensor network. Furthermore the authors combine the result they obtain with rate distortion theory [4]. This way the authors provide a mathematical analysis framework to study the energy consumption and network lifetime when there are arbitrary amount of data correlations between sensor nodes. The analysis framework allows to determine the optimal tuning of the cluster-head selection probability to balance the trade-off between energy consumption and network lifetime in clustering algorithms for WSNs.

To reduce the number of transmissions performed in the network, [23] devises the Clustered Aggregation CAG mechanism which provides approximate results to aggregate queries using the spatial data correlations among sensor nodes. CAG selects a set of cluster-heads, which correspond to a correlation-dominating set, using a simple localized scheme during the query propagation phase. The main pitfall of CAG is that it uses a simple notion of correlation, where the edges of the forwarding tree, constitute the correlations for the selection of cluster-heads and connecting sensor nodes.

A recent work on the subject, GRASS [2], provides exact and heuristic approaches to find a minimum number of aggregation points while routing data to the data gathering node such that the network lifetime is maximized. In GRASS, correlations refer to sensor nodes' readings which overlap statistically as they monitor the same event. These overlappings are used in GRASS to represent the relations among the gathered data. GRASS solves the aggregator selection and routing problems jointly at the data gathering node and then sends the results to the sensor nodes. This way, an optimal solution that is obtained by the data gathering node will result in an optimal routing and aggregation strategy.

Constructing data aggregation trees [9] [7] [16] is another approach to reduce the amount of data transmitted by the sensor nodes and prolong the network lifetime. Authors of [9] propose methods to construct efficient data aggregation trees which are rooted at the data gathering node. Data is aggregated at the intermediate nodes of the data aggregation tree. The authors of [7] propose a randomized tree construction algorithm that achieves a constant factor approximation of the optimal tree for grid network topologies. In both works, the correlations are specific to aggregation, where multiple data values can be compressed into a data value of defined size. The correlation structure that we consider is more general in the sense that the data of the given set of sensor nodes can be compressed depending on the correlation structure available in the network. Authors of [16] devise a randomized approximation algorithm, namely the minimum fusion Steiner tree (MFST), which takes into account not only the data transmission cost but also the data fusion cost.

Utilizing network coding to efficiently gather correlated data has been investigated by [22] [5] [3]. The authors of [22] propose two coding schemes: foreign-coding and self-coding. For these coding techniques, they devise algorithms to construct optimal (minimum weighted number of bit transmissions) and near-optimal data-gathering trees. [5] proposes a method to reduce the number of bits transmitted where the data gathering node is informed about the data correlations between sensor nodes. The data correlations that are realized by the data gathering node are then used to inform the sensor nodes about the number of bits they should use for encoding their sensed data. But this approach assumes a star topology and does not aim to reduce the number of bits transmitted in the network. The authors of [3] propose two approaches to optimize the transmission structure and the rate allocation determination at the sensor nodes. The first approach allows nodes to use joint coding of correlated data without explicit communication where routing and coding are separated. This results in complex data coding and also global network knowledge is needed for an optimal solution. The second approach allows nodes to exploit the data correlation only by receiving explicit side information from other nodes. This way, the correlation structure is exploited through communication and joint aggregate coding/decoding locally at each node. This results is easy data coding and relies only on locally available data as side information. But in this approach optimizing the routing structure becomes complex.

A very recent solution to the connected correlation-dominating set problem in the context of WSNs is given by [11]. The authors propose a centralized approximation algorithm called the L-hop centralized heuristic. The objective of the L-hop centralized heuristic is to find a correlation-dominating set with minimum number of nodes. The L-hop centralized heuristic is composed of two phases. The first phase constructs a correlation-dominating set and the second phase runs a Steiner tree approximation algorithm [17] to connect the correlation-dominating set constructed in the first phase. The complexity of the L-hop centralized heuristic is $O(nm^2g^L)$, where n is the number of sensor nodes in the network, m is the number of correlations, g is the maximum degree of a sensor node in the intersection graph of source sensor nodes and L is the hop count used

in the heuristic.

There are two main pitfalls of the L-hop centralized heuristic algorithm. The first pitfall is its high computational complexity. In a dense WSN, the execution time of the algorithm becomes unexpectedly high. The authors of [11] suggest that best results that are closest to the optimum solution set are obtained by taking the L value as 1. However our simulation results in Section 5.2.1 report that chosing the L value as 1 as opposed to 0, only performs a small increase in the network lifetime while having a dramatically low runtime performance. The second pitfall is the limited energy awareness of the L-hop centralized heuristic. The heuristic tries to increase the sensor network lifetime by only selecting the minimum number of sensor nodes. However, it does not consider the residual energy levels of the sensor nodes while constructing the correlation-dominating set. In this work, we develop an iterative improvement heuristic as a solution to the first pitfall by achieving an effective and runtime efficient correlation-dominating set and we devise an energy-aware benefit function as a solution to the second pitfall.

Chapter 4

Iterative Active Sensor Node Determination

4.1 Problem Definition

We represent the WSN as a two-tuple $\mathcal{W} = (\mathcal{N}, \mathcal{C})$. Here, $\mathcal{N}$ represents the set of sensor nodes and $\mathcal{C}$ represents the set of correlations among sensor nodes. In $\mathcal{C}$, each correlation is represented as two-tuple $C = (S, s)$, where source set S contains the source sensor nodes and s is the inferred node. The correlation $C = (S, s)$ means that when source sensor nodes in set S are active nodes in the WSN, sensor node s may stay inactive. This would result in energy saving in node s as it will not need to process, sense or transmit any data.

Let $Nodes(S)$ denote the set of sensor nodes constituting the source set S. We extend the $Nodes(.)$ operator to denote the sensor nodes that constitute a set $\tilde{S}$ of source sets, i.e.,

$$Nodes(\tilde{S}) = \bigcup_{S \in \tilde{S}} Nodes(S). \tag{4.1}$$

Let $Infer(S)$ denote the set of sensor nodes that are inferred by the source set

S, i.e.,

$$Infer(S) = \{s : (S, s) \in \mathcal{C}\}. \tag{4.2}$$

We extend the $Infer(.)$ operator to denote the set of nodes inferred by a set $\tilde{S}$ of source sets, i.e.,

$$Infer(\tilde{S}) = \bigcup_{S \in \tilde{S}} Infer(S). \tag{4.3}$$

Let $SrcSet(s)$ denote the set of source sets that contain node s, i.e.,

$$SrcSet(s) = \{S : (S, s) \in \mathcal{C}\}. \tag{4.4}$$

It should be noted that the correlations are not transitive. That is, $Infer(S_1) = S_2$ and $Infer(S_2) = S_3$ does not imply $Infer(S_1) = S_3$.

The problem of selecting the minimum number of sensor nodes while keeping the WSN fully operational can be formulated as an instance of the connected correlation-dominating set problem [11].

For a given sensor network $\mathcal{W} = (\mathcal{N}, \mathcal{C})$, a set $\mathcal{M}$ of source sets is called a connected correlation-dominating set if the following two conditions hold:

1. For each sensor node $s \notin Nodes(\mathcal{M})$, there is a source set $S \subseteq \mathcal{M}$ such that (S, s) is a correlation in $\mathcal{C}$.
2. The communication subnetwork induced by $Nodes(\mathcal{M})$ is connected, and $Nodes(\mathcal{M})$ contains the data-gathering node.

Here, $Nodes(\mathcal{M})$ denotes the set of sensor nodes that form the connected correlation-dominating set, i.e.,

$$Nodes(\mathcal{M}) = \bigcup_{S \in \mathcal{M}} Nodes(S). \tag{4.5}$$

The connected correlation-dominating set problem is NP-hard as the less general minimum dominating set problem is well known to be NP-hard [10]. Therefore, we should use heuristics for solving the problem.

4.2 Iterative Active Sensor Node Determination (IAND) Heuristic

In order to effectively and efficiently solve the connected correlation-dominating set problem, we devise a fast energy-aware greedy constructive heuristic which is followed by an iterative improvement heuristic. The proposed approach is referred to here as the iterative active node determination (IAND) heuristic. Both the greedy constructive heuristic and iterative improvement heuristic use an energy-aware benefit function for the determination of which nodes to keep active in the WSN.

4.2.1 Energy Aware Benefit Function

The benefit function $B(S, \mathcal{M})$ used by [11] determines the number of newly inferred nodes per new source node added to set $Nodes(\mathcal{M})$. Therefore the benefit function tries to select the highest number of newly inferred nodes while keeping the number of newly added source nodes to $Nodes(\mathcal{M})$ the smallest. This way set $Nodes(\mathcal{M})$ is constructed by selecting the minimum number of nodes, while inferring the maximum number of nodes. The benefit function $B(S, \mathcal{M})$ is as follows;

$$B(S,\mathcal{M}) = \frac{\text{Number of newly inferred nodes by } S}{\text{Number of new source nodes added to } Nodes(\mathcal{M})}$$
$$= \frac{|Infer(S) - Infer(\mathcal{M})|}{|Nodes(S) - Nodes(\mathcal{M})|} . \quad (4.6)$$

Rather than defining a totally different benefit function, we extend the benefit function in Equation (5) by adding energy awareness. For this purpose, we introduce an energy awareness function $E(S,\mathcal{M})$;

$$E(S,\mathcal{M}) = \frac{\text{Energy average of new source nodes added to } Nodes(\mathcal{M})}{\text{Energy average of newly inferred nodes by } S}$$
$$= \frac{E_{avg}(Nodes(S) - Nodes(\mathcal{M}))}{E_{avg}(Infer(S) - Infer(\mathcal{M}))} . \quad (4.7)$$

We obtain the new energy aware benefit function by combining $B(S,\mathcal{M})$ and $E(S,\mathcal{M})$, where the primary benefit value is considered as $B(S,\mathcal{M})$ and the secondary benefit value is considered as $E(S,\mathcal{M})$. The energy aware benefit function is outlined in Algorithm 1. The source set with the higher primary benefit value is assumed to have a higher benefit value. If two source sets have primary benefit values that are close to each other, i.e., their absolute difference is smaller than ϵ, then the secondary benefit value determines which source set has the higher benefit value. Consider a benefit value comparison of two source sets S_1 and S_2 for possible inclusion into M. If $abs(B(S_1,\mathcal{M}) - B(S_2,\mathcal{M})) < \epsilon$ then the source set with higher $E(S,\mathcal{M})$ is assumed to have a higher benefit value. Otherwise, source set with higher $B(S,\mathcal{M})$ value is assumed to have a higher benefit value. The purpose of the energy-aware benefit function is to select the minimum possible number of sensor nodes while preserving the energy quality of the selected nodes as high as possible.

We prefer geometric averaging scheme in the computation of $E(S,\mathcal{M})$. The geometric average of a given a set $\{e_1, e_2, ..., e_n\}$ of data is computed as $\sqrt[n]{e_1.e_2 \ldots e_n}$. Another approach could have been using arithmetic averaging

Algorithm 1: EnergyAwareBenefit Function

input : S_1, S_2, $\mathcal{M}$

```
if abs(B(S1, M) − B(S2, M)) ≤ ε then
    if E(S1, M) ≥ E(S2, M) then
        return S1
    else
        return S2
else
    if B(S1, M) ≥ B(S2, M) then
        return S1
    else
        return S2
```

scheme. Furthermore, instead of averaging, a min-max approach could have also been taken where $E(S, \mathcal{M})$ would be the minimum energy value of the new sensor node in the source set divided by the maximum energy value of the new sensor node in the newly inferred nodes set.

For a given dataset with a fixed arithmetic average, geometric averaging gives higher results for lower variations in the data values. That is why we prefer using the geometric averaging scheme rather than the arithmetic averaging scheme. For example, consider a source set S_1 with two new source nodes whose energy values are 1 and 19. Also consider a second source set S_2 with again two new source nodes whose energy values are 10 and 10. Assume that both source sets infer one new node whose energy value is 20. Because $B(S_1, \mathcal{M}) = B(S_2, \mathcal{M})$, the secondary metric will decide which source set to be selected. If arithmetic averaging would be used, this would have resulted in $E(S_1, \mathcal{M}) = E(S_2, \mathcal{M}) = 0.5$. However, it is obvious that S_1 should definitely have a lower benefit value since source set S_2 will likely be able to live longer than S_1. If geometric averaging would be used, this would have resulted in $E(S_1, \mathcal{M}) \simeq 0.2175$ and $E(S_2, \mathcal{M}) = 0.5$ which is desirable as selection of S_2 would likely result in a longer network lifetime.

When compared with the max-min approach, geometric averaging performs better in such cases; consider a source set S_3 with two new source nodes whose

energy values are 10 and 40. Also consider a second source set S_4 with again two new source nodes whose energy values are 10 and 10. Assume both source sets infer one new node whose energy value is 20. Because $B(S_3, \mathcal{M}) = B(S_4, \mathcal{M})$, the secondary metric will decide which source set to be selected. If max-min approach would be used, this would have resulted in $E(S_3, \mathcal{M}) = E(S_4, \mathcal{M}) = 0.5$. However, if geometric averaging would to be used, this would have resulted in $E(S_3, \mathcal{M}) = 1$ and $E(S_4, M) = 0.5$ which is desirable as selection of S_3 would likely result in a longer network lifetime.

During simulations, we observe that using geometric averaging in computing the $E(S, \mathcal{M})$ values prolongs the network lifetime of the WSN when combined with the iterative improvement heuristic the most. The details of the trade-off between these three benefit functions is given in Section 5.2.1.

4.2.2 Greedy Constructive Heuristic

We introduce the greedy constructive heuristic which generates a correlation-dominating set from the given set $\mathcal{C}$ of data correlations as the input. The constructed correlation-dominating set will be an input to the iterative improvement heuristic for refinement. The purpose of the greedy constructive heuristic is to perform the active sensor node selection as fast as possible for a large given data correlation input. The purpose of the greedy constructive heuristic is not to find the best or the minimum set of active sensor nodes. It is intended to be used together with the iterative improvement heuristic so that the energy quality of the selected active sensor nodes can be further improved. The greedy constructive heuristic uses the energy-aware benefit function for computing the benefit values of source sets.

The constructive heuristic briefly works as follows; it first computes the energy-aware benefit values for each source set through a single sequential pass over the given source sets. Then the source sets are sorted using a quicksort-based algorithm [18] according to the energy-aware benefit values in decreasing order. Finally source sets with higher benefit are added to set $\mathcal{M}$ until $\mathcal{M}$ becomes a

correlation-dominating set. The outline of the heuristic is given in Algorithm 2.

Algorithm 2: greedyConstructive Heuristic

input : $\mathcal{N}$, $\mathcal{C}$, *dataGatheringNode* d
output: $\mathcal{M}$

```
1  M ← ∅;
2  S_List ← ∅;
3  Nodes(M) ← d;
4  foreach correlation C = {S, s} ∈ C do
5      S.benefit1 ← B(S, M);
6      S.benefit2 ← E(S, M);
7      S_List ← S_List ∪ (S, S.benefit1, S.benefit2);
8  //sort in descending order;
9  S_SortedList ← Sort(S_List);
10 while IsCorrelationDom(M) = FALSE do
11     S ← next source set in S_SortedList;
12     M ← M ∪ {S};
```

For the sake of runtime efficiency, the source sets are maintained in compressed form in two one-dimensional arrays $srcNodeIndexArray$ and $srcNodeArray$. The IDs of the source sensor nodes that belong to the source set S are stored in $srcNodeArray$ at the indices beginning from $srcNodeIndexArray[S]$ to $srcNodeIndexArray[S+1]-1$. The inferred nodes are also maintained in compressed form in two one-dimensional arrays $inferredNodeIndexArray$ and $inferredNodeArray$. The IDs of the inferred sensor nodes by the source set S are stored in $inferredNodeArray$ at the indices beginning from $inferredNodeIndexArray[S]$ to $inferredNodeIndexArray[S+1]-1$. The data structures that are output of the greedy constructive heuristic are the $setMArray$ which corresponds to $\mathcal{M}$ and the $nodesInSetMArray$ which corresponds to $Nodes(\mathcal{M})$. $setMArray$ stores 1 in its ith index if the source set with ID i is in set $\mathcal{M}$ or 0 otherwise. Similarly $nodesInSetMArray$ stores 1 in its jth index if the node with ID j is inside a source set that is in $Nodes(\mathcal{M})$.

4.2.3 Iterative Improvement Heuristic

The selected set of active sensor nodes which constitute the correlation-dominating set found by the constructive heuristic is an initial solution to the iterative improvement heuristic. The purpose of the iterative improvement heuristic is to go through the initial solution and try to improve the quality of the selected active sensor nodes while preserving the correlation-dominating set property. The iterative improvement heuristic is outlined in Algorithm 3.

The iterative improvement heuristic is composed of 4 phases;

1. Induction of source sets that are not in $\mathcal{M}$ due to the sensor nodes of source sets in $\mathcal{M}$.
2. Identification and removal of redundant nodes in $Nodes(\mathcal{M})$.
3. Performing a sequence of swaps between selected sensor nodes and unselected source sets to improve the energy quality of $\mathcal{M}$.
4. Identification and removal of redundant nodes in $Nodes(\mathcal{M})$.

Algorithm 3: Iterative Improvement Heuristic

```
1 //First phase ;
2 sourceSetsInduction()
3 //Second phase;
4 eliminateRedundantNodes()
5 //Third phase;
6 performSwaps()
7 //Forth phase;
8 eliminateRedundantNodes()
```

For the first phase, a subset $\tilde{\mathrm{S}}$ of source sets in $\mathcal{M}$ may already contain the sensor nodes of another source set S_j which is not in $\mathcal{M}$. Source sets such as S_j are said to be induced by M. That is,

$$\bigcup_{S_i \in \tilde{S}} Nodes(S_i) \supseteq Nodes(S_j), \text{ where } S_j \notin \mathcal{M}. \tag{4.8}$$

These induced source sets are the ones that are not selected by the constructive heuristic but do exist. These induced source sets exist by the source sensor nodes in $Nodes(\mathcal{M})$ but which are probably in different source sets. Therefore without adding any further nodes to $Nodes(\mathcal{M})$, more source sets can be considered to exist in $\mathcal{M}$. Induction of new source sets increases the number of source sets in $\mathcal{M}$ and the number of sensor nodes inferred by $\mathcal{M}$. This increases the degrees of freedom of the iterative improvement heuristic which in turn increases the possibility of identification and deletion of redundant sensor nodes in phases 2 and 4, and increases the possibility of performing more swaps in phase 3 to enhance the energy quality of $\mathcal{M}$. Consider $S_1 \in \mathcal{M}$, $S_2 \in \mathcal{M}$ and $S_3 \notin \mathcal{M}$. Let $Nodes(S_1) = \{s_1, s_4, s_5\}$ and $Infer(S_1) = \{s_7\}$, and $Nodes(S_2) = \{s_2, s_9\}$ and $Infer(S_2) = \{s_{10}\}$. Let $Nodes(S_3) = \{s_1, s_2\}$ and $Infer(S_3) = \{s_3\}$. Since S_1 and S_2 induce S_3, S_3 can be added to $\mathcal{M}$ without any cost. This phase is outlined in Algorithm 4.

Algorithm 4: sourceSetsInduction function

```
input  : M
output: M
1 foreach source set S ∉ M do
2     existanceFlag ← TRUE;
3     foreach source node s ∈ S do
4         if s ∉ Nodes(M) then
5             existanceFlag ← FALSE;
6             break;
7     if existanceFlag = TRUE then
8         M ← M ∪ {S}
```

In the second phase of the algorithm, the redundant sensor nodes in $Nodes(\mathcal{M})$ are identified and removed from $\mathcal{M}$. A sensor node s is said to be redundant in $Nodes(\mathcal{M})$ if the following two conditions hold;

1. There is a source set $S \in \mathcal{M}$ where S infers sensor node s and does not contain s as its source sensor node. That is,
   $\exists\ S \in \mathcal{M}$ such that $s \notin S$ and $s \in Infer(S)$.

2. The sensor nodes that are inferred by the source sets that contain s are already inferred by other source set(s) in $\mathcal{M}$. That is,
   $\exists\ \bar{S} \subseteq \mathcal{M}$ such that $Infer(\bar{S}) \supseteq Infer(SrcSet(s))$ and $\bar{S} \cap SrcSet(s) = \emptyset$.

The number of active sensor nodes in a WSN is a very important factor on the application lifetime that runs on that WSN. If a sensor network has a large number of active sensor nodes, these sensor nodes will need to transmit their sensed data to the data gathering node. Due to multi-hop data routing, each forwarded data will reduce some amount of energy from the forwarder sensor node. This will affect the overall network lifetime as having more active sensor nodes will cause a faster reduction in the energy levels of the sensor nodes. Therefore it is very important to keep the number of active sensor nodes in the WSN as small as possible. For this purpose, in the iterative improvement heuristic, this phase allows to delete redundant sensor nodes from $Nodes(\mathcal{M})$. This phase deletes these redundant sensor nodes while preserving the correlation-dominating set property of the selected active sensor nodes set. Deletion of redundant sensor nodes will cause less network traffic without sacrificing the fully operability of the WSN. This will help the WSN to have a longer lifetime. The outline of this phase is provided in Algorithm 5.

In Algorithm 5, the first two for loops (lines 1-5) compute the inference count for each sensor node. Here, the inference count for sensor node s denotes the number of source sets that infer s. Then, the algorithm checks each sensor node s in $Nodes(\mathcal{M})$ whether it can be eliminated. For this purpose, the algorithm checks the inference count of each sensor node r which is inferred by the source sets that contain s. If the inference count of such a sensor node r is smaller than or equal to 1, it means that there is at most one source set that infers r. Therefore elimination of s from $Nodes(\mathcal{M})$ should not be allowed as it will leave r as an uninferred node. If r would remain as uninferred, set $\mathcal{M}$ would no longer be a correlation-dominating set. The if statement (lines 16-22) is executed if s

can be removed from $Nodes(\mathcal{M})$. In that case, s is removed from $Nodes(\mathcal{M})$, the source sets that contain s as a source sensor node are removed from $\mathcal{M}$ and finally the inference count of each sensor node that is no longer inferred is decremented. It should be noted here that the processing order of the sensor nodes in $Nodes(\mathcal{M})$ for elimination might affect the solution quality. Finding the maximum number of sensor nodes that can be eliminated from $Nodes(\mathcal{M})$ seems to be a hard problem. Therefore, for the sake of runtime efficiency, we prefered using a simple yet effective solution scheme.

Algorithm 5: eliminateRedundantNodes function

```
1  foreach sensor node s ∈ 𝒩 do
2      count[s] ← 0;
3  foreach source set S ∈ ℳ do
4      foreach sensor node s ∈ Infer(S) do
5          count[s] ← count[s] + 1;
6  foreach sensor node s ∈ Nodes(ℳ) do
7      foreach source set S ∈ SrcSet(s) do
8          removeFlag ← TRUE;
9          if S ∈ M then
10             foreach sensor node r ∈ Infer(S) do
11                 if count[r] ≤ 1 then
12                     removeFlag ← FALSE;
13                     break;
14             if removeFlag = TRUE then
15                 break;
16     if removeFlag = TRUE then
17         Nodes(ℳ) ← Nodes(ℳ) − {s};
18         foreach source set S ∈ SrcSet(s) do
19             if S ∈ M then
20                 ℳ ← ℳ − S;
21                 foreach sensor node r ∈ Infer(S) do
22                     count[r] ← count[r] − 1;
```

In the third phase of the algorithm, in order to improve the energy quality of selected nodes, the iterative improvement heuristic tries to perform swaps between the selected active sensor nodes and unselected source sets. For each sensor node s, whose residual energy level is less than the arithmetic average of the sensor nodes in $Nodes(\mathcal{M})$, the heuristic finds the set of sensor nodes that will remain uninferred if s is to be removed from set $Nodes(\mathcal{M})$. Then the heuristic tries to find a "good" subset of unselected source sets that can replace sensor node s in order to infer the uninferred sensor nodes when s is removed from set $Nodes(\mathcal{M})$. Here, goodness of a subset of source sets refers to containing small number of additional sensor nodes which have high residual energy levels.

Here we show that the solution to this swapping problem can be formulated as a subproblem of the original problem in a much smaller scale. We are again trying to select source sets for the inference of some nodes, but this time in a smaller scale. In a given correlation-dominating set solution $\mathcal{M}$ to the original problem $\mathcal{W}=(\mathcal{N},\mathcal{C})$, finding a "good" subset of unselected source sets to replace a source node s from $Nodes(\mathcal{M})$ can be formulated as finding a "good" correlation-dominating set of the following subproblem $\mathcal{W}_{sub}(s) = (\mathcal{N}_{sub}(s), \mathcal{C}_{sub}(s))$, where

$$\mathcal{C}_{sub}(s) = \{(S,r) : S \notin \mathcal{M} \wedge r \in Infer(SrcSet(s)) \\ \wedge\, r \notin Infer(\mathcal{M} - SrcSet(s))\} \tag{4.9}$$

$$\mathcal{N}_{sub}(s) = \bigcup_{(S,r)\in\mathcal{C}_{sub}} Nodes(S) \tag{4.10}$$

In the subproblem $\mathcal{W}_{sub}(s)$, $\mathcal{C}_{sub}(s)$ consists of the correlations among unselected source sets that infer the sensor nodes of already selected source sets that contains s and that are not inferred by the remaining source set in $\mathcal{M}$. $\mathcal{N}_{sub}(s)$ contains the sensor nodes of source sets in $\mathcal{C}_{sub}(s)$.

We use the 0-hop centralized constructive heuristic of [11] with our energy-aware benefit function defined in Section 4.2.1 for solving the above problem. The 0-hop centralized constructive heuristic is outlined in Algorithm 6. The

reason why we use the 0-hop centralized constructive heuristic rather than the greedy constructive heuristic (Algorithm 3) is because of the small scale of the subproblem. The scale of swapping problem is small because we are only trying to find source sets for inferring a small set of sensor nodes. Although 0-hop centralized constructive heuristic takes much more time than the greedy constructive heuristic for large scale problems, the running time of 0-hop centralized constructive heuristic is expected to be in acceptable levels for the small scale of the subproblem, and hence amortizing its better solution quality compared to the greedy constructive heuristic. The swapping of sensor nodes in $\mathcal{M}$ with unselected source sets is outlined in Algorithm 7.

In Algorithm 7, sensor nodes in $Nodes(\mathcal{M})$, whose residual energy levels are smaller than the average residual energy level of $\mathcal{M}$, are considered for swapping starting from the sensor node with the minimum residual energy level. We need to maintain a priority queue Q for the selection of sensor nodes with low energy levels because new sensor nodes that are added to $Nodes(\mathcal{M})$ due to the swap operations might be considered for swapping in the future iterations. The first two inner for loops (lines 7-14) construct the correlation-dominating set subproblem which will be solved for the swap of current sensor node s from $Nodes(\mathcal{M})$. The subproblem is solved at line 15 using 0-hop centralized constructive heuristic. At line 16, this subproblem solution is checked in order to see whether it improves the current quality of $\mathcal{M}$ in terms of average residual energy level. If the newly selected sensor nodes improve the overall solution quality of set $Nodes(\mathcal{M})$, then s is swapped with the newly selected source sets. The 0-hop centralized construtive heuristic may fail to find a solution for the subproblem, in which case the resulting M_{sub} is not swapped for the current solution $\mathcal{M}$. The last two for loops (lines 19-27) realize the swap operation together with inserting the new sensor nodes added to $Nodes(\mathcal{M})$ into Q. It should be noted that the $energy(s)$ function gives the residual energy level of sensor node s.

The fourth phase of the algorithm is the same as the second phase. The improved solution $Nodes(\mathcal{M})$ is pruned by identifying and deleting the nodes that become redundant after the swap phase.

Algorithm 6: 0HopCentralizedHeuristic

input : $\mathcal{N}_{sub}$, $\mathcal{C}_{sub}$
output: $\mathcal{M}$

```
1  M ← ∅;
2  foreach correlation C_cur = (S_cur, s_cur) ∈ C_sub do
3      if S_cur ∉ M then
4          S_max ← S_cur;
5          foreach correlation C_tmp = (S_tmp, s_tmp) ∈ C_sub do
6              if S_tmp ∉ M then
7                  S_max ← EnergyAwareBenefit(S_max, S_tmp, M);
8          M ← M ∪ {S_max};
9          if IsCorrelationDom(M) = TRUE then
10             break;
```

4.2.4 Minimum Steiner Tree Construction

After the execution of the iterative active sensor node determination heuristic, the correlation-dominating set is established. The correlation-dominating set is unaware of the network connectivity of the sensor nodes. It only guarantees that the constructed set is able to fully sense the necessary data of the WSN. In order for this data to be successfully collected at the data gathering node, the correlation-dominating set has to be fully connected. To establish the connected correlation-dominating set, we construct a minimum Steiner tree [17] which connects the sensor nodes in $Nodes(\mathcal{M})$ by adding necessary sensor nodes not in $Nodes(\mathcal{M})$ to achieve wireless connectivity. The sensor nodes that are added to $Nodes(\mathcal{M})$ after the minimum Steiner tree construction are called Steiner sensor nodes. The objective of minimum Steiner tree algorithm is to keep the number of Steiner nodes as small as possible. Note that the Steiner sensor nodes will not need to sense data from their environment although they will be active nodes in the network. The Steiner nodes will only be responsible for the routing of data packets towards the data gathering node.

Algorithm 7: performSwaps function

```
   input  : 𝒩, 𝒞, ℳ
   output: ℳ
 1 //Priority queue Q keyed with residual energy;
 2 Q ← Nodes(ℳ);
 3 s ← ExtractMin(Q);
 4 initialEnergyAvgM ← E_avg(M);
 5 while energy(s) < initialEnergyAvgM do
 6     𝒞_sub ← ∅;
 7     foreach correlation (S, r) ∈ 𝒞 do
 8         if E_avg(S) > initialEnergyAvgM then
 9             if source set S ∉ ℳ then
10                 if sensor node r ∈ Infer(SrcSet(s)) then
11                     𝒞_sub ← 𝒞_sub ∪ (S, r);
12     𝒩_sub ← ∅;
13     foreach correlation (S, r) ∈ 𝒞_sub do
14         𝒩_sub ← 𝒩_sub ∪ Nodes(S)
15     ℳ_sub ← OHopHeuristic(𝒩_sub, 𝒞_sub);
16     if E_avg(ℳ ∪ ℳ_sub) > E_avg(ℳ) then
17         //Swap s with Nodes(ℳ_sub);
18         Nodes(ℳ) ← Nodes(ℳ) − {s};
19         foreach correlation (S, r) ∈ 𝒞 do
20             if S ∉ ℳ then
21                 if r ∈ Infer(SrcSet(s)) then
22                     ℳ ← ℳ − {S};
23         ℳ ← ℳ ∪ {ℳ_sub};
24         foreach sensor node r ∈ Nodes(ℳ_sub) do
25             if r ∉ ℳ then
26                 Nodes(ℳ) ← Nodes(ℳ) ∪ {r};
27                 Insert(Q,r);
28         s ← ExtractMin(Q);
```

Chapter 5

Simulations and Evaluation

In this section, we report and discuss the results of the simulations we performed to test the validy of our proposed approach to the active sensor node determination problem in WSN. For this purpose, we first discuss the energy consumption model to be used in our simulations. Then, we report simulation results in different network topologies with different parameters to observe the performance of the proposed approach.

5.1 Energy Consumption Model

In order to determine the amount of energy that will be reduced from each selected sensor node, we define the following energy consumption model. After each configuration of set $\mathcal{M}$ as a connected correlation-dominating set, there are R data gathering rounds until the next configuration. During any given round, each selected active sensor node generates one packet towards the data gathering node. Let P be the amount of energy that is spent for transmitting one packet from one sensor node to its parent sensor node. Let G denote the number of descandants of an active sensor node. The total amount of energy spent by a selected active sensor node in a round is $P \times 2G + 1$ and the total amount of energy spent by a Steiner sensor node in a round is $P \times 2G$. Note that Steiner

nodes do not generate a data packet by themselves. They only act as routers for other packets. The total amount of energy that is consumed between two successive configurations is $R \times (P \times 2G + 1)$ for a selected active sensor node and is $R \times (P \times 2G)$ for a Steiner sensor node. We assume short-range radio transmitters and therefore the energy consumption for packet transmission is independent of the distance between sensor nodes. We also assume that the energy consumed in transmitting and receiving a packet is the same.

5.2 Assumptions and Parameter Values

The P value, which is the amount of energy that is spent for transmitting one packet from one sensor node to its parent sensor node, is selected as 0.01 energy units, where the initial energy of a sensor node is 100 energy units. The R value, which is the number of data gathering rounds between two configurations, is selected as 100. The communication range between sensor nodes is assumed to be 20 meters. The two main criteria for making performance comparisons between different approaches is sensor network lifetime and runtime of the active sensor node determination heuristics. We assume that the WSN is dead and cannot further operate once a connected correlation-dominating set cannot be constructed. Unless otherwise stated, the WSN topology is modelled according to Gaussian distribution with standard deviation (σ) set to 1 on a 150x150 meter^2 area, where the data gathering node is selected from the center of the network. It should be noted that in a WSN topology modelled according to Gaussian distribution more sensor nodes are placed around the center of the network as the σ value becomes smaller. The σ value indicates the variation of node positions around the data gathering node position.

The correlations that define the inference relationship among nodes are generated randomly in the simulations. The three parameters that effect the random correlation generation process are C_{per}, C_{maxsrc} and C_{maxhop}. A candidate correlation is accepted as a valid correlation if the correlation percentage value that is randomly selected in the scale of [0,100] is smaller than the defined C_{per} value

for correlation generation. The C_{maxsrc} parameter defines the maximum number of source sensor nodes that is allowed to be in a given correlation. Lastly, the C_{maxhop} parameter defines the maximum hop-count in the WSN for sensor nodes to infer one another. Unless otherwise stated, the correlation generation parameter values are $C_{per} = 50$, $C_{maxsrc} = 5$ and $C_{maxhop} = 3$.

For each simulation experiment, 10 different correlation sets are generated with different random seeds and the average of the 10 simulations on these correlation sets is reported in the following figures. This simulation scheme is used in order to provide average case results in the comparison of various active node determination heuristics.

5.2.1 Simulation Results

We first performed simulations to determine the parameters to be used in the comparison of IAND and L-hop centralized heuristics. Once we determined the parameters values to be used in these heuristics, we compared them in different network topologies. Finally, we changed the correlation generation parameter values to further observe and report the performance of the compared heuristics.

Figure 5.1 compares the performance of the 0-hop and 1-hop centralized heuristics with increasing number of nodes. The 0-hop and 1-hop heuristics are L-hop heuristics where L is 0 and 1, respectively. Figure 5.1(a) shows that the network lifetime performance of the 1-hop centralized heuristic is slightly better than that of the 0-hop centralized heuristic. The reason for the slightly better network lifetime performance of the 1-hop centralized heuristic is because of the fact that it includes a larger set of source sets into $\mathcal{M}$ through 1-hop union of source sets. However, in terms of runtime performance, Figure 5.1(b) shows that the runtime of the 1-hop centralized heuristic dramatically increases as the network becomes denser. Therefore, it becomes impractical to use the 1-hop centralized heuristic even for medium scale WSNs (and any L-hop heuristic with L larger than 1). For this reason, we compare our IAND heuristic against the 0-hop centralized heuristic in the rest of the experiments. The 0-hop centralized

heuristic achieves a solution of reasonably good quality with very short running time. It should be noted that the runtime of the 0-hop centralized heuristic is very small compared to that of the 1-hop centralized heuristic, so that its running time seems to lie on the x-axis of Figure 5.1(b). Because of the extremely long runtime of the 1-hop centralized heuristic, this simulation is performed on a 110x110 meter^2 area in which the number of sensor nodes in the network is varied between 125 and 350.

Figure 5.2(a) shows the effect of the ϵ parameter on the performance of our IAND heuristic. As seen in Figure 5.2(a), the network lifetime performance of IAND heuristic increases with increasing ϵ until $\epsilon = 0.5$, and then it begins to decrease for higher ϵ values. This is experimental finding is expected. For small ϵ values, the energy-aware benefit function gives more emphasis to the $B(S, \mathcal{M})$ function which considers the number of source nodes and inferred nodes. For large ϵ values, the energy-aware benefit function gives more emphasis to $E(S, \mathcal{M})$ function which considers the residual energy levels of the source nodes and inferred nodes.

Figure 5.2(b) shows the effect of three different energy averaging schemes proposed in Section 4.2.1 for our energy-aware benefit function in the performance of IAND heuristic. We compare the performance of our energy-aware benefit function against the benefit function of [11] which is referred to as "base" in Figure 5.2(b). As seen in the figure, although the difference is not large, all proposed energy-aware benefit function schemes perform better than the benefit function of [11]. Furthermore, Figure 5.2(b) confirms our expectation that the geometric averaging scheme performs better than the arithmetic averaging scheme and min-max scheme.

Figure 5.3 shows the performance of the IAND heuristic compared with the 0-hop centralized heuristic as the WSN becomes denser. In Figure 5.3, we also display the simulation results for an IAND version in which a random constructive heuristic is used instead of the greedy constructive heuristic given in Algorithm 2. This random constructive heuristic selects source sets randomly until the correlation-dominating set is constructed. IAND-rand results are given here

to show the effectiveness of iterative improvement heuristic even when a simple constructive heuristic is used for finding an initial solution. Thus, the IAND-rand heuristic is composed of the random constructive heuristic follwed by the iterative improvement heuristic. As seen in Figure 5.3(a), both IAND and IAND-rand perform considerably better than the 0-hop centralized heuristic in terms of average network lifetime, where IAND performs better than IAND-rand. As seen in Figure 5.3(b), the proposed IAND heuristics run drastically faster than the 0-hop centralized heuristic and the runtime performance gap increases considerably with the increasing network density in favor of IAND heuristics.

Figure 5.4 compares the performance of the IAND and IAND-rand heuristics with the 0-hop centralized heuristic as the σ value of the Gaussian distribution of the given WSN topology increases while the number of nodes stays as 500 in the same area. It should be noted here that the WSN topology becomes more uniform with increasing σ. Similarly, Figure 5.5 compares the performance of the IAND and IAND-rand heuristics against the 0-hop centralized heuristic in a uniform network topology as the number of nodes in the same area increases. As seen in Figure 5.4(a), the IAND approach is able achieve relatively much better network lifetime performance for small values of σ where the network topology is skewed and denser around the data gathering node. As also seen in Figure 5.4(a), the performance gap between the IAND heuristics and the 0-hop centralized heuristic becomes smaller with increasing σ. However, as seen in Figure 5.5(a), there is still considerable network lifetime performance difference between IAND approach and the 0-hop centralized heuristic in the uniform WSN topology. As seen in Figure 5.4(b), the runtime performance gap between IAND and the 0-hop centralized heuristic stays the same with increasing σ. As seen in Figure 5.5(b), the IAND heuristics run drastically faster than the 0-hop centralized heuristic as the number of nodes in the uniform WSN topology increases.

The main reason for the decrease in the network lifetime performance gap between IAND heuristics and 0-hop centralized heuristic with increasing σ is because of the fact that when the network topology is uniform, the nodes around the data gathering node constitute a bottleneck for the performance of all heuristics. The energy levels of the sensor nodes around the data gathering node deplete

faster than other nodes in the network because the sensor nodes around the data gathering node have more descendant sensor nodes and therefore they need to forward more packets than other sensor nodes. Since the sensor nodes around the data gathering node deplete their energy levels and become unusable, it becomes harder to construct a connected correlation-dominating set in a uniform topology. That is why all the approaches were not able to increase the network lifetime after a limited point. This experimental finding was also reported in [21], [6]. Figure 5.5(a) also shows that in uniform WSN topology, the network lifetime performance gap between the IAND and IAND-rand heuristics is quite small.

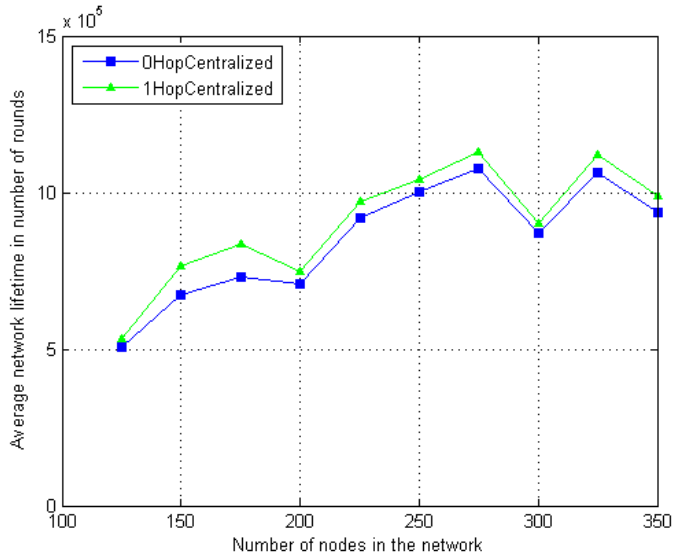
Figures 5.6–5.8 are presented to show the effect of the correlation-set generation parameters C_{per}, C_{maxsrc} and C_{maxhop} in the performance comparison of heuristics. Figures 5.6–8 shows the performance variation of IAND and 0-hop centralized heuristic with varying one of the parameter while fixing the other two. The simulations were performed with a 500 node WSN network.

As seen in .Figure 5.6(a), with increasing C_{per} value the network lifetime performance of all heuristics increase while the performance gap between IAND and 0-hop centralized heuristic remains nearly the same. Figure 5.6(b) shows that the runtime performance gap between IAND and 0-hop centralized heuristic becomes smaller as the correlation percentage increases. This is because, as the C_{per} value increases, the number of available candidate source sets increase and hence the number of passes over the candidate source sets performed by the 0-hop centralized heuristic decreases.

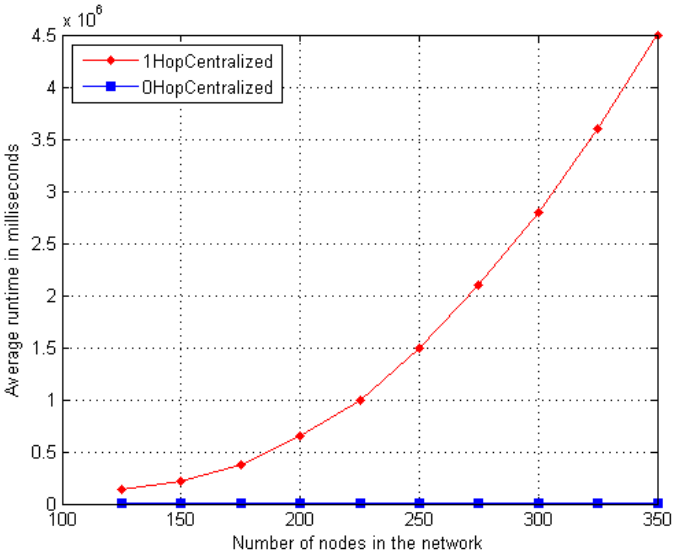
As seen in Figure 5.7, the C_{maxsrc} value does not affect the network lifetime and runtime performance of the heuristics considerably. This behaviour might be attributed to the possibility that the size of the union of source sets that constitute the connected correlation-dominating set remain nearly the same with varying C_{maxrsc} value.

As seen in Figure 5.8(a), the increase of the C_{maxhop} value increases the performance gap between IAND and 0-hop centralized heuristic in the favor of IAND. As the C_{maxhop} value increases, more sensor nodes that are distant from each

other are able to infer one another. This allows the iterative improvement heuristic to perform more swap operations which in turn increases the network lifetime performance. As seen in Figure 5.8(b), the runtime performance gap between IAND heuristics and 0-hop centralized heuristic remains nearly the same.

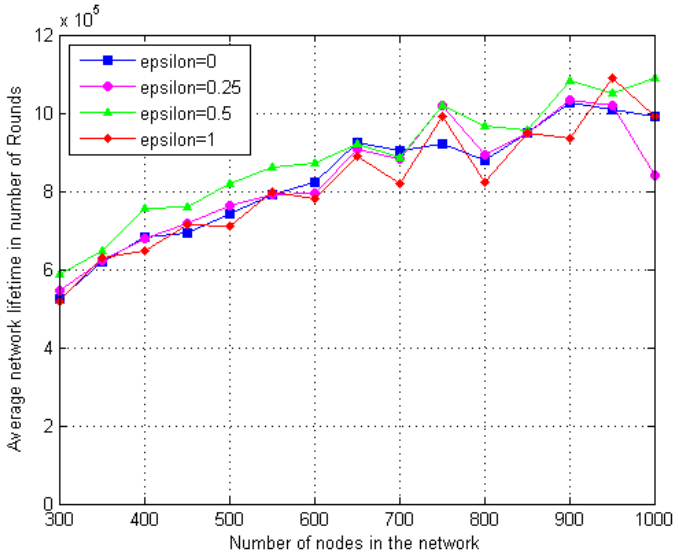


(a) Average network lifetime performance

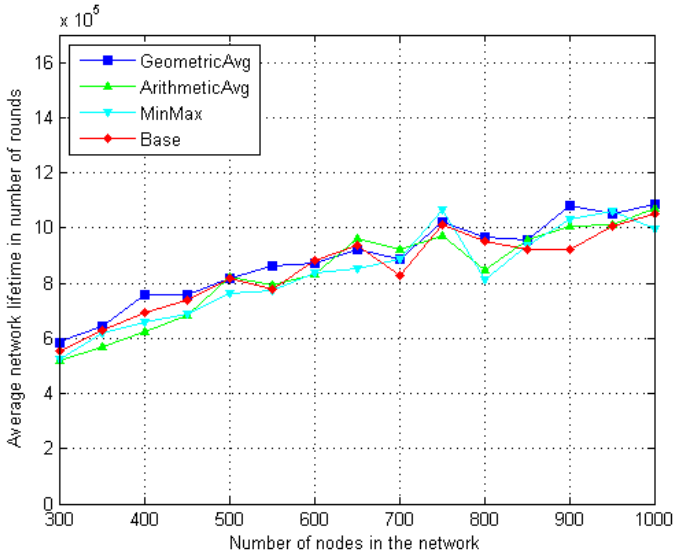


(b) Average runtime performance

Figure 5.1: Performance comparison of 0-hop and 1-hop centralized heuristics.

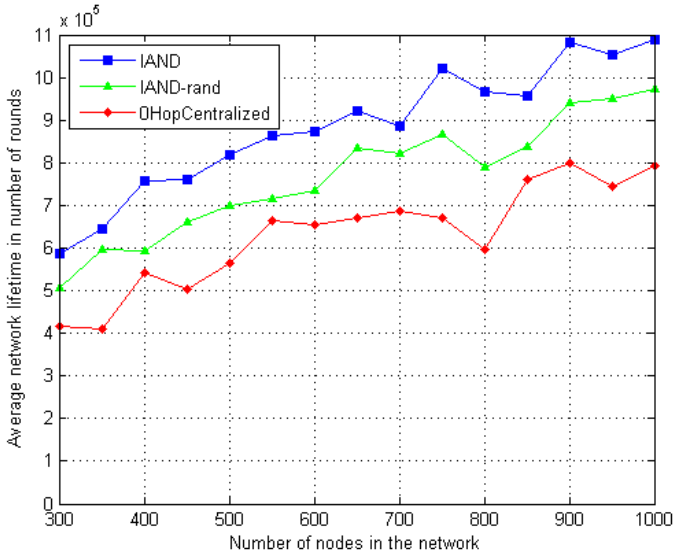


(a)

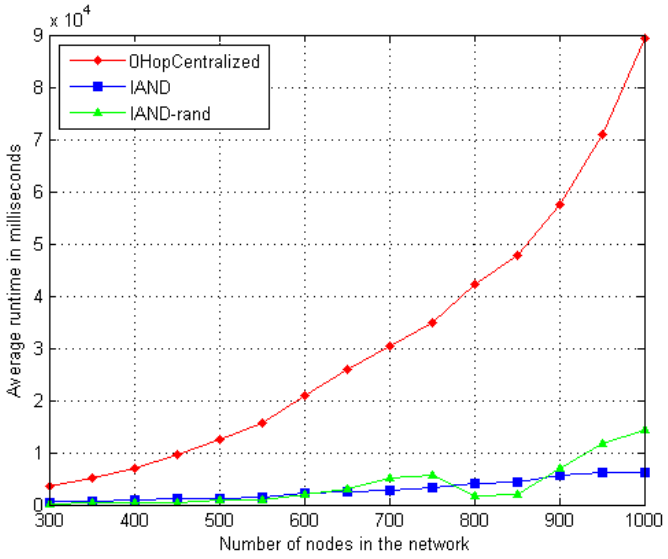


(b)

Figure 5.2: Effect of (a) ϵ parameter and (b) benefit function scheme on the performance of the IAND heuristic

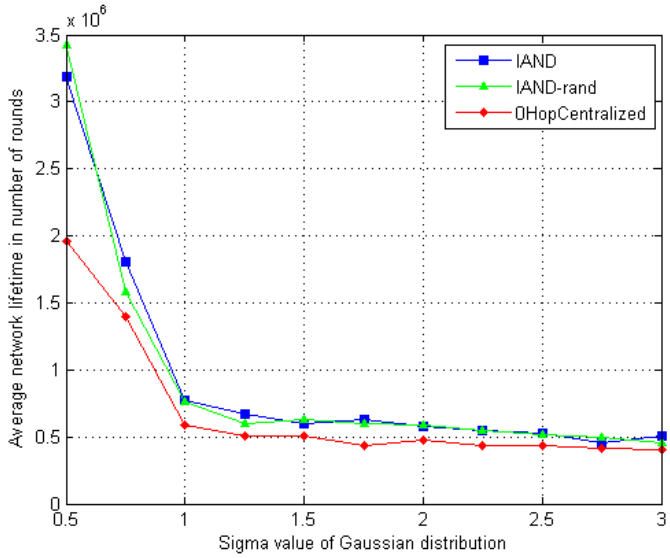


(a) Average network lifetime performance

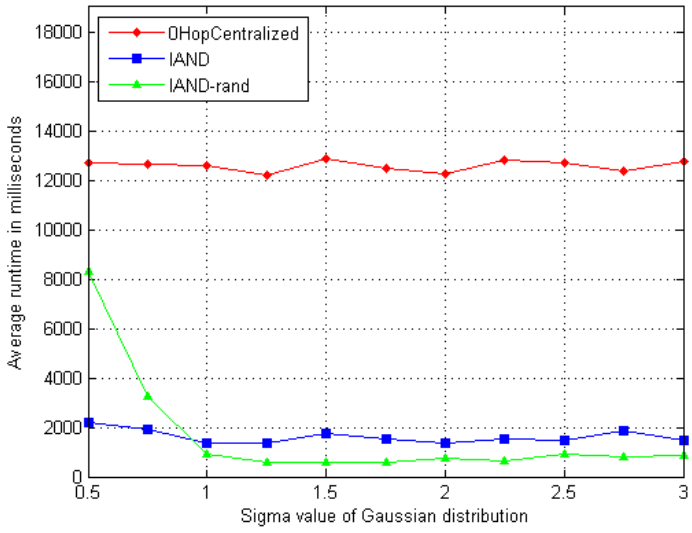


(b) Average runtime performance

Figure 5.3: Performance comparison of IAND, IAND-rand and 0-hop centralized heuristic with increasing number of nodes and Gaussian distribution with $\sigma = 1$.

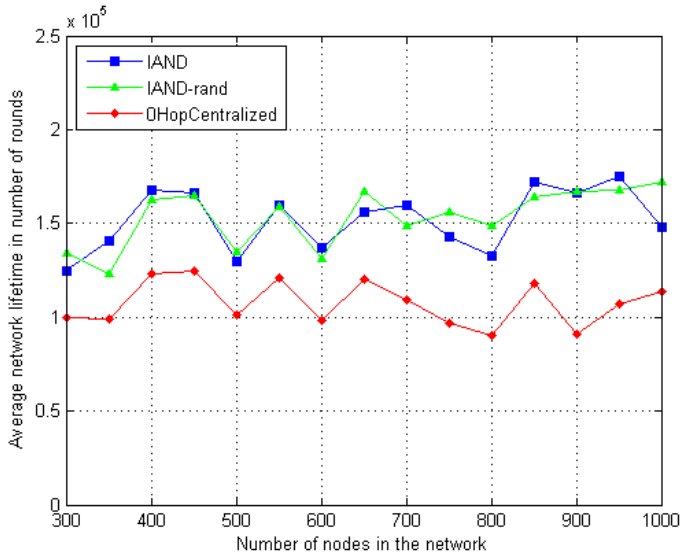


(a) Average network lifetime performance

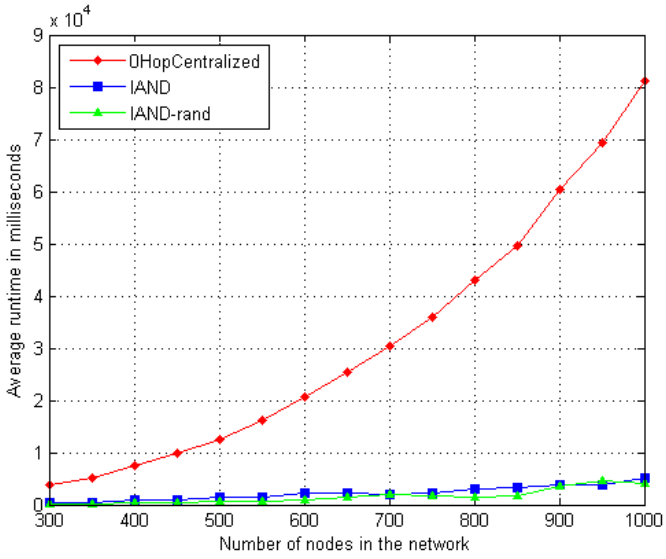


(b) Average runtime performance

Figure 5.4: Performance comparison of IAND, IAND-rand and 0-hop centralized heuristic with increasing Gaussian distribution σ.

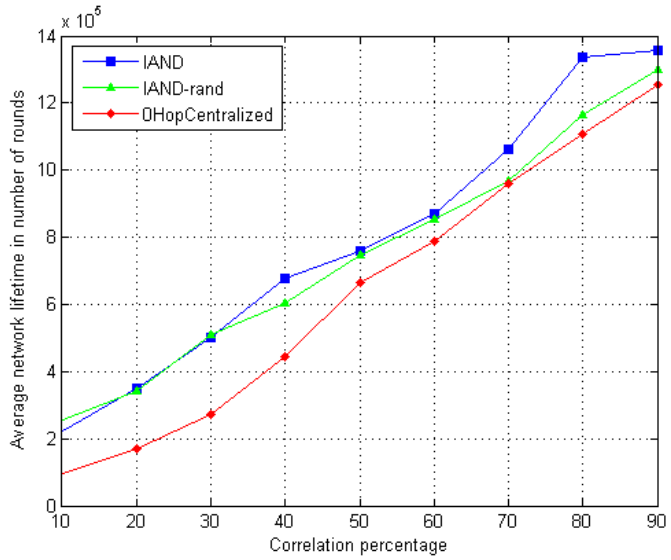


(a) Average network lifetime performance

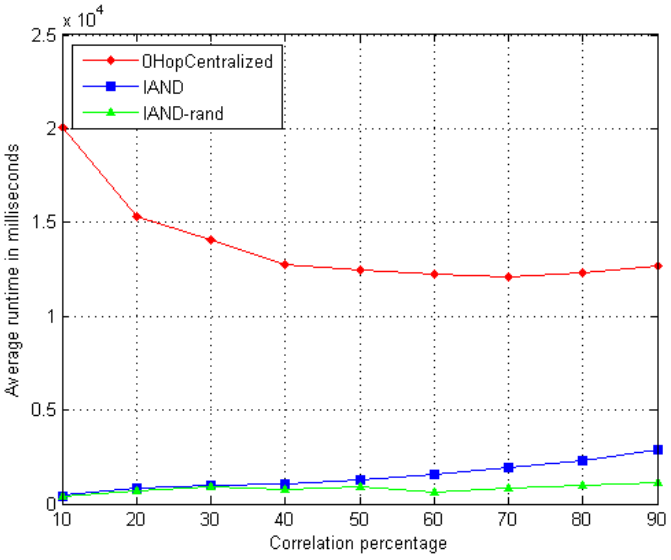


(b) Average runtime performance

Figure 5.5: Performance comparison of IAND, IAND-rand and 0-hop centralized heuristic in a uniform topology.

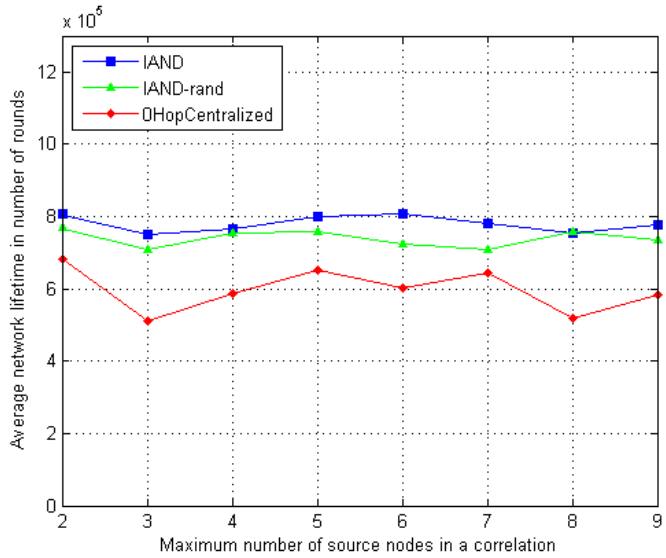


(a) Average network lifetime performance

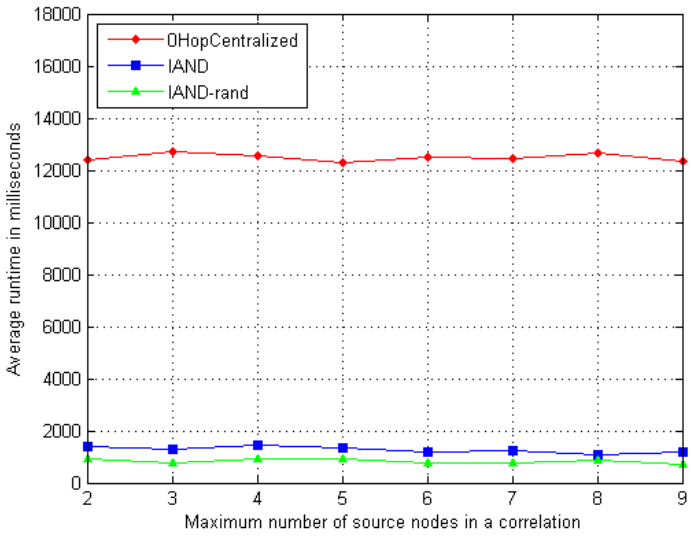


(b) Average runtime performance

Figure 5.6: Performance comparison of IAND, IAND-rand and 0-hop centralized heuristics as the C_{per} value is increased, where $C_{maxsrc} = 5$ and $C_{maxhop} = 3$.

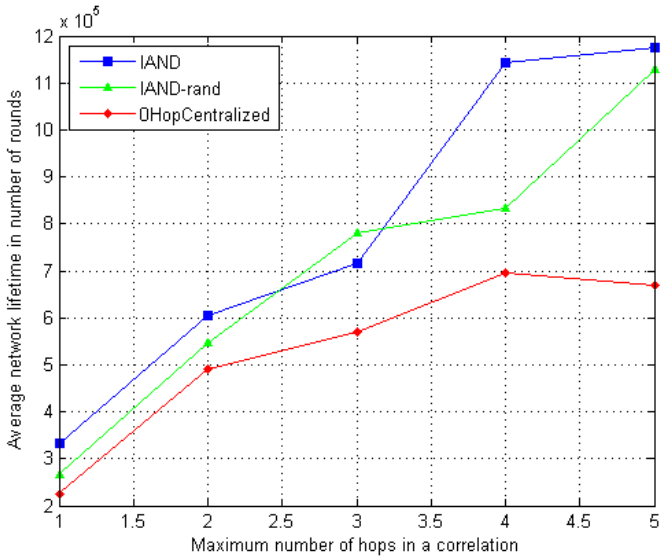


(a) Average network lifetime performance

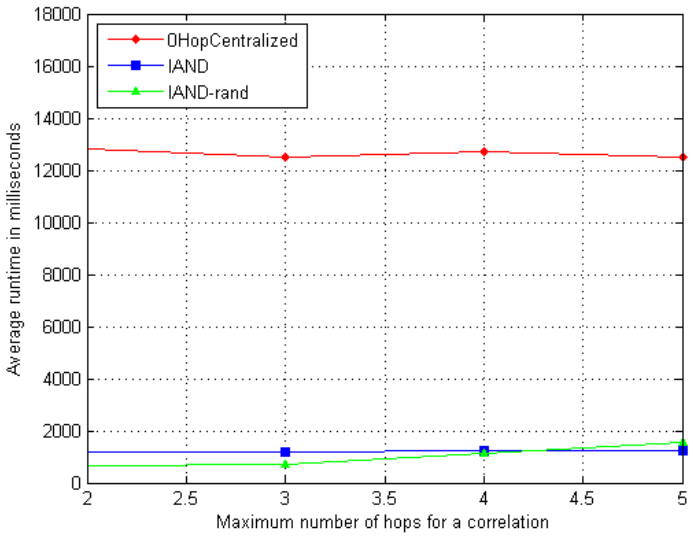


(b) Average runtime performance

Figure 5.7: Performance comparison of IAND, IAND-rand and 0-hop centralized heuristics as the C_{maxsrc} value is increased, where $C_{per} = 50$ and $C_{maxhop} = 3$.



(a) Average network lifetime performance



(b) Average runtime performance

Figure 5.8: Performance comparison of IAND, IAND-rand and 0-hop centralized heuristics as the C_{maxhop} value is increased, where $C_{per} = 50$ and $C_{maxsrc} = 5$.

Chapter 6

Conclusion

In wireless sensor network (WSN) applications where data gathered by different sensor nodes is correlated, all sensor nodes need not to be active for the WSN to be functional. In such WSN applications, selecting a set of active sensor nodes in the network is a critical issue for the performance of the WSN. In this work, we considered the problem of finding an active subset of nodes which are connected and can infer the correlated data of the inactive sensor nodes. This problem was formulated as an instance of the connected correlation-dominating set problem. In order to solve the connected correlation-dominating set problem in the context of WSNs, we have proposed and developed an Iterative Active sensor Node Determination (IAND) heuristic which is composed of a fast constructive heuristic followed by an effective and runtime efficient iterative improvement heuristic. The constructive heuristic is a fast algorithm that provides an initial solution for the iterative improvement heuristic. This initial solution is composed of selected active sensor nodes that constitute a correlation-dominating set for the given network.

The iterative improvement heuristic performs a sequence of swap operations to further improve the quality of active sensor nodes while preserving the correlation-dominating set property of the set of active sensor nodes. The swap operations take place between the selected sensor nodes in the current correlation-dominating set and the unselected source sets. The problem of finding a "good"

swap for a given selected source node was formulated as a subproblem of the original correlation-dominating set problem. We used the 0-hop centralized heuristic of [11] for solving this swap subproblem due to the small-size of the subproblem.

The extensive simulations that we performed showed that the proposed approach can efficiently compute an active sensor node set and can be effective in prolonging the network lifetime. We also compared our approach with a state-of-the-art approach. The simulation results showed that our approach can perform considerably better in terms of WSN lifetime than the existing approach, while achieving drastically better runtime efficiency.

Bibliography

[1] A. A. Abbasi and M. F. Younis. A survey on clustering algorithms for wireless sensor networks. *Computer Communications*, 30(14-15):2826–2841, 2007.

[2] J. N. Al-Karaki, R. Ul-Mustafa, and A. E. Kamal. Data aggregation and routing in wireless sensor networks: Optimal and heuristic algorithms. *Comput. Netw.*, 53(7):945–960, 2009.

[3] R. C. Baltasar, R. Cristescu, B. Beferull-lozano, and M. Vetterli. On network correlated data gathering. In *in IEEE InfoCom*, pages 2571–2582, 2004.

[4] T. Berger. Rate distortion theory: A mathematical basis for data compression. *Prentice-Hall series in information and system science*, 1971.

[5] J. Chou, D. Petrovic, and K. Ramchandran. A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks. In *INFOCOM*, 2003.

[6] Y. Ding, C. Wang, and L. Xiao. An adaptive partitioning scheme for sleep scheduling and topology control in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 20(9):1352–1365, 2009.

[7] M. Enachescu, A. Goel, R. Govindan, and R. Motwani. Scale-free aggregation in sensor networks. *Theor. Comput. Sci.*, 344(1):15–29, 2005.

[8] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman: San Francisco, 1979.

[9] A. Goel and D. Estrin. Simultaneous optimization for concave costs: single sink aggregation or single source buy-at-bulk. *SODA*, pages 499–505, 2003.

[10] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, 1998.

[11] H. Gupta, V. Navda, S. Das, and V. Chowdhary. Efficient gathering of correlated data in sensor networks. *ACM Trans. Sen. Netw.*, 4(1):1–31, 2008.

[12] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *Wireless Communications, IEEE Transactions on*, 1(4):660–670, 2002.

[13] A. Iyer, S. S. Kulkarni, V. Mhatre, and C. P. Rosenberg. A taxonomy-based approach to design of large-scale sensor networks. *Wireless Sensor Networks and Applications*, 2008.

[14] H. Karl and A. Willig. *Protocols and Architectures for Wireless Sensor Networks*. Wiley-Interscience, October 2007.

[15] M. Lotfinezhad and B. Liang. Effect of partially correlated data on clustering in wireless sensor networks. *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 172–181, Oct. 2004.

[16] H. Luo, Y. Liu, and S. Das. Routing correlated data with fusion cost in wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 5(11):1620–1632, Nov. 2006.

[17] K. Mehlhorn. A faster approximation algorithm for the steiner problem in graphs. *Inf. Process. Lett.*, 27(3):125–128, 1988.

[18] D. R. Musser. Introspective sorting and selection algorithms. *Software–Practice and Experience*, 8:983–993, 1997.

[19] G. Robins and A. Zelikovsky. Improved steiner tree approximation in graphs. In *SODA '00: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 770–779, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.

[20] S.G.Foss and S. Zuyev. On a voronoi aggregative process related to a bivariate poisson process. *Advances in Applied Probability*, 1996.

[21] R. Subramanian and F. Fekri. Sleep scheduling and lifetime maximization in sensor networks: fundamental limits and optimal solutions. In *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*, pages 218–225, New York, NY, USA, 2006. ACM.

[22] P. von Rickenbach and R. Wattenhofer. Gathering Correlated Data in Sensor Networks. *ACM Joint Workshop on Foundations of Mobile Computing (DIALM-POMC), Philadelphia, Pennsylvania, USA*, October 2004.

[23] S. Yoon and C. Shahabi. Exploiting spatial correlation towards an energy efficient clustered aggregation technique (cag) [wireless sensor network applications]. *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*, 5:3307–3313 Vol. 5, May 2005.

[24] O. Younis and S. Fahmy. Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *Mobile Computing, IEEE Transactions on*, 3(4):366–379, 2004.