DATA COLLECTION ALGORITHMS IN WIRELESS SENSOR NETWORKS EMPLOYING COMPRESSIVE SENSING

By

MINH TUAN NGUYEN

Bachelor of Electrical Engineering
University of Transport and Communications
Hanoi, Vietnam
2001

Master of Electrical Engineering
Military Technical Academy
Hanoi, Vietnam
2007

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2015

DATA COLLECTION ALGORITHMS IN WIRELESS SENSOR NETWORKS EMPLOYING COMPRESSIVE SENSING

Dissertation Approved:

Dr. Keith A. Teague

Dissertation Advisor

Committee Member: Dr. George Scheets

Committee Member: Dr. Qi Cheng

Committee Member: Dr. Johnson Thomas

ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Keith A. Teague for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study. I also would like to thank his wife, Mrs. Sherry Teague for everything she did for my family and myself. Thank you both very much for helping our colleagues from TNUT to visit OSU.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. George Scheets, Prof. Qi Cheng from School of Electrical and Computer Engineering and Prof. Johnson Thomas from Computer Science department for their insightful comments and encouragement, but also for the hard question which incented me to widen my research from various perspectives.

I also would like to thank some professors from Electrical and Computer engineering (ECE), Dr. Martin Hagan, George Scheets, Dr. James West, Dr. Guoliang Fan, Dr. Chandler, Dr. Qi Cheng, Dr. Rahnavard and Dr. Weihua Sheng for their classes and their knowledge they shared with me. I really appreciate that.

Being far away from my home country and my institute has been given me a big gap of culture. I would like to thank the department staffs, Lory Ferguson, Hellen Daggs, Sherrie Ishmael, Brian Ritthaler, especially Lory Ferguson for being supportive to me all the time.

I thank my fellow labmates, Ali Talari, Behzad Shahrasbi, Sheng Wang from the CWN lab for the stimulating discussions, for the tough time we were working together,

and for all the fun we have had in the last few years.

My sincere thanks to all Vietnamese families and friends here in Stillwater, Oklahoma. "chu Xuong", "chu Loi", "chu Nho", "anh Ky - chi Vinh", "em Dinh", "em Loan", Ha Do, Son Bui, Hung La, Hoa Nguyen, etc. You all have provided us the second family here in the US. I am grateful to the Vietnamese Student Association (VSA) with useful activities that bring us, our Vietnamese students at OSU, together.

Last but not the least, I would like to thank my little family, Ally Nguyen, Hang Nguyen and Thuong Nguyen for being with me, going together through tough time and enjoying happiness together. Without you, I would not have done this far including effort and succeed. I am very grateful to my big family, my parents, my brother, my sister in law, my nephew and niece for supporting me spiritually throughout writing this dissertation and my life in general. I would like to thank the big family in Ha Noi, especially grandpa Khien Trong Nguyen for being supportive to me while I was preparing to study abroad.

Thank you all very much!!! Thanks the others I did not list their names here. Five years generally may not be considered as a long time, but for me, we do not have many this five years in our lives. So, it is precious. Thanks Stillwater, the very peaceful land and suitable for studying. I may not see You again but I appreciate every moment here in Oklahoma, USA. Thank You!

Acknowledgements reflect the views of the author and are not endorsed by committee members or Oklahoma State University.

TABLE OF CONTENTS

Chapter Page

1 INTRODUCTION 1

2 BACKGROUND AND LITERATURE REVIEW 3

2.1 Wireless Sensor Network Overview 3

2.1.1 Introduction 3

2.1.2 Challenges for Data Collection Method Design in WSNs . . . 4

2.1.3 Data Collection Method Protocols in WSNs 7

2.2 Introduction to Compressive Sensing 31

2.2.1 Introduction 31

2.2.2 Vector Spaces 32

2.2.3 Sensing Matrices 35

2.2.4 Signal Recovery 41

2.3 Literature Review 45

2.3.1 CS Based Data Collection Algorithm in WSNs 46

2.3.2 Minimizing the Number of CS Measurements 51

3 RANDOM WALK BASED DATA GATHERING IN WIRELESS SENSOR NETWORKS 53

3.1 Introduction 53

3.1.1 Motivation 53

3.1.2 Related Work 55

3.2 Background and Problem Formulation 57

3.2.1 Random Walk 57
3.2.2 Problem Formulation 58
3.3 Compressive Sensing Based Random Walk Data Collection Algorithm (CSR) 60
3.3.1 System Model 60
3.3.2 The CSR Algorithm 60
3.3.3 Analysis of the Measurement Matrix: CS Recovery Performance and Network Coverage 62
3.3.4 Analysis of the Trade-off between the Transmission Range and the Random Walk Length 63
3.4 Directly Forwarding the CS Measurements to the Base-station (D-CSR) 63
3.4.1 Network Model 63
3.4.2 D-CSR Power Consumption Analysis 64
3.4.3 D-CSR Simulation Results 68
3.5 Multi-hop Relaying Data from Random Walks to the Base-station (M-CSR) 73
3.5.1 Network Model 73
3.5.2 Multi-hop Relaying Data Algorithm 73
3.5.3 M-CSR Power Consumption Analysis 75
3.5.4 M-CSR Simulation Results 77
3.6 Conclusion and Future Work 79

4 CLUSTER BASED DATA COLLECTION IN WIRELESS SENSOR NETWORKS 81

4.1 Introduction 81
4.1.1 Motivation 81
4.1.2 Related work 83
4.2 Problem Formulation 86

4.2.1 System Model 86
4.2.2 Block Diagonal Matrices 87
4.2.3 Problem Formulation 88
4.3 CCS: Cluster-Based Compressive Sensing for Data Collection in WSNs 88
4.4 Directly Send CS Measurements to the BS (DCCS) 92
4.4.1 Network Model 92
4.4.2 Power Consumption Analysis for DCCS 92
4.4.3 Simulation Results for DCCS 95
4.5 Inter-cluster Multi-hop Routing in CCS (ICCS) 104
4.5.1 Network Model 106
4.5.2 ICCS Power Consumption Analysis 108
4.5.3 ICCS Simulation Results 110
4.6 DCT Compression Transmitting only k Large Coefficients 112
4.6.1 Network Model 114
4.6.2 Communication Power Consumption 114
4.6.3 Simulation Results 115
4.7 Conclusion 120

5 TREE-BASED DATA GATHERING IN WIRELESS SENSOR NETWORKS 122

5.1 Introduction 122
5.1.1 Motivation 122
5.1.2 Related Work 124
5.2 Problem Formulation 127
5.2.1 Network Model 127
5.2.2 Tree-base Energy-Efficient Data Gathering (TCS) 127
5.2.3 Power Consumption Analysis 129
5.3 Simulation Results 131

5.3.1 Lattice Network 131
5.3.2 Arbitrary Network 131
5.4 Conclusions and Future Work 137

6 NEIGHBORHOOD BASED DATA COLLECTION IN WIRELESS SENSOR NETWORKS 138

6.1 Introduction 138
6.2 Problem Formulation 138
6.2.1 Network Model 138
6.2.2 Neighborhood Based Data Collection Algorithm (NeiCS) . . . 139
6.2.3 Power Consumption Analysis 141
6.3 Simulation Results 146
6.4 Conclusion and Future Work 151

7 CONCLUSIONS 152

BIBLIOGRAPHY 154

A RANDOM WALK BASED DATA GATHERING IN WIRELESS SENSOR NETWORKS 174

A.1 Additional Analysis for D-CSR to calculate E_{dtoBS} in order to compare with M-CSR 174

LIST OF TABLES

Table Page

4.1 Comparison between the existing data collection methods and CCS . 85

LIST OF FIGURES

Figure Page

2.1 K-means clustering algorithm with $k = 10$ clusters 9

2.2 EEHC algorithm with single level of clustering 12

2.3 FLOC program consists of 6 actions 13

2.4 Unequal size clusters in EEUC clustering algorithm 15

2.5 Packet transmission and global transmission schedule on location-aware in PEACH 16

2.6 Cluster structure of a network with MRPUC 17

2.7 S-Web clustering algorithm 18

2.8 HEECH divides WSN into six tracks with the same width 19

2.9 An illustration of Directed Diffusion in WSN 22

2.10 State transitions in GAF 28

2.11 Some common normed vectors 34

2.12 Random projection matrix 36

2.13 Restricted Isometric Constant Function 39

2.14 Sparsifying signals in a proper domain with ψ matrix 40

2.15 Compare four Compressive Sensing reconstruction schemes 46

3.1 Illustration of a simple RW routing in a WSN with 8 nodes and the projection matrix created from each RW. 59

3.2 Sensor neighborhoods defined by the sensor transmission range R . . 64

3.3 An illustration of RWs collecting data when BS at the center 66

3.4 RWs collecting data when the BS is outside the sensing area at $(L_i, \frac{L}{2})$. 67

3.5 The average number of neighbors of each sensor when changing the sensor transmission range R 68
3.6 The mixing time reduces as the sensor transmission range R increases 69
3.7 Sampling coverage the network with different number of random walks length 48 ($\tau = 48$) 70
3.8 The average square distance ($E[d^2_{toBS}]$) between RWs and the BS at different positions $L_i \geq 0.5L$ up to $L_i = 5L$ 70
3.9 Total power consumption of the network versus sensor transmission ranges when BS at the center of the sensing area 71
3.10 Comparison between the full dense Gaussian and the sparse binary matrix collected different number of RWs: random walk length $\tau = 48$ with different number of measurements 72
3.11 Total power consumption through all data collection processes with M = 90 measurements, transmission range R* = 14 in different RW's lengths when the BS at the center of the sensing area 72
3.12 Tree-based relaying measurements after each RW to the base-station formed with 500 nodes and transmission range $R = 14$. 75
3.13 Total number of hops from all sensor nodes to the BS as we increase the transmission range R 77
3.14 The total power consumption applied M-CSR versus different transmission ranges R when BS at the center; R* = 12 78
3.15 Compare the total power consumption in two random walk routing method when BS at the center and $R = 14$. 79

4.1 Average reconstruction error versus the fraction of the measurements collected from the first cluster ($T = M_1/M$). The error is minimize when T is equal to the fraction of the nodes in the first cluster ($N_1/N = 0.7$). 91

4.2 A clustered WSN with BS outside the sensing area ($L_i > L$). 94
4.3 Histogram of number of sensors in each cluster for K-means and LEACH. 96
4.4 Number of measurements required to satisfy target error = 0.1 for a 100-sparse signal (sparse in canonical basis). 97
4.5 Total power consumption when BS at the center of the sensing area. Here, $N_c^* = 14$. 98
4.6 Total power consumption when BS at 1L ($L_i = L$). Here, $N_c^* = 9$. . . 99
4.7 Total power consumption when BS at 2L ($L_i = 2L$). Here, $N_c^* = 4$. . 99
4.8 Total power consumption when BS at 3L ($L_i = 3L$). Here, $N_c^* = 2$. . 100
4.9 Number of measurements required when Wavelet is considered as the sparsifying basis. 101
4.10 Total power consumption when the BS is at the center of the sensing area. Here, $N_c^* = 18$. 101
4.11 Total power consumption when $L_i = L$. Here, $N_c^* = 12$. 102
4.12 Total power consumption when $L_i = 3L$. Here, $N_c^* = 2$ *or* 3 (depending on the clustering scheme). 102
4.13 Total power consumption when $L_i = 5 \times L$. Here, $N_c^* = 2$. 103
4.14 Number of measurements required when DCT is considered as the sparsifying basis. 103
4.15 Total power consumption when the BS at the center of the sensing area. 104
4.16 Total power consumption when the BS outside the sensing area at $L_i = 3L$. 105
4.17 All transmissions in the clustered network with inter-cluster multi-hop routing when the BS at the center. 105
4.18 Total number of hops routing when changing the broadcasting radius R 110
4.19 The total power consumption when change the broadcast radius R . 111

4.20 Intra-cluster power consumption when BS at the center in a circle sensing area . . . 112
4.21 Inter-cluster power consumption when BS at the center in a circular sensing area . . . 113
4.22 Total power consumption for ICCS and DCCS in a circular area network with $R_0 = 50$ . . . 113
4.23 Unsorted sensory readings from 2000 sensors and the DCT transformed coefficients . . . 116
4.24 Descending sorted readings from 2000 sensors and the DCT transformed coefficients . . . 117
4.25 Reconstruction error versus number of measurements with different of clusters . . . 118
4.26 Reconstruction error versus number of clusters with different of measurements . . . 118
4.27 CS reconstruction error versus measurements with noise . . . 119
4.28 DCT compression reconstruction error versus measurements with noise and noiseless . . . 119

5.1 An example of a tree formed by MTT algorithm with 1000 nodes deployed in a arbitrary network when BS at the center . . . 123
5.2 A simple example illustrates TCS algorithm with 8 sensors . . . 127
5.3 Compare total numbers of transmissions between three algorithms in different lattice topology networks . . . 132
5.4 Total energy consumption in arbitrary networks with different numbers of nodes with M = 500, p = 1/3 . . . 133
5.5 The reduction ratio of power consumption of TCS over MTT in arbitrary networks versus the various number of sensors(M = 500, p = 1/3) . . . 134

5.6 Total number of transmission hops in various transmission range (N = 2000; M = 500; p = 1/3) 134
5.7 Power consumption affected by increasing transmission range (N = 2000; M = 500; p = 1/3) 135
5.8 Power consumption reduced with sparser projection matrices in both MTT and TCS in arbitrary networks (N = 2000; M = 500) 136
5.9 CS reconstruction error when reducing the probability of non-zero elements in sparse measurement matrices 136

6.1 M random neighborhoods are sampled in an arbitrary network with 500 sensors; transmission range $R = 9$ defines N neighborhoods in the graph G(V, E). 140
6.2 Tree formed by the greedy algorithm with 500 nodes and transmission range R = 14 to relay measurements from random sensors to the BS. 144
6.3 Comparison between full Gaussian measurement matrix and the one created by NeiCS 147
6.4 Total power consumption within difference number of neighborhoods; $N = 500$, $R_0 = 50$ and $R = 9$. 147
6.5 Total consumption transmits M measurements directly to the BS; $N = 500$, $R_0 = 50$ and $R = 9$. 148
6.6 Total power consumption when NeiCS transmits measurements directly to the BS; $N = 500$, $R_0 = 50$ and $R = 9$. 149
6.7 Total power consumption to relay multi-hop M measurements through intermediate nodes to the BS; $N = 500$, $R_0 = 50$ and $R = 9$. 149
6.8 Total power consumption to relay multi-hop M = 100 measurements through intermediate nodes to the BS with different transmission ranges; $N = 500$, $R_0 = 50$. 150

6.9 Total consumption with NeiCS when multi-hop relaying M measurements through intermediate nodes to the BS, compared to one-hop NeiCS; $N = 500$, $R_0 = 50$ and $R = 9$. 150

A.1 Real distances from any random node to the BS in a circle shape area arbitrary network 174

CHAPTER 1

INTRODUCTION

Wireless sensor networks (WSN) facilitate many application areas in the real world [1, 2]. The networks consist of small inexpensive sensors deployed randomly in geographical areas to monitor (e.g., temperature, humidity, acoustic, vibration) or detect events (e.g., intruders, chemical leak, vehicle passing, fire or flood detection). The sensors typically operate on battery power and they communicate wirelessly within a communication range, and have some level of computational capability. In monitoring applications, sensors send their readings to the sink or base-station (BS) for data analysis or mapping. Since the sensors operate on low power and may not be easily accessible by people, the network lifetime depends on sensor connections in the entire area.

There have been many data collection methods exploring different network topologies to minimize the total consumed power for such networks [3, 4]. These methods focus on balancing energy between sensors and spending less power on transmitting data. But, those methods still have to ensure delivery of all the sensory readings from sensors to the BS which could result in an energy imbalance, especially on the sensors close to the BS which play a role of relaying data.

Compressive sensing (CS) [5, 6, 7, 8] is a mathematical technique in signal processing focused on representing and reconstructing a signal through undersampling and optimization. CS allows for sampling and recovering a signal at a sampling rate lower than allowed by the Nyquist-Shannon sampling theorem based on knowledge about a signals sparsity. Since the sensory readings in WSNs are often highly cor-

related, CS can be considered as a potential framework for data collection in such networks [9, 10, 11]. With CS the BS only needs a small number of CS measurements collected from the networks compared to the total number of sensors to reconstruct all data from the sensing area. CS based data collection methods in WSNs have been shown to be energy efficient.

In this dissertation, four new CS based data collection methods are propose called CS based random walk (CSR) [12, 13, 14], Cluster-based CS data collection (CCS) [15, 16, 17, 18], Tree-based data gathering (TCS) [19] and Neighborhood-based data collection (NeiCS) [20], respectively. The methods exploit the existing network topologies and common connection between sensors in WSNs including random walk and tree routing, cluster network or undirected graph, and utilize CS to reduce the data collecting in such networks. The total power consumption for data transmission in the networks are analyzed and formulated. In each specific case, optimal points are suggested to minimize the total power consumption to prolong the network lifetime.

This dissertation is organized as follows. In Chapter 2, the background including the overview of WSNs and CS and the literature review are presented. The literature review section addresses existing work related to applying CS into WSNs and our proposed methods. In Chapters 3, 4, 5 and 6 we propose four data collection methods. In each method, the problem formulation, power consumption analysis and simulation results are provided. We further suggest optimal cases for each method to consume the least power in order to prolong the network lifetime. Conclusions and suggestions for future work for each data collection method are presented at the end of each chapter. Finally, Chapter 7 summarizes the dissertation and describes future work.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

2.1 Wireless Sensor Network Overview

2.1.1 Introduction

Wireless sensor networks (WSNs) facilitate many application areas. The network is the collaboration of a large number of sensor nodes which are deployed in a sensing area that needs to be observed. The sensors are typically low-cost, low-power, multi-functional, and small devices that can calculate/measure/process sensed data and communicate to each other or the base-station (BS) for data collection. Sensor nodes can be considered as randomly and densely deployed in a sensing area, inside a phenomenon, or close to it. They may be working in battlefield beyond the enemy lines, at the bottom of an ocean, inside a tornado, attached to animals or moving vehicles, in a biologically or chemically contaminated field, etc. They are usually small in size, sometimes even smaller than a cubic centimeter [21]. These sensors consume extremely low power [22] and the cost of each sensor could be less than one dollar [23].

Composed of a large number of sensors, WSNs may consist of many different types of sensors and may be able to accommodate different applications in diverse areas including military applications, environmental applications, health applications, etc. In military applications, as mentioned in [1, 24], sensors are deployed for battlefield surveillance, monitoring force, nuclear, biological and chemical attack detection and reconnaissance, etc. In environmental applications, a WSN can be deployed in a

forest to detect fire. Other applications include flood detection, animal tracking, bio-complexity mapping of the environment, and precision agriculture [25, 26, 27, 28]. In health applications, sensors can be used to track doctors and patients in a hospital, or to send patients' behaviors for help if needed [22, 29, 30]. Home applications of sensors have a lot of attention with smart or automation home. Almost the electronic devices in the house can be under control/adjust with optimize solutions [31, 32, 33]. Sensor networks are being developed to satisfy the human needs in present and in the future.

2.1.2 Challenges for Data Collection Method Design in WSNs

Energy saving is a critical issue for any WSN. Many routing, power management and data dissemination protocols have been proposed to reduce power consumption for such networks. Typically, WSNs contain hundreds or thousands of sensors. The sensors are often densely deployed in a sensing area that needs to be observed. The greater the number of sensors, the greater will be the accuracy of the observed information. As mentioned above, the cost for each sensor is typically very small due to restrictions, such as limited energy supply, limited computing power, and limited bandwidth of the wireless links connecting sensor nodes. Under the objectives of transmitting data to a data processing center in an energy-efficient manner, saving sensor energy consumption without losing accuracy, and preserving network lifetime, designing WSNs involves several difficult challenges.

Sensor node deployment: Sensor nodes can be either manually placed or randomly dropped in a sensing area to be observed. With manual deployment, data is collected at the sink with predetermined routes. Most networks involve randomized deployment with all sensors scattered randomly, creating an ad hoc routing infrastructure.

Balance and minimize energy consumption : In order to maintain the

network connections or to prolong the network lifetime, a network design should have an energy consideration to consume the least power. Inter-sensor communication is often over sort distances due to limited energy and bandwidth limitations. Transmitting data to the sink prefers multi-hop routing which normally consumes less energy than direct communication. Besides, designed routes should deplete equally power from all sensors deployed in the sensing area.

Data reporting method : Depending on the specific application and the time criticality of sensing data, data reporting in WSNs can be categorized as time-driven, event-driven, query-driven or a hybrid of some or all the methods. In the time-driven method, sensors collect and send their data periodically. In event-driven and query-driven methods, sensor nodes react when an event occurs and send data to the sink or the BS. Some networks use hybrid data delivery models to facilitate sensors.

Sensor capability : In many research studies, all the sensor nodes deployed in a sensing area are assumed to be homogeneous. This means that they have equal capacity in terms of pre-charged battery, communication and computation. But in some networks, sensors can be heterogeneous due to different roles. For example, there may be different types of data to be collected such as temperature, pressure and humidity. Furthermore, with pre-chosen cluster-heads (CH) in clustered networks, the CHs have higher power capacity than others since the burden of data transmission often falls on them.

Fault tolerance : Sensors may change from active status or fully functioned to be blocked due to lack of energy. The malfunctioned nodes are isolated but might still be used for relaying data in the network. The fully functioned nodes may cover the inactivated nodes and this failure would not affect the network in collecting data at the BS. This requires more capacity for each sensor to be able to work in a fault-tolerant network. Such as sensors might adjust transmitting power, signal rates, etc.

Sensor coverage : Due to the limitations of sensing range and transmission

range, sensors only can cover a limited region. Network coverage is highly dependent on the number of sensors, types of sensors, and coverage algorithms in order to solve the best coverage problem.

Network dynamics : In many applications, sensors may not be fixed all the time. Sensors may take turns to be mobile to collect data from static sensors. In some cases, the phenomenon may be mobile in tracking target applications. Dynamic network structures become flexible and challenge data routing algorithms. Dynamic networks may require additional energy, bandwidth, and so forth.

Data aggregation : Sensors may generate significant redundant data due overlapped regions covered by more than one sensors Similar packets from multiple nodes can be aggregated to reduce the number of transmissions in the network. Data aggregation or data fusion is the combination of data from different sources with sensed data being processed before it is sent ot the BS.

Quality of service : Beside the accuracy of data transmitting to the BS, latency is another condition for time-constrained applications. Data reporting time and quality of sensed data, critical in some applications, and conservation of energy, which is closely related to network lifetime, are in competition. Balancing quality of service to prolong network lifetime is a challenge for designing WSNs.

Other than the challenges and design issues listed above, other factors that must be considered in network design include sensor scalability, transmission media, connectivity, etc and others. Based on these design constraints, many data collection methods have been proposed in order to solve the issues and challenges. The methods are generally categorized as hierarchical routing, flat routing and local-based routing as follows.

2.1.3 Data Collection Method Protocols in WSNs

Hierarchical Routing

Hierarchical or cluster-based routing is utilized to perform energy-efficient routing in WSNs. In order to keep sensors in WSNs alive longer in their tasks, numerous clustering algorithms have been developed and refined in research. Sensors are divided into clusters regionally with an appropriate number of clusters. Each cluster chooses one of member leader, called the cluster head (CH), which will take the role to forward all aggregated data from the cluster to the sink or BS. The non-cluster head sensors only send their data to their own CHs.

There are many different clustering algorithms. Some focus on balancing energy for the networks, or distances between non-CH sensors and CHs and distances between CHs and BS; some others optimize the number of clusters in WSNs; and others identify energy efficient topologies for the network. The hierarchical data collection methods have general feature as follows.

- Cluster head (CH) may be pre-determined by a network designer before being deployed to a sensing area [34]. These CHs may have richer resources than non-cluster head sensors because they have to expend more power to transmit aggregated data from clusters to the BS while all other sensors only send their readings over a shorter distance to the CHs. This configuration can help to make a network operate longer but will be a challenge to deploy those CHs uniformly in the sensing area. However, the network is not flexible as intended or may be out of order when some CHs fail to function properly.

- Role of CHs can be exchanged (tolerant) by algorithm: in the most cases, CHs are some of the sensors deployed to sensing areas and determined after landing and clustering. It depends on a specific algorithm, a CH is chosen to satisfy a network's requirements and works until running out of power. To avoid the network becoming

disconnected as sensors deplete their power, especially CHs since they work for the longest distance with all cluster gathered data, in many algorithms, the role of being CHs will be changed frequently based on low energy notices within a cluster [35],[36], [37].

- Multi-hop or single hop routing within a cluster could be applied: In general, CHs often locate in the middle of clusters and minimize the total distance between non-CH and CHs smallest [38]. If clusters are large, the direct links between sensors and CHs may consume a lot of energy. In this case, we call single hop data transmission. To reduce energy consumption, multi-hop links enable sensors to transfer their data through adjacent nodes and finally reach a CH. These methods are mentioned in [39, 40].

- Clustering in WSNs with multiple objectives: under the common purpose of saving transportation cost and energy, and prolonging the network lifetime, the objectives can be load balanced between clusters, optimal number of clusters [36], fault-tolerance [37], increased connectivity and reduced latency. These aspects are addressed in the next sections.

K-means clustering algorithm: K-means is a very simple but effective algorithm in WSNs [41, 38, 42]. Suppose we have a set of sensor nodes $\underline{X} = [x_1\, x_2 \ldots x_N]$, and want them arranged into N_c clusters; each cluster has one cluster head (CH) at the center. The algorithm has only four simple steps as follows.

1) Randomly choose N_c centroid points for N_c clusters (or we can base on some prior knowledge); it really does not matter in choosing these positions at first. Calculate the cluster prototype matrix $\underline{M} = [m_1\, m_2 \ldots m_{N_c}]$.

2) Assign each object in the data set to the nearest cluster C_w, i.e.

$x_j \in C_w \quad if \,||\, x_j - m_w \,||\, < \,||\, x_j - m_i \,||$

$for\, j = 1, \ldots, N, \quad i \neq w\,, \quad and\, i = 1, \ldots, N_c$

In this step, we rearrange clusters based on distances between a CH and non-CH

sensors. A sensor will choose the closest CH to be with and new CHs have to be at the center of clusters.

3) Recalculate the cluster prototype matrix based on the current partition.

4) Repeat steps 2 - 3 until there is no change for each cluster;

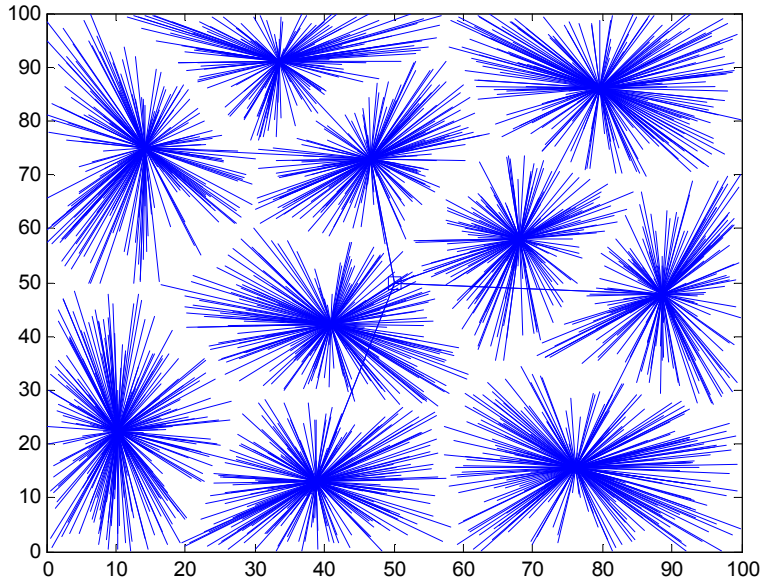


Figure 2.1: K-means clustering algorithm with $k = 10$ clusters

Figure 2.1 illustrates a WSN deployed in a square sensing area (100×100) with 500 sensors that are divided into 10 clusters by the K-means clustering algorithm. Besides some advantages, K-means still has some limitations. All the centroid points vary with different initial assignments. This means that each time we choose different centroid positions, we will get different converged points in the same network. According to [43], K-means cannot guarantee convergence to a global optimum. It is sensitive to outliers and noise and the definition of means limits the application to numerical variables. Some work on advanced K-means clustering can be found in [44] and [45].

Fuzzy C-means clustering algorithm According to [46, 47, 48], the FCM or Fuzzy C-means clustering algorithm works better than K-means; it may converge faster and dissipates energy less than K-means.

With FCM, one sensor can belong to more than one cluster head (CH) based on a

relationship called degree between them. If we have N sensor nodes which are divided into c clusters, the purpose of this algorithm is to minimize the total energy within clusters called J_m as follows.

$$J_m = \sum_{i=1}^{c} \sum_{j=1}^{N} u_{ij}^m d_{ij}^2, \tag{2.1}$$

where:

u_{ij} is node $j's$ degree that related to cluster i^{th}.

d_{ij} is the distance between the centroid of cluster i^{th} and node j.

$\underline{M} = [m_1\, m_2 \,..\, m_c]$ is the prototype matrix or cluster centroid points for our WSNs.

$m \in [1,\, \infty)$, in general, m is selected as 2.

This algorithm contains 4 simple steps:

1) Randomly select C central point for C clusters and choose a value for ϵ as a stop condition for our algorithm.

2) Calculate the matrix $U = [u_{ij}]$ by

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} (\frac{d_{ij}}{d_{kj}})^{\frac{2}{m-1}}}, \tag{2.2}$$

where $i = 1, \ldots, c$ and $j = 1, \ldots, N$.

3) Update to calculate the prototype for clusters in the next step by

$$m_i = \frac{\sum_{j=1}^{N} u_{ij}\, x_j}{\sum_{j=1}^{N} u_{ij}}, \tag{2.3}$$

where x_j is the position data of sensor j^{th}.

4) Repeat the steps 2 to 3 until $|| \, M^{(t+1)} - M^{(t)} \, || < \epsilon$.

The c central points are determined when the algorithm converges. These clusters are joint (overlapped) and not adaptable to our problem, but we can use the central points and then apply the idea of K-means to separate the clusters. In our simulation, this algorithm produces the same results to K-means.

Low-energy adaptive clustering hierarchy (LEACH) In LEACH [49, 50], nodes are organized into local clusters. In each cluster, all non-cluster head sensors

transmit their data to the CH. This CH collects data from all nodes in its cluster including its own data. The CH may perform processing on the data and then sends it to the BS. Based on this point, the energy burden falls on CHs because they have to transfer not only a larger amount of data than other nodes but also over a longer distance from its position to the BS. These CHs expend energy faster than non-CH nodes leading to earlier network disconnection.

LEACH is designed to expect that all nodes in WSNs have a chance to consume energy equally. Every sensor takes turns being a CH with a probability. After one round, the role of being a CH will be moved to another node.

* If we assume that all nodes start with equal energy available, then the probability $P_i(t)$ for becoming a CH can be calculated as follows.

$$P_i(t) = \begin{cases} \frac{N_c}{N - N_c(r \bmod \frac{N}{N_c})} & : \ C_i(t) = 1 \\ 0 & : \ C_i(t) = 0 \end{cases} \tag{2.4}$$

where

$C_i(t)$ is the indicator function determining whether or not node i has been a CH.

r is number rounds of our algorithm.

N_c: indicates the number of clusters using in our network.

And we also have the expectation of the number of cluster as

$$E[\#CH] = \sum_{i=1}^{N} P_i(t) \times 1 = N_c \tag{2.5}$$

* If the WSNs have sensor nodes with different amount of available energy, the probability will be calculated based on every node's energy as follows.

$$P_i(t) = min \left\{ \frac{E_i(t)}{E_{total}(t)} \times N_c, 1 \right\} \tag{2.6}$$

where $E_i(t)$ is the current energy of node i.

And the average number of CHs in this case is

$$E[\#CH] = (\frac{E_1(t)}{E_{total}} + \ldots + \frac{E_N(t)}{E_{total}}) \times N_c. \tag{2.7}$$

Summarizing, nodes in WSNs select themselves to be CHs based on two different cases mentioned above. In the cluster formation algorithm of LEACH, each non-CH sensor chooses its CH based on the minimum communication energy; the decision is based on the received signal strength of broadcast messages to determine which cluster that sensor should belong to. This step is similar to step 2 in the K-means clustering algorithm.

Energy Efficient Hierarchical Clustering (EEHC) The EEHC algorithm [51] is based on the Max-Min d-cluster algorithm [52] in which clusters are formed of collections of nodes that are up to d hops away from CHs. In EEHC, k is used to denote the number of hops to collect nodes to form clusters. A sensor becomes a CH itself with probability p and then sends advertisements to inform nodes within radius k about the new role. Any sensor receiving the advertising message will join the cluster. Those that do not receive any advertisement become CHs themselves resulting in two kinds of cluster heads, volunteer CHs and forced CHs.

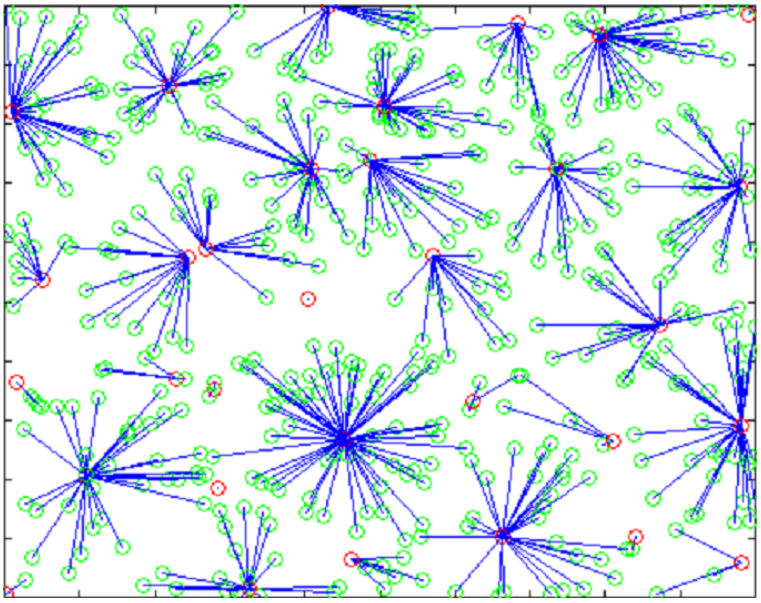


Figure 2.2: EEHC algorithm with single level of clustering

As shown in Figure 2.2, there are some nodes who are single and then become forced CHs. Energy is consumed quite differently at this single level. To balance

energy in the entire network, in the second phase, a multi-level clustering creates h levels of cluster hierarchy. There are h hops of connectivity between CHs and the BS. The algorithm also ensures that CHs far from BS can consume less energy by transferring data to another CH, not by sending directly to the BS. EEHC is a distributed, randomized clustering algorithm.

Fast Local Clustering service (FLOC) This clustering algorithm uses a wireless radio-model that has double bands to arrange sensor nodes in the entire network [53]. Sensors are in communication within each other using inner-band range. I-band radius is a unit distance can be determined. Similarly, we have outer-band range that nodes can communicate unreliably.

Nodes can be determined into a cluster whether they fall within i-band or o-band for a certain node. FLOC is a fast and scalable algorithm that creates non-overlapped clusters with approximately equal radius.

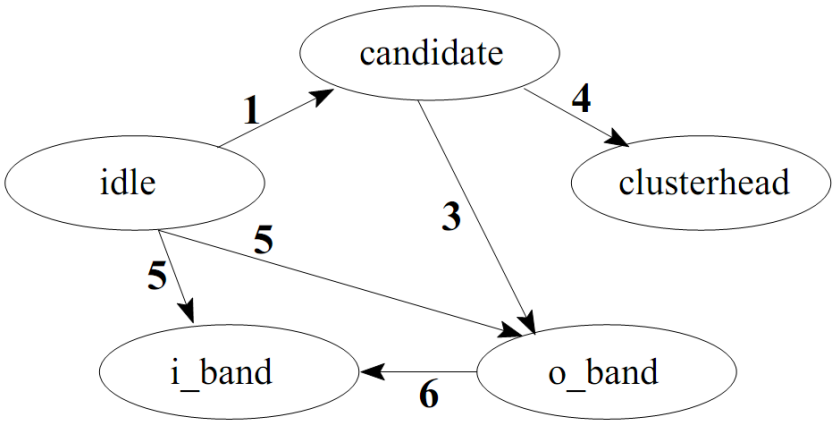


Figure 2.3: FLOC program consists of 6 actions

Hybrid Energy-Efficient Distributed Clustering (HEED) In HEED [54, 55], CHs are chosen from deployed sensors working in WSNs. The algorithm considers hybrid energy and cost while selecting CHs. The CHs are chosen based on their residual energy, not randomly. The energy can be estimated based on the consumption for sensing, processing and communication. The probability of becoming a CS can

be calculated as follows.

$$CH_{prob} = C_{prob} \times \frac{E_{residual}}{E_{max}}, \tag{2.8}$$

where $E_{residual}$ is the estimated current residual energy of a node; E_{max} is a full charged battery of each sensor. C_{prob} is a initial percentage value of CHs among all nodes.

During any iteration, every "uncovered" node elects to become a CH with CH_{prob}. A node selects its CH with the least communication cost. If it does not hear any CH, a sensor then selects itself to be a CH and sends an announcement message to its neighbors informing them about the changed status. Every sensor doubles its CH_{prob} and goes to the next iteration step. A node will finish HEED execution if its CH_{prob} reaches 1 that will make two status: Tentative ($CH_{prob} < 1$) and Final ($CH_{prob} = 1$).

Note that a node can be chosen to become a CH at consecutive clustering intervals if it has high residual energy and low cost. HEED is improved in [56] that considers nodes that did not hear from any CH.

Energy-Efficient Unequal Clustering Mechanism (EEUC) EEUC [57] attempts to balance energy consumption in the entire WSN to prolong the network lifetime. As shown in Figure 2.4, a multi-hop WSN has some clusters that consume energy differently. For more details, the ones closer to the BS consume more energy than the further ones because the ones nearer the BS have to transmit not only their own data but also the relayed data. To make every cluster deplete power equally, EEUC proposed the idea to create unequal clustering for WSN in which the clusters closer to the BS have smaller size than the ones far away from the BS.

In EEUC, data readings are transferred from clusters to the closest cluster to the BS and then the BS. With the unequal size clusters, the small clusters consume less energy for inter-cluster communication but larger energy on transferring data, which is the inverses of the larger sized clusters.

The role of CHs is rotated among sensors in each data gathering round through

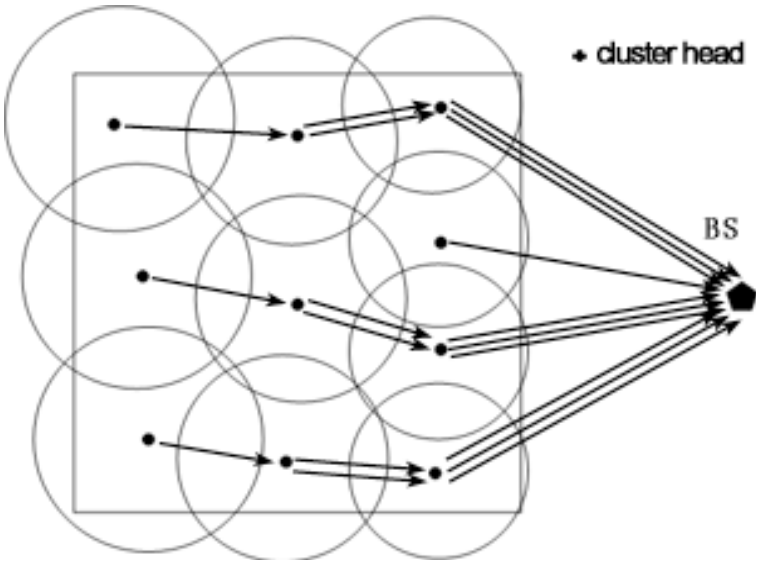


Figure 2.4: Unequal size clusters in EEUC clustering algorithm

the network. CHs are selected based on the residual energy of each node. Due to the different cluster sizes, there is a large number of small sized clusters which are close to the BS. And there are fewer clusters with large size far from the BS. Tentative CHs are selected at first with the same probability T. There is a competition range R_{comp} which is a function of the distance between nodes and the BS that decides that a node can becomes a tentative CH as follows.

$$s_i R_{comp} = (1 - c\frac{d_{max} - d(s_i, BS)}{d_{max} - d_{min}})R^0_{comp}, \tag{2.9}$$

where d_{max} and d_{min} denote the maximum and minimum distance between sensors and the BS. s_i is a tentative CH and $d(s_i, BS)$ is the distance between s_i and the BS. c is a constant coefficient between 0 and 1.

The final CHs are selected at the end of a competition algorithm. There are more CHs closer to the BS. Non-CH sensors choose the closest CH to join and then clusters are formed. A sleeping mode is mentioned in this algorithm to save energy. EEUC contributes a good proportion between cluster size and the distance from clusters to the BS.

Power-Efficient and Adaptive Clustering Hierarchy PEACH [58] pro-

poses the idea of saving energy for each sensor node that can distinguish the transmitters or other sensors broadcasting their data. Sensors in PEACH can avoid additional overheard. PEACH is applicable in both location-unaware and location-aware networks and this helps it become dynamic and effective to prolong network lifetime.

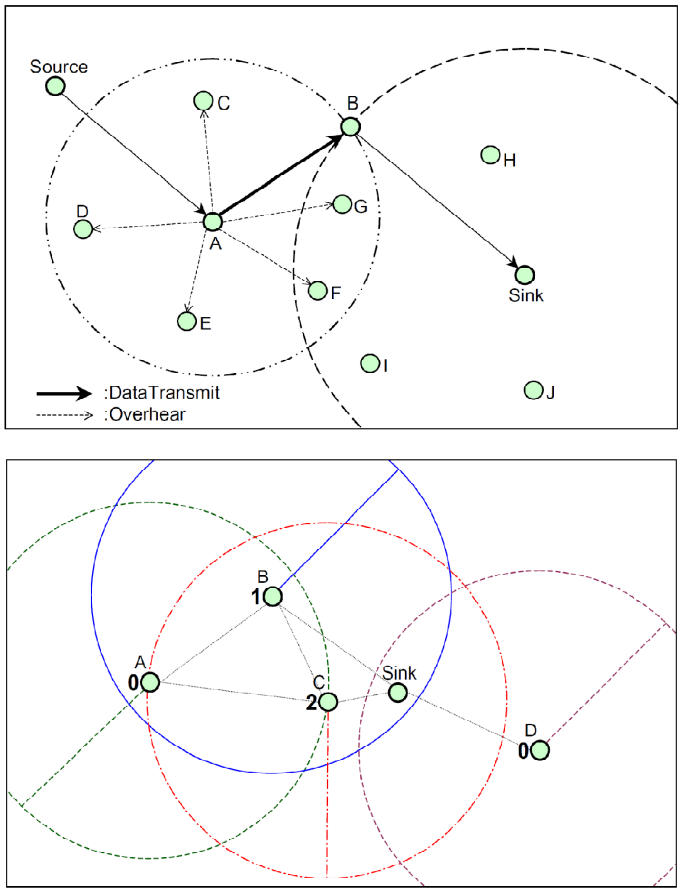


Figure 2.5: Packet transmission and global transmission schedule on location-aware in PEACH

PEACH has two definitions: $NodeSet(N_iN_j)$ is a set of sensor in a circle with center point N_i and N_j is a radius distance far from N_i. All sensor nodes in $NodeSet$ can overhear messages from N_i to N_j. $ClusterSet(N_iN_j)$ is a set that includes both nodes and the sink or BS.

In the location-unaware algorithm, when a node receives a packet, if it is noticed

to be the destination, it becomes a CH during T_{delay} and after $T_{delay,}$, it transmits the packet to the next hop; if the node is not a CH, it will join the cluster of the destination node.

When the location is aware, each node knows the locations of all the nodes. A node can calculate a global transmission schedule without communicating with the others. The farthest node away from the sink node must initiate the packet transmission.

Multi-hop Routing Protocol with Unequal Clustering (MRPUC) This method [59] is quite similar to EEUC [57]; they both divide a WSN into unequal clusters with the same purpose which is to balance energy consumption of the network. They are also multi-hop methods to prolong the network lifetime. MRPUC can be considered as an updated version of EEUC. It ensures that after clustering, there is no sensing hole in the entirety of the network. This means that all sensors belong to clusters and all sensor readings are sent to the BS. There are three phases in MRPUC: Cluster setup, inter-cluster multi-hop routing formation and data transmission.

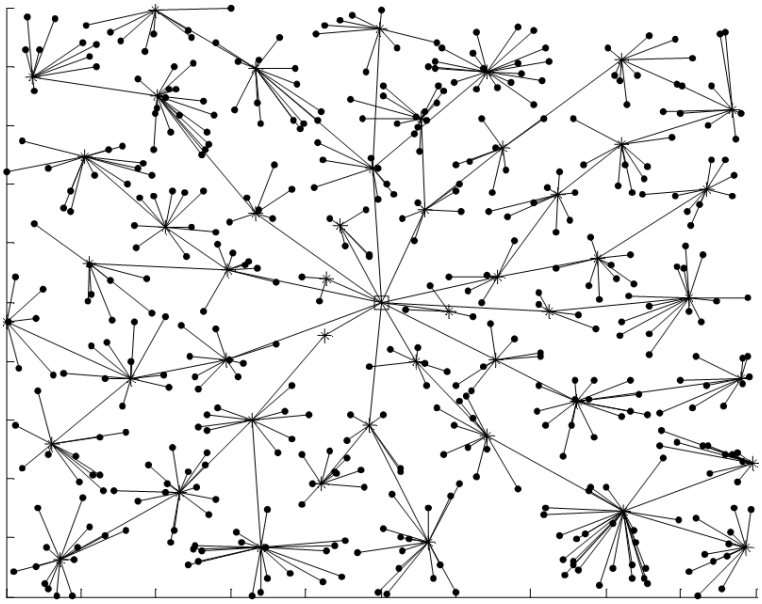


Figure 2.6: Cluster structure of a network with MRPUC

The BS is assumed to be at the center of the sensing area. Each node calculates approximately its distance d to the BS based on a broadcast message sent by the BS.

We have the maximum distance d_{max}, and the maximum and minimum radius are predefined as R_{max} and R_{min}, respectively. The cluster radius R^i of node i is set as

$$R^i = \frac{d(i, BS)(R_{max} - R_{min})}{d_{max}} + R_{min}. \tag{2.10}$$

Each node gathers correlative information of its neighbors and elects a node having maximum residual energy to be the CH. Clusters closer to the BS have smaller size than the ones father from the BS to balance between inter-cluster communication energy and energy to transmit data to the BS.

In the multi-hop routing phase, each CH has to choose another CH to transfer its data following a rule to minimize communication cost. The cost depends on two factors: relay energy consumption and residual energy of neighbor CHs. After each CH has chosen a parent node, an inter-cluster tree rooted at the BS is constructed.

During a round, a CH aggregates data packets into a single packet and then sends to its parent node that will forward the packet to the BS.

S-Web: An Efficient and Self-organizing WSN Model This algorithm [60] organizes sensors into clusters based on their geographical location without requiring those sensors to have GPS or any localization mechanism supported.

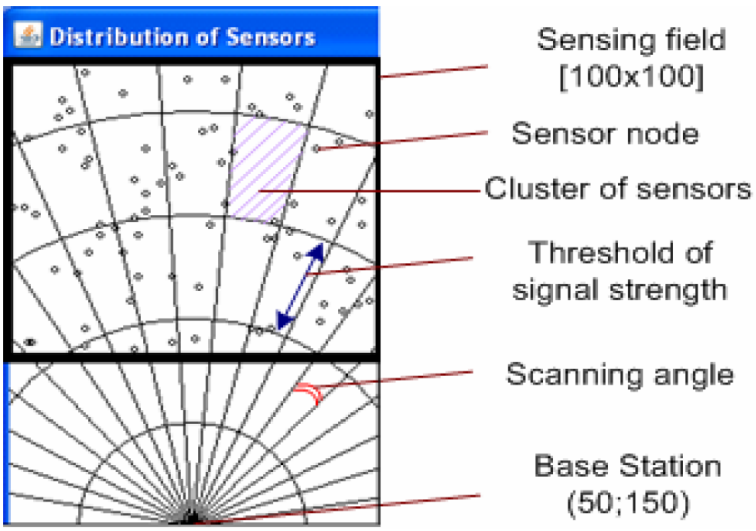


Figure 2.7: S-Web clustering algorithm

The geographical location is determined by two factors: distance and angle. Sensors that receive beacon signals from the BS can measure those factors.

In S-Web, a node chosen as a CH has highest residual energy to work as a router for the cluster that it belongs to. The role of being CH will be rotated in the same cluster to balance load and also energy.

When a packet is transmitted to a cluster, non-CH nodes will transfer to the CH, and this CH will forward the packet to a neighbor closer to the BS. The closeness is determined by the two factor mentioned above.

Unequal cluster size is considered in this algorithm to balance energy consumption in the network.

Hybrid Energy Effective Clustering Hierarchical Protocol (HEECH) HEECH [61] is a multi-hop algorithm that has been shown to increase network lifetime by about 56% and 9% compared to LEACH [50] and HEED [55], respectively.

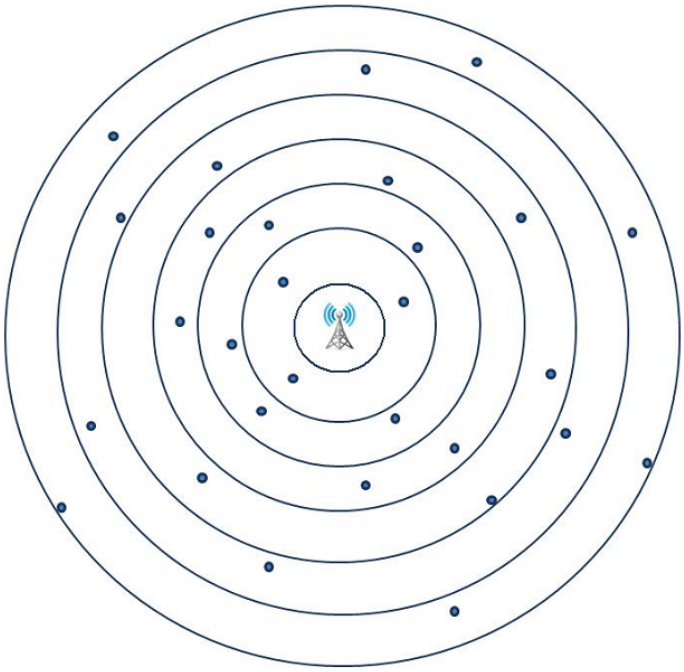


Figure 2.8: HEECH divides WSN into six tracks with the same width

HEECH works to solve the unbalanced energy consumption problem not only by choosing CHs but also dividing sensing area. There are five phases to the algorithm.

The first phase is dividing the sensing area into six tracks with the same width, shown in figure 2.8, called the configuration phase.

The announcement phase is the second one to choose CHs for each track based on the residual and the maximum energy of each node as follows.

$$Prob_{CH} = \alpha(\frac{E_r}{E_m}) + \beta(\frac{D_{LCH-BS} - D_{LCH-HCH}}{\frac{n*R}{6}}), \tag{2.11}$$

where α is impact factor of energy and β is impact factor of distance.
E_r is remaining energy of sensor node and E_m is maximum or initial energy of sensor.
R is network radius.
$D_{LCH-HCH}$ is the distance between the desired node and the high level CH node closer to it.
D_{LCH-BS} is distance between the desired node and the BS.

The third and the forth phase are called cluster formation and schedule creation, respectively. They are similar to LEACH but HEECH considers the distance between CHs and BS in multi-hop transmissions and this point solves the unbalancing energy problem.

In the fifth phase, called data transmission, the low level CH sends its data to the high level one.

Flat Routing

In flat WSNs, every sensor typically plays the same role. The sensors also collaborate to perform the sensing tasks in such networks. Due the large number of sensors deployed in the sensing area, it is not feasible to assign global identifiers to all the sensor nodes. This leads to the difficulty of collecting of specific sets of nodes to be queried. This consideration has led to data-centric routing, which is different from traditional address-based routing where routing links are created between addressable nodes managed in the network layer. In data-centric routing, the BS sends queries

to certain regions and waits for data from the sensors in the selected regions. We describe some algorithms as follows.

Flooding and Gossiping: Flooding and gossiping [62, 63] are two classical methods to relay data in WSNs. In the flooding method, each sensor keeps sending broadcast messages to its neighbors within a sensor transmission range until it receives data packets or the maximum number of hops for the packet is reached. On the other hand, gossiping is a slightly enhanced version of flooding where the receiving node sends the packet to a randomly selected neighbor. This neighbor will pick another random neighbor to forward the data to, and so on.

Sensor Protocol for Information via Negotiation (SPIN): SPIN is a family of negotiation-based information dissemination protocols suitable for WSNs. In these protocols, all the information is disseminated at each sensor to every node in the network [64, 65]. As assumed, all the sensor nodes could be able to be the sink node or BS. In SPIN, sensor nodes name their data using high-level data descriptors, also called meta-data, to eliminate the transmission of redundant data throughout the network. Before transmission, meta-data are exchanged among sensors via a data advertisement mechanism. Each sensor upon receiving new data, advertises it to its neighbors and interested neighbors. Sensors which do not have data retrieve the data by sending a request message.

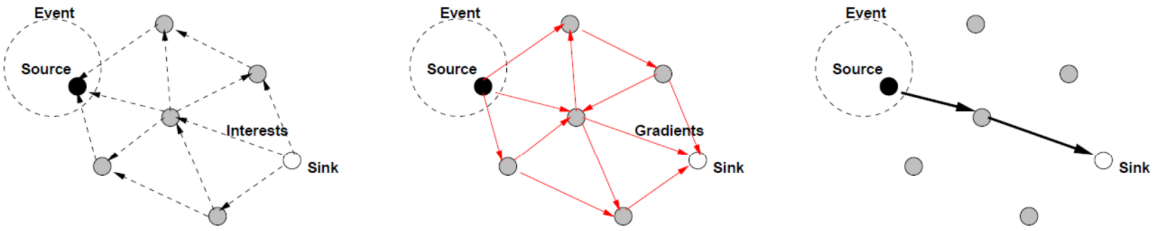
Using meta-data names, sensor nodes negotiate with each other about the data they process. These negotiations ensure that nodes only transmit data when necessary and never waste energy on useless transmissions. SPIN's meta-data negotiation solves the problems of flooding such as redundant information passing, overlapping of sensing area and resource blindness.

One of the advantages of SPIN is that topological changes are localized since each node needs to know only its neighbors with single-hop communications. However, the data advertisement mechanism cannot guarantee delivery of data. For example,

in considering the application of intrusion detection where data should be reported over periodic intervals, and assume that nodes interested in the data are located far away from the source node, etc. such data would not be delivered to the destination.

Directed Diffusion: This is a data-centric (DC) routing algorithm in which all communication is for named data. All nodes in a directed diffusion-based network are application-aware [66]. Data generated by sensor nodes is named by attribute-value pairs. The main idea of the DC paradigm is to combine the data coming from multiple sources by eliminating redundancy, minimizing the number of transmissions, thus saving network energy and prolonging the network lifetime.

In DC, sensor nodes detect events and create gradients of information in their respective neighborhoods. The BS requests data by broadcasting interests. An interest describes a task required to be done by the network. The interest diffuses through the network hop by hop from neighbor to neighbor. As the interest is broadcast in the network, gradients are set up to draw data satisfying the query toward the requesting node. Each sensor that receives the interest sets up a gradient toward the sensors from which it receives the interest. This process continues until gradients are set up from the sources back to the BS.



(a) Interest propagation (b) Initial gradients set up (c) Data delivery along reinforced path

Figure 2.9: An illustration of Directed Diffusion in WSN

A gradient specifies an attribute value and a direction. The strength of the gradient may be different due to different neighbors that results in different information flow. Figure 2.9 depicts an example of directed diffusion with sending interests, set-

ting up gradients and data dissemination, respectively. When interests fits gradients, paths for information flow are formed from multiple paths, and then the best paths are reinforced to prevent further flooding according to a local rule. Data is aggregated to reduce communication cost. The goal is to find an aggregation tree to transmit data from source nodes to the BS. The BS periodically refreshes and resends the interest when it starts to receive data from the sources.

Rumor Routing: This is a logical compromise between flooding queries and flooding event notifications. The main idea in rumor routing [67] is to route the queries to the nodes that have observed a particular event rather than flooding the entire network to retrieve information about the occurring events. Each node maintains a list of its neighbors as well as an events table. When a sensor node generates a query for an event, the ones that know the route may respond to the query by inspecting its event table. Note that any node may generate a query, which should be routed to a particular event. It is not necessary to flood the whole network. Rumor routing maintains only one path between source and destination as opposed to directed diffusion where data can be routed through multiple paths at low rates.

Rumor routing can only perform well when the number of events is small. For a large number of events, the cost for maintaining agents and event tables in each node becomes infeasible if there is not enough interest in these events from the BS. In addition, the overhead associated with rumor routing is controlled by different parameters used in the algorithm such as time to live pertaining to queries and agents.

Minimum Cost Forwarding Algorithms (MCFA): In this method [68], each sensor should know the least cost path estimated from itself to the BS. The BS broadcasts a message with the cost set to zero, while every sensor initiates its least cost to the BS to infinity. Each sensor node, upon receiving the broadcast message originated at the BS, checks to see if the estimate in the message plus the link on which it is received is less than the current estimate for updating. In this case, the

nodes far away from the BS get more updates than the ones closer to the BS. Once the cost field is established, any sensor can deliver the data to the sink along the minimum cost path. Each intermediate node forwards the message only if it finds itself on the optimal path for this message based on the messages cost states.

Gradient-based Routing (GBR): In GBR [69, 70], each sensor calculates a parameter called the height of the node, which is the minimum number of hops to reach the BS. The difference between a sensor's height and that of its neighbor is considered the gradient on that link. A packet is forwarded on a link with the largest gradient. GBR uses auxiliary techniques such as data aggregation and traffic spreading in order to uniformly divide the traffic over the network.

In GBR, three different data dissemination techniques have been discussed. Stochastic scheme, where a sensor picks one gradient randomly when there is more than one next hops that have the same gradient. In energy-based scheme, the sensor increases its height when its energy drops below a certain threshold. In the stream-based scheme, new streams are not routed through nodes that are currently part of the path of other streams. The main objective of these schemes is to balance the traffic in the network, to prolong the network lifetime.

Information-driven Sensor Querying and Constrained Anisotropic Diffusion Routing: The paper [71] describes two techniques, information-driven sensor querying (IDSQ) and constrained anisotropic diffusion routing (CADR), for energy-efficient data querying and routing in ad-hoc sensor networks. The idea is to query sensors and route data in a network in order to maximize the information gain, while minimizing the latency and bandwidth.

In CADR, each node evaluates an information objective and routes data based on the local information gradient and end-user requirements. The information utility measure is modeled using standard estimation theory. CADR diffuses queries by using a set of information criteria to select which sensors can get the data. This is achieved

by activating only the sensors that are close to a particular event and dynamically adjusting data routes.

In IDSQ, the querying node can determine which node can provide the most useful information with the additional advantage of balancing the energy cost. While IDSQ provides a way of selecting the optimal order of sensors for maximum incremental information gain, it does not specifically define how the query and the information are routed between sensors and the BS. Therefore IDSQ can be seen as a complementary optimization procedure.

Active Query Forwarding (ACQUIRE): In ACQUIRE [72] an active query is forwarded through the network, and intermediate nodes use cached local information (within a look-ahead of d hops) in order to partially resolve the query. When the query is fully resolved, a completed response is sent directly hack lo the querying node.

Similar to COUGAR [73], ACQUIRE views the network as a distributed database where the complex queries can be further divided into several sub-queries. The operation of this method can be addressed as follows. The BS sends a query to other sensors. During this progress, each sensor node tries to respond to the query partially by using its pre-cached information and then forwards it to another node. If the pre-cached information is not up-to-date, the nodes gather information from their neighbors within a lookahead of d hops. Once the query is resolved completely, it is sent back through either the reverse or shortest path to the BS. Hence, ACQUIRE can deal with complex queries by allowing many sensors to send responses.

ACQUIRE selects the next node to forward a query that has two options, chooses randomly or selects based on maximum potential query satisfaction. The problem of selecting the next node is also mentioned in CADR [71].

Energy-Aware Routing:The goal of this method [74] is try to find the minimum energy path to optimize energy usage at a node to increase the network lifetime. It

maintains a set of paths instead of maintaining or enforcing one optimal path at higher rates. These paths are maintained and chosen by means of a certain probability. The probability depends on how low the energy consumption is that each path can achieve. By having paths chosen at different times, the energy of any single path will not deplete quickly. In addition, the energy is dissipated equally among all sensors that balances the energy consumption in the network and prolonging the network lifetime.

Routing Protocols with Random Walks: This method has been mentioned in many research studies including [75, 76, 77, 78, 79]. The objective of random walk based data gathering methods is to achieve load balancing in a statistical sense. Since all sensors deplete energy equally, the connections in the network created by the sensors can last longer.

Random walk (RW) on a graph can be modeled as a Markov Chain as mentioned in [75]. Walking steps jump from node to node randomly based on probabilities generated based on sensor neighborhoods, called transition probabilities. Specifically, the next node j in the sequence is selected from the set of neighbors of the previous node i in the sequence with probability P_{ij}. The probabilities form a transition matrix $P = [P_{ij}]_{N \times N}$.

For example, a simple random walk, at time k and at vertex i needs to move to one of its adjacent vertices j with a probability P_{ij}. The transition probability is calculated as follows

$$P_{i,j} = P(X_{k+1} = j \,|\, X_k = i) = \begin{cases} \frac{1}{d(i)}, & if\ (i,j) \in E \\ 0, & others \end{cases} \tag{2.12}$$

where $d(i)$ denotes the degree of vertex i.

Sensed data from the network is forwarded to the BS through intermediate nodes which are chosen randomly as mentioned above. RW routing does not need global or local information. Furthermore, the methods are suitable to apply sleeping schedules

and fault tolerance schemes that can prolong the network lifetime.

Location-Based Routing

There are many data collection methods for WSNs that require location information for sensors to calculate the data transmission distance between two sensor or sensors and the BS. Based on that, the energy consumption for data transmission can be estimated. The local information of sensors can be used in routing in an energy-efficient manner. For instance, the query can be diffused only to a particular region that can reduce the number of transmissions between sensors.

MECN and SMECN: Minimum energy communication network [80] sets up and maintains a minimum energy consumption for a WSN by utilizing low power GPS [81]. The paper describes a distributed network protocol optimized for achieving the minimum energy for randomly deployed ad-hoc networks. The network protocol not only maintain a globally connected network in spite of possible module failure, but also defines the major power management strategy based on low-power RF transceiver design. Although the algorithm is designed for mobile networks, it is applicable to WSNs.

MECN identifies a relay region for every node. The relay region consists of nodes in the surrounding area where transmitting through those nodes is more energy efficient than direct transmission. The main idea of MECN is to find a sub-network, which will have a smaller number of nodes and require less power for transmission between any two particular nodes. In this way, global minimum power paths are found without considering all the nodes in the network.

MECN is self-reconfiguring and thus can dynamically adapt to node failure or deployment of new sensors. In SMECN [82] which is an extension to MECN, possible obstacles between any pair of nodes are considered. The network is always connected as in MECN, but SMECN consumes less energy than MECN.

Geographic Adaptive Fidelity (GAF): GAF [83] is an energy-aware location-based routing algorithm designed primarily for mobile ad-hoc networks. It could be applicable to WSNs as well. The sensing area is divided into fixed regions which form a virtual grid. Inside each region, sensor nodes collaborate with each other to play different roles, sleep and awake for example. Each node uses its GPS to associate itself with a point in the virtual grid. Nodes associated with the same point on the grid are considered equivalent in terms of the cost of packet routing. Such equivalence is exploited in keeping some nodes located in a particular grid area in a sleeping state in order to save energy.

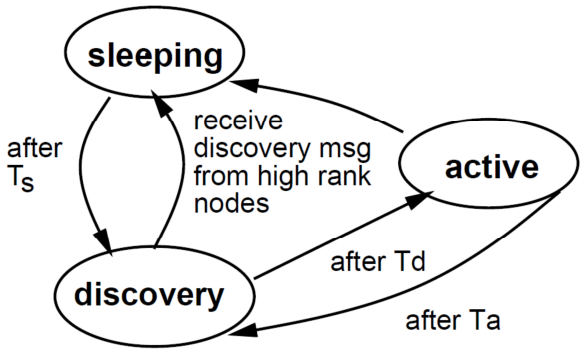


Figure 2.10: State transitions in GAF

There are three states defined in GAF as shown in Figure 2.10, discovery for determining the neighbors in the grid; active for reflecting participation in routing; and sleep when the radio is turned off for energy saving. Nodes change states from sleeping to active in turn so that the load is balanced to prolong the network lifetime. GAF strives to keep the network connected by keeping a representative node always in active mode for each region on its virtual grid. GAF performs at least as well as a normal ad-hoc routing protocol in terms of latency and packet loss and increases the lifetime of the network by saving energy. GAF maybe considered a hierarchical protocol, where the clusters are based on geographic location.

Geographic and Energy Aware Routing (GEAR): GEAR [84] uses energy-

aware and geographically informed neighbor selection heuristics to route a packet toward the destination region. The key idea is to restrict the number of interest in directed diffusion by only considering a certain region rather than sending the interest to the whole network. GEAR can conserve more energy than directed diffusion.

In GEAR, each sensor keeps an estimated cost and a learning cost of reaching the destination through its neighbors. The estimated cost is a combination of residual energy and distance to destination. There are two phases: (1) *Forwarding packets toward the target region*: Upon receiving a packet, a node checks its neighbors to see if there is one neighbor that is closer to the target region than itself. If there is no more than one, the nearest neighbor is chosen as the next hop. If they are all further than the node itself, one of the neighbors is picked to forward the packet based on the learning cost function. (2) *Forwarding the packets within the region*: If the packet has reached the region, it can be diffused in that region by either recursive geographic forwarding or restricted flooding.

GEAR is compared to GPSR [85] in solving the problem of holes. GEAR not only reduces energy consumption for the route setup but also outperforms GPSR in terms of packet delivery.

MFR, DIR, and GEDIR: These protocols deal with basic distance, progress, and direction-based methods [86]. The key ideas are forward and backward directions. A source node or any intermediate node will select one of its neighbors according to a certain criterion. The routing methods that belong to this category are Most Forward within Radius (MFR), Geographic Distance Routing (GEDIR), that is a variant of greedy algorithms, the two-hop greedy method, alternate greedy method, and DIR (a compass routing method).

GEDIR is a greedy algorithm that always moves the packet to the neighbor of the current vertex whose distance to the destination is minimized. The algorithm fails when the packet crosses the same edge twice in succession. In most case, MFR and

greedy methods have the same path to the destination. In DIR, the best neighbor has the closest direction toward the destination. GEDIR and MFR are loop-free, while DIR may create loops unless past traffic is memorized or a time-stamp is enforced [86].

Greedy other Adaptive Face Routing: This is a new geometric ad-hoc routing algorithm combining greedy and face routing, named GOAFR [87], which combines greedy and face routing. It is shown that GOAFR is asymptotically optimal with respect to the competitive ratio with the shortest path.

The greedy algorithm always picks the neighbor closest to a node to be next for routing. However, it can easily be stuck at some local minimum, for example, no neighbor is closer to a node than the current node. It was shown that GOARF can achieve both worst-case optimality and average-case efficiency. Based on the simulation results of GOARF, there are several ways to further improve the average-case performance. It was also shown that GOAFR outperforms other prominent algorithms, such as GPSR and AFR.

SPAN: SPAN is an energy-efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks [88] that selects some nodes as coordinators based on their positions. SPAN adaptively elects coordinators from all nodes in the network, and rotates them in time. The coordinators form a network backbone used to forward messages. A node should become a coordinator if two neighbors of a non-coordinator node cannot reach each other directly or via one or two coordinators. New and existing coordinators are not necessarily neighbors, which in effect makes the design less energy-efficient because of the need to maintain the positions of two or three-hop neighbors in the complicated SPAN algorithm. SPAN coordinators stay awake and perform multi-hop packet routing within the ad hoc network, while other nodes remain in power-saving mode and periodically check if they should awaken and become a coordinator.

SPAN not only preserves network connectivity, it also preserves capacity, decreases

latency, and provides significant energy savings. The amount of energy that SPAN saves increases only slightly as density increases. This is largely due to the fact that the current implementation of SPAN uses the power saving features of 802.11, in which nodes periodically wake up and listen for traffic advertisements.

2.2 Introduction to Compressive Sensing

2.2.1 Introduction

Compressed Sensing (CS) [5, 6, 7, 8] is considered a very strong tool in signal processing. CS builds upon the fundamental fact that we can represent many signals using only a few non-zero coefficients in a suitable basis or dictionary. The sparse representation will be mentioned later on but we call the signals to be applied CS sparse signals with a small number of non-zero elements. Based on this representation, with a very few measurements, the signal will be recovered precisely. This is the key point of CS.

CS differs from Nyquist [89], which requires a high rate of sampling; CS shows its ability to recover perfectly with a small number of measurements, also called projections. This is a turning point for many applications which can apply CS. Reducing measurements means that the cost for signal processing and data transmission can also be reduced significantly. These costs are main components in any network design. Many application areas including imaging, video, medical imaging, remote surveillance, spectroscopy, etc. can apply CS and become more effective with CS. These applications will be mentioned again in detail in the signal recovery section to show how the signals are sampled and reconstructed.

We consider the signals to which we apply CS are sparse and have k non-zero elements out of n elements in the data vector length n, where $k \ll n$. The second condition we assume in order to apply CS is that the signal must be compressible. Both sparse and compressible signals can be represented with high fidelity by preserving

the values of the largest coefficients of the signal. This point helps the reconstruction process to recover the original signal with only a certain number of CS measurements.

There are some points in CS we need to consider to differentiate from classical sampling methods. First, CS focuses on measuring finite-dimensional vectors in R^n while other method such as Nyquist samples continuous and infinite signals. Second, other than sampling the signal at specific points in time, CS systems acquire measurements in the form of inner products between the signal and a more general test function. The inner products are very important to mutual coherence between the ϕ and ψ matrices that will be mentioned in the next section. Third, in CS, signal recovery is achieved using nonlinear methods, while in the Nyquist-Shannon framework, recovery is based on interpolation. These are distinct differences between traditional Nyquist sampling and CS.

In this section, the overview of CS theory with fundamental basic and updated techniques is addressed. Vector spaces, sensing matrices and signal reconstruction techniques will be mentioned respectively.

2.2.2 Vector Spaces

Background

A vector space over a field F is a set of vectors V and a set of scalars F, vector addition $(+)$, scalar multiplication $(.)$ satisfy

1) $(V, +)$ is a commutative group

2) Scalar multiplication satisfies

(i) Closed

$a \in F,\ X \in V$

$a.X \in V$

(ii) Associativity

$(ab)X = a(bX)$

(iii) Unity

$1.X = X$

3) Distribution

(i) $a(X + Y) = aX + aY$

(ii) $(a + b)X = aX + bX$

Definition 1:

The span $\{V_1, \ldots V_m\}$ is a set of all linear combinations of $V_1, \ldots V_m = a_1V_1 + a_2V_2 + \ldots + a_mV_m$. Where $a_1, a_2, \ldots, a_m \in F$

Definition 2:

$\{V_1, \ldots V_m\}$ is linearly dependent when one of the vector can be expressed as a linear combination of others.

Definition 3:

A basis of a vector space V is a linearly independent set of vector that span V, $Span(Basis) = V$.

Definition 4:

- The dimension of a vector space is the number of a vectors in its basis, $dim(V) = |basis|$.

- Relation between $dim(V)$ and $|V|$: for a vector space, we have dimension of V $dim(V)$ = number of vectors in basis = $|basis| = k$.

$$Basis = \left\{ \begin{array}{c} V_0 \\ V_1 \\ \vdots \\ V_k \end{array} \right\}$$

$V = a_0V_0 + a_1V_1 + \ldots + a_{k-1}V_{k-1}$; $a_i \in GF(2)$. We can have 2^k different combinations, then we have the relationships: $|V| = 2^{dim(V)}$ or $dim(V) = log_2|V|$

Definition 5:

A subset $S \subseteq V$ of a vector space V is a subspace only if S is a vector space.

Definition 6:

The inner product: Let two vectors $\underline{X} = (x_0, \ldots, x_{n-1})$ and $\underline{Y} = (y_0, \ldots, y_{n-1})$ are in a vector space over F. The inner product is defined as follows

$$< \underline{X}.\underline{Y} >= \underline{X}.\underline{Y} = \sum_{i=0}^{n-1} x_i y_i$$

Properties of inner product:

(i) Commutative: $\underline{X}.\underline{Y} = \underline{Y}.\underline{X}$ or $a(\underline{X}.\underline{Y}) = (a\underline{X}).\underline{Y}$

(ii) Distributive: $\underline{X}.(\underline{Y} + \underline{Z}) = \underline{X}.\underline{Y} + \underline{X}.\underline{Z}$

Normed vector spaces

In discrete and finite domains, our signals can be viewed as vectors in an n-dimensional Euclidean space R^N. We have the definition of the norm p that will be used very often in the CS process, where $p \in [1, \infty]$ as follows.

$|| X ||_p = (\sum_{i=1}^{N} | x_i |^p)^{\frac{1}{p}}$

or $l_p = (| x_1 |^p + | x_2 |^p + \ldots + | x_N |^p)^{1/p}$

We also have:

$l_0 = || X ||_0$ = number of non-zero elements

and $l_\infty = || x ||_\infty = max | x_i |$.

We have some figures that show the Unit ball in some normed vectors:

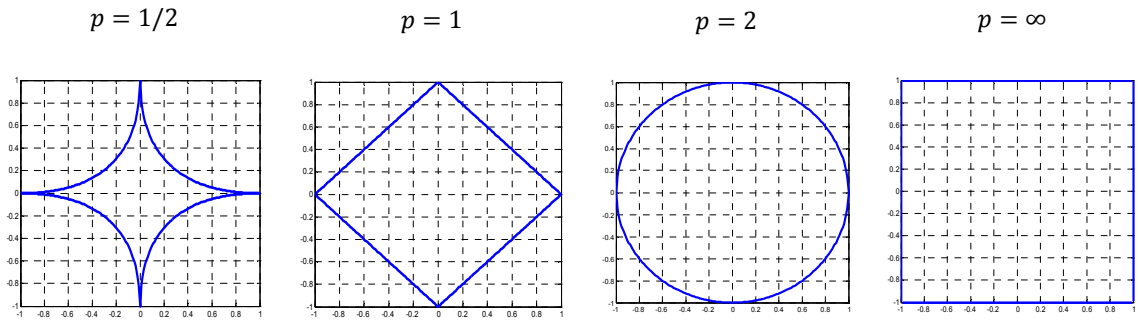


Figure 2.11: Some common normed vectors

Sparse vectors

We have some definitions as follows.

Definition 1: A vector signal $\underline{X}$ is k-sparse if it has k non-zero coefficients as $|| \underline{X} ||_{l_0} = k$.

Definition 2:

Sparsity rate S of a k-sparse vector $X \in R^N$ is defined as $S = \frac{k}{N}$, where n is the length of vector X.

Definition 3: A signal $X \in R^N$ is approximately k-sparse if it has $k \ll N$ large coefficients while the remaining coefficients are small but not necessarily zero.

****Compressible signals***:

Our signal must be compressible if CS is to be applied. Compressible signals are well approximated by sparse signals in the same way that signals close to a subspace are well approximated by the first few principle components.

We can quantify the compressibility by calculating the error incurred by approximating a signal X by some $\hat{X} \in \sum_k$: $\delta_k(X)_p = min_{\hat{X} \in \sum_k} || X - \hat{X} ||_p$.

We have another way to judge compressibility of a signal; it is the rate of decay of their coefficients which will be addressed as follows: If $\underline{X} = \Psi \underline{\theta}$, we sort the coefficients θ_i such that $|\theta_1| \geq |\theta_2| \geq \cdots\cdots \geq |\theta_N|$. We can say that the coefficients obey a power law decay if there exist constants θ_1 and q such that $|\theta_i| \leq \theta_1 i^{-q}$. It is clear that the larger q, the faster the magnitudes decay, and the more compressible a signal is. Because the magnitudes of their coefficients decay so rapidly, compressible signals can be represented accurately by $k \ll N$ coefficients.

2.2.3 Sensing Matrices

Also called measurement matrices or projection matrices, sensing matrices are the most important ones to decide how effective CS works. It is known as Φ matrix in

CS's representation in the equation as follows.

$$\underline{Y} = \Phi \underline{X}, \tag{2.13}$$

where $\underline{Y}$ is measurement vector, $Y \in R^M$ and Φ is $M \times N$ sensing matrix.

We will show the structure of these matrices and some properties that affect directly CS process in the following sections.

Structure of measurement matrices

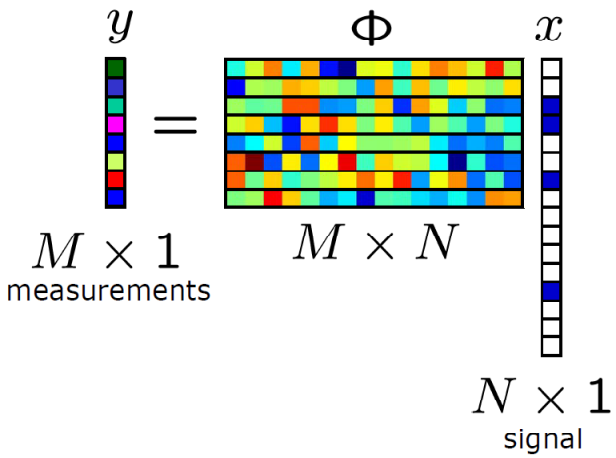


Figure 2.12: Random projection matrix

As mentioned, CS can be applied to signals that can have sparse representation in a proper basis. We have:

- CS sampling (encoding): $\underline{Y} = \Phi \underline{X}$

- CS recovery (decoding): $\hat{\underline{X}} = arg\,min||\underline{X}||_{l_1}\ subject\,to\ \underline{Y} = \Phi \underline{X}$

or $\hat{\underline{\theta}} = arg\,min||\underline{\theta}||_{l_1}\ subject\,to\ \underline{Y} = \Phi\Psi\underline{\theta}$, where $\underline{X} = \Psi\underline{\theta}$

As shown in Figure 2.12, measurement matrix Φ collects data from sensor nodes for each measurement that is recorded as each row of the matrix. We have M rows corresponding to M times collecting data or M measurements.

Coefficients of the Φ matrix are generated differently depending on which matrix is chosen to be applied. There are some common matrices often used with CS.

* Gaussian matrix: all coefficients in this matrix are randomly generated following Gaussian coefficients with zero mean and covariance $(1/M)$.

$\Phi_{ij} \sim N(0, \delta^2 = 1/M)$

$M > Cklog(N/k) = O(klogN)$

* Binary matrix: Φ matrix with iid elements

$$\phi_{ij} = \begin{cases} 1/\sqrt{M} & p = 1/2 \\ -1/\sqrt{M} & p = 1/2 \end{cases} \tag{2.14}$$

* Bernoulli matrix: Coefficients are only (+1) and (-1) or (0) with the same probability as follows

$$\phi_{ij} = \begin{cases} +1 & p = 1/2 \\ -1 & p = 1/2 \end{cases} \tag{2.15}$$

or

$$\phi_{ij} = \begin{cases} 1 & p = 1/2 \\ 0 & p = 1/2 \end{cases} \tag{2.16}$$

* Scrambled Fourier matrix: our Φ matrix is created as follows

$$\Phi = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \xi & \cdots & \xi^{(N-1)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \xi^{(M-1)} & \cdots & \xi^{(M-1)(M-1)} \end{bmatrix}_{M \times N} \tag{2.17}$$

where $\xi = e^{-j\frac{2\pi}{N}}$

* Sparse matrices: these matrices are different from the others. Each row has a certain and small number of non-zero elements randomly distributed. There are three cases of this type of matrix:

- Constant column weight: All columns have the same number of non-zero elements.

- Constant row weight: rows in Φ have the same number of non-zero elements.

- Constant row and column weight: a constant r is assigned for our Φ matrix that each row or each column has only r non-zero elements.

Null space conditions

Null space of a matrix is denoted as $\mathcal{N}(A)$ and is defined as a set of all vectors $\underline{X}$ that satisfy $A\underline{X} = 0$, where A is a measurement matrix with dimension $M \times N$.

$\mathcal{N}(A) = \{\underline{X} \mid A\underline{X} = 0\}$

if A is full-rank, then $\mathcal{N}(A) = \{\underline{0}\}$

We have: $Rank(A) + dim(\mathcal{N}(A)) = N$

Lemma: If X_0 is a set of solution for $\underline{Y} = \Phi\underline{X}$ then $N(\Phi) + X_0$ is the set of all possible solutions.

Returning to our k-sparse signal $\underline{X}$ that is sampled with measurement vector Y in $\underline{Y} = \Phi\underline{X}$, the null space of Φ must NOT include a vector whose sparsity is $2k$ or less. Further, any $2k$ columns of ϕ must be linearly independent and any submatrix of Φ with $2k$ columns must be full rank.

As defined as in [90], the *spark* of a given matrix is the smallest number of columns of that matrix that are linearly dependent. We have a theorem as follows:

Theorem: For any vector $Y \in R^M$, there exists at most one signal $X \in \sum_k$ such that $\underline{Y} = A\underline{X}$ if and only if $spark(A) > 2k$.

Based on all conditions addressed, we can pick M such that $M \geq 2k$.

The Restricted Isometry Property (RIP)

In [91], the following isometry condition on matrices Φ and established its important role in CS. The definition is as follows.

Definition: A matrix Φ satisfies the restricted isometry property (RIP) of order k if there exists a $\delta_k \in (0,1)$, called restricted isometry constant, is the smallest quatity such that

$$(1-\delta_k)||X||_2^2 \leq ||\phi \underline{X}||_2^2 \leq (1+\delta_k)||\underline{X}||_2^2 \tag{2.18}$$

We also have

$$1-\delta_k \leq \lambda_{min}(\Phi_T^T \Phi_T) \leq \lambda_{max}(\Phi_T^T \Phi_T) \leq 1-\delta_k \tag{2.19}$$

where T is all subset of $\{1, ..., N\}$ with $|T| \leq k$

We strive to work on the Φ matrix to get columns of Φ_T close to being orthogonal to make CS perform better. We finally have the definition for RIP as follows:

Definition: A matrix $\Phi \in R^{M \times N}$ is said to satisfy RIP of order $k \in N$ if its RIP constant δ_k is less than 1, $(0 \leq \delta \leq 1)$.

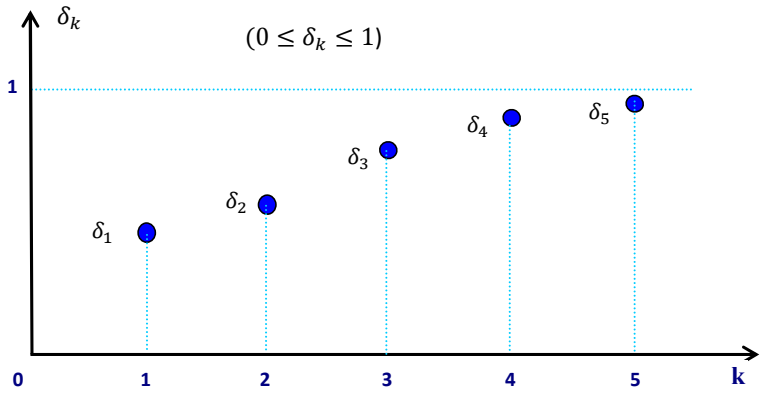


Figure 2.13: Restricted Isometric Constant Function

Mutual Coherence

The coherence between Φ and Ψ is defined as

$$\mu(\Phi, \Psi) = \sqrt{N} max| < \varphi_i, \psi_j > |, \tag{2.20}$$

where $i = 1 \div M,\ j = 1 \div N$ and

φ_i: row of Φ

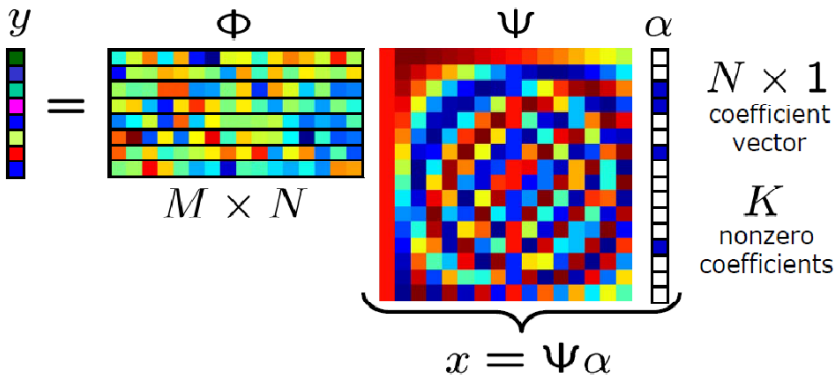


Figure 2.14: Sparsifying signals in a proper domain with ψ matrix

ψ_j: column of Ψ

The coherence measures the largest correlation between rows of Φ and columns of Ψ. If they contain correlated vectors, μ is large, otherwise, it is small. In CS, we expect to have μ is as small as possible. For orthonormal Ψ and Φ normalized rows, we have a limited range for coherence $\mu \in [1, \sqrt{N}]$.

Theorem 1: If φ_i's and ψ_j's have unit norm $||\varphi_i||_2 = 1$ and $||\psi_i||_2 = 1$ then

$1 \leq \mu(\varphi, \psi) \leq \sqrt{N}$

Theorem 2: Suppose $\underline{X} \in R^N$, $\underline{X}$ is k-sparse in Ψ domain. Set M measurement $\underline{Y} = \Phi\underline{X}$ if $M \geq C\mu^2(\varphi, \psi)klog(N/k)$ for $C \geq 0$. The solution to l_1 minimization is exact with high probability.

$\hat{\underline{\theta}} = arg\,min\,||\underline{\theta}||_{l_1}$ *subject to* $\underline{Y} = \Phi\Psi\underline{\theta}$, $\underline{X} = \Psi\underline{\theta}$

As mentioned, we expect μ small to get better CS reconstruction. For μ to be close to its minimum value of 1, each of the row of Φ must spread out in the Ψ domain.

Optimizing sensing matrices

Based on analysis above, the performance of CS can be improved by optimizing the Φ matrices. This is an open topic for additional research. There has been some work improving the projection matrix to make CS work better [92]. In WSNs, ϕ is created by the way we collect data. In other words, the matrix created depends

on the network topology and routing methods applied in a WSN. We will focus on different forms of the matrix in the next chapters associated with our data collection algorithms.

2.2.4 Signal Recovery

There are many studies on different reconstruction algorithms. All of these algorithm strive to recover the original signals more precisely. This dissertation will briefly address only some popular algorithms: L1-optimization [5, 6], OMP [93], ROMP [94], CoSaMP [95]. At the conclusion this section, there is an example which compare these recovery methods under one particular assumption.

L1 optimization

The number of measurements required to reconstruct the original signal perfectly with high probability is $M = O\left(k\, log\, N\right)$ following the l_1 optimization problem given by [5, 6].

$$\underline{\hat{X}} = arg\, min\, ||\, \underline{X}\, ||_1,\ subject\, to\ \ \underline{Y} = \Phi \underline{X}. \tag{2.21}$$

In case $\underline{X}$ is not sparse in canonical basis but sparse in a proper domain (Ψ can be Wavelet or DCT depending on signal properties). We have

$$\underline{\hat{\theta}} = arg\, min\, ||\, \underline{\theta}\, ||_1,\ subject\, to\ \ \underline{Y} = \Phi\Psi \underline{\theta}, \tag{2.22}$$

where $||\, \underline{\theta}\, ||_1 = \sum_{i=1}^{N} |\theta_i|$. The l_1 optimization problem can be solved with linear programming techniques such as Basis Pursuit (BP) [5].

In reality, we have to consider noise while sampling and sending the measurements (in our case we collect measurements and send to the base-station) : $Y = \Phi X + \underline{e}$, with $||\, \underline{e}\, ||_2 < \epsilon$ and recover

$$\underline{\hat{X}} = arg\, min\, ||\, \underline{X'}\, ||_1,\ subject\, to\ \ ||\underline{Y} - \Phi \underline{X'}||_2 < \epsilon. \tag{2.23}$$

Orthogonal Matching Pursuit (OMP)

Given measurements $\underline{Y}$ and the knowledge that our signal $\underline{X}$ is sparse or compressible, we can consider that $\underline{Y}$ is a linear combination of k columns of Φ. In OMP algorithm [93], we need to determine which columns of Φ contribute to $\underline{Y}$.

The idea of OMP is to pick columns in a greedy fashion. At each iteration, we choose a column of Φ that is most strongly correlated with remaining part of $\underline{Y}$, then we subtract its contribution to $\underline{Y}$ and iterate on the residual for k iterations. The OMP will find the correct set of columns.

Before starting OMP, the columns of Φ should be normalized as $||\varphi_i||_2 = 1$. Then we have OMP algorithm with 5 steps as follows:

1) Initialize:

Residual $\underline{r}_0 = \underline{Y}$

I_t= index set of column (support) = {}

Iteration $t = 1$

$\phi_0 = \{\}$ ←set of columns

2) Find the index I_t that solves

$I_t = arg\ max_{i=1,\ldots,n} | < \varphi_i, r_{t-1} > |$

$I_1 = arg\ max | < \varphi_1, r_0 > |$

If the max occurs for multiple indices, then pick one of them. This step find the largest correlation with residual r_{t-1}(Note: $t = 1,\ r_0 = y$).

3) Argument the indices

Find the index for support: $col_t = col_{t-1} \cup \{I_t\}$

then add the column to ϕ_t: $\phi_t = [\varphi_{t-1} \quad \varphi_{I_t}]$

4) Solve a least square problem to obtain a new signal estimate

$\underline{x}_t = arg\ min||\phi_t \underline{x} - \underline{y}||_2$

we have closed form solution for the least square problem:

$x_t = (\phi_t^T \phi_t)^{-1} \phi_t \underline{y}$

$\underline{r}_t = \underline{y} - \phi_t \underline{x}$

5) Increase $t = t + 1$, go to step 2. with the condition: *while* $t < k$.

Note: at step 4, we look for a linear combination of a chosen column in ϕ_t such that $\underline{r}_t = \underline{y} - \phi_t \underline{x}$ has the smallest norm 2. It is known that r_t is orthogonal to ϕ_t.

In general, greedy methods are usually fast and easy to implement. OMP only needs k iterations to find support and each of them requires matrix multiplication, solving one least square problem.

Regularized OMP (ROMP)

OMP does not typically perform well in the presence of noise. A similar algorithm called Regularized OMP [94] is created to solve the noise problem. In ROMP, at each iteration, more than one support will be collected.

In ROMP, instead of picking one support like OMP, ROMP picks more than one support at each iteration.

$u = \phi^T \underline{r}_{t-1}$

Choose a set J with $|J| = k$ biggest absolute values of $\underline{u}$.

Regularize: Derive set J into subsets J_0 with comparable coordinate that satisfy:

$|U(j)| \geq \frac{1}{2}|U(i)| \qquad \forall i, j \in J_0$

- Sorting $U(i)$

- Choose the subset J_0 with the maximum energy

- Update $col_t = col_{t-1} \cup J_0$

- The next steps are similar to step 4 and 5 in OMP.

The condition to stop: $||r_t||_2 < 10^{-6}$ or number of selected indices for support $\geq 2k$.

Compressive Sampling Matching Pursuit (CoSaMP)

We know that the most important part of the reconstruction process is to identify the support of $\underline{X}$. In CoSaMP [95], an approach inspired by RIP is used.

Suppose Φ has $\delta_k \ll 1$, then for a k-sparse signal $\underline{X}$, we can have $\underline{Z} = \Phi^T \Phi \underline{X} = \Phi^T \underline{Y}$.

For more details, the largest k entries of $\underline{Z}$ point toward the largest k entries of $\underline{X}$. So we can obtain a proxy of $\underline{X}$ by applying Φ^T of $\underline{Y}$. At each iteration, the current approximation induces a residual, the part of the signal that has not been approximated.

As the algorithm progress, the samples are updated as follows:

Initialize:

$r_0 = \underline{Y}$

$t = 1$

$a^0 = 0$ (zero vector)

Repeat:

$\underline{Z} = \Phi^T \underline{r}_{t-1}$ (form proxy)

$\Omega = supp(\underline{Z}_{2k})$ (identify large components)

$\Omega \cup supp(a^{t-1})$ (merge support)

Least square:

$x_t = arg\ min||\underline{Y} - \Phi_T \underline{X}||$

$x_t = (\Phi_T^T \Phi_T)^{-1} \Phi_T^T \underline{Y}$

- Prune to have k largest elements: $a^t = \hat{x_k}$

$\underline{r}_t = y - \Phi a^t$

$t = t + 1$

Go back!

To sum up, with CoSaMP, we have five major steps:

1) Identification: The algorithm forms a proxy of the residual and locates $2k$ largest components of the proxy.

2) Support merge: The set of newly identified components is united with the set of component that appear in the current approximation.

3) Estimation: The algorithm solves a least square problem to estimate the target signal on the merged set.

4)Pruning: The algorithm produces a new approximation by keeping only k largest entries in least square approximation.

5)Update: Form the residual and go back to 1.

- Stop criteria is set as the number of iteration $> 6(k+1)$ or $||r_t||_2 < 10^{-6}$.

Comparison of reconstruction algorithms in compressed sensing

There are many more CS algorithms that have been applied to different areas that are not listed here. The success of CS depends on properties of the signals to which it is being applied. In this subsection, we only give one simple example to show how these four CS reconstruction schemes work on a specific signal. We assume to have a k-sparse signal with length $N = 1000$ and $k = 50$. We sample our $\underline{X}$ signal with additive noise having zero mean and unit variance. The number of measurement is variable from $(100 : 350)$.

From 2.15, we can briefly say that OMP is not good with noise and L1 is still the best one in this case.

2.3 Literature Review

In this section, work which focuses on two different points to improve data collection algorithms in WSNs is reviewed. The first is applying CS into data collection methods. As discussed in 2.1.3, there are many methods to collect data in WSNs to be sent to the sink or BS. When these methods employ CS, the common ways of collecting data

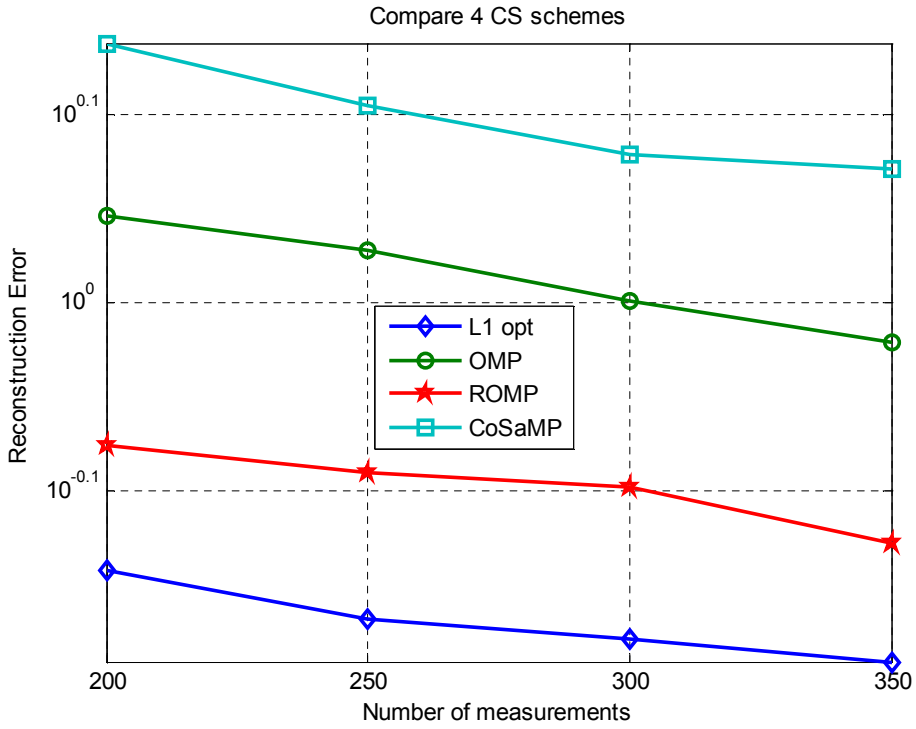


Figure 2.15: Compare four Compressive Sensing reconstruction schemes

from sensors changes depending on each specific method. Measurement matrices in CS recovery algorithms at the BS show records of how CS measurements are collected. They are also called routing matrices. We want to ensure that a measurement matrix created by each routing method can work well in a CS recovery process. In this review, measurement matrices created by data collection methods have been shown to satisfy RIP.

The second point will be addressed in this review is improving the measurement matrix for better CS performance. Based on that, the number of CS measurements could be reduced which may reduce transportation cost in WSNs.

2.3.1 CS Based Data Collection Algorithm in WSNs

There are three goals we always expect when CS-based routing algorithms are designed; balance energy for sensor, reduce total power consumption for networks and measurement matrices created by collecting CS measurements do not degrade CS

performance.

Random routing based methods

Random routing, also called random walk (RW), is considered as a routing method that balances energy for sensors in a WSN. Sensors are chosen randomly to send their data to be forwarded to the sink node or BS. There is not routing table or global information for sensors to send data to the BS. In this type of routing, the existing work focuses on how to send data to the BS or how to form CS measurements to be forwarded to the BS for data reconstruction.

In [96] RWs start from random sensors in a WSN. They choose the next nodes to collect data based on a fixed probability until they meet the BS or the sink. As mentioned in [97], the RW length can be considered as the cover time in random geometric graphs as $O(NlogN)$. In this case, the RW's lengths are different to each other and extendable into many steps that the sensors consume more energy for transmitting data. In [98] a mobile sink is used to collect CS measurements from storage nodes after nodes forward data based on RWs. In [99, 12] RWs are created that sensors choose the next nodes based on the Metropolis-Hastings algorithm as mentioned in [100, 97, 101, 102]. Based on the probability mentioned in [100], every sensor has an equal chance to be visited with smaller RW length. Each sensor also needs to verify the number of its neighbors from others to decide the next nodes to forward data.

In [13, 103, 14], simple RWs are used to collect data in which the next nodes for RWs are chosen based on a uniform probability [13, 14] or pseudo-random seeds [103]. Each CS measurement is created at the last node visited by each RW with a predefined length. CS measurements from RWs are sent directly [13] or through intermediate nodes in multi-hop [103, 14] to the BS.

Clustering based methods

Generally, a WSN has one sink node or BS as the data processing center. The network could be clustered in different ways as mentioned in 2.1.3. Cluster-heads (CH) can be chosen from random nodes to balance energy for the network since they deplete energy more quickly than common nodes. There are two types of measurement matrices which are applied, random Gaussian with all entries i.i.d. zero-mean random variables with variance $\frac{1}{M}$ and block diagonal matrix (BDM).

In [104, 105, 106] both raw data and combined samples are in traffic. In [104] the CS operation requires each node in the WSN to send exactly M packets. M is denoted as the number of CS measurements required to reach a given reconstruction quality. A spanning tree is used to partition the network into sub-nets. [105] and [106] propose a scheme called hybrid CS aggregation that combines the non-aggregation and plain CS mentioned in [104] to reduce the traffic loads sending from each node. The non-aggregation method is used if a node receives less than $(M-1)$ raw readings from its downstream nodes. Otherwise, plain CS is used. In [104, 105, 106], each sensor needs to send up to M samples to the BS to contribute M CS measurements for signal recovery. The BDM is mentioned in [106] to reduce partially the total samples being sent from each sub-net. In [107], a WSN is partitioned into clusters. Sensor readings are sent to CHs and the CHs send the received data to the BS. Since the measurement matrix is full Gaussian, each cluster needs to generate M samples to contribute to M CS measurements at the BS.

The block diagonal matrices are used in [15, 16, 17] for more energy saving in data transmission in clustered WSNs utilizing CS. Sensors take turns to become CHs to share the burden of the role which has to transmit data much more than common nodes. In [16], non-CH sensors multiply their readings with Gaussian random coefficients and keep sending data to CHs they belong to. The CHs add up the received data including their own and finally send all data from their clusters to the

BS for data reconstruction. In this work, each sensor still has to send its own data a certain number of times corresponding to the number of CS measurements required from its cluster. Note that as shown in [15], the number of CS measurements required from a cluster is linearly proportional to the number of sensors in the cluster. In this case, sensors from cluster ith do not have to send M times, just M_i times, where $\frac{M_i}{M} = \frac{N_i}{N}$. ($N_i$ is number of sensors in i_{th} cluster).

In [15, 17], each sensor only has to send its data once to its CH. A sub-matrix is created at every cluster to form CS measurements to be sent to the BS. The sub-matrix is created to be contributed to the BDM at the BS. The dimensions of the sub-matrices are also linearly proportional to the number of sensors in the clusters. After CS measurements are created at CHs, they can be forwarded to the BS directly [15] or in multi-hop through CHs [17] to the BS. By utilizing CS in data collection in clustered WSNs, the total power consumption could be reduced significantly. As mentioned in [15, 17], reducing the cluster size could either reduce intra-cluster power consumption or total power consumption for such networks.

Tree based methods

As mentioned in 2.1.3 tree can be formed for data collection in WSNs. When CS is employed in collecting data, there are two types of measurement matrices which are created by different methods: random sparse measurement matrix and full dense Gaussian matrix.

C. Luo [108] and J. Luo [109] contribute routing methods to collect data in a WSN utilizing CS. All sensors in the methods have to send either their own readings or combined messages to their parent node or the sink M times corresponding to the number of measurements required. The methods only help to balance energy in such networks since all sensors need to transmit data a certain number of times corresponding to the number of CS measurement required. Measurement matrices

created in the methods are full dense Gaussian since all sensors contribute their readings for each CS measurement.

Wang [110] mentioned a distributed algorithm with the sparse random projections that motivated a different way to collect measurements from sensors in a WSN. At a sampling time, only some random sensors send their data to the BS. Their indices correspond to the non-zero entries in each row of the measurement matrix at the BS. For the others which do not contribute data to a CS measurement, the entries in the corresponding row in the measurement matrix are zeros. This significantly reduces a huge number of transmissions or power consumption for data collection in such networks. We need to make sure that this type of sparse random matrix does not degrade the CS performance.

A novel idea from Xie called MTT [111] significantly reduces transportation cost. By using the sparse random matrix, only some random nodes send their own data to build a CS measurement. The other nodes relay data if they are along the routing path or they may not participate in some rounds of transmission. The routing tree is designed to eliminate some residual transmissions since nodes that have more rounds to transmit their own data are a priority to choose while building the tree.

In TCS [19], an algorithm is proposed based on an existing tree which can be formed by any routing tree. The algorithm assumes that each sensor could be able to store and generate accumulative measurements to be forwarded to the BS. Each node sends its own readings to its parent node only once. Based on a generated sparse measurement matrix, the parent nodes know when their child nodes contribute their data to CS measurement and they generate measurements to send to upper nodes or the BS for the CS recovery process. The results in [19] have shown that TCS overcomes MTT [111] in performance.

2.3.2 Minimizing the Number of CS Measurements

Exploiting Inner Products

As mentioned in Section 2.3.1, data collection algorithms are designed under routing mechanism to reduce transportation cost or power consumption for such networks in order to prolong the network lifetime.

In [112], a different mutual coherence is considered. The goal is to minimize the mutual coherence for a better CS performance. It is known that each row of the measurement matrix represents a routing path to collect data from sensors contributed data for that corresponding measurement. The CS measurements collected at the BS for signal reconstruction can be addressed as $\underline{Y} = [y_1\, y_2 \ldots y_M]^T$ or $\underline{Y} = \Phi\Psi\underline{\Theta}$, where $\underline{\Theta} = \Psi^T\underline{X}$. The sparsity of measurement matrices or projections have limitation. So we only can exploit the incoherence between Φ and Ψ for more energy saving. The coherence μ can be calculated as

$$\mu = max_{i \neq j} \langle H_i, H_j \rangle. \tag{2.24}$$

The inner product

$$\langle H_i, H_j \rangle = \Psi_i^T \Phi^T \Phi \Psi_j. \tag{2.25}$$

For small coherence value, we require $\Phi^T\Phi \approx I$. In [113], Lee proposed a low coherence projection for efficient routing in WSNs, called LCPR. At each routing path, a random node chooses a next node to forward data based on a greedy choice which minimizes the "intermediate coherence" of the spatial measurement matrix Φ.

In [114], "maximum energy overlap" is calculated as

$$\beta(\Psi) = \beta(P\tilde{\Psi}) = \max_{i,j} \sum_{l} \Psi_i^2(i,j), \quad \beta(\Psi) \in [0,1], \tag{2.26}$$

where β represents the maximum amount of energy of a basis functions captured by a single cluster. As basis functions are overlapped with more clusters, we will have a

potentially higher chance to reconstruct the signal correctly. As discussed, the more energy overlapped, the better CS performance.

Mapping sensor nodes

As mentioned, sensor readings are often highly correlation which is suitable to apply CS. While collecting data in a WSN from a sensing area, there are many way to find a mapping between sensors and the data vector ($\underline{X} = [x_1\, x_2 \ldots x_N]^T$) that is supposed to be sparse or compressible. If the sensor nodes are indexed properly, the sensory data collected from them can be more compressible in a sparsifying domain (Discrete Cosine Transform - DCT or wavelet). As mentioned in [115], the coherence between the measurement matrix (Φ) and the sparsifying matrix (Ψ) is exploited. The smaller the coherence, the fewer CS measurements are required. This means that the WSN collects less CS measurements from the sensors that reduces power consumption for data transmission from sensors to the BS.

In [116], an algorithm is designed to compress and reconstruct sensor observations in an energy efficient manner by exploiting spatial correlation between the sensed data. In [117] Leinonen reduced the number of transmissions by exploiting certain correlation structures of a multi-dimensional signal. Instead of conveying original sensor readings, the sensors deliver CS-aggregated measurements to the sink, leading to a reduced number of incurred transmissions in the WSN. With a sufficient number of measurements, the sink can accurately reconstruct the data via CS decoding.

CHAPTER 3

RANDOM WALK BASED DATA GATHERING IN WIRELESS SENSOR NETWORKS

3.1 Introduction

3.1.1 Motivation

Saving power consumption in monitoring data in wireless sensor networks (WSNs) is always a critical issue. Sensors in the networks are randomly deployed to sensing areas that need to be observed. They are often deployed in harsh conditions without maintenance or a replaceable power supply [118]. Therefore, the connection of the network relies on these small and inexpensive devices under a severe energy constraint.

Random walk routing (RW) [75, 119, 120] has been shown to balance sensors' consumed energy in WSNs since each node is randomly chosen to transmit or relay data. Based on the sensor broadcasting radius, also called transmission range, a WSN is connected as an undirected graph $G(V, E)$, where V is the vertex set and E is the edge set. Depending on the transmission range, each node may connect to many or a few other nodes with different real distances. The method itself prolongs network lifetime.

Compressive sensing (CS) [5, 6, 7, 8, 121] provides a novel data processing technique for sparse or compressible signals. Based on the spatial correlation of the sensory data in WSNs, the technique offers to reconstruct all sensor readings from N nodes based on only $M = \mathcal{O}(klogN/k)$ CS measurements collected from the sensing area and sent to the sink or base-station (BS) [9, 11, 10]. The sensory data is assumed

to be k-sparse signal in canonical basis or other sparsifying bases such as *Discrete Cosine Transform* (DCT) or *Wavelet*. Each CS measurement can be collected from all sensing nodes or from some random nodes depending on data collection methods in the network utilizing CS. In recent years, there are several studies on the integration of CS and data collection in WSNs [122, 123, 99, 124, 13, 15, 20]. In these methods, sensor readings are multiplied by some coefficients and are sent to the BS following some chosen routing method such as gossip-based, random walk, tree-based, and cluster-based. The CS measurements are collected at the BS as $\underline{Y} = \Phi\underline{X}$, where Φ is the measurement matrix, also called the projection matrix and vector $\underline{X}$ presents all unknown readings from all the sensors. The resulting measurement matrices can be sparse or dense with Gaussian coefficients depending on the underlying routing methods.

In order to apply CS in RW routing in WSNs in which all sensors are randomly deployed in the sensing area, we need to initiate a certain number of RWs with a predefined length to collect sensor readings from random nodes. Those readings are added together as a CS measurement to be sent to the BS for the CS recovery process. Each RW may or may not sample all sensors depending on the RW's length or the measurement matrices we want to apply in the CS recovery. Since Radu [125] states that random sparse measurement matrices can work as well as the full Gaussian ones in the CS recovery processes, then the sparse binary matrix is applied in the proposed RW routing. Thus, each RW only needs to collect sensory data from some random nodes, not from the entire network, which results one spare binary row of the measurement matrix. M rounds of RW collect M random measurements as $\underline{Y}_{M\times 1} = \Phi_{M\times N}\underline{X}_{N\times 1}$ to be sent to the BS for the CS recovery process.

The length of RWs mainly determines the energy consumption for each measurement collected. In order to have each RW visit nodes with an equal probability, we choose the length as the mixing time based on RW theory. The mixing time is dif-

ferent in each particular network and depends on the transition probability matrix which will be addressed in the next section. It is known that sensors communicate to each other based on a transmission range. This range decides the size of each sensor's neighborhood. We analyze the trade-off between the length of RWs and the transmission range and suggest an optimal case to minimize the energy consumption for such networks.

3.1.2 Related Work

Random walk has been used effectively for data gathering in WSNs. It does not need global information as the shortest path routing [126] or other methods. Furthermore, RW achieves load balancing for the networks based on the probabilities to transmit or relay sensory data. [127] provides some results on regular and irregular grid networks to contribute a construction of RWs on one particular family of random graphs. [128] uses some mobile patrol nodes to collect data from random static sensors uniformly distributed in a square network. The paper provides the concept of node coverage to calculate the expected number of sensors captured within a given time frame. [129] works on lattice networks to collect data based on oriented routing from a source to a destination. The probabilities of wrong and right directions is mentioned in the paper. [119] considers a given packet from a random sensor sending until it reaches the sink or BS. Only grid networks are considered. [120] addresses different RWs as Blind RW, RW with memory, and a new proposed RW. It is proposed that a node chooses a next node based on a probabilities created by the distances between its neighbors. None of them consider the length of RWs routing in WSNs in calculating the power consumption.

RW becomes more powerful when it combines with CS [5, 6, 7] since the sparse random projection has been shown to work as well as the dense full Gaussian matrix in [125, 130, 110]. In [96] RWs choose the next nodes to collect data based on a fixed

probability until they meet the BS or the sink. In this case, the RW's lengths are different to each other and extendable into many steps that the sensors consume more energy for transmitting data. In [98] a mobile sink is used to collect CS measurements from storage nodes after nodes forward data based on RWs. In [99, 12] RWs are used where sensors choose the next nodes based on the Metropolis-Hastings algorithm mentioned in [100, 97, 101, 102]. They also verify the number of neighbors from others to decide the next nodes to forward data. In [13, 103, 14] simple RWs are used to collect data in which the next nodes for RWs are chosen based on an uniform probability [13, 14] or pseudo-random seeds [103]. Each CS measurement is created at the last node visited by each RW with a predefined length and is sent directly [13] or through intermediate nodes in multi-hop [103, 14] to the BS.

The approach here is based on our previous work in [13, 14]. We study RWs and the integration between RWs and CS to build an energy-efficient data collection method in WSNs employing CS. In this CSR algorithm, M RWs start from M random nodes which are chosen with probability M/N. These RWs randomly choose the next nodes as neighbors and visit random nodes in a certain number of steps which are defined as RW's length to collect sensory readings, and finally send directly (D-CSR) or in multi-hop routing (M-CSR) the CS measurements to the BS. All power consumptions for data collection in the networks are formulated, simulated and analyzed. The trade-off between the RW length and the sensor transmission range is investigated to minimize the network power consumption.

The main contributions in this work are summarized as follows:

1- Two versions of the CSR algorithm (D-CSR and M-CSR), which combine RW and CS in collecting data in WSNs. CSR significantly reduces transmission energy consumption in the networks.

2- Energy consumptions in D-CSR and M-CSR are formulated and simulated.

3- A simple greedy distributed data gathering tree algorithm is proposed to for-

ward CS measurements in multi-hop routing to the BS.

4- The trade-off between the RW length (Mixing time) and the sensor transmission range is investigated for further energy saving.

3.2 Background and Problem Formulation

3.2.1 Random Walk

Random walk (RW) on a graph can be modeled as a Markov chain as noted in [75, 97, 131]. Walking steps jump from node to node randomly based on probabilities generated based on sensor neighborhoods, called transition probabilities. Specifically, the next node j in the sequence is selected from the set of neighbors of the previous node i in the sequence with probability p_{ij}. Such probabilities form a transition probability matrix $P = [p_{ij}]_{N\times N}$, where N is the total number of vertexes. For example, a simple random walk at a step of time k needs to move from vertex i to one of its adjacent vertices j with a probability p_{ij}. The transition probability is calculated as follows

$$p_{i,j} = P(X_{k+1} = j \,|\, X_k = i) = \begin{cases} \frac{1}{d(i)}, & if\ (i,j) \in E \\ 0, & others, \end{cases} \tag{3.1}$$

where $d(i)$ denotes the degree of vertex i. This matrix characterizes the Markov chain that is an important model for random walks on graphs which satisfies

$$0 \leq p_{\{ij\}} \leq 1, \quad i,j = 1,2,...N; \quad \sum_{j=1}^{N} p_{\{ij\}} = 1, \quad i = 1,2,...N. \tag{3.2}$$

The length of RWs is often considered based on two factors, the cover time and the mixing time which are defined as follows.

Definition 1: *Cover time is the expected number of steps to each every node in the connected graph. The cover time of the graph is $\mathcal{O}(NlogN)$, where there are N vertices in the graph ($|V| = N$).*

The cover time refers to a long RW length. We prefer to choose the mixing time that can approach uniform stationary distribution for all sensor nodes as

Definition 2: *Mixing time is the number of steps for the distribution of a random walk to be stationary. It shows how fast a RW converges to its stationary distribution.*

The mixing time of RWs has been studied well in [97, 132, 133] showing that it significantly depends on the transmission range or the transition probability p_{ij}. It measures the number of steps for the distribution to reach the stationary distribution. As mentioned in [97], the asymptotic rate of convergence of the Markov chain to the uniform equilibrium distribution is determined by the second largest eigenvalue of the transition probability matrix P as follows

$$\mu(P) = max_{i=2,\ldots,n}|\lambda_i(P)| = max[\lambda_2(P), .\lambda_n(P)]. \tag{3.3}$$

Since the graph is irreducible and aperiodic, then $\mu(P) < 1$ and the distribution converges to uniform asymptotically. The mixing time τ can be calculated as

$$\tau = 1/log(1/\mu). \tag{3.4}$$

3.2.2 Problem Formulation

We assume to have a WSN in which the sensors are randomly distributed in the sensing area. An appropriate transmission range is chosen for all sensors to be connected to each other as a undirected graph $G(V, E)$. The edge set E represents all communication links between sensors that can be changed based on the sensor transmission range, denoted as R.

Based on the CS theory, we need a number of measurements required ($M = \mathcal{O}(klogN/k)$) from the network to recover all raw sensor readings at the BS. M nodes are chosen randomly to initiate M data collection walks. In each RW, a chosen node adds its own reading to the RW's message and continue to choose another node to forward the combined message to. Its ID corresponds to the non-zero coefficient in a

row of the measurement matrix ϕ. The sampling processes finish when the number of walking steps reaches the length of the RWs which is chosen as the mixing time of the graph $G(V, E)$.

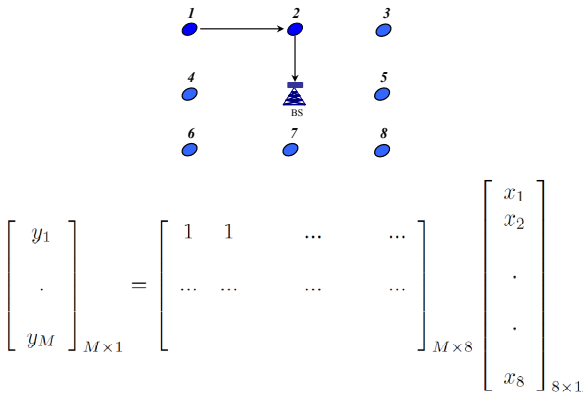


Figure 3.1: Illustration of a simple RW routing in a WSN with 8 nodes and the projection matrix created from each RW.

Figure 3.1 shows a simple example of RW routing and building the matrix ϕ. The first RW samples only two nodes, numbered as 1 and 2. It starts from node 1 and finishes at node 2. The readings from the two nodes are added together to be sent to the BS as the first CS measurement. This is illustrated as a record of collecting data in the measurement matrix at the 1^{st} row. The next $(M-1)$ nodes are chosen randomly to initiate $(M-1)$ RWs that also create $(M-1)$ rows for the measurement matrix. For simplicity, we choose all coefficients equal to "1" that results in a sparse binary matrix. We assume that M RWs will collect enough of the number of measurements required to reconstruct precisely all raw data from the network at the BS.

3.3 Compressive Sensing Based Random Walk Data Collection Algorithm (CSR)

3.3.1 System Model

We model a WSN with N sensors which are uniformly randomly distributed in either a square area $L \times L$ or circular area with radius R_0. The topology of the connected network is a random geometric graph $G(V, E)$. We assume that each sensor has the same broadcasting radius, also called transmission range (R), calculated as a Euclidean distance. It is also assumed that sensors can adjust their consumed energy to transmit data based on real distances between them.

In reality, our network is not a regular network where each sensor has a different number of neighbors within its range of the radius R. If we change the value of R, the connection of the graph will change as follows: the set of vertex V is the same based on the total number of sensors, but the edge set E changes based on the communications between sensors in each neighborhood. We assume that a proper value of R is chosen that each node connect to $\mathcal{O}(logN)$ neighbors for connectivity of the network [134].

3.3.2 The CSR Algorithm

The proposed algorithm is summarized below as Algorithm 1.

The number of measurements required M is now the number of RWs that decides the reconstruction error when the BS recovers all sensor readings $\underline{X}$. It can be determined prior to sampling the data. The more CS measurements, the more accuracy of the recovered data.

In a WSN with a large number of nodes, the transmission range determines number of neighborhoods of each sensor as shown in Figure 3.5 in Section 3.4.3. It is highly related to the mixing time (τ) which is chosen as the RW length. The *LengthCounter*

Algorithm 1: Compressive Sensing Based RW Data Collection Algorithm

Initial phase:

- $\underline{\mathbf{X}} = [x_1 \dots x_N]^T$ represents N unknown values to the BS.

- $\underline{\mathbf{Y}} = [y_1 \dots y_M]^T = zeros(M, 1)$; % CS measurements collected from M RWs.

- $RWlength = \tau$; % The RW length is chosen as the mixing time τ

- $LengthCounter = RWlength$;

- $ID[i] = zeros(1, RWlength)$ % shows IDs of all nodes visited by i^{th} RW.

- $\Phi = zeros[M, N]$; % The measurement matrix before the RWs proceed.

Data collection phase:

for *i = 1 to M* **do**

  while $LengthCounter \geq 0$ **do**

    if *Node* j^{th} *is chosen based on Equation (*3.1*)* **then**

      if $j \notin ID[i]$ **then**

        $Y(i) = Y(i) + X(j)$; % Add new data to the combined message

        Update $ID[i]$;

      else

        $Y(i) = Y(i)$; % Do not add data or update ID

      end

      $LengthCounter = LengthCounter - 1$;

      $\Phi[i, :] = ID[i]$;

    end

  end

  One RW complete: CS measurement $Y(i)$ is sent to the BS.

end

Data recovery phase

All unknown values ($\underline{\mathbf{X}}$) are reconstructed at the BS based on M CS measurements ($\underline{\mathbf{Y}}$) as

$\hat{\underline{\mathbf{X}}} = arg\,min\,||\,\underline{\mathbf{X}}\,||_1$, *subject to* $\underline{\mathbf{Y}} = \Phi\underline{\mathbf{X}}$.

is set as the length when a random node is chosen to start a RW. It count-downs when a RW reaches a next node. It is possible that a RW may visit a node more than once. The node cannot add again its own reading to the combined packet and update its ID. It only receives and forwards the same packet to a neighbor chosen randomly based on Equation (3.1). Each CS measurement is built when a RW reaches the last node or $LengthCounter = 0$. The CS measurements are sent directly or in multi-hop to the BS for reconstructing all the unknown readings from the network.

3.3.3 Analysis of the Measurement Matrix: CS Recovery Performance and Network Coverage

Each row of the measurement matrix is built from a RW with length τ. It shows the node each RW visited corresponding to the non-zero elements in the matrix as follows

$$\Phi = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & ... & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & ... & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & ... & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ 1 & 0 & 1 & 0 & 0 & 1 & ... & 0 \end{bmatrix}_{M \times N} \tag{3.5}$$

Some row weight may be less than τ since a RW may visit a sensor node more than once that does not affect the CS recovery process. As shown in [125, 130], this sparse binary matrix satisfies the Restricted Isometry Property (RIP) that can be applied in the CS recovery algorithm to reconstruct signals. We also show in the simulation section that a sparse binary matrix formed from RWs in an arbitrary network can work as well as a full dense Gaussian matrix in Figure 3.10.

We also consider sensor coverage for the networks while using a certain number of RWs with a limited RW length as τ. Coverage is usually interpreted as how well a sensor network will monitor a field of interest. It can be thought of as a measure of quality of service. Back to our problem, we show the rate of sampling all sensors

in the network by different numbers of RWs with a fixed RW length in Figure 3.7. With the received measurement matrix Φ, if there is a column weight ≥ 1 then the corresponding node is visited at least once, or vice versa.

3.3.4 Analysis of the Trade-off between the Transmission Range and the Random Walk Length

As mentioned in Section 3.2.1, the RW length or the mixing time (τ) is calculated based on the transition probability matrix P. The matrix is created based on the sensor connections which are determined by the sensor transmission range R. Our sampling goal is to have RWs visit all nodes with the same probability that also means the stationary distribution is approximately uniform. If we increase R, each sensor node will have more neighbors as shown in Figure 3.5. It can reach the others in a smaller number of steps or in other words, the mixing time reduces, as shown in Figure 3.6. When the RW length reduces, each RW spends transmission energy on the smaller number of walking steps but longer distances. We investigate the trade-off between these two important factors, mixing time and transmission range that can provide an optimal point for the networks to consume the least power, as shown in Figures 5.4 and 4.19 in D-CSR and M-CSR in the next sections, respectively.

3.4 Directly Forwarding the CS Measurements to the Base-station (D-CSR)

3.4.1 Network Model

In this section, we choose a square sensing area dimensioned $L \times L$ with N sensor nodes randomly deployed in the area. An appropriate transmission range R is chosen for all sensors that the network is connected as a graph $G(V, E)$. Given the number of RWs (M) and the RW length (τ), M random nodes are chosen with the probability $\frac{M}{N}$ to initiate RWs. At the end of each walk, the last node obtains one CS measurement

and sends the measurement directly to the BS as shown in Figures 3.3 or 3.4. The power consumption of the power amplifier is a function of transmitting distances which we consider to formulate based on a stochastic problem.

3.4.2 D-CSR Power Consumption Analysis

The total power consumption for collecting measurements at the BS generally contains the consumed power for random walks and the power to send directly the measurements to the BS, that is specified as

$$E_{total} = (E_{RW} + E_{to\,BS}). \tag{3.6}$$

E_{RW} is the consumed power for M random walks with length τ that can be calculated as follows

$$E_{RW} = M \times \sum_{i=1}^{\tau} r_i^{\alpha}, \tag{3.7}$$

where r represents real transmitting distances between sensors in RW routing, and α is the path-loss exponent ($\alpha \geq 2$). It is shown in [135] that $\alpha = 2$ and $\alpha = 4$ in free space and multipath fading channels, respectively.

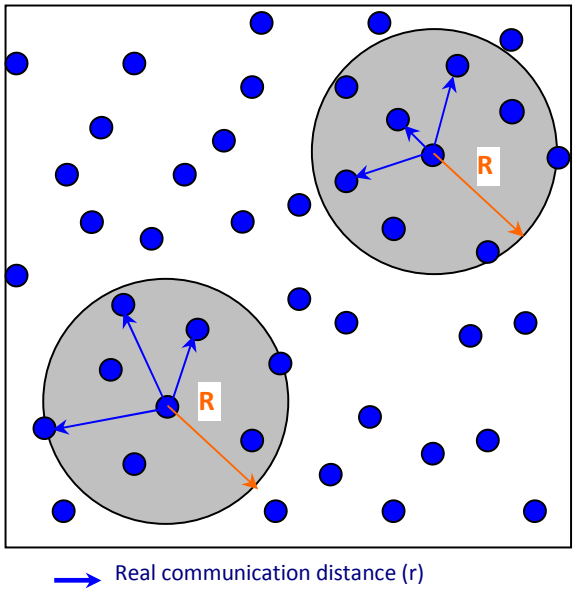


Figure 3.2: Sensor neighborhoods defined by the sensor transmission range R

Since sensors are uniformly distributed in an area covered by R as shown in Figure 3.2, and one of them is chosen based on the transition probability mentioned in Equation (3.1), r is also a random variable presenting the real distance between consecutive sensors along a RW (Figure 3.2).We can calculate the mean communication distance statistically as follows

$$\mathrm{E}[r^2] = \int\int (x^2 + y^2)\, \rho(x, y)\, dx\, dy, \tag{3.8}$$

where $\rho = 1/(\pi R^2)$ is the joint probability (pdf) with two random variables x and y. We can change Equation (3.8) into polar coordinates as

$$\mathrm{E}[r^2] = \int\int r'^2 \rho(r', \theta)\, r'\, dr'\, d\theta \tag{3.9}$$

$$\mathrm{E}[r^2] = \frac{1}{\pi R^2} \int_{\theta=0}^{2\pi} \int_{r'=0}^{R} r'^3 dr'\, d\theta. \tag{3.10}$$

Finally, we obtain

$$\mathrm{E}[r^2] = \frac{R^2}{2}, \tag{3.11}$$

or

$$r_{mean} = \frac{R}{\sqrt{2}}. \tag{3.12}$$

So, the total consumed power for M random walks with length τ is

$$E_{RW} = M \times \tau\, (\frac{R^2}{2})^{\alpha/2}. \tag{3.13}$$

Since the nodes sending CS measurements to the BS are randomly distributed, we can calculate $E_{to\,BS}$ statistically as

$$E_{to\,BS} = \sum_{i=1}^{M} d_i^{\alpha} \;=\; M \times d_{mean}^{\alpha}, \tag{3.14}$$

where d presents the transmitting distance between the last node of a RW and the BS that can be considered as a random variable. Since sensors and RWs are initiated

randomly, we can calculate the mean value of d (d_{mean}) approximately in two common positions for the BS in WSNs: BS at the center and outside the sensing area.

BS at the center of the sensing area

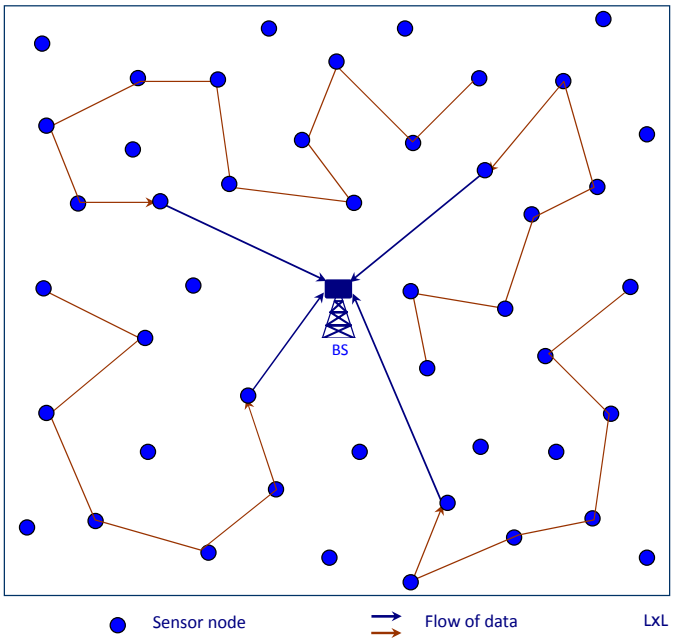


Figure 3.3: An illustration of RWs collecting data when BS at the center

In Figure 3.3, a RW ends at a node that is also randomly distributed in the whole network. We can calculate the mean square distance as follows

$$E[d^2] = \int_0^L \int_0^L [(x - \frac{L}{2})^2 + (y - \frac{L}{2})^2] f(x,y) dxdy, \tag{3.15}$$

where $f(x,y) = \frac{1}{L^2}$ is the joint probability function (pdf). We can obtain the expected square distance from RWs to the BS

$$E[d^2] = \frac{L^2}{6}. \tag{3.16}$$

We finally derive the total power consumption for WSNs from Equation (3.6) as follows

$$E_{total} = M(\tau (\frac{R^2}{2})^{\alpha/2} + (\frac{L^2}{6})^{\alpha/2}). \tag{3.17}$$

BS outside the sensing area

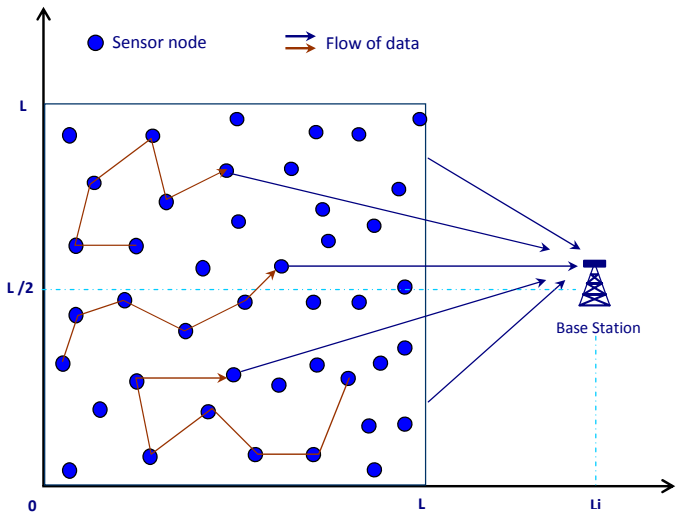


Figure 3.4: RWs collecting data when the BS is outside the sensing area at $(L_i, \frac{L}{2})$.

As shown in Figure 3.4, the BS is located outside the sensing area. We set a fixed position for BS $(L_i, \frac{L}{2})$ in this analysis section. It means L_i can be changed versus L in real applications. The expected square distance between RWs and BS can be calculated as

$$E[d^2] = \int_0^L \int_0^L [(x - L_i)^2 + (y - \frac{L}{2})^2] f(x, y) dx dy. \tag{3.18}$$

Similarly, we obtain the mean square distance between RWs and BS as follows

$$E[d^2] = \frac{1}{L}[\frac{(L - L_i)^3}{3} + \frac{L_i^3}{3}] + \frac{L^2}{12}. \tag{3.19}$$

From Equations (3.6) and (3.19), we derive the formula for the total power consumption when BS is outside the sensing area as

$$E_{total} = M[\tau\,(\frac{R^2}{2})^{\alpha/2} + (\frac{(L - L_i)^3 + L_i^3}{3L} + \frac{L^2}{12})^{\alpha/2}]. \tag{3.20}$$

We now have built the required total power consumption formulas. They will be applied for an arbitrary network in the simulation section.

3.4.3 D-CSR Simulation Results

In this section, we consider a WSN with $N = 500$ sensors uniformly randomly distributed in a square sensing area 100×100 ($L = 100$). The path-loss exponent for all communication distances is chosen as $\alpha = 2.5$. As we assumed, all sensors have the same transmission range R but they can adjust their power while communicating. We work on both sparse signals and real sensor readings. The real sensor readings we chose to use are collected from Sensorscope: Sensor Networks for Environmental Monitoring [136].

Since R significantly affects the transmission energy which sensors spend on communicating with each other, we first simulate this case with different values of R. We start with a smallest $R = 10$ to satisfy the network being always connected. As we increase R, the number of sensors in each neighborhood in the graph increases as shown in Figure 3.5. The maximum number of sensors in each neighborhood is equal to the total number of sensors in the whole network.

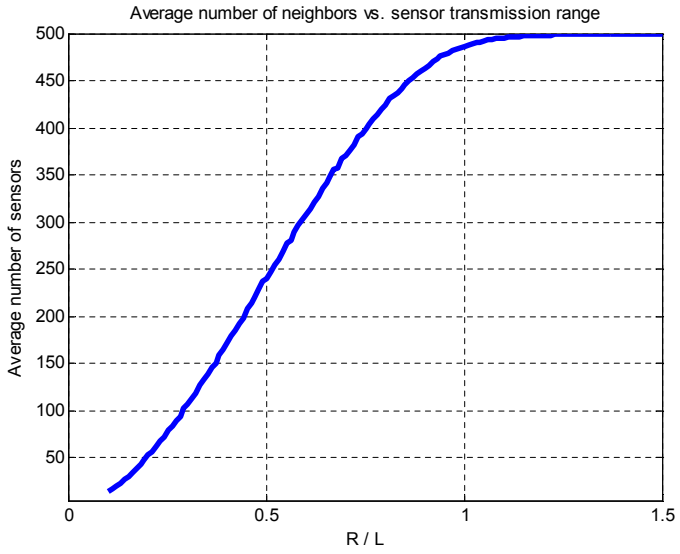


Figure 3.5: The average number of neighbors of each sensor when changing the sensor transmission range R

Based on the connected graph, we derive the probability transition matrix $[p_{ij}]$

that follows Equation (2.12). As shown in Figure 3.6 the mixing time required for all sensors to achieve the uniform distribution asymptotically reduces as we increase the transmission range R. It leads to a trade-off to choose an optimal value of R for the least energy consumption.

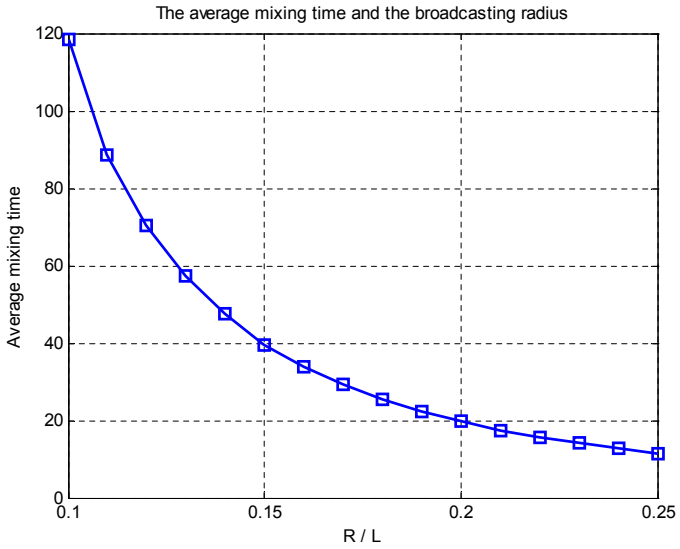


Figure 3.6: The mixing time reduces as the sensor transmission range R increases

Figure 3.7 illustrates the sensor coverage of the network. We need to consider how many RWs can sample the entire network area with the chosen RW's length τ. It shows that 90 RWs with the length of 48 can visit all nodes in the network.

In order to check the accuracy of the statistical calculation of the distance between RWs and the BS, we consider the BS at many different positions and compare the analysis and simulation results. As shown in Figure 3.8, Equations (3.16) and (3.19) are calculated precisely when either the BS is at the center ($L_i = 0.5L$) or outside the sensing area ($L_i > L$).

As we receive the lengths of RWs corresponding to the different transmission ranges as mentioned above, we can calculate the total power consumption for the network. Figure 3.9 shows the average power consumption at each value of transmission range. We can see that the network consumes the lowest energy at $R^* = 14$ or at

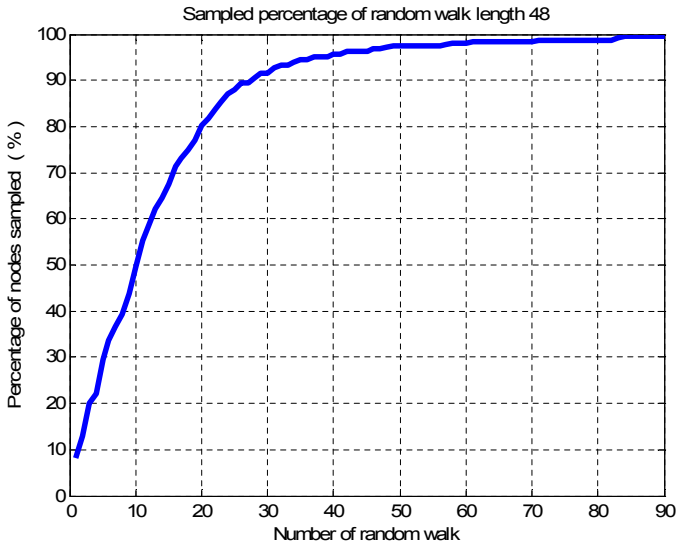


Figure 3.7: Sampling coverage the network with different number of random walks length 48 ($\tau = 48$)

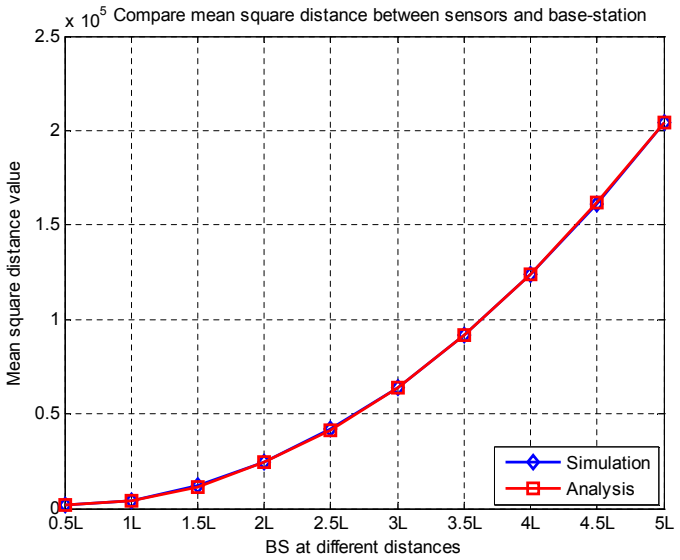


Figure 3.8: The average square distance ($E[d^2_{toBS}]$) between RWs and the BS at different positions $L_i \geq 0.5L$ up to $L_i = 5L$

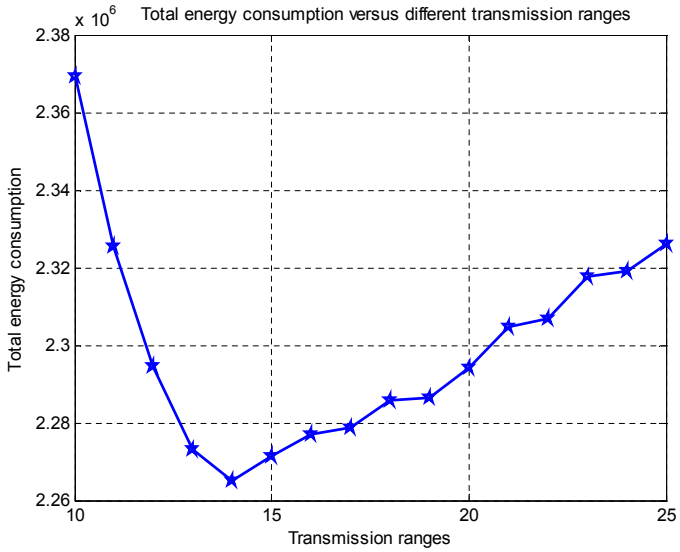


Figure 3.9: Total power consumption of the network versus sensor transmission ranges when BS at the center of the sensing area

the rate $R/L = 0.14$. Based on these results, we can suggest it as the optimal range for the sake of prolonging the network lifetime.

The total power consumption is minimized at $R^* = 14$ when the RW length is $\tau = 48$ as shown in Figure 3.6. Each RW only visit at most 48 random nodes for building one CS measurement. It also means that the corresponding row in the projection matrix has only 48 or less non-zero elements since a RW may visit a node more than once. As mentioned in [125], an appropriate sparse binary projection matrix can work as well as a full-Gaussian matrix for k-sparse signals with different k values. We also have worked on real temperature sensor readings with different number of CS measurements from 60 to 90. The normalized reconstruction errors ($\frac{\|\underline{X}-\hat{\underline{X}}\|_2}{\|\underline{X}\|_2}$) are used to compare the two different measurement matrices, the full dense Gaussian and the sparse binary matrix collected from the RWs. It is shown in Figure 3.10 that the optimal length of RWs ($\tau = 48$) works well for either energy saving or CS reconstruction.

Figure 3.11 illustrates the total energy consumption for 90 RWs with different RW

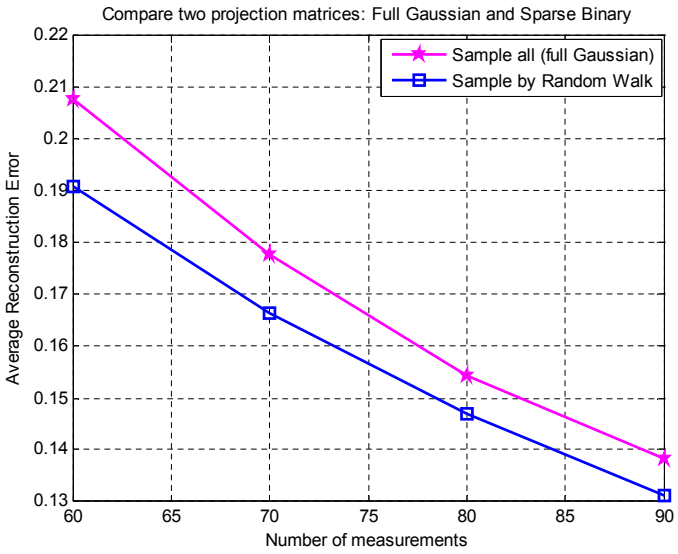


Figure 3.10: Comparison between the full dense Gaussian and the sparse binary matrix collected different number of RWs: random walk length $\tau = 48$ with different number of measurements

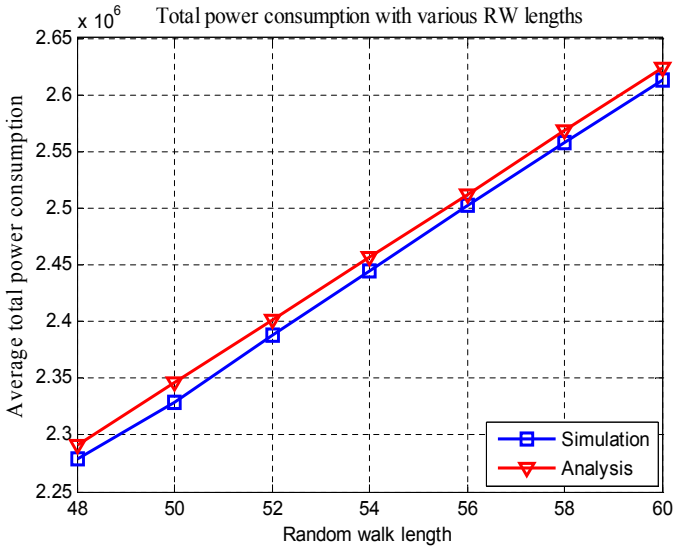


Figure 3.11: Total power consumption through all data collection processes with M = 90 measurements, transmission range R* = 14 in different RW's lengths when the BS at the center of the sensing area

lengths. It is clear that the total energy is a linear function of the RW length τ.

3.5 Multi-hop Relaying Data from Random Walks to the Base-station (M-CSR)

In this section, we extend our previous work by proposing a multi-hop transmission method to relay CS measurements though intermediate nodes to the BS instead of sending them directly. We also propose a distributed greedy algorithm to form a spanning tree with the root at the BS. In the method, sensors do not require global information from the others as the minimum spanning tree [137] or the shortest path tree [138]. After a RW finishes collecting data, the CS measurement is transmitted following the tree to the BS. We formulate the total energy consumption for the network and also suggest an optimal sensor transmission range R^*.

3.5.1 Network Model

In this model, we assume that the sensing area has a circular shape in which the BS is at the center. We also assume that sensors still can adjust their power level to collect data while walking through the number of sensors corresponding to the length of RWs. After that, sensors use transmission range, denoted as R to connect to their parent nodes in the tree to relay CS measurements to the BS. We still have N sensors uniformly distributed in the circular area of radius R_0, and also, the path-loss exponent is still assumed to be equal to 2.5 ($\alpha = 2.5$) as used in the previous sections.

3.5.2 Multi-hop Relaying Data Algorithm

We propose a distributed greedy algorithm to form Multi-hop Relaying for RWs, named M-CSR: We assume all sensors have an equal transmission range (R) that allows them to communicate to each other within range R. An appropriate R should be chosen depending on the node density of the entire network ($\frac{N}{\pi R_0^2}$) to ensure that

all sensors are connected as an undirected geometry graph. Based on the graph, we can deploy M-CSR to form the routing paths for the sensors as: All nodes broadcast their information about their number of hops away from the BS to their neighbors. At the first iteration, only nodes close to the BS (the R overlap the BS) have the number of hops (NoH). They name their NoH as "1" and broadcast to their neighbors in the next iterations. A node choose to connect to one of its neighbors which has the smallest NoH. After a few iterations, the routing paths may be formed but not completely done because a sensor only chooses one of its neighbors having NoH while the rest may not have one after a few iterations. So the algorithm keeps running until there is no change of routing paths between all sensors. This algorithm can be written shortly as below:

All nodes connected as a graph with the same range R

1. ***While*** *(the routing paths is changing)*
2. *NoH(BS) = 0; i ∈ N nodes*
3. *Nei = set of i's neighbors*
4. ***if*** $d_{ij} < R$, *where j ∈ Nei*
5. *sensor(i) chooses sensor(j) when NoH(j) = min{NoH(Nei)}*
6. *name NoH(i) = NoH(j) + 1*
7. ***end if***
8. ***end while*** *(Until no change of routing paths between all nodes)*

Figure 3.12 is created by running the algorithm with 500 sensor randomly deployed in the circular area with radius $R_0 = 50$. The transmission range is chosen based on the optimal one in the previous part (D-CSR), $R^* = 14$. The optimal transmission range in M-CSR could be different based on the multi-hop routing part that will be shown later in the simulation results.

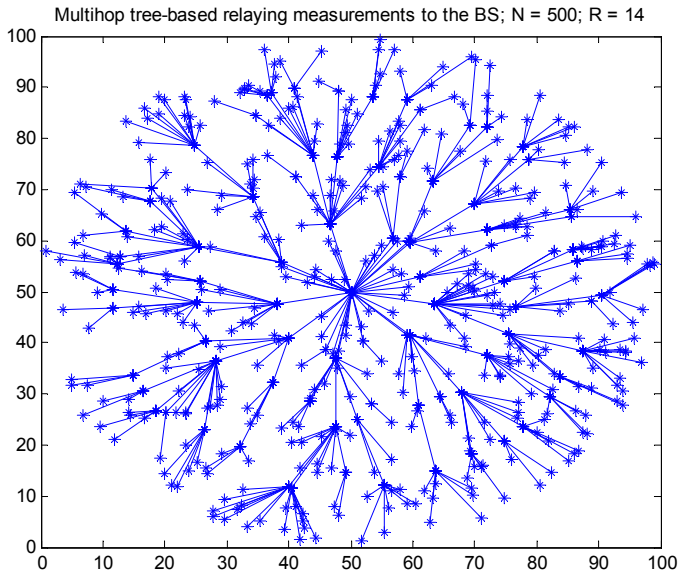


Figure 3.12: Tree-based relaying measurements after each RW to the base-station formed with 500 nodes and transmission range $R = 14$.

3.5.3 M-CSR Power Consumption Analysis

The total power consumption generally in WSNs using RWs is $E_{total} = (E_{RW} + E_{to\,BS})$ as mentioned in Equation (3.6). The consumed power for only RW part is already calculated in Equation (3.13) as $E_{RW} = M\tau(\frac{R^2}{2})^{(\alpha/2)}$, where R is the transmission range and M is the number of measurements required or the number of RWs.

$E_{to\,BS}$ is calculated after we have the tree-based multi-hop routing formed. Since we use multi-hop transmission rather than directly transmitting data from RWs to the BS, we need to formulate this consumed energy as follows

$$E_{toBS} = \sum_{i=1}^{M} NoH(i) \times R^{\alpha}, \tag{3.21}$$

where R^{α} is the energy consumption based on the transmission range spent on each hop relaying measurements.

In [139], Chandler calculated the average number of relay hops in randomly located radio network. Based on the idea, equation (4.29) can be written as

$$E_{toBS} = NoH_{ave} \times R^{\alpha} \times M, \tag{3.22}$$

where NoH_{ave} is the average number of hops calculated as E[n] in [139]. The expectation of the number of hops is calculated based on the probability of being able to make a connection between a random node and the BS. Sensors are supposed to have an equal transmission range. If an area covered by a transmission range does not include a destination as the BS, there must be at least one node existing in the radio overlapped area called A to relay data.

The number of sensors existing in the area A follows Poisson distribution with the mean value $\lambda = \frac{N_c}{\pi R_0^2} \times A$. The probability of being able to make a connection between a random node and the BS is

$$P(\#ofnodes \geq 1) = 1 - P(\#ofnodes = 0) \tag{3.23}$$

$$= 1 - e^{-\frac{N}{\pi R_0^2} \times A}, \tag{3.24}$$

where $A = 2R(2\theta - sin\theta cos\theta)$ and $\theta = cos^{-1}(x/2R)$.

Since sensor nodes are randomly distributed and are chosen in the sensing area, the distance between any sensor and the BS denoted as x can be considered as a random variable. The probability of being able to make a connection at distance x using NoH or less hops is denoted by $P_{NoH}(x)$. Chandler [139] provides the expectation value of the number of hops in a random network as follows

$$\mathrm{E}[NoH] = \sum_{NoH=1}^{max(NoH)} n[P_{NoH}(x) - P_{NoH-1}(x)]/P_{max(NoH)}(x) \tag{3.25}$$

$$= max(NoH) - \sum_{NoH=1}^{max(NoH)-1} \frac{P_{NoH}(x)}{P_{max(NoH)}(x)}, \tag{3.26}$$

where $max(NoH)$ is the maximum number of hops allowed. Finally, we obtain the power consumption for RWs relaying M measurements to the BS formulated as

$$E_{toBS} = \left\{ NoH_{max} - \sum_{NoH=1}^{NoH_{max}-1} \frac{P_{NoH}(x)}{P_{NoH_{max}}(x)} \right\} R^{\alpha} \times M. \tag{3.27}$$

Now we analyze the transmission range R. In each routing path, the number of hops is directly related to R. If we increase R, a sensor could reach further nodes

to choose one of them to forward measurements. This means that total number of hops can be reduced or increased with variable values of R, which may effect power consumption. For example, if we increase R, the number of hops in each routing path might be reduced. But we have to deal with longer hop distance that consumes more power. In addition, R is strongly related to the RW length τ as shown in Figure 3.6 that might provide us a trade-off of choosing an optimal R^*.

3.5.4 M-CSR Simulation Results

These conditions are simulated using an arbitrary circular network having radius $R_0 = 50$. All 500 sensors are uniformly randomly distributed and have the same transmission range R. The path-loss exponent is still assumed as $\alpha = 2.5$.

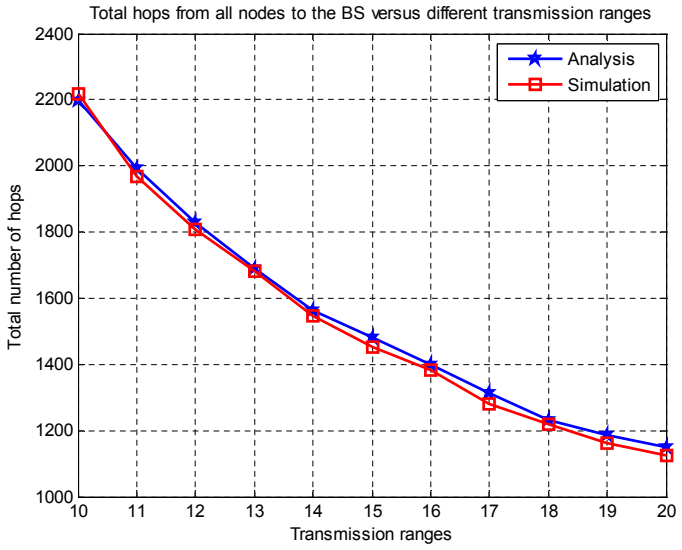


Figure 3.13: Total number of hops from all sensor nodes to the BS as we increase the transmission range R

In Figure 3.13, the transmission ranges are chosen as $R = \{10 \div 20\}$ to consider the total number of hops from all nodes on the relaying tree to the BS. It shows the accuracy of Equation (3.26) and clarifies that the total number of hops degraded corresponding to the transmission range increased. As mentioned in the M-CSR

analysis section, increasing R might increase E_{toBS} but also reduces the RW length τ as calculated in the D-CSR section.

Figure 3.14 shows the total power consumption for the network while different values of transmission ranges are chosen for the sensors in the network. This provides us the optimal transmission range in this case, $R^* = 12$.

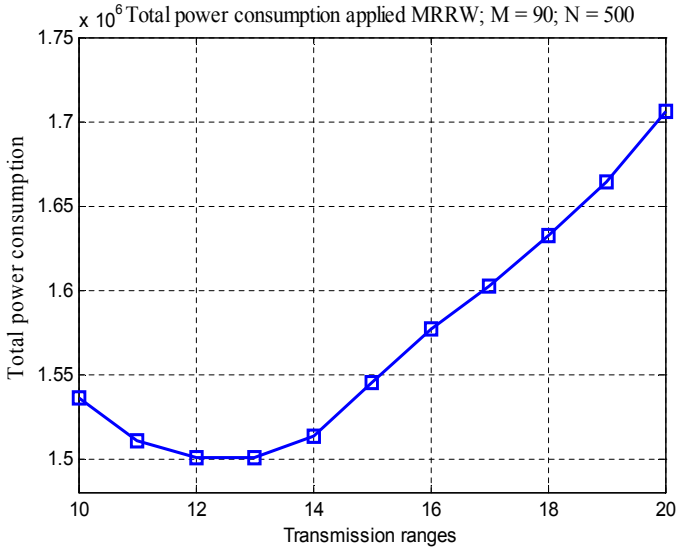


Figure 3.14: The total power consumption applied M-CSR versus different transmission ranges R when BS at the center; R* = 12

Based on Figures 3.13, 3.14, we should choose the optimal transmission range $R^* = 12$ that results the least consumed power for the network.

Figure 3.15 compares the total power consumption from the two methods, D-CSR and M-CSR. Please note that our calculation for power consumption for D-CSR was based on the assumption of a square sensing area. We extended our results for the case of a circular sensing area with BS at the center in Appendix A. In Figure 3.15, the total consumed power is a linear function of the number of measurements required or the number of RWs. We can save up to $20 - 30\,\%$ when using M-CSR instead of D-CSR.

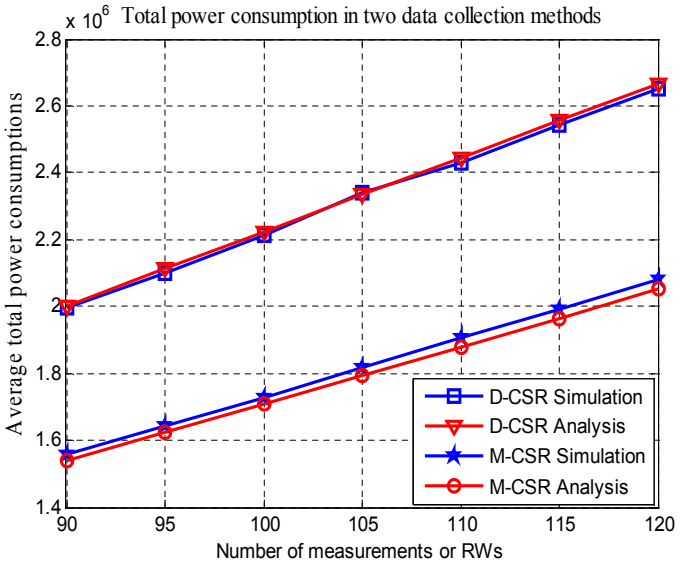


Figure 3.15: Compare the total power consumption in two random walk routing method when BS at the center and $R = 14$.

3.6 Conclusion and Future Work

In this chapter, the integration between CS and RW for the purpose of saving energy for monitoring data in WSNs was presented and analyzed. It is well known that natural signals have spatial correlation and therefore the sensor readings in a WSN are sparse in a proper basis that facilitates the utilization of CS for energy-efficient data collection in such networks. In our proposed method, a certain number of sensors are chosen to initiate RWs. Each RW keeps adding sensory data from nodes it visits and finally forward the CS measurement to the BS for the CS recovery processes in two ways: directly (one hop) and multi-hop fashion, called D-CSR and M-CSR, respectively. The RW length is chosen as the mixing time that allows the RWs to visit sensors node with an approximately equal probability.

We investigated the trade-off between the transmission range R and the mixing time τ to suggest an optimal R^* for the networks to consume the least power. Furthermore, we analyzed and formulated all power consumptions collecting data in such networks based on stochastic problems. We also compared both models of transmit-

ting CS measurements to the BS, D-CSR and M-CSR. The total power consumption in M-CSR reduces up to 30% compared to D-CSR in both analysis and simulation. Since we do not consider other characteristics such as latency or capacity in the networks, the multi-hop routing (M-CSR) is suggested for energy saving.

In future work, we will exploit how the sparsity of the projection matrix to save the power consumption for routing methods in WSNs utilizing CS.

CHAPTER 4

CLUSTER BASED DATA COLLECTION IN WIRELESS SENSOR NETWORKS

4.1 Introduction

4.1.1 Motivation

Wireless sensor networks (WSNs) have found numerous uses in both military and civilian applications [118]. Sensors in WSNs are usually randomly dropped/deployed in a sensing area that needs to be monitored. They are often deployed in harsh conditions without maintenance or renewable power supply. Therefore, the connection and operation of these networks rely on these small and inexpensive devices under a severe energy constraint. Saving energy in data collection in such networks is always a critical problem that directly impacts network lifetime.

The spatial correlation of the sensor readings in WSNs results in an inherent sparsity of data in a proper basis. This sparsity facilitates the application of the compressive sensing (CS) [5, 6, 8] technique in data collection in WSNs [9, 10, 140]. CS offers a novel framework to reconstruct all sensor readings based on a small number of CS measurements, which creates an opportunity to significantly reduce power consumption.

In recent years, there have been several studies on the integration of CS and data collection in WSNs, e.g., [122, 123, 124, 13, 15]. In these methods, sensor readings are multiplied by a selected set of coefficients and are sent to the base station (BS) following some routing methods such as gossip-based, random walk, tree-based, or

cluster-based. The CS measurements are collected at the BS as $\underline{Y} = \Phi \underline{X}$, where Φ is called the measurement matrix and vector $\underline{X}$ represents all unknown readings from all the sensors. The resulting measurement matrices can be sparse or dense with Gaussian coefficients depending on the underlying routing method.

In this chapter we combine the clustering technique, which has been shown to save and balance energy consumption for WSNs, and block diagonal matrices (BDMs) as the CS measurement matrices. We propose an algorithm called Cluster-Based Compressive Sensing Data Collection (CCS) in which **the CS measurements are generated at each cluster-head** (CH) in the clustered networks. We consider two methods to send these measurements to the BS: directly (one-hop) and multi-hop relaying through the intermediate CHs. In the BDM, the size of each sub-matrix (block) depends on the size of each cluster. We formulate the total power consumption and discuss the effect of different sparsifying bases on the CS performance as well as the optimal number of clusters for reaching the minimum power consumption. For our formulations, two common positions for the BS are considered: the BS located at the center and outside the sensing area. Based on that, we can obtain the optimal number of clusters that provides the minimum power consumption for our networks. In our simulation, we consider both random sparse signals in canonical basis and real sensor readings. The real sensor readings, although not sparse in canonical domain, are sparse in frequency domains (DCT or Wavelet). We compare different combinations between the measurement matrices and the sparsifying matrices in our theoretical and simulation results. Our work shows promise not only in WSNs but also in mobile sensor networks or vehicle networks for data monitoring or similar purposes. The power saving in communications prolongs the lifetime in such networks.

4.1.2 Related work

Clustering is an effective way to enhance the performance and lifetime of a WSN and many different clustering algorithms have been studied [34, 43, 51, 61, 141]. Each cluster has a cluster head (CH) and CHs can be pre-determined [34] or be selected while doing clustering as in the following algorithms. K-means [41, 38, 42] is a well-known and simple clustering algorithm that chooses CHs for K clusters at the central point of each cluster. Since power consumption is dependent on transmission distance, this helps to minimize the intra-cluster power consumption. In general, CHs consume power much more than other sensors as they transmit the entire cluster's data to the BS. In LEACH [141], sensor nodes randomly elect themselves to be CHs. This way, the high power consumption related to communication with the BS will be distributed among the nodes in the network. The HEED algorithm [142] chooses CHs based on the highest residual energy of sensors to balance network energy. EEUC [57] makes unequal size clusters and muti-hop links between CHs to reduce and balance the power consumption. Fault-tolerant clustering is considered in [37] in order to recover sensors in a failed cluster. Load-balancing clustering [35] makes the whole network consume power equally and [50] finds the optimal number of clusters to get the lowest power consumption for WSNs.

Utilizing CS is also an effective way to reduce the number of required samples from a sparse signal. Due to the correlation between the sensor readings in a WSN, the monitored signal can have a sparse representation in a proper domain such as DCT or wavelet. Accordingly, CS has found applications in data collection in WSNs [9, 10, 140]. In [123] a tree-based algorithm called CDG is proposed to balance the payload falling on nodes close to the BS. The measurement matrix is a full Gaussian one that consumes more power than the sparse binary measurement matrices [125]. To reduce power consumption, other methods based on sparse CS matrices have been proposed [143, 19]. In [143] the authors proposed MTT, which is an heuristic

algorithm to compute a minimum transmission spanning tree for data collection in WSNs using CS. Another tree-based study is TCS [19] that utilizes sensor storage to reduce the number of transmissions in a routing tree.

Hybrid data collection schemes where both raw data and combined samples are in traffic are mentioned in [104, 105, 106]. In [104] CS operation requires each node in the WSN to send exactly M packets. M is denoted as the number of CS measurements required to reach a given reconstruction quality. A spanning tree is used to partition the network into sub-nets. [105] and [106] propose a scheme called hybrid CS aggregation that combines the non-aggregation and plain CS mentioned in [104] to reduce the traffic loads sending from each node. The non-aggregation method is used if a node receives less than $(M-1)$ raw readings from its downstream nodes. Otherwise, plain CS is used. In [104, 105, 106], each sensor needs to send up to M samples to the BS to contribute M CS measurements for signal recovery. The BDM is mentioned in [106] to reduce partially the total samples being sent from each sub-net. In [107], a WSN is partitioned into clusters. Sensor readings are sent to CHs and the CHs send the received data to the BS. Since the measurement matrix is full Gaussian, each cluster needs to generate M samples to contribute to M CS measurements at the BS.

This work extends our previous studies [15, 17]. In our proposed CCS algorithm, all non-CH sensors send their own readings to their corresponding CHs *only once* during M rounds of measurement collection. The CHs generate sub-matrices with Gaussian coefficients (ϕ_i) and generate M_i CS measurements using $\underline{y}_i = \phi_i \underline{x}_i$, where $\underline{y}_i$ is the measurement vector collected from the i^{th} cluster, and $\underline{x}_i$ represents all readings in the i^{th} cluster. The CS measurements are either sent directly from CHs to the BS or in a multi-hop fashion. We provide Table 4.1 to compare our data collection method with other related studies. In the table, we focus on the network structures, the number of times each sensor sends its reading to its CH or the subnet-head to

contribute to M CS measurements, and the number of combined data packets being sent from each CH or subnet-head to contribute to M CS measurements.

Table 4.1: Comparison between the existing data collection methods and CCS

Related papers	[104, 105, 106]	Partly [106]	[107]	CCS
Network structure	Spanning tree	Spanning tree	Cluster	Cluster
Measurement matrix	Full dense Gaussian	BDM	Full dense Gaussian	BDM
Number of times sending data from each sensor	Up to M times	M_i times	Only once	Only once
Number of packets sending from each cluster or subnet	M	M_i	M	M_i ($\frac{M_i}{M} = \frac{N_i}{N}$)
CS measurements are generated at	Base-station	Subnets	Base-station	Cluster-heads

The main contributions in this chapter are summarized as follows:

1- Two versions of the CCS algorithm (D-CCS and I-CCS), which combine clustering and BDMs as CS matrices, are proposed. CCS significantly reduces transmission power consumption in WSNs.

2- Expressions for power consumption in D-CCS and I-CCS are formulated.

3- The optimal numbers of clusters are suggested for the networks in different scenarios to consume the least power.

The remainder of this chapter is organized as follows. Problem Formulation is addressed in Section II. The CCS algorithm is stated in Section III. In Section IV and

Section V, the two models to forward the CS measurements to the BS are presented with corresponding transmission power consumption analysis and simulation results. In Section VI use of the DCT and sending only k large transformed coefficients is considered for comparison purpose since $k \ll M \ll N$. Finally, conclusions and suggestions for future work are presented.

4.2 Problem Formulation

4.2.1 System Model

In the network model considered here, we assume that N sensors have been distributed uniformly at random in a sensing area. N_c out of N nodes in the network are selected uniformly at random as CHs with probability $\frac{N_c}{N}$ and the other nodes connect to the closest CH, as mentioned in LEACH [141]. This creates N_c non-overlapped clusters with each cluster having one CH and $(\frac{N}{N_c} - 1)$ non-CH sensors on average. Each CH is assumed to have enough capacity to store the data vector collected from its own non-CH sensors and to generate a number of CS measurements required based on the number of sensors in the cluster.

We assume that each node can adjust its power level based on its distance from its CH and this can be done based on the received signal strength [144]. The consumed power for reaching a destination node j with distance d_{ij} from the node i is[1] $P_{ij} = d_{ij}^{\alpha}$. Parameter α is called the path loss exponent, which is usually between 2 and 4, depending on the characteristics of the channel [135]. In this chapter, we assume $\alpha = 2$. For the reconstruction error related to CS signal recovery we consider the normalized reconstruction error $\frac{\|\underline{X}-\underline{\widehat{X}}\|_2}{\|\underline{X}\|_2}$.

[1]In fact, we have $P_{ij} \propto d_{ij}^{\alpha}$ [144]. However, since we are interested in a comparison of different schemes, and not the exact values of the power, without loss of generality, we can consider the constant factor as one.

4.2.2 Block Diagonal Matrices

As mentioned before, our goal is to utilize block-wise CS for data collection in clustered WSNs. Since CS measurements are formed at each CH, the overall CS measurement matrix formed at the BS will no longer have the form of the conventional CS matrices, such as a *dense* matrix with all the entries being i.i.d. Gaussian or Rademacher [2]. Instead, our CCS algorithm results in *block diagonal matrices* (BDMs), in which ϕ_i, the i^{th} block in ϕ, corresponds to the i^{th} cluster and has i.i.d. Gaussian entries. Let $\underline{x}_i$ denote a vector of size N_i consisting of the sensor readings of the nodes in the i^{th} cluster, and $\underline{y}_i = \phi_i \underline{x}_i$ denote a vector including M_i CS measurements collected from the i^{th} cluster. We have

$$\underbrace{\begin{bmatrix} \underline{y}_1 \\ \underline{y}_2 \\ \vdots \\ \underline{y}_{N_c} \end{bmatrix}}_{\underline{Y}:\, M\times 1} = \underbrace{\begin{bmatrix} \phi_1 & & & \\ & \phi_2 & & \\ & & \ddots & \\ & & & \phi_{N_c} \end{bmatrix}}_{\Phi:\, M\times N} \underbrace{\begin{bmatrix} \underline{x}_1 \\ \underline{x}_2 \\ \vdots \\ \underline{x}_{N_c} \end{bmatrix}}_{\underline{X}:\, N\times 1} \tag{4.1}$$

The restricted isometry property (RIP) of BDMs has been studied in [145, 146] and it has been shown that BDMs can satisfy RIP and therefore can be used as efficient measurement matrices. The required number of the measurements though depends on the basis in which the signal is sparse. According to [145, 147], the number of measurements required for a BDM, consisting of N_c blocks with Gaussian entries, to satisfy RIP with high probability is given as [145]

$$M = O(k\,\widetilde{\mu}^2 log^2(k)\, log^4(N)), \tag{4.2}$$

where $\widetilde{\mu} = \min\{\sqrt{N_c}, \mu\}$ and $1 \leq \mu \leq \sqrt{N}$ is the coherence between ψ and canonical basis and defined as $\mu = \sqrt{N} \max |\psi_{ij}|$. From (4.2) several very interesting points can be concluded. If the sparsifying basis has a small coherence with the canonical

[2] A Rademacher random variable takes a value of +1 or -1 with equal probability.

basis (such as the Fourier basis or DCT basis), then increasing N_c (which results in a more sparse matrix) does not increase M. On the other hand, if the sparsifying basis has a large coherence with the canonical basis (such as Wavelet or Canonical bases), then for $N_c < \mu^2$, M is a linear function of N_c. These results will be seen in our simulations later (Figures 4.4, 4.9, 4.14).

4.2.3 Problem Formulation

Consider a vector $\underline{X} = [\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_{N_c}]$ in which $\underline{x}_i$ represents unknown sensor readings from cluster i. We assume the BS needs M CS measurements collected from the network to recover precisely all raw readings. The i^{th} CH ($i = 1, 2, \ldots, N_c$) generates an $M_i \times N_i$ block of Gaussian coefficients (ϕ_i), where N_i is the number of sensors in the i^{th} cluster. The CH then generates M_i CS measurements using $\underline{y}_i = \phi_i \underline{x}_i$ and sends them to the BS. The exact value of M_i out of M is calculated based on $\frac{N_i}{N}$) which is shown in Lemma 1 in the next section.

The BS receives random seeds to generate the BDM and the measurement vector $\underline{Y} = [\underline{y}_1, \underline{y}_2, \ldots, \underline{y}_{N_c}]$ separately from N_c CHs. The greater number of measurements, the better the accuracy of the reconstruction. In the following sections we will analyze the use of BDMs in the CS recovery processes with different types of signals; sparse in canonical basis or in frequency domain. Transmission power consumptions for data collection in the networks are formulated, analyzed and finally simulated in arbitrary networks. The optimum number of clusters such that power consumption is minimized will be determined for each network.

4.3 CCS: Cluster-Based Compressive Sensing for Data Collection in WSNs

The proposed CCS algorithm is summarized in Algorithm 2 below. CCS is divided into two parts. The first is the underlying clustering that can be based on different

Algorithm 2: CCS - Cluster-Based Compressive Sensing Algorithm

(1)-**Clustering phase:**

- N_c out of N sensors are randomly chosen as CH_i $(i = 1, ...N_c)$ with probability $p = \frac{N_c}{N}$.

- Assign each sensor (S_j) in the entire network to the nearest cluster C_i as $S_j \in C_i$ if $\parallel S_j - CH_i \parallel < \parallel S_j - CH_t \parallel$ $for j = 1, ..., N$, $i \neq t$ and $i = 1, ...N_c$

(2)-**Measurement generating phase**

- Non-CH sensors send their data once to their CHs. Data vector $\underline{x_i}$ is stored at CH_i

- CS measurements are generated at each CH as

for *i = 1 to* N_c **do**

- $\phi_i = randn(M_i, N_i)$ % sub-matrix is created at each cluster

- $\underline{y}_i = \phi_i \underline{x}_i$ % M_i measurements are created at CH_i

end

(3)-**Measurement collection and Data recovery phase**

- $M = \sum_i^{N_c}(M_i)$ CS measurements are forwarded separately from CHs to the BS.

- Given the BDM ϕ, all unknown values $(\underline{\mathbf{x}})$ are reconstructed based on $(\underline{\mathbf{y}})$.

methods such as K-means [38] and LEACH [141] which we use for comparison purposes in our simulations. The second is the generation of CS measurements based on BDMs and forwarding them to the BS either directly or in multiple hops to be addressed in Sections 4.4 and 4.5, respectively. In a real WSN each cluster may have a different number of sensors and accordingly different numbers of measurements are required from each CH. The following lemma relates N_i and M_i.

Lemma 1 *Let M be the number of required measurements to be taken from all clusters to satisfy the RIP for a block diagonal matrix with blocks ϕ_i of size $M_i \times N_i$. To get the best CS performance in term of the reconstruction error, the number of measurements from the i^{th} cluster (M_i) should be linearly proportional to the number of sensors in the cluster (N_i). In other words, $\frac{M_i}{M} = O(\frac{N_i}{N})$.*

Proof. According to [7], the number of CS measurements required to reconstruct a k-sparse signal of length N using a dense Gaussian measurement matrix of size $M \times N$ is given as

$$M = O(klog\frac{N}{k}). \tag{4.3}$$

Now assume the k non-zero elements are uniformly distributed in the vector $\underline{X}$, and $\underline{X}$ has been partitioned into sub-vectors of size N_i. Therefore, we have

$$\frac{N_i}{N} = \frac{k_i}{k}. \tag{4.4}$$

Using (4.3), and considering that ϕ_i is a dense Gaussian measurement matrix we have:

$$M_i = O(k_i log\frac{N_i}{k_i}). \tag{4.5}$$

From (4.3), (4.4) and (4.5), we obtain

$$\frac{M_i}{M} = \frac{O(k_i logN_i/k_i)}{O(klogN/k)} = O(\frac{k_i}{k}) = O(\frac{N_i}{N}).$$

Hence,

$$\frac{M_i}{M} = O(\frac{N_i}{N}). \tag{4.6}$$

■

Although this lemma has been proven for the case that the signal $\underline{X}$ is sparse in the canonical domain, our simulation results, below, show that the lemma holds even when $\underline{X}$ is sparse in another domain. Here is an example. Assume a network of size N is divided into two clusters with sizes of $N_1 = 0.7N$ and $N_2 = 0.3N$. A fraction T and $1 - T$ of the total measurements are collected from cluster 1 and 2, respectively. Signal $\underline{x}$ is assumed to be sparse in DCT. Figure 4.1 depicts the reconstruction error versus T, when $N = 1000$ and $M = 250$. As we see the minimum error occurs when $T = M_1/M$ is exactly equal to $N_1/N = 0.7$. This is the same result stated in Lemma 1. Experiment with real sensor readings illustrates that the number of acquired measurements from a cluster should be proportional to the cluster size for the best CS performance.

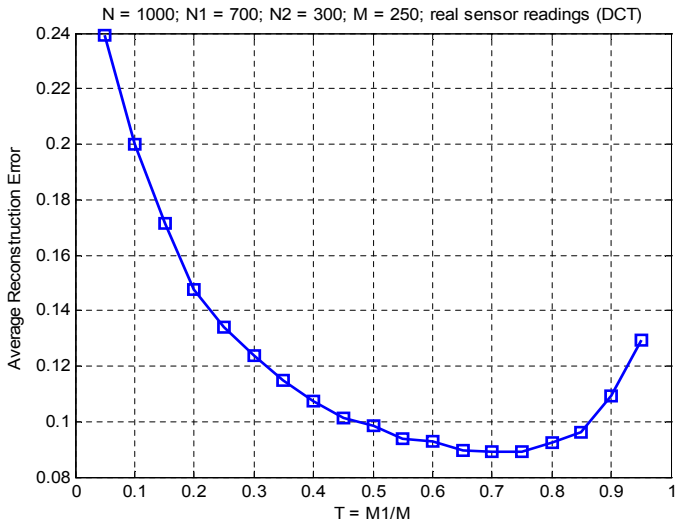


Figure 4.1: Average reconstruction error versus the fraction of the measurements collected from the first cluster ($T = M_1/M$). The error is minimize when T is equal to the fraction of the nodes in the first cluster ($N_1/N = 0.7$).

4.4 Directly Send CS Measurements to the BS (DCCS)

4.4.1 Network Model

In this model, we assume a WSN with N sensors deployed in a square sensing area sized $L \times L$ distance unit^2. Non-CH sensors send their readings directly to the CHs they belong to based on real distances (r). The CHs generate CS measurements and send them directly to the BS. The position of BS is changeable which can be inside or outside the sensing area.

4.4.2 Power Consumption Analysis for DCCS

We refer to the communication cost associated with the communication between the non-CH nodes and their CHs as the *intra-cluster* power consumption which is denoted as $P_{intra-cluster}$. The CHs create the CS measurements as combinations of all received data within each cluster ($\underline{y}_i = \phi_i \underline{x}_i$) and send the measurements directly to the BS. The corresponding power consumption is referred to as $P_{to\,BS}$. The total power consumption is formed as

$$P_{total} = (P_{intra-cluster} + P_{to\,BS}). \tag{4.7}$$

Analysis of $P_{intra-cluster}$

We assume a uniformly distributed WSN divided into N_c clusters with the same number of sensors as N/N_c, consisting of one CH and $(\frac{N}{N_c} - 1)$ non-CH nodes. We have

$$P_{intra-cluster} = N_C(\frac{N}{N_c} - 1)\, E[r^{\alpha}], \tag{4.8}$$

where r is a random variable representing the distance of a non-CH sensor to its corresponding CH and α is the path loss exponent that we assume to be 2 throughout

the chapter. We can calculate $E[r^2]$ as following:

$$E[r^2] = \int\int (x^2 + y^2)\, \rho(x, y)\, dx\, dy \tag{4.9}$$

$$= \int\int r'^2 \rho(r', \theta)\, r'\, dr'\, d\theta, \tag{4.10}$$

in which $\rho(x, y)$ is the node distribution. To make the analysis tractable, similar to [50], we assume each cluster area is a circle with radius $R = L/\sqrt{\pi N_c}$ and the density of the nodes is uniform throughout the cluster area, i.e. $\rho(r', \theta) = 1/(L^2/N_c)$. We have [50]:

$$E[r^2] = \frac{1}{(L^2/N_c)} \int_{\theta=0}^{2\pi} \int_{r'=0}^{R} r'^3 dr'\, d\theta = \frac{L^2}{2\pi N_c}. \tag{4.11}$$

and accordingly

$$P_{intra-cluster} = (\frac{N}{N_c} - 1)\frac{L^2}{2\pi}. \tag{4.12}$$

As we see, the total intra-cluster power consumption is a decreasing function of the number of clusters.

Analysis of $P_{to\,BS}$

Next, we need to determine $P_{to\,BS}$, which is based on the distances between CHs and the BS and the total number of measurements M required to be transmitted from each CH to the BS. We assume the BS is located at the location $(L_i, \frac{L}{2})$ with respect to our reference point (see Figure 4.2). The average consumed power by all CHs is given by

$$P_{to\,BS} = ME[d^2], \tag{4.13}$$

where d is a random variable representing the distance between the CHs and BS. Assuming that all CHs are randomly distributed in the sensing area, the expected

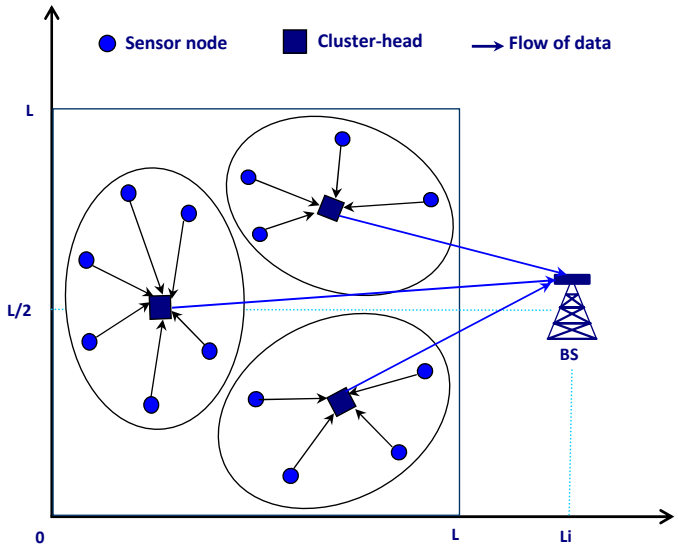


Figure 4.2: A clustered WSN with BS outside the sensing area ($L_i > L$).

squared distance between CHs and the BS is given by

$$E[d^2] = \int_0^L \int_0^L [(x - L_i)^2 + (y - \frac{L}{2})^2] f(x,y) dx dy \tag{4.14}$$

$$= \frac{1}{L}[\frac{(L - L_i)^3}{3} + \frac{L_i^3}{3}] + \frac{L^2}{12}, \tag{4.15}$$

in which $f(x,y) = \frac{1}{L^2}$ (uniform distribution of CHs). From Equations (4.13) and (4.15) we conclude that $P_{to\,BS}$ is independent of the number of the clusters. Using (4.7), (4.12), (4.13), and (4.15), the total power consumption can be formulated as

$$P_{total} = (\frac{N}{N_c} - 1)\frac{L^2}{2\pi} + \frac{M}{L}[\frac{(L - L_i)^3 + L_i^3}{3}] + \frac{ML^2}{12} \tag{4.16}$$

We usually have two common positions for the BS, at the center of the sensing area ($L_i = L/2$) and outside the sensing area ($L_i \geq L$). For the former case, (4.16) is simplified as

$$P_{total} = (\frac{N}{N_c} - 1)\frac{L^2}{2\pi} + \frac{ML^2}{6}. \tag{4.17}$$

According to (4.2) [145], we can see that for canonical and wavelet bases, the number of required measurements is a linear function of N_c. Based on this, we can state the following lemma to find the optimal number of clusters N_c^* for minimizing the power consumption.

Lemma 2 *Assume the number of required measurements is a linear function of the number of clusters, i.e., $M = aN_c + b$, where a and b are appropriate constants. In order to achieve the lowest power consumption with CCS, the optimal number of clusters is given by*

$$N_c^* = \sqrt{C\,N} = O(\sqrt{N}), \tag{4.18}$$

where

$$C = \frac{6L^3}{4\pi a\,[(L-L_i)^3 + L_i^3] + \pi a\,L^3}. \tag{4.19}$$

Proof. Adding the linear function of M mentioned in the lemma into the general equation of the total power consumption (4.16), we have

$$P_{total} = (\frac{N}{N_c} - 1)\frac{L^2}{2\pi} + \frac{aN_c + b}{L}[\frac{(L-L_i)^3 + L_i^3}{3}] + \frac{(aN_c + b)L^2}{12}. \tag{4.20}$$

We have

$$\frac{dP_{total}}{dN_c} = -\frac{NL^2}{N_c^2 2\pi} + \frac{a}{L}[\frac{(L-L_i)^3 + L_i^3}{3}] + \frac{aL^2}{12}. \tag{4.21}$$

By forcing $\frac{dP_{total}}{dN_c} = 0$, we can obtain the optimal number of cluster N_c^* calculated as

$$N_c^* = \sqrt{\frac{6L^3N}{4\pi a[(L-L_i)^3 + L_i^3] + \pi aL^3}} = \sqrt{C \times N}. \tag{4.22}$$

So,

$$N_c^* = O(\sqrt{N}). \tag{4.23}$$

■

4.4.3 Simulation Results for DCCS

In this section, we work with both random k-sparse signals (sparse in a canonical basis, i.e, ψ is the identity matrix) and real sensor readings (which are sparse in DCT or wavelet bases). We create a random network with number of sensors $N = 2000$ and size $L = 100$ according to the network model from Sections 4.2.1 and 4.4.1. We use K-means and LEACH clustering algorithms to arrange sensors into N_c

clusters. Then, we apply our CS-based data collection and calculate the total power consumption of the network for collecting M CS measurements that is required for reaching a target error rate of 0.1. The number of measurements from each cluster is linearly proportional to the size of the cluster based on Lemma 1. Simulation results based on K-means and LEACH clustering as well as the analytical results derived in Section 4.4.2 are provided below.

Figure 4.3 shows the histogram for the number of sensors in each cluster for both K-means and LEACH when $N_c = 10$. K-means generates clusters that are more uniform in size, resulting in a lower expected intra-cluster power consumption since it aims to minimize the within-cluster sum of squares [38]. Next we find the number of

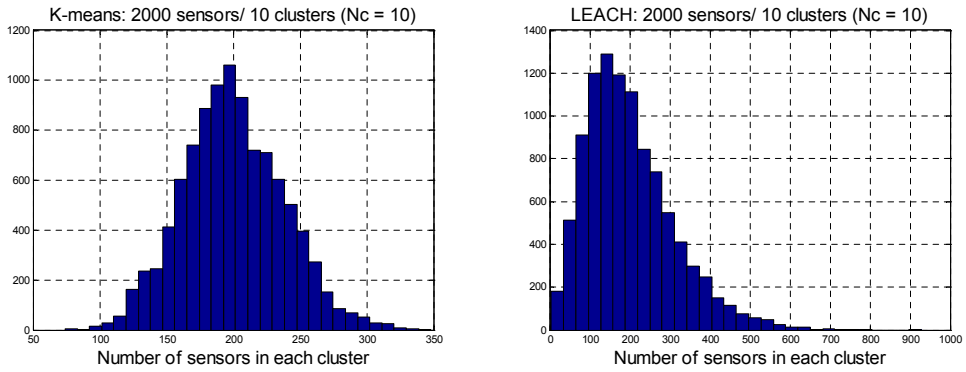


Figure 4.3: Histogram of number of sensors in each cluster for K-means and LEACH.

measurements required based on CS to satisfy a *target error* for our network when it is clustered into different number of clusters ($N_c = [1\ 2\ ...\ 50]$). Each clustering method provides a different BDM as the measurement matrix. For deriving our analytical results, we assumed clusters with equal size, while K-means and LEACH have different size blocks. For comparison, we also generated a BDM with all equal size blocks as the measurement matrix and found the number of required measurements to reach the target error. This is referred to as CS-based uniform clustering to compare with other methods. We choose a fixed target error in all our simulations as *target error*= 0.1. After finding the number of measurements required, we will find the power

consumption for different choices for the location of the BS as mentioned before. We present our first simulation with random k-sparse signals, then real sensor readings as actual temperatures.

$\underline{X}$ as a random k-sparse vector

In this example, we consider $\underline{x}$ to be sparse in the canonical basis. We create a 100-sparse vector $\underline{X}$ with length $N = 2000$. The measurement matrix is an $M \times N$ BDM, where M is the number of measurements required to satisfy the target error of 0.1. We obtain the number of required measurements for three algorithms as shown in Figure 4.4.

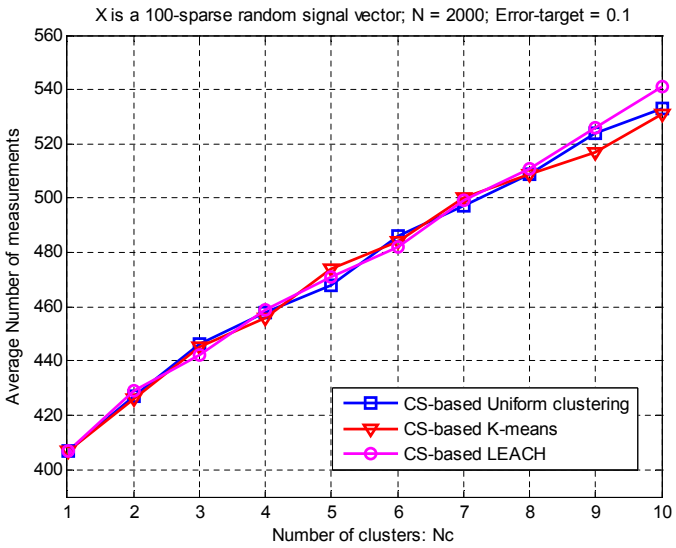


Figure 4.4: Number of measurements required to satisfy target error = 0.1 for a 100-sparse signal (sparse in canonical basis).

As shown in Figure 4.4, increasing N_c leads to a degradation in the CS performance and an approximately linear increase in the number of required measurements (as discussed in Section 4.2.2). This increases $P_{to\,BS}$. On the other hand, $P_{intra-cluster}$ is a decreasing function of N_c. Therefore, there is an optimal N_c^*, for which the total power consumption is minimized. Figure 4.5 depicts P_{total} when the BS is at the center of the sensing area. In this case, we have $N_c^* = 14$.

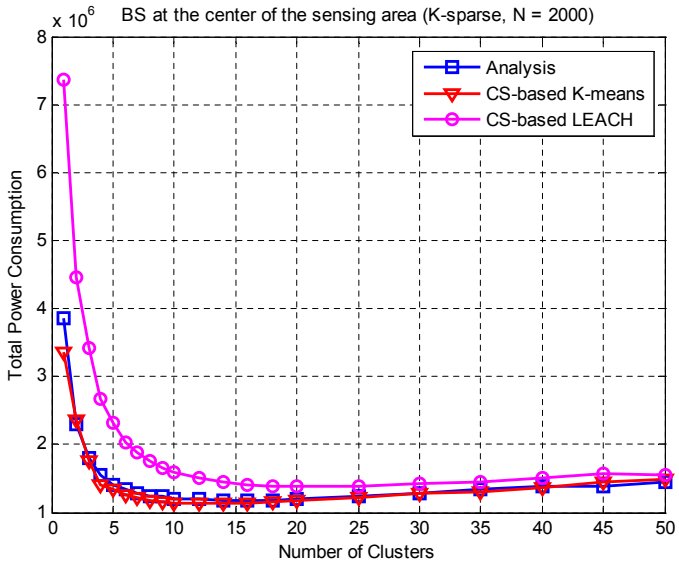


Figure 4.5: Total power consumption when BS at the center of the sensing area. Here, $N_c^* = 14$.

Figures 4.6, 4.7 and 4.8 depict P_{total} when BS is outside the sensing area at different locations. In the first case, the minimum power consumption occurs for $N_c^* = 9$. Figures 4.7 and 4.8 show P_{total} when the BS is far from the sensing area. The optimal number of clusters is $N_c^* = 4$ and $N_c^* = 2$ for $L_i = 2L$ and $L_i = 3L$, respectively. In such cases, $P_{to\,BS}$ will be the dominating factor in P_{total} and accordingly N_c^* becomes smaller. It is worth nothing that the results with K-means clustering match the analytical results much better than the case with LEACH clustering. This is due to the nonuniform cluster sizes in LEACH compared to K-means (Figure 4.3).

$\underline{X}$ as real sensor readings

We use real sensor readings from Sensorscope: Sensor Networks for Environmental Monitoring [136]. $\underline{x}$ is dense in the canonical domain. In order to apply CS, as mentioned in the background section, we need a sparsifying basis. Next, we will consider the utilization of both DCT and wavelet bases for this purpose.

**** Wavelet as the sparsifying basis*** : This case is similar to the case dis-

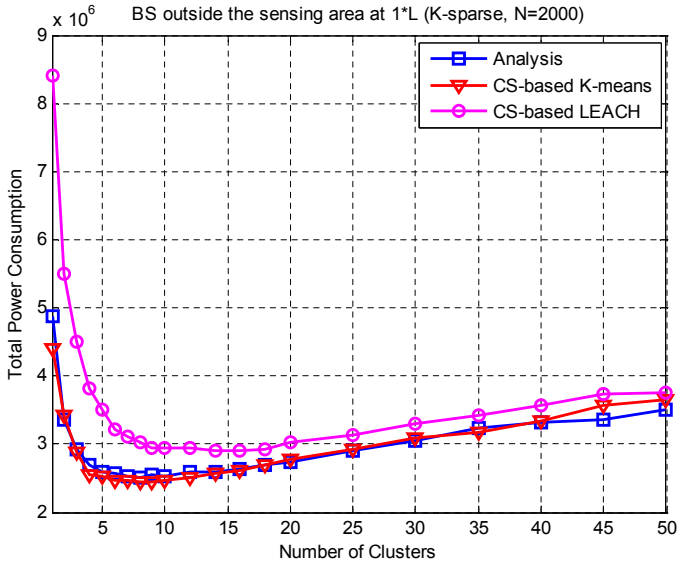


Figure 4.6: Total power consumption when BS at 1L ($L_i = L$). Here, $N_c^* = 9$.

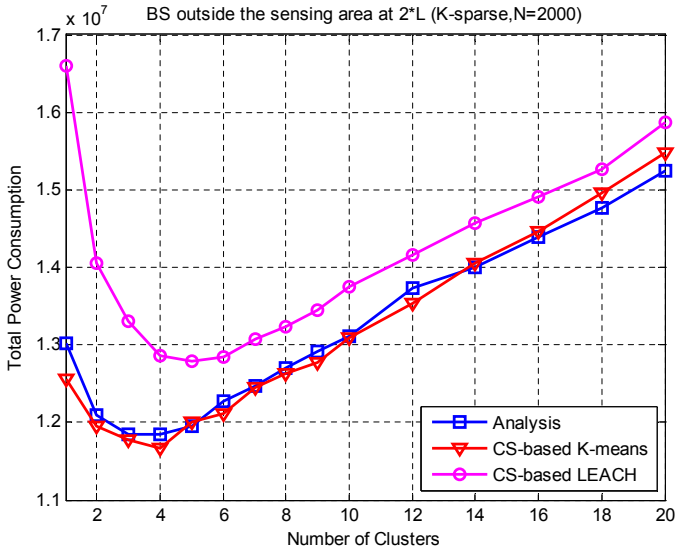


Figure 4.7: Total power consumption when BS at 2L ($L_i = 2L$). Here, $N_c^* = 4$.

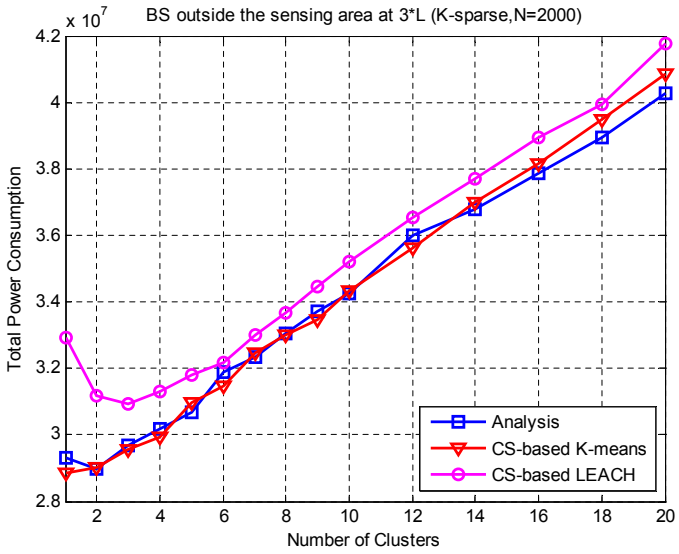


Figure 4.8: Total power consumption when BS at 3L ($L_i = 3L$). Here, $N_c^* = 2$.

cussed in Section 4.4.3 in the sense that the wavelet basis also has a large coherence μ. As discussed in Section 4.2.2, this causes a linear increase in the number of required measurements versus N_c. Our simulation results in Figure 4.9 depict this fact. Similarly, there will be an optimal N_c^*, for which the total power consumption is minimized. Figures 4.10, 4.11, 4.12 and 4.13 depict P_{total} when the BS is at the center, $Li = L$, $Li = 3L$, and $Li = 5L$, respectively. The optimal number of clusters are $N_c^* = 18$, $N_c^* = 12$, $N_c^* = 2$ *or* 3 (depending on the clustering scheme), and $N_c^* = 2$, respectively. As we make the BS farther from the sensing area, $P_{to\,BS}$ becomes a more dominating factor in P_{total} and this leads to a decrease in N_c^*.

**** DCT as the sparsifying basis*** : In this case we employ DCT as the sparsifying basis. As discussed in Section 4.2.2, the DCT is incoherent with the canonical basis and the CS performance does not degrade with increasing N_c. This can be seen in our simulation results shown in Figure 4.14. The number of required measurements to reach a target reconstruction error is almost constant versus changing N_c. Given that M does not change with N_c, we can see from Equation (4.16) that P_{total} is a decreasing function of N_c. This is also shown in Figure 4.15 and 4.16 for the BS being

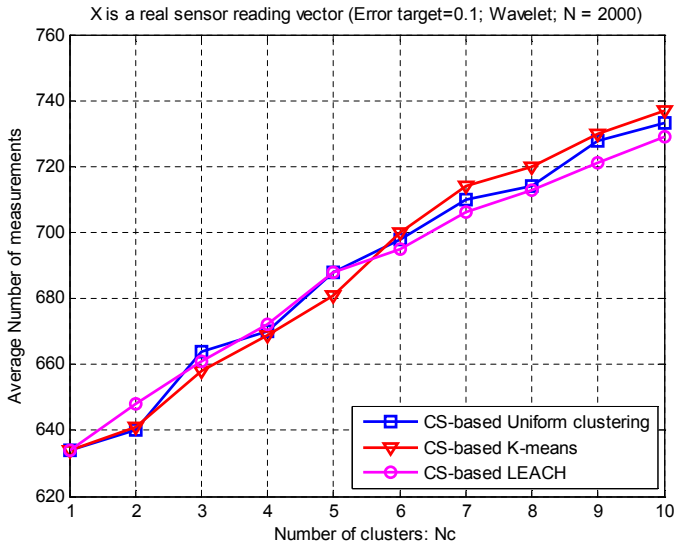


Figure 4.9: Number of measurements required when Wavelet is considered as the sparsifying basis.

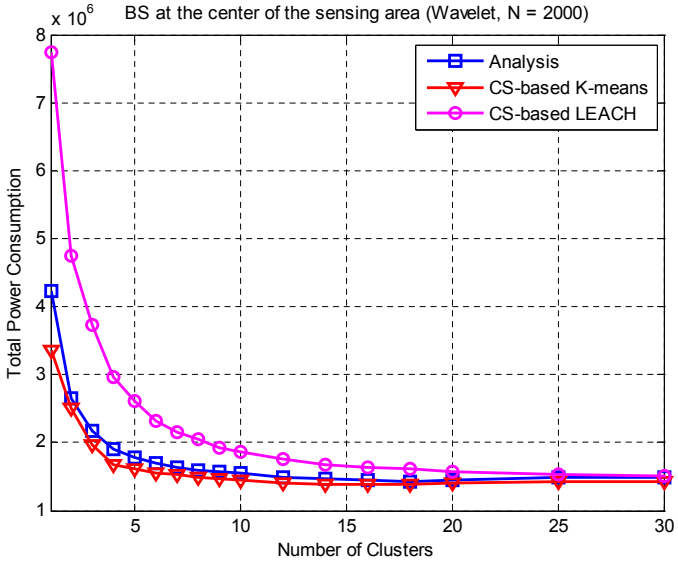


Figure 4.10: Total power consumption when the BS is at the center of the sensing area. Here, $N_c^* = 18$.

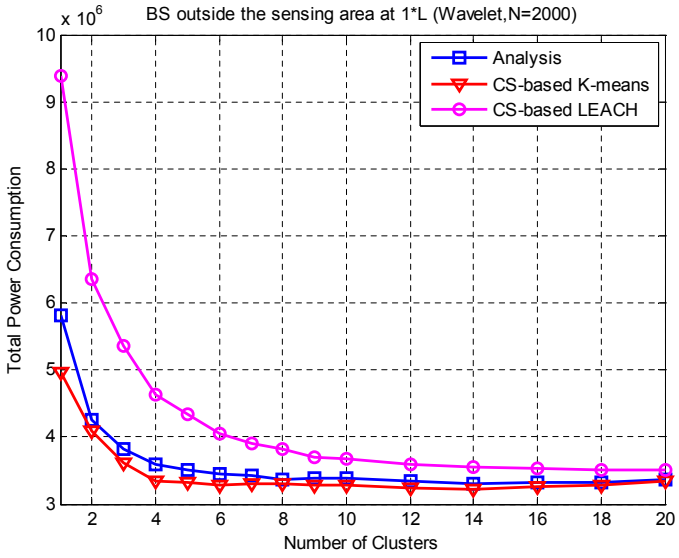


Figure 4.11: Total power consumption when $L_i = L$. Here, $N_c^* = 12$.

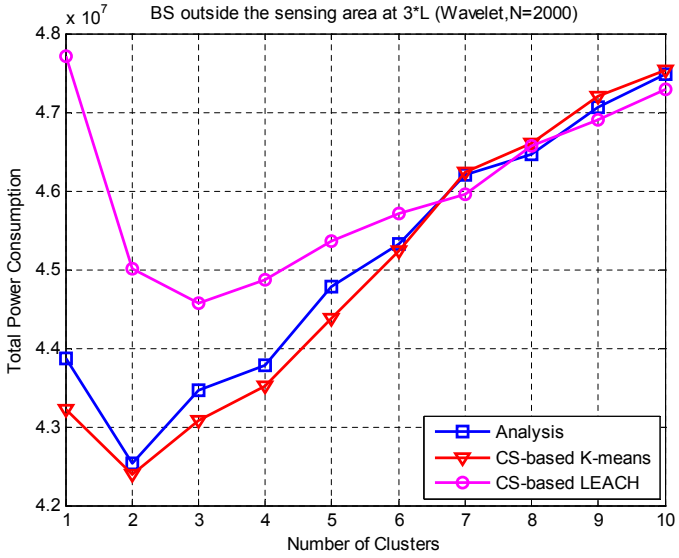


Figure 4.12: Total power consumption when $L_i = 3L$. Here, $N_c^* = 2$ *or* 3 (depending on the clustering scheme).

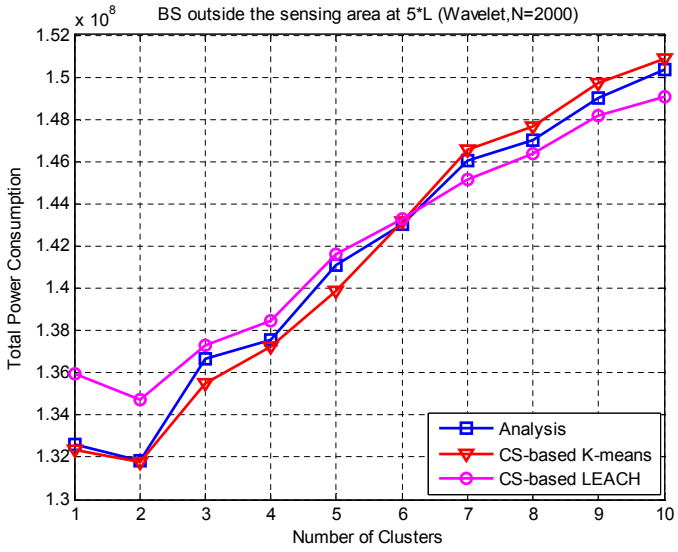


Figure 4.13: Total power consumption when $L_i = 5 \times L$. Here, $N_c^* = 2$.

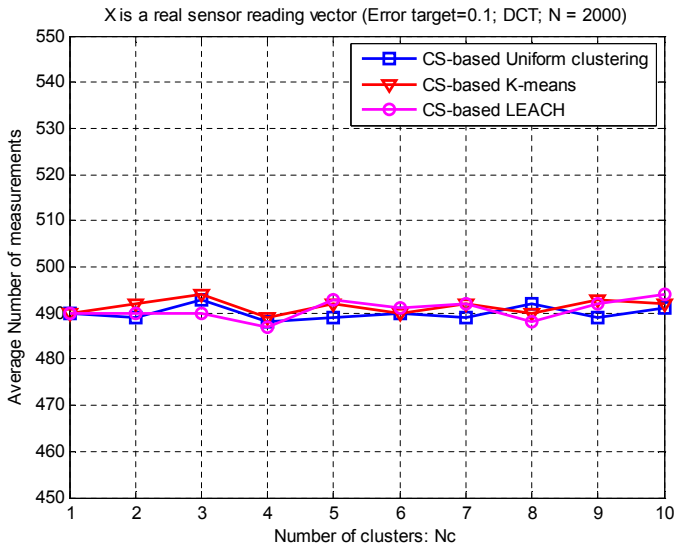


Figure 4.14: Number of measurements required when DCT is considered as the sparsifying basis.

at the center and $L_i = 3L$, respectively. On the other hand, since we are collecting M measurements from the networks, we have $N_c \leq M$. Therefore, $N_c^* = M$ and the smallest size of each cluster on average is N/M sensors.

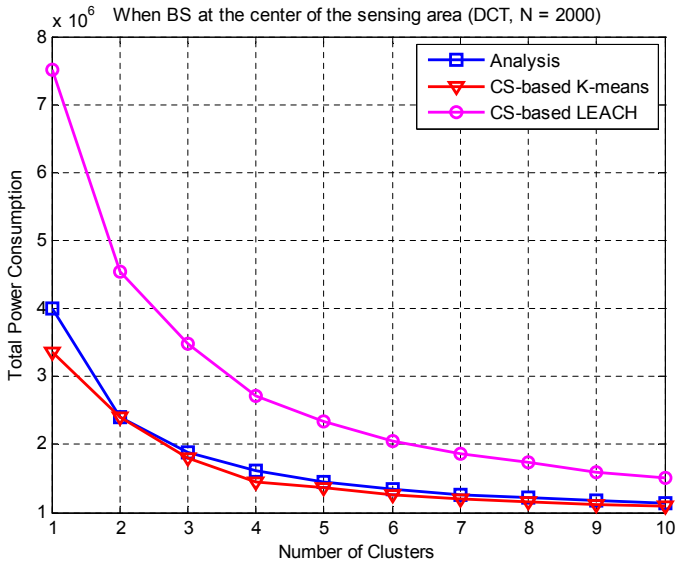


Figure 4.15: Total power consumption when the BS at the center of the sensing area.

**** Remarks on the effect of the sparsifying basis on performance*** : Based on our discussion, we can conclude that under the given cluster scenarios and assuming that the signal of interest is sparse in both wavelet and DCT bases, employing the DCT will be more energy efficient. This is because when ψ is a DCT matrix, ϕ can become very sparse (by increasing N_c) without a considerable loss in CS performance. Our analytical and simulation results showed that in this case the consumed power is a decreasing function of N_c and more clusters results in more power savings and $N_c^* = M$.

4.5 Inter-cluster Multi-hop Routing in CCS (ICCS)

In this section, we propose a method for further energy saving during data collection where by CHs transmit the CS measurements through intermediate CHs to the BS. We refer to this method as inter-cluster multi-hop routing in CCS (ICCS). For

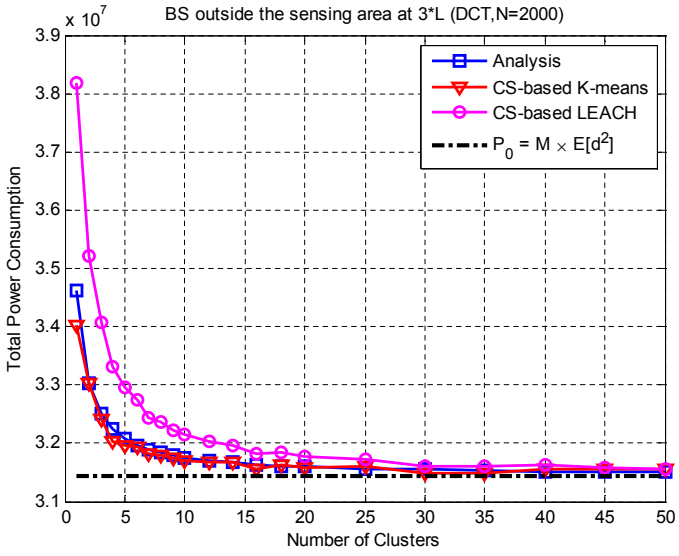


Figure 4.16: Total power consumption when the BS outside the sensing area at $L_i = 3L$.

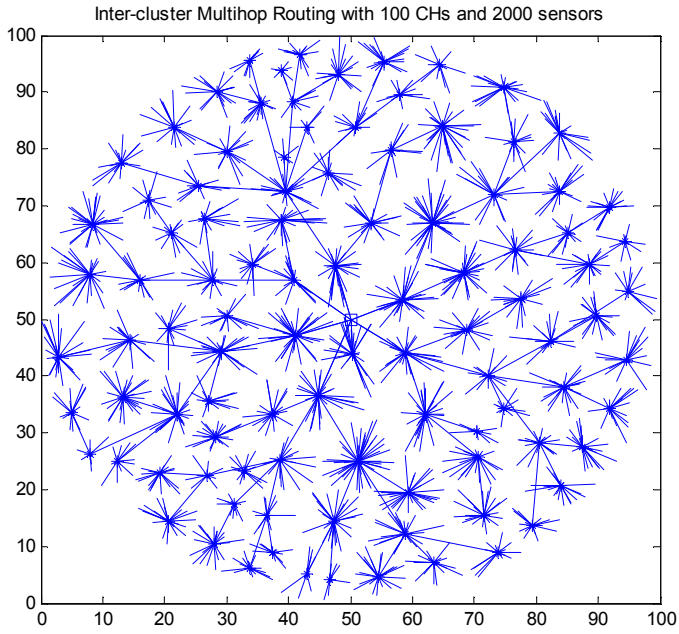


Figure 4.17: All transmissions in the clustered network with inter-cluster multi-hop routing when the BS at the center.

networks with a small number of clusters, ICCS may not help because the multi-hop routing paths might require more power than transmitting directly. But with a large number of CHs, ICCS can significantly reduce the power needed to transmit the CS measurements.

Since we already have clusters formed by K-means or LEACH, we develop an iterative greedy distributed algorithm to form a tree that connects all CHs with the root at the BS. We assume all the CHs have the same transmission range (R) and that CHs within that range can communicate with one another. An appropriate R should be chosen based on the number of CHs formed so that all CHs can be connected as an undirected geometry graph G(V,E), where V is the set of vertices referred to the number of CHs, and E is the set of edges referred to the number of communications links between CHs. Based on the graph, we can deploy the GDA to form the routing paths for the CHs: All CHs broadcast their information about the number of hops away from the BS to their neighbors. At the first iteration, only the CHs which are close to the BS (their transmission ranges cover the BS) have the number of hops (NoH). They name their NoH as "1" and broadcast their own updated information to their neighbors in the next iterations. The algorithm iterates running until there is no change in the communication links between all CHs. This algorithm is shown below as Algorithm 3.

4.5.1 Network Model

To simplify the problem, in this model, we assume that the sensing area has a circular shape with radius R_0 in which the BS is at the center. With intra-cluster transmission, sensors still can adjust their power level to transmit data to the CHs, while all CHs only use one transmission range, denoted as R, to connect to other CHs and the BS. We still have N sensors uniformly distributed in the area, and the path-loss exponent is assumed to be equal to 2 ($\alpha = 2$).

Algorithm 3: Distributed Tree-based Routing Algorithm

Initialization phase:

- N_c clusters marked as CH_i, $i = 1, ...N_c$; CH's transmission range is R.

- NoH(BS) = 0; Nei = set of CH_i's neighbors;

Tree forming phase

while *the routing paths are changing* **do**

  for i = *1 to* N_c **do**

    for j = *1 to* N_c **do**

      if $d(i, j) < R$ **then**

        CH(i) chooses CH(j) if NoH(j) = min{NoH(Nei)};

        NoH(i) = NoH(j) + 1;

      end

    end

  end

end

As shown in Figure 4.17, all the transmissions show CHs receive readings from their cluster members as non-CH sensors and they transmit CS measurements through other CHs or directly to the BS at the center depending on their positions and R.

4.5.2 ICCS Power Consumption Analysis

As before, we refer to the communication cost associated with the communication between the non-CH nodes to CHs as the *intra-cluster* power consumption and denote it as $P_{intra-cluster}$. The CHs create the CS measurements as the combinations of all sampled data within each cluster ($\underline{y}_i = \phi_i \underline{x}_i$) and send the measurements to the BS in a multi-hop fashion. The corresponding power consumption is referred to as $P_{to\,BS}$. The total power consumption is formed as

$$P_{total} = (P_{intra-cluster} + P_{to\,BS}). \tag{4.24}$$

Analysis of $P_{intra-cluster}$

Similar to the assumption for Equation (4.8), we have

$$P_{intra-cluster} = N_C(\frac{N}{N_c} - 1)\, E[r^2], \tag{4.25}$$

and

$$E[r^2] = \int\int r'^2 \rho(r', \theta)\, r'\, dr'\, d\theta. \tag{4.26}$$

We assume each cluster area is a circle with radius $R = R_0/\sqrt{N_c}$ and the density of the nodes $\rho(r', \theta) = 1/(\pi R_0^2/N_c)$. Hence,

$$E[r^2] = \frac{1}{(\pi R_0^2/N_c)} \int_{\theta=0}^{2\pi} \int_{r'=0}^{R} r'^3 dr'\, d\theta = \frac{R_0^2}{2N_c}, \tag{4.27}$$

and accordingly

$$P_{intra-cluster} = (\frac{N}{N_c} - 1)\frac{R_0^2}{2}. \tag{4.28}$$

Equation (4.28) shows that $P_{intra-cluster}$ is a decreasing function of N_c.

Analysis of $P_{to\,BS}$

$P_{to\,BS}$ is calculated based on the inter-cluster multi-hop routing as follows

$$P_{toBS} = \sum_{i=1}^{N_c} NoH_i \times R^2 \times M_i, \tag{4.29}$$

where M_i is the number of measurements required taken from the i^{th} cluster, and R^2 is the power consumption based on the CH's radius spent on each hop with path-loss exponent $\alpha = 2$. For analysis we assume equal size clusters (equal number of sensor nodes). According to Lemma 1, the number of measurements required taken from each cluster should be linearly proportional to the number of sensors in each cluster or $M_i = \frac{M}{N_c}$. So, (4.29) can be written as

$$P_{toBS} = R^2 \times \frac{M}{N_c} \sum_{i=1}^{N_c} NoH_i, \tag{4.30}$$

where, M is the total number of measurement required from the network to satisfy an error-target. In [139], Chandler calculated the *average* number of relay hops in randomly located radio networks. Based on this, (4.30) is given by

$$P_{toBS} = NoH_{avg} \times R^2 \times M, \tag{4.31}$$

where NoH_{avg} is the average number of hops equal to $E[NoH]$ as mentioned in [139]. This expectation of the number of hops is calculated as

$$E[NoH] = max(NoH) - \sum_{NoH=1}^{max(NoH)-1} \frac{P_{NoH}(x)}{P_{max(NoH)}(x)}, \tag{4.32}$$

where *max(NoH)* is the maximum number of hops allowed and $P_{NoH}(x)$ is the probability of being able to send data from a CH to BS at distance x using NoH or less hops. Interested readers are referred to [139] for the details on the calculation of $P_{NoH}(x)$.

Analysis of CH's transmission range R

In each routing path, the number of hops is directly related to the broadcast radius R. If we increase R, a CH could reach more distant CHs and choose one of them to

forward measurements. This means that the total number of hops can be reduced or increased with variable values of R which may affect power consumption. For example, if we increase R the number of hops in each routing path might decrease. But we have to deal with longer hop distance that consumes more power. In Figures 4.18 and 4.19, we have a 2000-node sensor network with 500 clusters formed by K-means and LEACH. We chose $R = \{10, 12, 14, 16, 18, 20\}$. Figure 4.18 shows the total number of hops reduced corresponding to the increase in the radius. Figure 4.19 shows

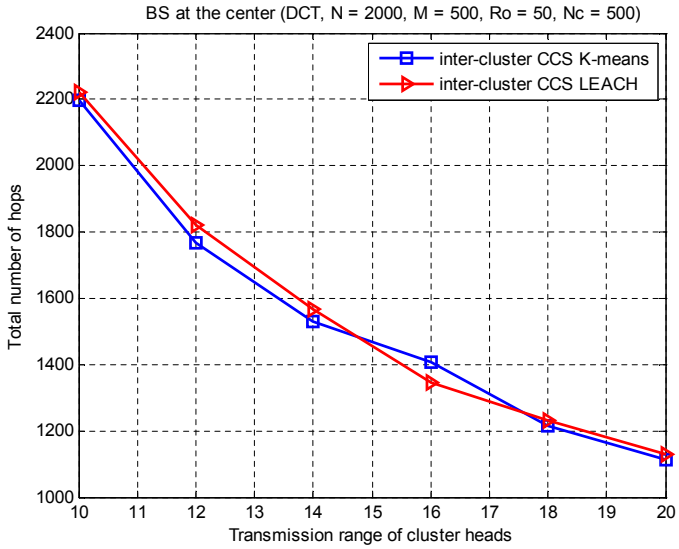


Figure 4.18: Total number of hops routing when changing the broadcasting radius R

that the total consumed power increases as we increase R. Based on Figures 4.18 and 4.19 we should choose the smallest R that results in the least consumed power for the network.

4.5.3 ICCS Simulation Results

In this simulation we form a network consisting of 2000 sensors randomly distributed in a circular area with radius $R_0 = 50$. The BS is set at the center of the sensing area. We use real sensory data collected from [136] and the sparsifying matrix ψ as the DCT. In this case, as discussed in the previous sections, the total number of

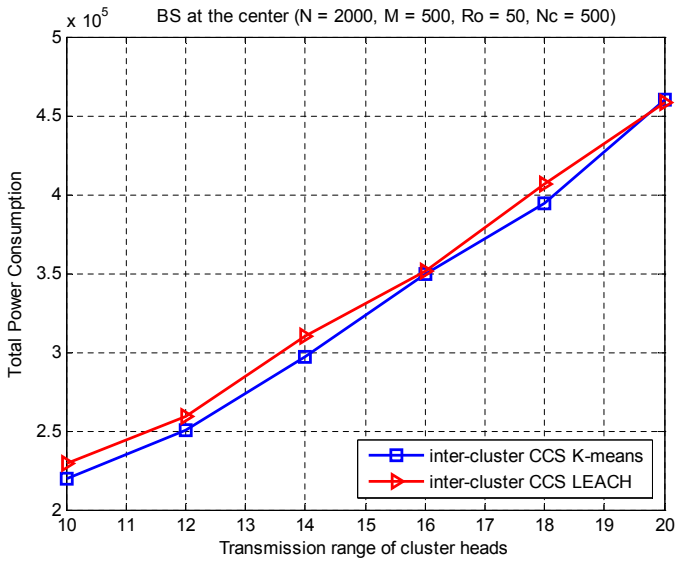


Figure 4.19: The total power consumption when change the broadcast radius R

measurements required does not change as we increase the number of clusters. Hence, for any number of clusters, we chose $M = 500$ to satisfy the error-target of 0.1. We only consider the maximum number of clusters up to $N_c = 500$ since $M = 500$. This means that each cluster should send at least one measurement to the BS for the data recovery process.

We apply K-means and LEACH clustering algorithms to form two different clustered networks. Figure 4.20 shows the total intra-cluster power consumption due to the total consumed power required to transmit data from all non-CH sensors to their CHs within all clusters. As shown in Figure 4.20, the intra-cluster consumed power becomes very small if the network is divided into many clusters. In this case, the total power consumption will be dominated by the power corresponding to the inter-cluster routing paths.

In Figure 4.21, the total inter-cluster power consumption is reduced as the network is divided into a larger number of clusters. For a large value of N_c, the density of CHs in the sensing area is large. Therefore, the CH's transmission range needed to maintain inter-cluster connection becomes smaller. These numbers of clusters of $N_c =$

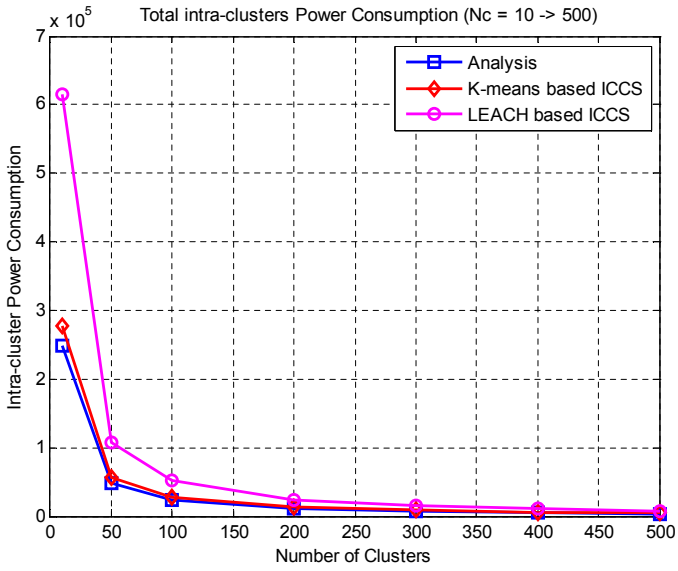


Figure 4.20: Intra-cluster power consumption when BS at the center in a circle sensing area

10, 100, 200, 300, 400, 500 correspond to values of $R = 50,\ 30,\ 25,\ 22,\ 18,\ 14,\ 11,$ respectively, which explains the reduced inter-cluster consumed power.

As shown in Figures 4.20 and 4.21, the total power consumption is reduced by both intra-cluster and inter-cluster transmissions as we increase N_c. As compared with DCCS in Figure 4.22, ICCS significantly reduces the power consumption when the network is arranged into a large number of clusters ($N_c \geq 100$). Note that our calculation for power consumption for DCCS was originally based on the assumption of a square sensing area. These results are extended to the case of a circular sensing area with BS at the center in our previous work [17].

4.6 DCT Compression Transmitting only k Large Coefficients

In this section, we consider transmitting only k large DCT coefficients as proposed in RIDA [148]. All raw readings from non-CH sensors are sent to their respective CH and sorted at each CH in descending or ascending order. Either DCT or wavelet transform is used as the sparsifying matrix to achieve a k-sparse data vector. The mapping

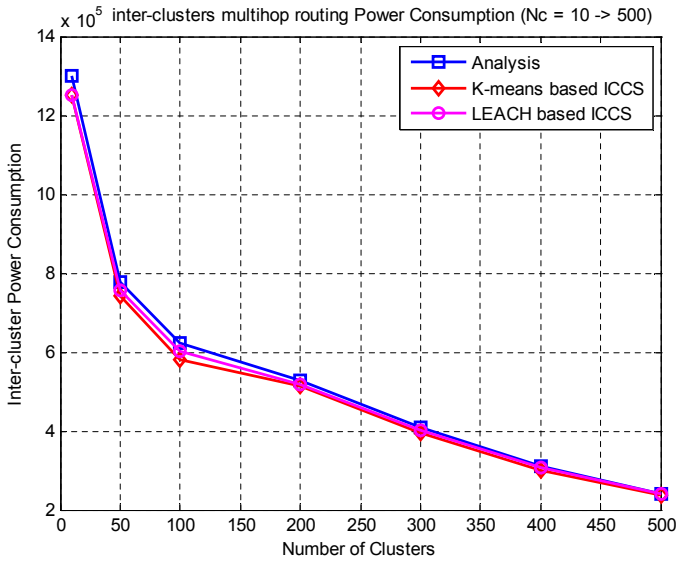


Figure 4.21: Inter-cluster power consumption when BS at the center in a circular sensing area

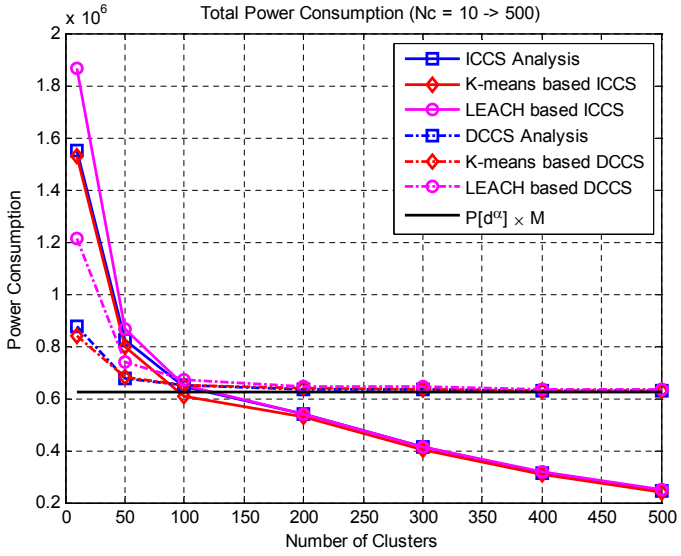


Figure 4.22: Total power consumption for ICCS and DCCS in a circular area network with $R_0 = 50$

process given in [148] is used to match sensors to virtual indices. These sensors will multiply their readings with DCT coefficients and then only send k significant large coefficients to the BS. The rest of the coefficients are considered as zeros and not sent to the BS. At the BS, all k large coefficients are mapped to zero-coefficients and recovered to return all the raw data.

In order to reduce transmission cost in RIDA, the idea of transmitting measurements in CCS [15, 17] is applied. First, all sensor readings from non-CH sensors are sent to their own CH. The data is sorted at the CHs. After being multiplied with a sparsifying DCT matrix, a large proportion of the signal energy is focused on the very k first large coefficients. Instead of sending M CS measurements from the CHs as with in CCS, only these k large coefficients are sent directly to the BS for the recovery process as mentioned in RIDA. All calculations in this section are based on the network model in the DCCS section, and k is much less than M.

4.6.1 Network Model

We assume the WSN is deployed in a square sensing area sized $L \times L$. This model is similar to the model defined for the analysis of the DCCS algorithm. The power consumption for each transmission is calculated as it was with DCCS, except for the number of coefficients k transmitted from CHs to the BS. We will not compare the total power consumption between DCCS and DCT compression because of $k \ll M$. We formulate and also simulate the problem to show how this compression method works with clusters, noiseless or noisy signals.

4.6.2 Communication Power Consumption

We assume that all clusters have the same number of sensors. Hence, the number of large coefficients collected from all clusters should be equal. Hence, the total number of large coefficients is calculated as $k = \sum_{i=1}^{N_c} k_i$, where k_i is number of coefficients

collected from the i^{th} cluster. Similar to Equation (4.16) in Section 4.4.2, the total power consumption for this method can be formulated in general as

$$P_{total} = (\frac{N}{N_c} - 1)\frac{L^2}{2\pi} + \frac{k}{L}[\frac{(L - L_i)^3 + L_i^3}{3}] + \frac{kL^2}{12}. \tag{4.33}$$

When the BS at the center of the sensing area ($L_i = L/2$), (4.33) is simplified as

$$P_{total} = (\frac{N}{N_c} - 1)\frac{L^2}{2\pi} + \frac{kL^2}{6}. \tag{4.34}$$

4.6.3 Simulation Results

In this section, we consider both sorted and unsorted signals generated from 2000 sensors uniformly distributed in a square sensing area. These types of data provide different values of k that affects either the transmitting cost from the CHs to the BS or the reconstruction error at the BS.

Figure 4.23 shows unsorted sensor readings collected from a WSN [136] and their transformations in the DCT domain. All signal energy is preserved in the transformed vector but is now focused in a relatively small number of large coefficients. If we transmit only these k large valued coefficients to the BS, this results in much less consumed power than transmitting all the values as was done earlier with CCS.

Figure 4.24 shows sorted signals in decreasing order and the DCT coefficients. The large coefficients are concentrated in the lower numbered coefficients. The transmission cost can be reduced based on the smaller values of k compared to that in unsorted signals.

Both Figures 4.25 and 4.26 show that increasing the number of clusters or reducing the total number of coefficients k transmitted to the BS will increase the reconstruction error. Transmitting more of the larger DCT coefficients to the BS can compensate for the error as we increase the number of clusters.

In a noiseless environment, using DCT compression consumes less energy than CCS since CCS transmits M measurements to the BS while DCT compression only

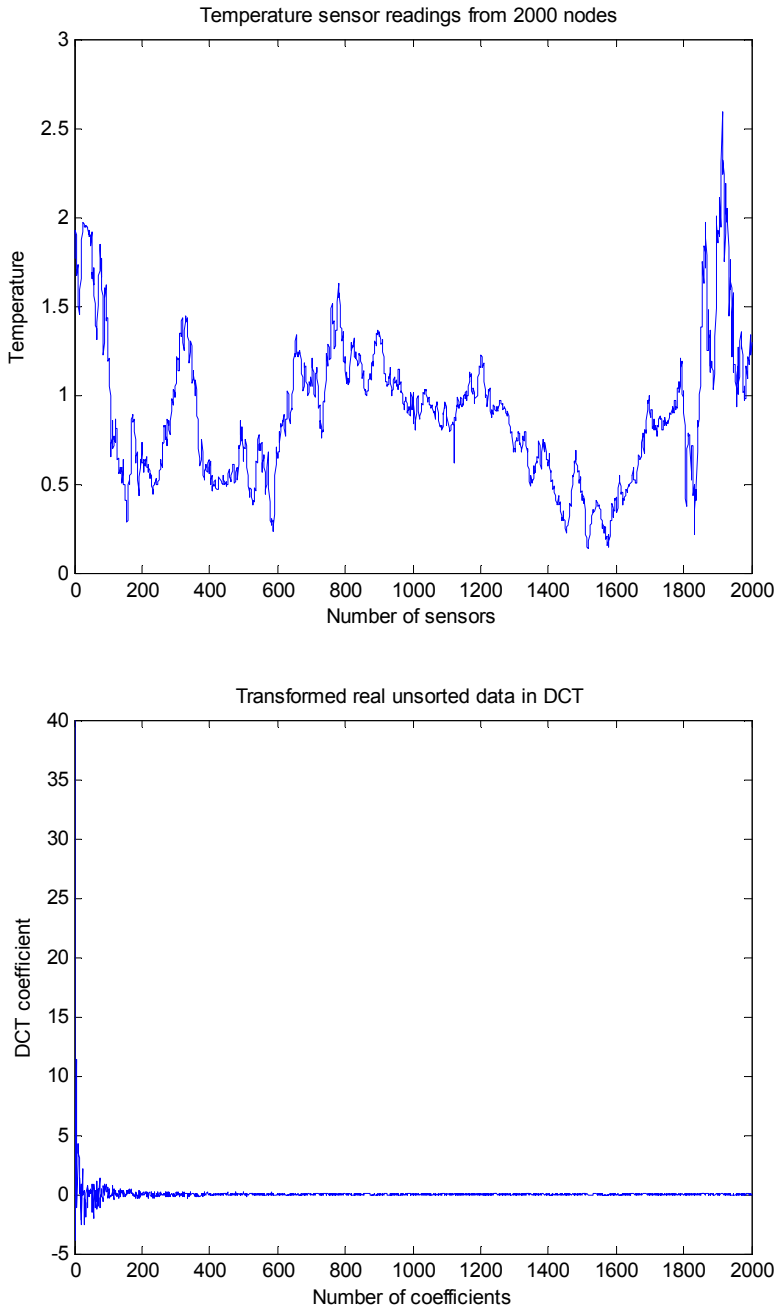


Figure 4.23: Unsorted sensory readings from 2000 sensors and the DCT transformed coefficients

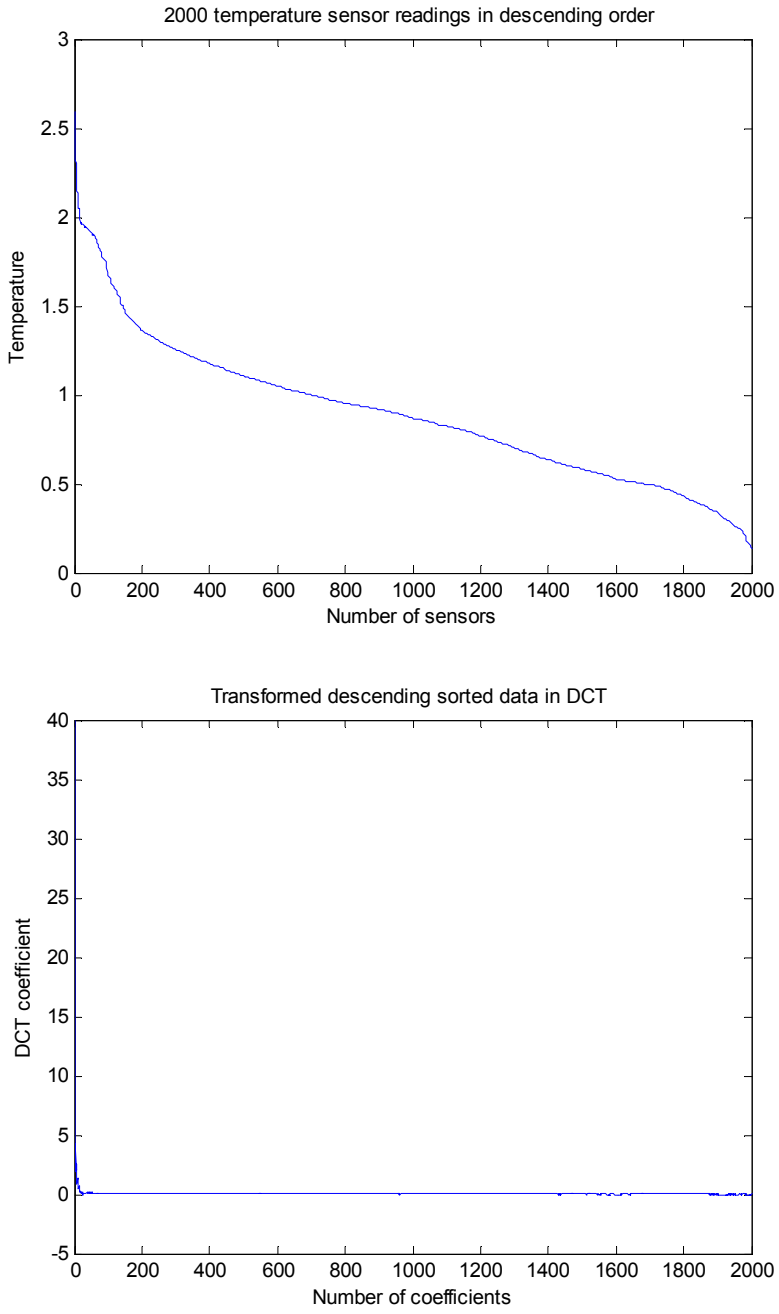


Figure 4.24: Descending sorted readings from 2000 sensors and the DCT transformed coefficients

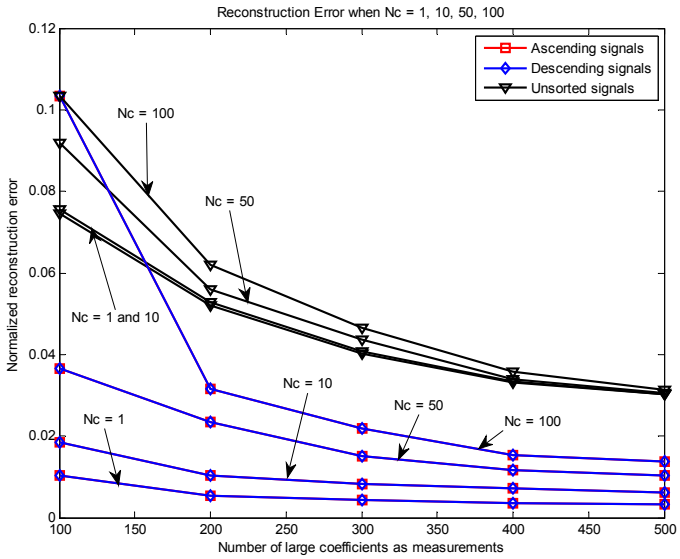


Figure 4.25: Reconstruction error versus number of measurements with different of clusters

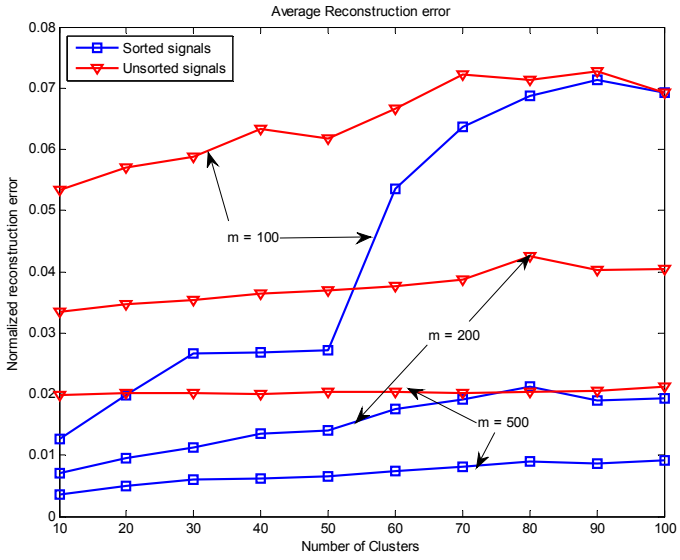


Figure 4.26: Reconstruction error versus number of clusters with different of measurements

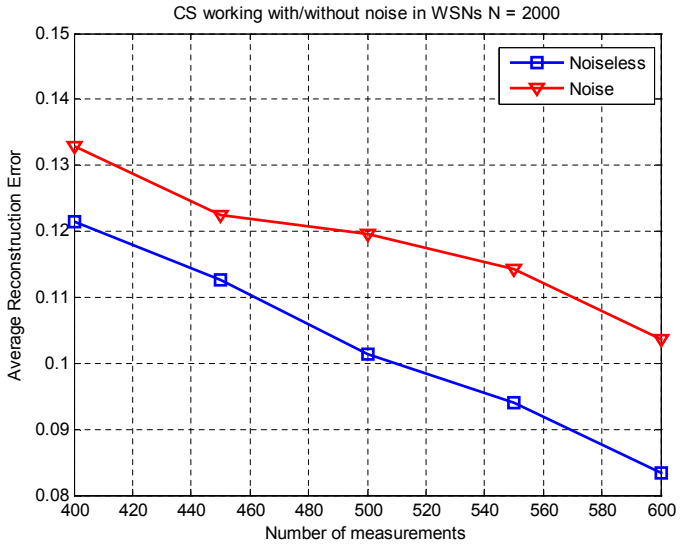


Figure 4.27: CS reconstruction error versus measurements with noise

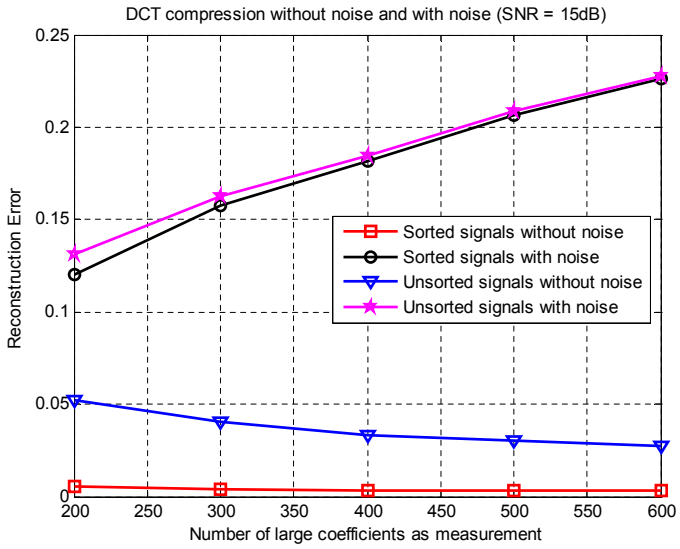


Figure 4.28: DCT compression reconstruction error versus measurements with noise and noiseless

sends k large transformed coefficients ($k \ll M$). As shown in our simulation results, k is generally only about 20% as large as M to satisfy the error-target in signal recovery processes. In practical networks noise is problematic. CCS can work with noise contaminated measurements while DCT compression is quickly degraded. As shown in Figure 4.27, sensor readings can be recovered at the BS based on different numbers of noisy measurements. Increasing the CS measurements can recover the original signals with less error.

Figure 4.28 shows that with DCT compression in the presence of noise the reconstruction error increases as the total number of measurements is increased. So, in practical applications of WSNs DCCS and ICCS should be considered.

4.7 Conclusion

In this chapter we presented an energy-efficient data collection method applied in WSNs that is based on an integration of clustering and block-wise CS, called CCS. It is well known that natural signals have spatial correlation and therefore the sensor readings in a WSN are sparse in a proper basis such as DCT or wavelet. This sparsity facilitates the utilization of CS for energy-efficient data collection in such networks. In contrast to previous work in this area, we introduced CCS in which all non-CH sensors send their readings only once to the CH they belong to. The CS measurements required are generated at the CH before being sent to the BS for the CS recovery processes in two possible ways: directly (one hop) or multi-hop, called DCCS and ICCS, respectively.

We formulated the total power consumption and discussed the effect of different sparsifying bases on CS performance as well as the optimal number of clusters for reaching the minimum power consumption. We employed K-means, LEACH, and uniform clustering techniques in our simulations and found the optimal cluster size when the signal of interest is sparse in canonical, wavelet, and DCT bases. After

choosing DCT as the best sparsifying basis for CCS, we showed choosing a larger number of clusters can achieve less power consumption utilizing DCT with real sensor readings as the intra-cluster power consumption is reduced. The optimum number of clusters was determined to be $N_c^* = M$. Furthermore, as we employ many clusters, ICCS outperforms DCCS based on multi-hop routing.

As a final case to compare with DCCS and ICCS, we considered transmitting only k large coefficients in DCT transformed signals. This method cannot work in noisy environments as mentioned in simulation Section 4.6.

CHAPTER 5

TREE-BASED DATA GATHERING IN WIRELESS SENSOR NETWORKS

5.1 Introduction

5.1.1 Motivation

As noted before data gathering in wireless sensor networks (WSNs) relies on small and inexpensive devices with severe energy constraints. Our goal is to prolong the network lifetime based on saving energy for the sensors while collecting the required CS measurements. In this chapter, tree-based data gathering for WSNs is considered.

In order to apply compressive sensed data gathering in WSNs, only M CS measurements need to be collected from the network and sent to the base-station (BS) to reconstruct all N raw sensor readings ($M \ll N$). Each measurement is a linear combination that might be collected from all sensors as mentioned in [108, 109] or from only a certain small number of random active sensors in the network [111]. The network topology can be formed as a spanning tree with parent and child nodes counted from the sink, also called BS. Once the parent node receives data from all its descendant nodes, it adds its own data sample and all received data together and then sends the combined data to its upper parent node or the sink node. Radu [125] has shown that the sparse binary measurement matrix can work as well as the dense Gaussian one in CS recovery processes, so not all sensors must participate in adding their readings together to make each CS measurement. We are motivated to work on the sparse compressed networked data for more power savings. We need to find a rout-

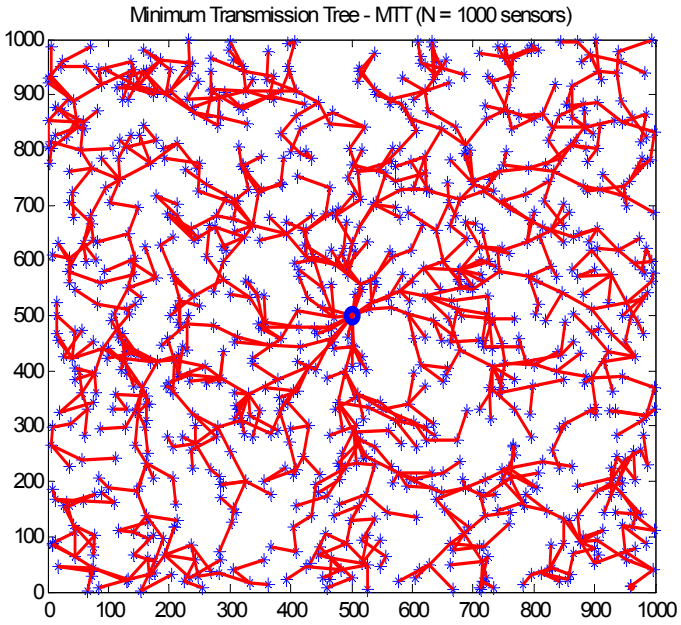


Figure 5.1: An example of a tree formed by MTT algorithm with 1000 nodes deployed in a arbitrary network when BS at the center

ing method that consumes energy the least. In this chapter, we consider a method called the tree-based energy-efficient method (TCS) that reduces significantly the consumed power for data transmission between sensors. A sensor (instead of sending many times its own data to a parent corresponding to the number of times to collect M measurements) only sends its data once. The raw data is processed by the parent node: multiplying the coefficients based on a measurement matrix and adding all received data together and eventually forwarding one combined data sample to an upper parent or the sink.

The minimum transmission data gathering tree method, called MTT [111], that has been known as an energy-efficient data collection method. It forms the data gathering tree based on not only the shortest path, but also the number of times that nodes transmit their own data. As shown in [111], MTT consumes power significantly less than both methods: the minimum spanning tree (MST) [137] and the shortest path tree (SPT) [138]. So, in our experiments with both lattice and arbitrary net-

works, we consider MTT as a baseline and apply our algorithm to compare results related to consumed power.

In addition, we analyze the networks' characteristics in terms of saving energy, then choose an appropriate transmission radius for sensors. And, we exploit the sparsity of the measurement matrix for the networks to consume less power, which does not affect the reconstruction error at the BS.

5.1.2 Related Work

An important issue that needs to be considered for applying CS in the data collection problem is the underlying routing mechanism. Wang [110] mentioned the distributed algorithm with sparse random projections that provides the idea for how to collect measurements from sensors in a WSN. C. Luo [108] and J. Luo [109] contribute routing methods to collect data but all sensors have to send either their own readings or combined messages to their parent node or the sink M times corresponding to the number of measurements required. Some other work focused on random walk routing [12, 13] or cluster-based data collection [15, 16, 17] as discussed previously. We propose applying CS using a tree-based algorithm to reduce the power consumption based on Radu's [125] results with sparse binary matrices. This means that each measurement could be collected from some random sensed data, not from all sensors. This motivates us to focus on the routing problem to save consumed power for WSNs to prolong the network lifetime. A novel idea from Xie [111] significantly reduces transportation cost. By using the sparse random matrix, only some random nodes send their own data to build one measurement. The other nodes relay data if they are along the routing path or they may not participate in some rounds of transmission. The routing tree is designed to eliminate some residual transmissions since nodes that have more rounds to transmit their own data are a priority to choose while building the tree.

**** Minimum Transmission data gathering Tree (MTT) [111]*** This algorithm provides a novel idea to form a spanning tree as an energy efficient tree. The tree is built from the sink node as shown in Figure 5.1 based on a random connected network and a projection matrix Φ. The network can be considered as a undirected graph G(V,E), where V is the vertex set and E is the edge set. The graph is created by the broadcasting radius between nodes, called R. A subtree at first has only the sink node. Each iteration considers the lowest cost edge within its transmitting range to extend the tree one more node. In addition, based on the projection matrix that shows the number of times each node sends it own data, the algorithm calculates a cost called the incremental cost for every edge connected to the subtree, and chooses the lowest cost edge and adds the node in the tree. These steps help to reduce redundant transportation energy from relaying data between nodes.

The heuristic MTT algorithm[111]:

Input: $G =< V, E >$ and Φ;

1. Initialization: $T' = \{v_0\}, V' = \{v_0\}, U' = V \setminus \{v_0\}$
2. Get $R(u)$ from Φ, for $u \in U'$;
3. **while** $U' \neq \emptyset$ **do**
4. **for** $e = (u, v) \in E$, where $u \in U'$, $v \in V'$
5. Calculate incremental cost $C^+(T', e)$;
6. Calculate average incremental cost $\bar{C}^+(T', e)$;
7. **end for**
8. Choose the edge e^*, whose average incremental cost

$\bar{C}^+(T', e)$ is minimum, and add the edge e^* into T';

9. Update the related variable U', V';
10. **end while**

Output: Data gathering tree $T = T'$ with the root of v_0 and connecting the nodes in V.

Given a graph $G =< V, E >$ and the measurement matrix Φ, the iterative MTT algorithm adds one node into the subtree T' constructed by each iteration. V' stands for the set of nodes in the subtree T', and U' denote the set of nodes unconnected to T'. At each one out of N iterations, the algorithm chooses one edge which connects a node in U' to T' based on the average incremental cost of each edge calculated as

$$\bar{C}^{+}(T', e) = \sum_{i \in R(u)} C_i^{+}(T', e) / \mid R(u) \mid, \tag{5.1}$$

where $\mid R(u) \mid$ presents the number of rounds in which node u has its own data to transmit. This corresponds to the number of non-zero elements in the column u in the projection matrix ϕ.

After N iterations, $U' = \emptyset$, then the tree is completely built. For the sake of simplicity, MTT assumes the cost of each edge for one transmission is 1 and applies this for building the trees.

The MTT algorithm initiates collecting data based on ϕ. The non-zero elements in each row ϕ_{ij} present sensor nodes that send their own data to upper parents. The rest of the sensors may not participate in a routing path or just relay data if they are located between the routing paths. The participation costs energy for transmitting. All the combined data is gathered at the BS to build one measurement that corresponds to one row in ϕ.

In this chapter, we offer a new algorithm to improve the generation of CS measurements for more energy saving, called TCS. The algorithm can apply to any tree that is formed by MTT, SPT or MST. It significantly diminishes the total number of transmissions in tree-based routing in WSNs. TCS will be addressed and analyzed in the next section.

5.2 Problem Formulation

5.2.1 Network Model

We model a WSN with N sensors which are uniformly randomly distributed to a square area $L \times L$. The topology of the connected network is a random geometric graph $G(V, E)$. Any algorithm creates a tree based on the graph. We assume that each sensor has the same maximum broadcasting radius called R calculated as a Euclidean distance. The sensor nodes can adjust their power to transmit to each other based on real distances between them. In reality, this network is not a regular network where each sensor has a different number of neighbors within its broadcasting radius R. If we change the value of R, the connection of the graph will change as follows: the set of vertices is the same based on the total number of sensors, but the edge set changes based on the communication between sensors in each neighborhood. We exploit this in the next section.

5.2.2 Tree-base Energy-Efficient Data Gathering (TCS)

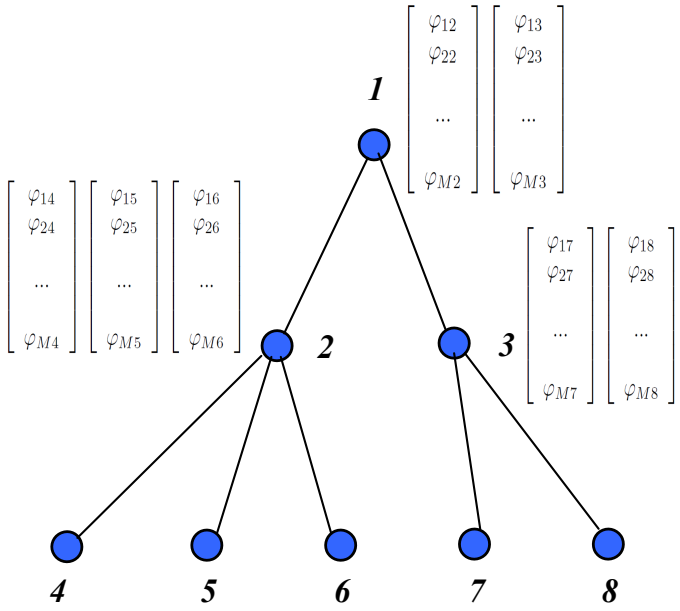


Figure 5.2: A simple example illustrates TCS algorithm with 8 sensors

The algorithm: There are three sensor status conditions in TCS:

Definition 1: S_T: Nodes that have their own data to transmit in a round of transmission are corresponding to non-zero elements in a row of the projection matrix ϕ. In TCS, these nodes send their readings only once at the first time. Then, they might relay data or may not participate in rounds of transmission.

Definition 2: S_R: Nodes that relay data do not have their own data to transmit but still participate transmitting data in each round if they are distributed in a routing path between S_T and the sink. They have three roles: generating data if their children's vectors have at least one non-zero element, adding received data from downstream nodes including the one it might have, and then forwarding the combined data to its parent or the BS.

Definition 3: S_N: Nodes do not participate in a round of collecting data corresponding to zero elements in the vectors stored at sensors. The status of sensors randomly changes between three of them depending on the Φ matrix.

TCS only exploits the measurement collecting part that significantly reduces a certain number of transmissions for WSNs using tree-based routing. Given a spanning tree, sensor nodes store their children's measurement vectors instead of their own vectors as shown in Figure 5.2. Each node only sends its data to the parent node once. A parent node stores a number of column sparse binary vectors corresponding to the number of its descendant sensors. At each round of collecting data, it adds stored readings from nodes that have non-zero coefficients in the same index row of the stored vectors. It also adds the received data from the downstream nodes, then finally sends the combined data to its parent. The message might add more data through the routing path to the BS.

In Figure 5.2, node 2 stores three sensor readings from nodes 4, 5, 6 and their column measurement vectors. Node 1 stores 2's and 3's while node 3 stores 7's and 8's. Since all the coefficients are zero and one, a node only generates a measurement

and then transmits it to the BS if it has at least one non-zero coefficient in the same row of stored vectors corresponding to the round of collecting data.

The total power consumption is calculated based on the total number of transmissions in the network. In TCS, although a S_T node transmits it own data to the upper node once through M rounds, it may work as S_R in some rounds that count for power consumption. So, TCS obviously reduces transmissions with S_T which are not between any S_T and the sink at each round. The reduction rate will be shown in the simulation section.

5.2.3 Power Consumption Analysis

Based on CS theory, a certain number of measurements is required (M) from the network to recover all raw sensor readings at the BS. To build one measurement, some sensors S_T that correspond to the non-zero coefficients in a row of ϕ matrices send their own data to the parent node. In addition, a number of sensors along the routing path work as S_R. If we call the total number of transmissions N_T, the total power consumption for collecting M measurements to the BS can be calculated as

$$P_{total} = N_T \times r^{\alpha}, \tag{5.2}$$

where r presents real transmitting distances between consecutive nodes in the routing tree while we assume sensors can adjust power to transmit data to a destination. α is the path loss exponent that is $\alpha = 2$ and $\alpha = 4$ in free space and multipath fading channels, respectively [135]. For simplicity, we assume $\alpha = 2$ throughout this discussion.

Power consumption formulas that will be applied for a grid network and also an arbitrary network are presented in the simulation section. In both networks, the total number of hops N_T determines the total energy consumption. There are two steps to eliminate numbers of hops and choose an optimal transmission range for sensors.

Reduce the sparsity of measurement matrices

Originally, the MTT tree is formed as a spanning tree with a large number of non-zero entries of the projection matrix as follows

$$\phi_{ij} = \begin{cases} 1 & with\ p = 1/3 \\ 0 & with\ p = 2/3. \end{cases} \tag{5.3}$$

We know that p results from the number of nodes sending their own data to create one measurement corresponding to each row of Φ. It means that at each round of collecting one measurement, only $p \times N = N/3$ sensors sending their data to the BS, on average. The rest ($2N/3$ sensors) could be in standby or sleep to save energy if the sleeping schedule is applicable. Based on Radu's works on sparse measurement matrices [125], it could be possible to reduce the probability of non-zero element p in Φ matrix as much as it does not affect the reconstruction error. Furthermore, since the sensing data is highly correlated, all raw signals are reconstructed accurately at the BS even if a few nodes may not be sampled. Figure 5.8 in the simulation section shows the improvement from the probability reductions.

Choosing an optimum transmission range for sensors

Since all sensors are connected as a spanning tree, evaluating the transmission range is a huge step in building a WSN. It is not only about energy saving but also related to other coefficients of the network. Each sensor consumes more energy when its transmission range is increased. But the number of hops transmitted from its position to the sink is reduced. Based on equation (5.2), the transmission distances between consecutive nodes with exponent α dominates the total consumed power, even with the total number of hops reduced. So, the optimal transmission range should be as small as possible only to keep the network connected. Further, if we consider interference, increasing transmission range can interfere with other nodes within the

range. Figures 5.6 and 5.7 in the simulation section show the total number of hops and consumed power when we change the transmission range.

5.3 Simulation Results

In our simulations, we first use the sparse binary matrices generated by the following equation (5.3). We work on lattice networks and then arbitrary networks. In both types of network, different numbers of sensors are deployed. We consider normalized reconstruction error $\frac{||\underline{x}-\widehat{\underline{x}}||_2}{||\underline{x}||_2}$ in CS signal recovery.

5.3.1 Lattice Network

We create a lattice topology that has the same number of rows and columns. Nodes at the corner have only two neighbors while nodes at the border have three and the others have four neighbors. The transmission range R and the distance r are both equal to 1. The sink node is one of the nodes and it locates at the center point. We choose different numbers of nodes as shown in Figure 5.3 that results in a different size of the lattice. We apply algorithms SPT and MTT to form the data gathering trees at each network size. In MTT, we use $M = N/5$ to generate ϕ before building the MTT tree as mentioned in [111]. After the trees are formed, we calculate the number of transmissions N_T following SPT, MTT and TCS based on MTT tree. Figure 5.3 shows that TCS significantly reduces the total number of transmissions at each lattice size.

Since $R = 1$, $r = 1$ in lattice topologies, we only compare the number of transmissions. We calculate and compare the power consumptions in the arbitrary network.

5.3.2 Arbitrary Network

As briefly mentioned in the network model, we deploy different numbers of sensors $N = 1000, ..., 3000$ which are randomly distributed in a square area 1000×1000 $unit^2$.

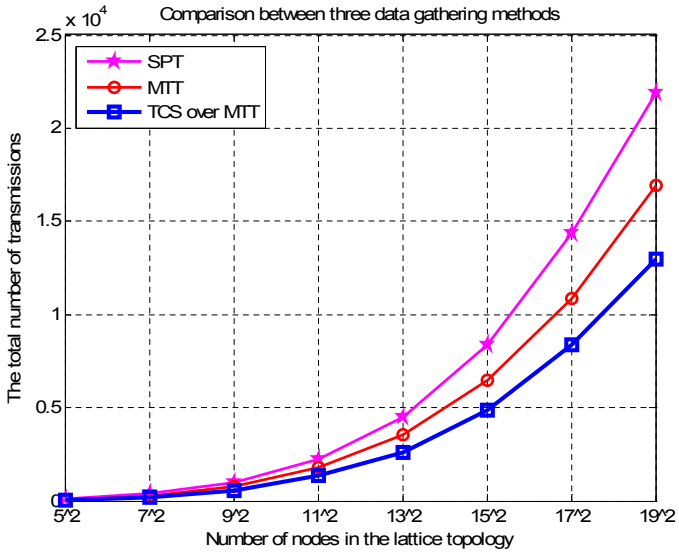


Figure 5.3: Compare total numbers of transmissions between three algorithms in different lattice topology networks

We use real sensor readings from Sensorscope: Sensor Networks for Environmental Monitoring [136]. $\underline{x}$ is dense in the canonical domain. In order to apply CS, as mentioned in the background section, we need a sparsifying basis such as DCT. In each network, we collect the same number of measurements $M = 500$. We choose the same transmission range $R = 60$ *units* for the sensors throughout all networks. Based on Figure 5.3, MTT used fewer number of hops than SPT. So, with random networks, we only work with MTT and apply TCS on the tree to show the results mentioned in the previous section. After using MTT to form the routing tree, we can calculate the total power consumption for both MTT and TCS. As shown in Figure 5.4, TCS consumes less power than MTT. TCS can gain more energy reduction when the networks are more dense. In other words, with the same M and p applied, each parent node has more children nodes when sensor density is increased. TCS benefits at this point since all child nodes send their readings to their parent only once. It also results in Figure 5.5 that the power consumption reduction ratio of TCS over MTT increases when the networks deployed more sensors. It could save from 30%

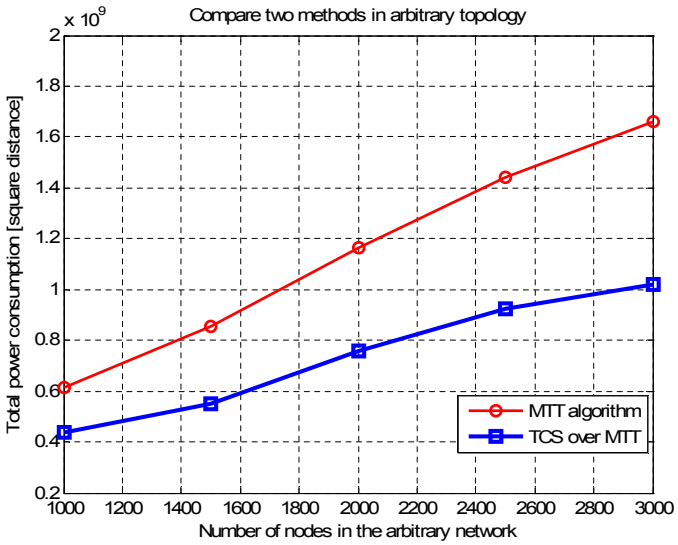


Figure 5.4: Total energy consumption in arbitrary networks with different numbers of nodes with M = 500, p = 1/3

to 37% energy compared to MTT in such networks. In order to save more power in gathering data, we can consider both of these ideas: Find an appropriate transmission range R and increase the sparsity of the measurement matrix Φ. As mentioned in section 5.2.3, we can find an optimal value of R. Figure 5.6 shows the number of hops reduced when we increase R since a node can reach further along the route to transmit data to the BS. In addition to this slow reduction, we have to deal with the energy increase of R^2.

In Figure 5.7, when R is increased, the total power consumption is also increased following our discussion in section 5.2.3. Without considering other characteristics of the network such as capacity and latency, this figure leads to a conclusion that we should choose a smallest R as the optimal R or $R^* = min\{R\}$.

We prefer to suggest the second idea: using small p that can significantly reduce the number of transmissions. We need to ensure in getting p reduced that the projection matrices are getting sparser does not affect CS performance. In other words, if the CS recovery processes do not degrade, we can keep reducing p as small as possible

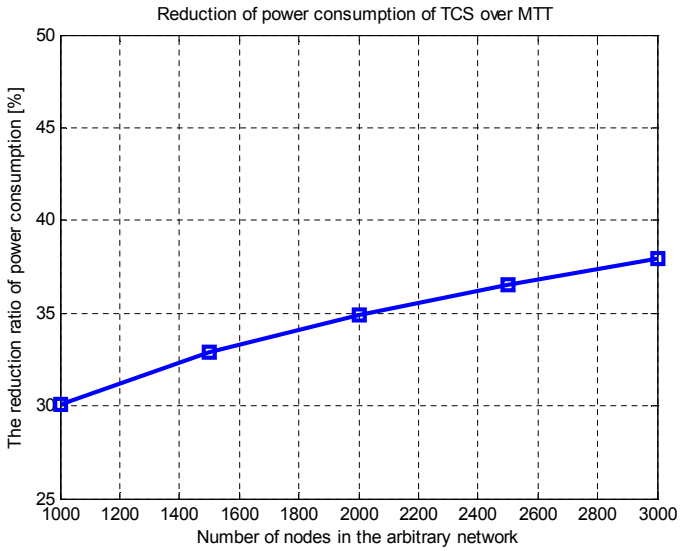


Figure 5.5: The reduction ratio of power consumption of TCS over MTT in arbitrary networks versus the various number of sensors(M = 500, p = 1/3)

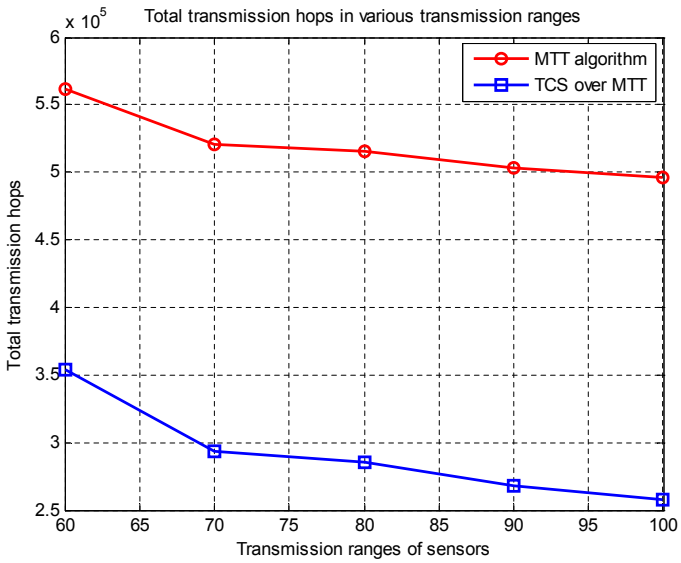


Figure 5.6: Total number of transmission hops in various transmission range (N = 2000; M = 500; p = 1/3)

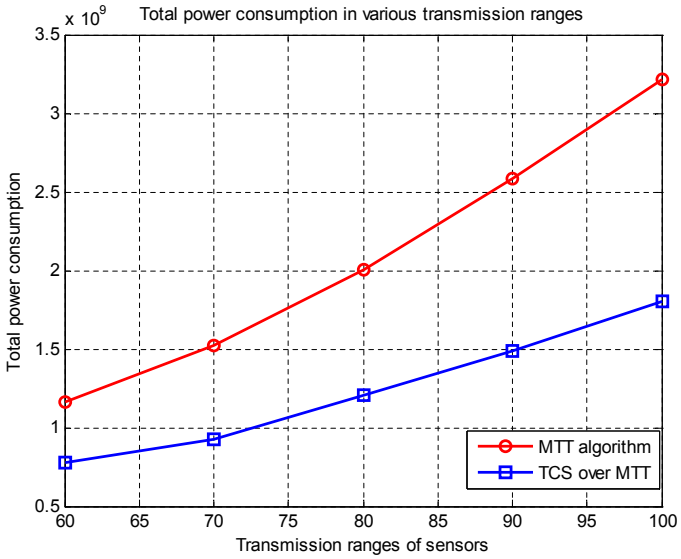


Figure 5.7: Power consumption affected by increasing transmission range (N = 2000; M = 500; p = 1/3)

to save more energy for WSNs.

In Figure 5.8, we chose one specific arbitrary network with 2000 sensors and $R = 60$. Sparse projection matrices with the same dimension (500×2000) but different sparsity are created to form the routing trees. As we calculated, the total consumed power reduces as long as we reduce p. Since p is very small, the number of nodes transmitting their own data at each time of collecting a measurement also becomes to small ($p \times N$, on average). And the gain of TCS over MTT becomes smaller since TCS only reduces the number of times each node transmits its own data. If we keep reducing p across a threshold, we cannot get much more gain but it may effect the CS reconstruction error that will be shown in Figure 5.9. Figure 5.9 shows the probability p we could apply to achieve the smallest consumed power. According to Figure 5.9, we should only reduce p until $p = 1/500$ that does not affect the CS recovery performance. If we choose $p < 1/500$, the reconstruction error increases and the number of measurements required, M, should be increased to compensate for the error target which we want to achieve. It absolutely consumes much more energy. So

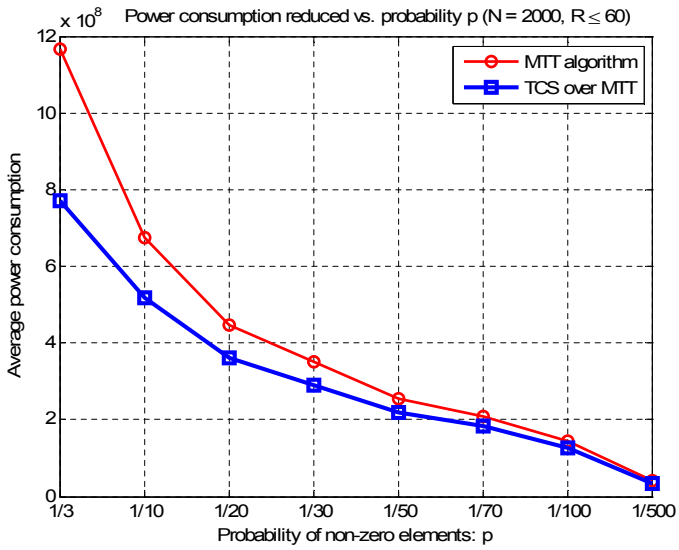


Figure 5.8: Power consumption reduced with sparser projection matrices in both MTT and TCS in arbitrary networks (N = 2000; M = 500)

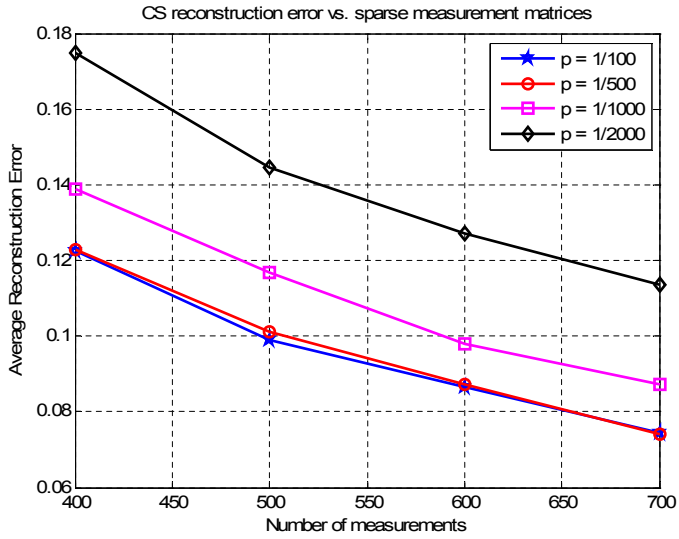


Figure 5.9: CS reconstruction error when reducing the probability of non-zero elements in sparse measurement matrices

$p = 1/500$ could be the optimal sparsity for matrix Φ or $p^* = 1/500$ in this example.

5.4 Conclusions and Future Work

In the chapter, we introduce a tree-based energy-efficient data gathering algorithm, called TCS. We exploit the sensor-storage to store measurement vectors and generate measurements at parent nodes in order to reduce the number of transmissions due to collecting data. All sensors send their readings to their parents only once. Each parent node stores a certain number of measurement vectors corresponding to its descendant nodes. It generates samples for its children, adds more data received from other sensors and eventually forwards the combined measurement to the upper node. All data collected at the sink node for one measurement at each round corresponds to one row of the projection matrix. What we achieve from TCS is that S_T nodes only transmit their data once which significantly reduces the number of transmissions. We provide experiments in both lattice and arbitrary networks to show the advancement of TCS. In addition, we suggest two ideas to save more consumed power by choosing an optimal transmission range R^* and reducing the probability of non-zero elements p for the Φ matrix. Based on the reconstruction error at the BS, we can suggest approximately the optimal sparsity for the projection matrix. In future work, we plan to exploit the boundary of sparsity of the projection matrix in formulas for further consumed power saving in tree-based routing methods in WSNs utilizing CS and optimize the transmission range R based on other network's characteristics.

CHAPTER 6

NEIGHBORHOOD BASED DATA COLLECTION IN WIRELESS SENSOR NETWORKS

6.1 Introduction

In this chapter, we propose a new paradigm of generating measurements combined with two ways of delivering and collecting those CS measurements to the BS. Based on a geographic graph G(V, E) based on an appropriate transmission range R between sensors, the network is supposed to be connected. The set of vertices is always equal to N but the set of edges E increases or decreases depending on R. Each sensor has its own neighborhood which is different to the others'. When a sensor is randomly picked to generate CS measurements, all nodes within its neighborhood send their readings to it. This node generates a linear measurement by multiplying the received data including its own reading with a random Gaussian vector. The measurement is sent to the BS by two proposed methods: directly to the BS or through intermediate sensors. We formulate the total power consumption for each method and then compare them in both simulation and analysis results. The network's characteristics are considered and suggested for further energy saving.

6.2 Problem Formulation

6.2.1 Network Model

We assume to have N sensors randomly distributed in a circular shaped area, as shown in Figure 6.1. Based on a pre-chosen sensor transmission range R, the network

is connected as a graph G(V, E), where V is the vertex set that represents N sensors, and E is the edge set that depends on the value of R. In other words, the number of edges is increased as we extend R and vice versa. We also assume that sensors communicate with each other within their transmission range R. Therefore, sensor nodes can only contact within their neighborhoods, but each sensor might have a different neighborhood.

6.2.2 Neighborhood Based Data Collection Algorithm (NeiCS)

We assume a graph G(V, E) associates all N sensors in forming a connected network. Based on an appropriate transmission range R, the set of edges E could be changed, but the set of vertices V is always equal to N. Our algorithm is referred as NeiCS is addressed as follows:

1. M sensors are chosen randomly with probability $\frac{M}{N}$.

2. A chosen sensor broadcasts a beacon to its neighbors asking for data. The neighbors transmit directly their readings to requesting node.

3. The node generates a random measurement vector, then multiplies with the received data including its own data to create one CS measurement, and finally sends the measurement accompanied with the seed used for generating the measurement vector to the BS.

4. The BS implements a CS reconstruction algorithm to find sensor readings $\underline{x}$, given the measurement matrix ϕ and M measurements collected.

As shown in Figure 6.1. M random chosen sensors send data from their neighborhoods defined by the transmission range R to the BS. In NeiCS, nodes which do not get any beacon asking for data after the listening period can go to sleep to save energy. In addition, M sensors are picked randomly at each surveillance time

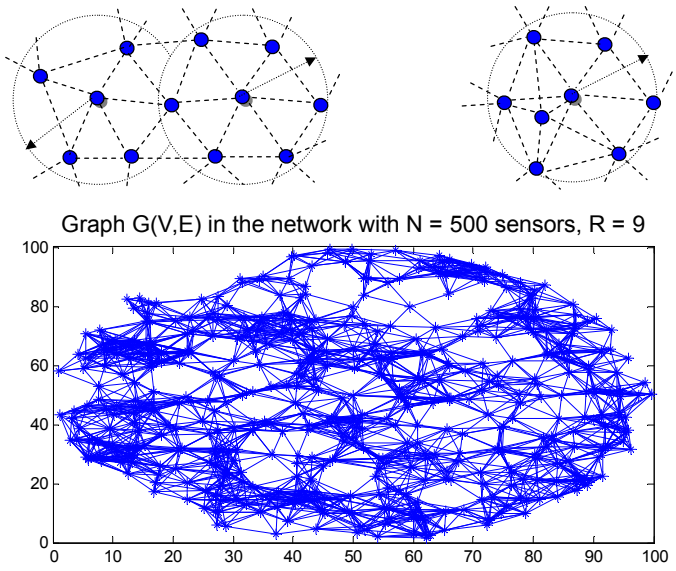


Figure 6.1: M random neighborhoods are sampled in an arbitrary network with 500 sensors; transmission range $R = 9$ defines N neighborhoods in the graph G(V, E).

which helps balance energy for such networks. Additionally, we consider two ways to transmit measurements from each random neighborhood to the BS: one-hop directly and multi-hop through active nodes, which are analyzed and formulated in the next section.

This algorithm can work well with fault tolerance in the network. Since we only take one measurement from each neighborhood separately, each node is only responsible to the others within its neighborhood. Network or node faults could be detected and recovered by fault tolerant algorithms for clustered networks [149] since each neighborhood can be considered as a cluster, or for tree-based networks [150] since we apply multi-hop routing to relay measurements. In that case, malfunctioned nodes are isolated but could be used for relaying data in the network if possible. This could be an open area for our future research.

6.2.3 Power Consumption Analysis

The total power consumption for such networks based on the NeiCS algorithm can be addressed as

$$P_{total} = (P_{nei} + P_{to\,BS}). \tag{6.1}$$

P_{nei} represents the total consumed power for neighbor nodes sending data to M sensors that get picked with probability M/N.

$$P_{nei} = \omega \times R^{\alpha} \times M, \tag{6.2}$$

where ω is the average number of neighbors of each node. α is the path loss exponent that $\alpha = 2$ and $\alpha = 4$ in free space and multipath fading channels, respectively [135]. For simplicity, we assume to have $\alpha = 2$. As assumed, sensors are randomly distributed in the area, we can find the average number of nodes deployed in the area covered by each sensor transmission range R as $\frac{N}{R_0^2} \times R^2$. Based on this, we can approximately obtain the mean value of ω as

$$\omega = (N\frac{R^2}{R_0^2} - 1). \tag{6.3}$$

Hence, the total consumed power for data gathering in M neighborhoods is calculated as

$$P_{nei} = (N\frac{R^2}{R_0^2} - 1)R^2 M, \tag{6.4}$$

that will be verified in figure 6.2 in the simulation section.

As previously mentioned, we formulate the power consumption P_{toBS} for two cases: transmit measurements directly to the BS and relay them through intermediate nodes to the BS.

Transmit CS measurements directly from random nodes to the BS

Based on the idea in [13], the expectation of the square distance between random nodes distributed in a square shape area, and the BS at the center can be calculated

as follows

$$E[d_{toBS}^2] = \int\int (x^2 + y^2)\, \rho(x, y)\, dx\, dy. \tag{6.5}$$

In our case, as assumed, the sensors are uniformly distributed in the circular area with the radius R_0. $\rho(r', \theta) = 1/(\pi R_0^2)$ is the joint probability function (pdf). We can apply Equation (6.5) into polar coordinates with r a random variable representing distance from any node to the BS

$$E[d_{toBS}^2] = \int\int r'^2 \rho(r', \theta)\, r'\, dr'\, d\theta \tag{6.6}$$

$$= \frac{1}{\pi R_0^2} \int_{\theta=0}^{2\pi} \int_{r'=0}^{R_0} r'^3 dr'\, d\theta. \tag{6.7}$$

Finally, we obtain

$$E[d_{toBS}^2] = \frac{R_0^2}{2}, \tag{6.8}$$

and

$$P_{to\,BS} = \frac{R_0^2}{2} \times M. \tag{6.9}$$

From Equations (6.4) and (6.9), the total power consumption in this case is calculated as

$$P_{total} = M[(N\frac{R^2}{R_0^2} - 1)R^2 + \frac{R_0^2}{2}]. \tag{6.10}$$

Forward CS measurements through intermediate nodes to the BS

In order to forward M measurements to the BS, we need a spanning tree to connect all sensors and a routing to forward measurements. Then, we formulate the consumed power for multi-hop transmission.

We first propose a greedy distributed algorithm to form multi-hop relaying CS measurements to the BS: We assume all sensors have the same transmission range R that enables them to communicate with each other within range R. An appropriate R that should be chosen depends on the number of sensors in the entire network to ensure that all sensors are connected as an undirected geometric graph G(V,E). Based

on the graph, we can form the routing paths for the sensors: All nodes broadcast their information about the number of hops away from the BS to their neighbors. At the first iteration, only nodes close to the BS (the R overlap the BS) have the number of hops (NoH). They name their NoH as "1" and broadcast to their neighbors in the next iterations. A node connects to one of its neighbors broadcasting their NoHs which is closest. Or it chooses the one with smaller NoH to connect. After a few iterations, the routing paths may be formed but not completely because a sensor only chooses one of its neighbors having NoH while the rest may not have one after a few iterations. So the algorithm keeps running until there is no change of routing paths between all sensors. This algorithm can be written concisely as below:

All nodes connected as a graph with a same value of R

1. ***While*** *(the routing paths is changing)*
2. *NoH(BS) = 0; $i \in N$ nodes*
3. *Nei = set of i's neighbors*
4. ***if*** *distance $[i, j] < R$, where $j \in Nei$*
5. *sensor(i) chooses sensor(j) when NoH(j) = min{NoH(Nei)}*
6. *name NoH(i) = NoH(j) + 1*
7. ***end if***
8. ***end while*** *(Until no change of routing paths between all nodes)*

$P_{to\,BS}$ is calculated after we have the tree-based multi-hop routing formed. Since we use a multi-hop transmission for relaying data from random sensors to the BS, so we need to formulate this consumed power as follows:

$$P_{toBS} = \sum_{i=1}^{M} NoH(i) \times R^2, \tag{6.11}$$

where M is the total number of measurements required, and R^2 is the power consumption based spending on each hop to relay CS measurements.

In [139], Chandler calculated the average number of relay hops in the randomly

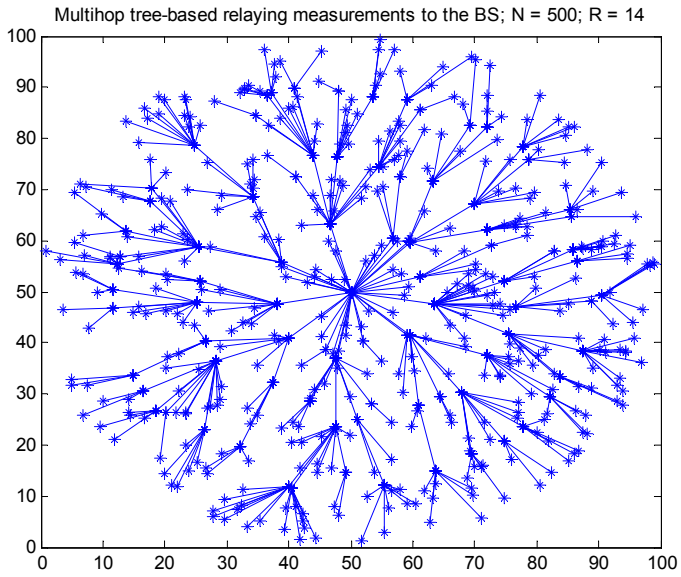


Figure 6.2: Tree formed by the greedy algorithm with 500 nodes and transmission range R = 14 to relay measurements from random sensors to the BS.

located radio network. Based on the idea, Equation (6.11) can be written as

$$P_{toBS} = NoH_{ave} \times R^2 \times M, \tag{6.12}$$

where NoH_{ave} is the average number of hops mentioned as $E[n]$ in [139]. The expectation of the number of hops is calculated based on the probability of being able to make a connection between random nodes. These nodes have an equal transmission range. If an area covered by a sensor's transmission range does not include its destination, there must be at least one node exist in the area called A to relay data.

The number of sensors existing in the area A follows a Poisson distribution with the mean value $\lambda = \frac{N}{\pi R_0^2} \times A$. The probability of being able to make a connection between a source node and a destination node is

$$P(\#ofnodes \geq 1) = 1 - P(\#ofnodes = 0) \tag{6.13}$$

$$= 1 - e^{-\frac{N}{\pi R_0^2} \times A}, \tag{6.14}$$

where $A = 2R(2\theta - sin\theta cos\theta)$ and $\theta = cos^{-1}(x/2R)$.

Since sensors are randomly distributed and are chosen in the sensing area, the distance between a random node and the BS is a random variable, denoted as x. The probability of being able to make a connection at distance x using n or less hops is denoted by $P_n(x)$. In [139] the mean value of the NoH is calculated as follows

$$E[n] = max(NoH) - \sum_{n=1}^{max(NoH)-1} \frac{P_n(x)}{P_{max(NoH)}(x)}, \tag{6.15}$$

where *max(NoH)* is the maximum number of hops allowed. Finally, we obtain the total consumed power for relaying M measurements to the BS as

$$P_{toBS} = \left\{ NoH_{max} - \sum_{n=1}^{NoH_{max}-1} \frac{P_n(x)}{P_{max(NoH)}(x)} \right\} R^2 \times M. \tag{6.16}$$

We now have the total power consumption for data collection when relaying measurements to the BS

$$P_{total} = R^2[(N\frac{R^2}{R_0^2} - 1) + NoH_{ave}]M \tag{6.17}$$

Compare NeiCS multi-hop power consumption with tree-based data gathering in formula

We simply compare the results of our analysis with a general tree-based data aggregation network applying CS. In a tree-based network, M CS measurements are collected from ω sensors at each time of sampling data. The measurement matrix created is a sparse binary matrix as mentioned in [151, 111]. We can formulate the total power consumption in this case as follows:

$$P_{total(tree-based)} = R^2[\omega \times NoH_{ave}]M, \tag{6.18}$$

where ω can be considered as the average row weight of the measurement matrix. Our NeiCS multi-hop provides the total consumed power as addressed in Equation (6.17), that can be simplified as

$$P_{total(neiCS)} = R^2[\omega + NoH_{ave}]M. \tag{6.19}$$

In WSNs, ω could not be equal to 1 since the measurement matrix has to satisfy sparsity in the CS recovery process [125]. NoH_{ave} also could not equal to 1 when the networks have many nodes with multi-hop routing. Hence, the total consumed power in Equation (6.18) is always much greater than the one in Equation (6.19).

6.3 Simulation Results

In this section, we created an arbitrary network and deployed 500 sensors ($N = 500$) randomly distributed in a circular area with radius $R_0 = 50$. N neighborhoods are connected based on a given transmission range R. We first ensure that the measurement matrix created by NeiCS can work as well as the standard full Gaussian matrix. We calculate the total power consumption for the network with the two different data collection methods. We use real sensor readings from Sensorscope: Sensor Networks for Environmental Monitoring [136]. Since our data vector $\underline{x}$ is dense in the canonical domain, in order to apply CS recovery algorithms, we use DCT as a sparsifying matrix (ψ).

As shown in Figure 6.3, the measurement matrix created by NeiCS achieves the reconstruction error as low as the Gaussian one. The error decreases as we collect more CS measurements from the network.

Figure 6.4 shows the total consumed power in M neighborhoods in which all the neighbors send their readings to the chosen nodes before the CS measurements are generated and are sent to the BS.

In the case all the CS measurements are directly transmitted to the BS, the total consumed power referred as $P_{to\,BS}$ is calculated based on the distances between M random nodes to the BS as shown in Figure 6.5.

Figure 6.6 represents the total consumed power (P_{total}) when using one-hop to forward CS measurements directly to the BS. It shows with the same transmission range R, this total power is a linear function of the total number of measurements

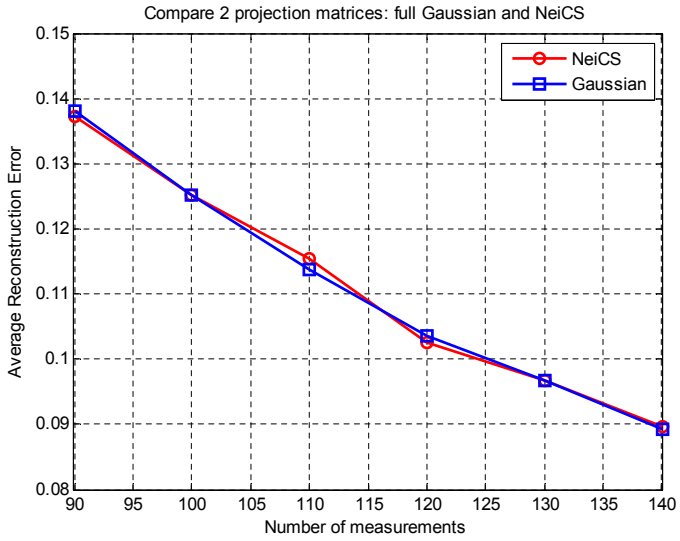


Figure 6.3: Comparison between full Gaussian measurement matrix and the one created by NeiCS

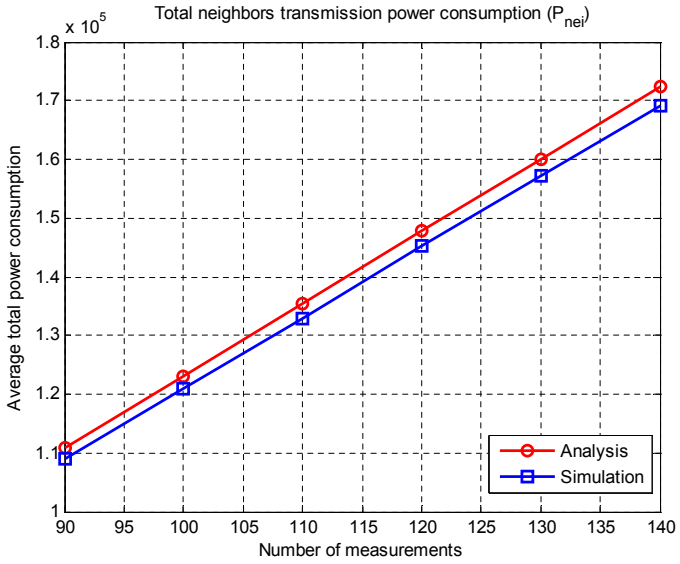


Figure 6.4: Total power consumption within difference number of neighborhoods; $N = 500$, $R_0 = 50$ and $R = 9$.

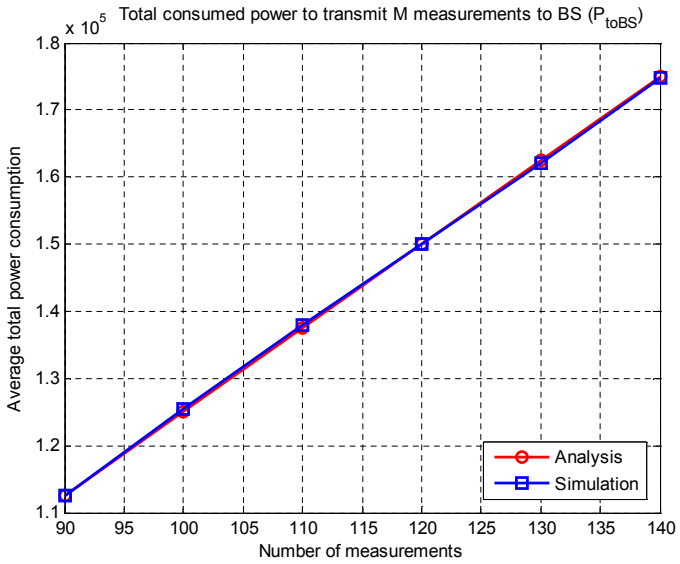


Figure 6.5: Total consumption transmits M measurements directly to the BS; $N = 500$, $R_0 = 50$ and $R = 9$.

M.

When NeiCS is applied multi-hop for relaying the measurements to the BS. P_{toBS} depends on either the average number of hops (NoH_{ave}) or the transmission range (R). These two variables are inverse to each other. The number of hops is reduced as we increase the transmission range and vice versa. Figure 6.7 shows the P_{toBS} when we chose a fixed $R = 9$ with different number of measurements. Figure 6.8 specifies that, even we increase the transmission range, the P_{toBS} cannot be increased as a general parabolic line since the number of hops is also reduced.

Finally, Figure 6.9 presents the total power consumption for the network NeiCS applying multi-hop forwarding measurements, and compares with NeiCS one-hop. If we do not consider latency or capacity for the network, the best way to save energy is using NeiCS with multi-hop routing.

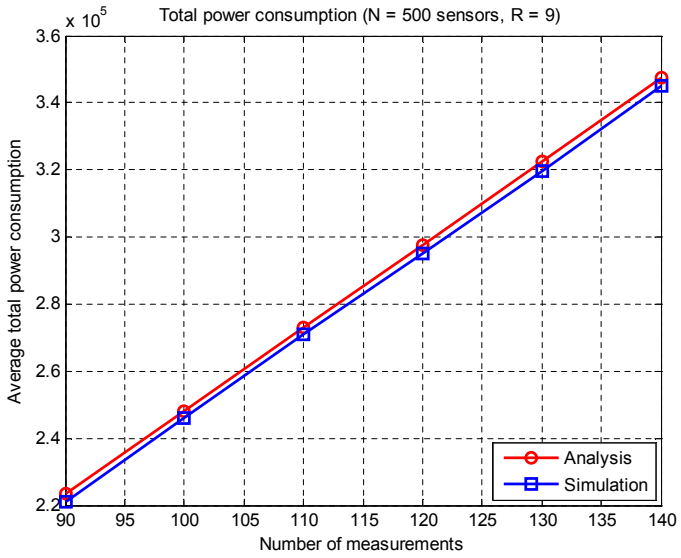


Figure 6.6: Total power consumption when NeiCS transmits measurements directly to the BS; $N = 500$, $R_0 = 50$ and $R = 9$.

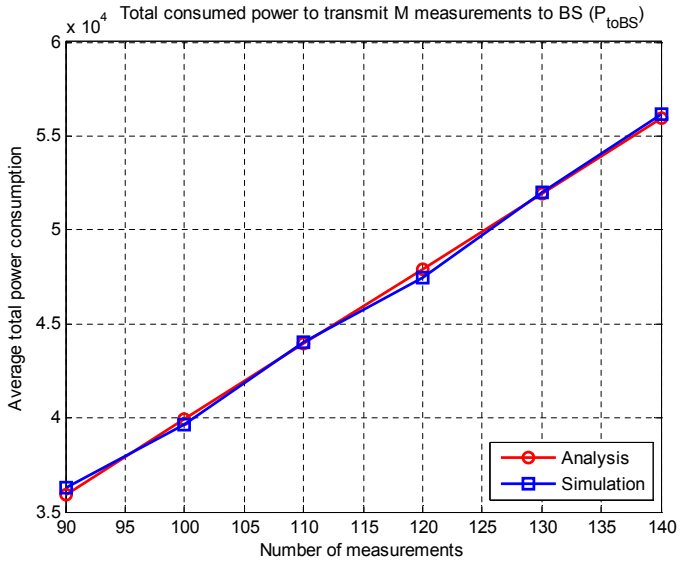


Figure 6.7: Total power consumption to relay multi-hop M measurements through intermediate nodes to the BS; $N = 500$, $R_0 = 50$ and $R = 9$.

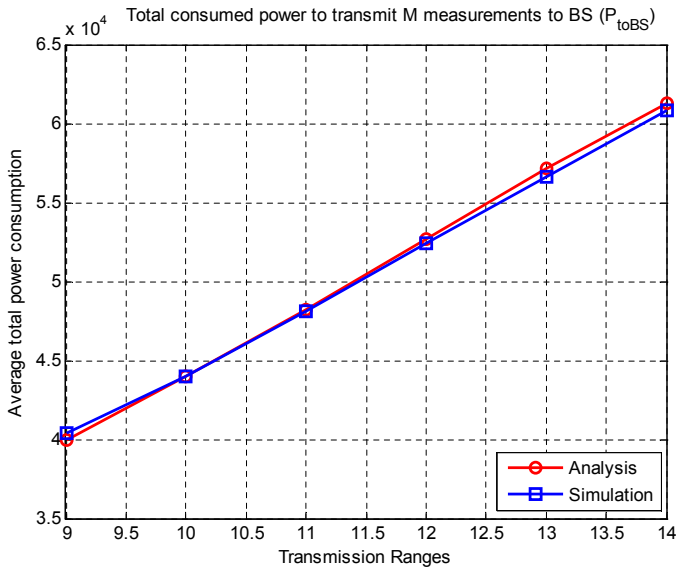


Figure 6.8: Total power consumption to relay multi-hop M = 100 measurements through intermediate nodes to the BS with different transmission ranges; $N = 500$, $R_0 = 50$.

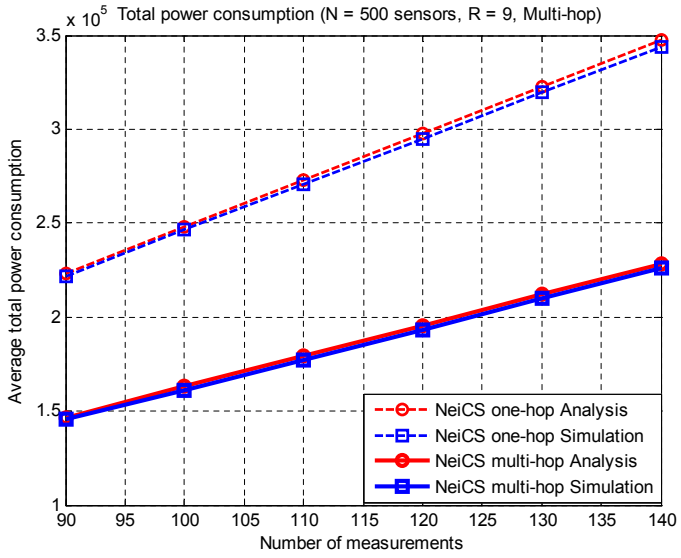


Figure 6.9: Total consumption with NeiCS when multi-hop relaying M measurements through intermediate nodes to the BS, compared to one-hop NeiCS; $N = 500$, $R_0 = 50$ and $R = 9$.

6.4 Conclusion and Future Work

In this chapter, we propose neighborhood based paradigms for sampling and collecting data in WSNs applying CS. Sensors in WSNs are connected as different neighborhoods based on an appropriate transmission range. Each neighborhood is sampled randomly and its CS measurement is sent to the BS by two common ways, directly and relaying through intermediate nodes. The algorithm, called NeiCS is proposed to collect data in each neighborhood. An iterative greedy algorithm is also proposed to form a routing tree for relaying the CS measurements to the BS. We have verified the performance of the measurement matrix to ensure that it works well with CS recovery algorithms. All consumed powers for each transmission method are formulated and compared with simulation results. We suggest the multi-hop NeiCS for further power saving for the networks since it outperforms NeiCS one-hop. In future work, we want to study the capacity and latency of the networks based on the NeiCS algorithm. Fault tolerance should also be considered for more details. Our goal is to optimize the total power consumption in order to extend the network lifetime.

CHAPTER 7

CONCLUSIONS

Wireless sensor networks (WSNs) facilitate many applications in sensing, observing and detecting areas. Since the networks in service are often based on sensors having limited power capacity, our goal is to improve such network lifetime by designing data collection algorithms that consume less energy for data transmission. We employed Compressive Sensing (CS) to our algorithms to reduce the number of measurements collected from the networks which also reduces the number of transmissions. The algorithms were addressed in chapters from Chapter 3 through Chapter 6.

In Chapter 3, the algorithm called Random Walk Based Data Gathering in WSNs was proposed. With this algorithm sensors are sampled randomly and sparsely in order to balance power consumption between sensors in the network. The predefined RW length was chosen so as to visit equally all sensors. Each CS measurement is created after each RW to be forwarded to the BS in two ways, directly (D-CSR) and in multi-hop through intermediate nodes (M-CSR).

In Chapter 4, Cluster Based Data Collection in WSNs was proposed. With this algorithm, non-CH sensors only send their data to their CHs once. CS measurements are generated at the CHs to be forwarded to the BS directly (DCCS) or by inter-CH multi-hop (ICCS) routing for data recovery.

Tree Based Data Gathering in WSNs for energy-efficient manner was proposed in Chapter 5. The algorithm exploits storage capacity of sensor nodes to reduce the number of transmissions in a routing tree. Each parent node stores its children nodes' data and generates and sends their accumulative data to an upper node each time they

contribute to form CS measurements corresponding to a sparse binary measurement matrix. The combined measurements are sent instead of individual sensor readings which significantly reduces power consumption for the networks.

In Chapter 6 Neighborhood Based Data Collection in WSNs was proposed to exploit the short communication between sensors and their neighbors to sample randomly the networks. A certain number of sensors is chosen randomly to create CS measurements to be sent to the BS. The CS measurements are created only between sensor neighborhoods with a limited communication range that reduces power consumption for such networks.

These algorithms all focused on data collection methods employing CS in WSNs. They did not address improvement of CS performance in reconstructing data and they also did not degrade it. The algorithms exploited common ways sensors can send data to each other and combined with the ways which CS measurements are created to reduce power consumption for WSNs. The measurement matrices in Chapters 3 and 5 are sparse binary. In Chapter 4 the measurement matrix is a block diagonal matrix. In Chapter 6, the measurement matrix is an asymptotically sparse binary matrix. This matrix may degrade the CS performance when it becomes excessively sparse. In all chapters, all the matrices formed by the algorithms are tested to be able to perform as well as the full dense Gaussian random matrix which corresponds to full sampling all sensor nodes for one CS measurement. Furthermore, power consumption for data transmission in such networks was analyzed, formulated and simulated, and suggestions were presented for the networks to consume the least power in terms of prolonging the network lifetime.

BIBLIOGRAPHY

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393 – 422, 2002.

[2] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292 – 2330, 2008.

[3] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: a survey," *Wireless Communications, IEEE*, vol. 11, pp. 6–28, Dec 2004.

[4] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 325 – 349, 2005.

[5] D.L.Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, pp. 1289 – 1306, 2006.

[6] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, pp. 489 – 509, Feb. 2006.

[7] R. Baraniuk, "Compressive sensing [lecture notes]," *Signal Processing Magazine, IEEE*, vol. 24, pp. 118 –121, July 2007.

[8] E. Candes and M. Wakin, "An introduction to compressive sampling," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, 2008.

[9] J. Haupt, W. Bajwa, M. Rabbat, and R. Nowak, "Compressed sensing for networked data," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 92–101, 2008.

[10] G. Quer, R. Masiero, G. Pillonetto, M. Rossi, and M. Zorzi, "Sensing, compression, and recovery for wsns: Sparse signal modeling and monitoring framework," *Wireless Communications, IEEE Transactions on*, vol. 11, pp. 3447–3461, October 2012.

[11] M. Davenport, J. Laska, J. Treichler, and R. Baraniuk, "The pros and cons of compressive sensing for wideband signal acquisition: Noise folding versus dynamic range," *Signal Processing, IEEE Transactions on*, vol. 60, pp. 4628–4642, Sept 2012.

[12] M. Nguyen and Q. Cheng, "Efficient data routing for fusion in wireless sensor networks," in *The 25th International Conference on Computer Applications in Industry and Engineering (CAINE), New Orleans, LA, 2012*, Nov 2012.

[13] M. T. Nguyen, "Minimizing energy consumption in random walk routing for wireless sensor networks utilizing compressed sensing," in *System of Systems Engineering (SoSE), 2013 8th International Conference on*, pp. 297–301, June 2013.

[14] M. T. Nguyen and K. Teague, "Compressive sensing based energy-efficient random routing in wireless sensor networks," in *Advanced Technologies for Communications (ATC), 2014 International Conference on*, pp. 187–192, Oct 2014.

[15] M. T. Nguyen and N. Rahnavard, "Cluster-based energy-efficient data collection in wireless sensor networks utilizing compressive sensing," in *Military Communications Conference, MILCOM 2013 - 2013 IEEE*, pp. 1708–1713, Nov 2013.

[16] M. T. Nguyen and K. Teague, "Compressive sensing based data gathering in clustered wireless sensor networks," in *Distributed Computing in Sensor Systems (DCOSS), 2014 IEEE International Conference on*, pp. 187–192, May 2014.

[17] M. T. Nguyen, K. Teague, and N. Rahnavard, "Inter-cluster multi-hop routing in wireless sensor networks employing compressive sensing," in *Military Communications Conference (MILCOM), 2014 IEEE*, pp. 1133–1138, Oct 2014.

[18] M. T. Nguyen and K. Teague, "Distributed dct based data compression in clustered wireless sensor networks," in *Design of Reliable Communication Networks (DRCN), 2015 11th International Conference on the*, pp. 255–258, March 2015.

[19] M. T. Nguyen and K. Teague, "Tree-based energy-efficient data gathering in wireless sensor networks deploying compressive sensing," in *Wireless and Optical Communication Conference (WOCC), 2014 23rd*, pp. 1–6, May 2014.

[20] M. T. Nguyen and K. Teague, "Neighborhood based data collection in wireless sensor networks employing compressive sensing," in *Advanced Technologies for Communications (ATC), 2014 International Conference on*, pp. 198–203, Oct 2014.

[21] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Commun. ACM*, vol. 43, pp. 51–58, May 2000.

[22] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next century challenges: Mobile networking for “smart dust”," in *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom '99, (New York, NY, USA), pp. 271–278, ACM, 1999.

[23] J. Rabaey, J. Ammer, J. da Silva, J.L., and D. Patel, "Picoradio: Ad-hoc wireless networking of ubiquitous low-energy sensor/monitor nodes," in *VLSI, 2000. Proceedings. IEEE Computer Society Workshop on*, pp. 9–12, 2000.

[24] S. H. Lee, S. Lee, H. Song, and H. S. Lee, "Wireless sensor network design for tactical military applications : Remote large-scale environments," in *Military Communications Conference, 2009. MILCOM 2009. IEEE*, pp. 1–7, Oct 2009.

[25] J. Gehrke and P. Seshadri, "Querying the physical world," *IEEE Personal Communications*, vol. 7, pp. 10–15, 2000.

[26] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat monitoring: Application driver for wireless communications technology," *SIGCOMM Comput. Commun. Rev.*, vol. 31, pp. 20–41, Apr. 2001.

[27] M. P. Hamilton and M. Flaxman, "Scientific data visualization and biological diversity: new tools for spatializing multimedia observations of species and ecosystems," *Landscape and Urban Planning*, vol. 21, no. 4, pp. 285 – 287, 1992.

[28] P. Corke, T. Wark, R. Jurdak, W. Hu, P. Valencia, and D. Moore, "Environmental wireless sensor networks," *Proceedings of the IEEE*, vol. 98, pp. 1903–1917, Nov 2010.

[29] N. Noury, T. Herve, V. Rialle, G. Virone, E. Mercier, G. Morey, A. Moro, and T. Porcheron, "Monitoring behavior in home using a smart fall sensor and position sensors," in *Microtechnologies in Medicine and Biology, 1st Annual International, Conference On. 2000*, pp. 607–610, 2000.

[30] C. Baker, K. Armijo, S. Belka, M. Benhabib, V. Bhargava, N. Burkhart, A. Der Minassians, G. Dervisoglu, L. Gutnik, M. Haick, C. Ho, M. Koplow, J. Mangold, S. Robinson, M. Rosa, M. Schwartz, C. Sims, H. Stoffregen, A. Waterbury, E. Leland, T. Pering, and P. Wright, "Wireless sensor networks for home health care," in *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, vol. 2, pp. 832–837, May 2007.

[31] E. Petriu, N. D. Georganas, D. Petriu, D. Makrakis, and V. Groza, "Sensor-based information appliances," *Instrumentation Measurement Magazine, IEEE*, vol. 3, pp. 31–35, Dec 2000.

[32] C. Herring and S. Kaplan, "Component-based software systems for smart environments," *Personal Communications, IEEE*, vol. 7, pp. 60–61, Oct 2000.

[33] D.-M. Han and J.-H. Lim, "Smart home energy management system using ieee 802.15.4 and zigbee," *Consumer Electronics, IEEE Transactions on*, vol. 56, pp. 1403–1410, Aug 2010.

[34] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications*, vol. 30, no. 1415, pp. 2826 – 2841, 2007. ¡ce:title¿Network Coverage and Routing Schemes for Wireless Sensor Networks¡/ce:title¿.

[35] A. Amis and R. Prakash, "Load-balancing clusters in wireless ad hoc networks," in *Application-Specific Systems and Software Engineering Technology, 2000. Proceedings. 3rd IEEE Symposium on*, pp. 25 –32, 2000.

[36] G. Gupta and M. Younis, "Load-balanced clustering of wireless sensor networks," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 3, pp. 1848 – 1852 vol.3, may 2003.

[37] G. Gupta and M. Younis, "Fault-tolerant clustering of wireless sensor networks," in *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, vol. 3, pp. 1579 –1584 vol.3, march 2003.

[38] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability* (L. M. L. Cam and J. Neyman, eds.), vol. 1, pp. 281–297, University of California Press, 1967.

[39] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," 2003.

[40] S. Banerjee and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks," 2001.

[41] H. Steinhaus, "Sur la division des corp materiels en parties," *Bull. Acad. Polon. Sci*, vol. 1, pp. 801–804, 1956.

[42] S. P. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, pp. 129–137, 1982.

[43] R. Xu and I. Wunsch, D., "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on*, vol. 16, pp. 645 –678, may 2005.

[44] P. Hansen and N. Mladenovi, "J-means: a new local search heuristic for minimum sum of squares clustering," *Pattern Recognition*, vol. 34, no. 2, pp. 405 – 413, 2001.

[45] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: Analysis and implementation," 2000.

[46] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338 – 353, 1965.

[47] J.-S. Zhang and Y.-W. Leung, "Improved possibilistic c-means clustering algorithms," *Fuzzy Systems, IEEE Transactions on*, vol. 12, pp. 209 – 217, april 2004.

[48] D. Hoang, R. Kumar, and S. Panda, "Fuzzy c-means clustering protocol for wireless sensor networks," in *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, pp. 3477 –3482, july 2010.

[49] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, p. 10 pp. vol.2, jan. 2000.

[50] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *Wireless Communications, IEEE Transactions on*, vol. 1, pp. 660 – 670, oct 2002.

[51] S. Bandyopadhyay and E. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, pp. 1713 – 1723 vol.3, march-3 april 2003.

[52] A. Amis, R. Prakash, T. Vuong, and D. Huynh, "Max-min d-cluster formation in wireless ad hoc networks," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, pp. 32 –41 vol.1, 2000.

[53] M. Demirbas, A. Arora, and V. Mittal, "Floc: A fast local clustering service for wireless sensor networks," in *Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS/DSN*, 2004.

[54] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," 2004.

[55] O. Younis and S. Fahmy, "Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 3, pp. 366 – 379, oct.-dec. 2004.

[56] H. Huang and J. Wu, "A probabilistic clustering algorithm in wireless sensor networks," in *Vehicular Technology Conference, 2005. VTC-2005-Fall. 2005 IEEE 62nd*, vol. 3, pp. 1796 – 1798, sept., 2005.

[57] C. Li, M. Ye, G. Chen, and J. Wu, "An energy-efficient unequal clustering mechanism for wireless sensor networks," in *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pp. 8 pp. –604, nov. 2005.

[58] S. Yi, J. Heo, Y. Cho, and J. Hong, "Peach: Power-efficient and adaptive clustering hierarchy protocol for wireless sensor networks," *Computer Communications*, vol. 30, no. 1415, pp. 2842 – 2852, 2007. ¡ce:title¿Network Coverage and Routing Schemes for Wireless Sensor Networks¡/ce:title¿.

[59] B. Gong, L. Li, S. Wang, and X. Zhou, "Multihop routing protocol with unequal clustering for wireless sensor networks," in *Computing, Communication, Control, and Management, 2008. CCCM '08. ISECS International Colloquium on*, vol. 2, pp. 552 –556, aug. 2008.

[60] H. Le, D. Hoang, and R. Poliah, "S-web: An efficient and self-organizing wireless sensor network model," in *Network-Based Information Systems* (M. Takizawa, L. Barolli, and T. Enokido, eds.), vol. 5186 of *Lecture Notes in Computer Science*, pp. 179–188, Springer Berlin / Heidelberg.

[61] S. Heikalabad, N. Firouz, A. Navin, and M. Mirnia, "Heech: Hybrid energy effective clustering hierarchical protocol for lifetime prolonging in wireless sensor networks," in *Computational Intelligence and Communication Networks (CICN), 2010 International Conference on*, pp. 325 –328, nov. 2010.

[62] S. M. Hedetniemi, S. T. Hedetniemi, and A. L. Liestman, "A survey of gossiping and broadcasting in communication networks," *Networks*, vol. 18, no. 4, pp. 319–349, 1988.

[63] A. G. Dimakis, A. D. Sarwate, and M. J. Wainwright, "Geographic gossip: Efficient aggregation for sensor networks," in *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, IPSN '06, (New York, NY, USA), pp. 69–76, ACM, 2006.

[64] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, MobiCom '99, (New York, NY, USA), pp. 174–185, ACM, 1999.

[65] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks," *Wirel. Netw.*, vol. 8, pp. 169–185, Mar. 2002.

[66] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, (New York, NY, USA), pp. 56–67, ACM, 2000.

[67] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA '02, (New York, NY, USA), pp. 22–31, ACM, 2002.

[68] F. Ye, A. Chen, S. Lu, and L. Zhang, "A scalable solution to minimum cost forwarding in large sensor networks," in *Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on*, pp. 304–309, 2001.

[69] C. Schurgers and M. Srivastava, "Energy efficient routing in wireless sensor networks," in *Military Communications Conference, 2001. MILCOM 2001. Com-*

munications for Network-Centric Operations: Creating the Information Force. IEEE, vol. 1, pp. 357–361 vol.1, 2001.

[70] F. Ye, G. Zhong, S. Lu, and L. Zhang, "Gradient broadcast: A robust data delivery protocol for large scale sensor networks," *Wirel. Netw.*, vol. 11, pp. 285–298, May 2005.

[71] M. Chu, H. Haussecker, F. Zhao, M. Chu, H. Haussecker, and F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks," *International Journal of High Performance Computing Applications*, vol. 16, 2002.

[72] N. Sadagopan, B. Krishnamachari, and A. Helmy, "The acquire mechanism for efficient querying in sensor networks," in *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, pp. 149–155, May 2003.

[73] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," *SIGMOD Rec.*, vol. 31, pp. 9–18, Sept. 2002.

[74] R. Shah and J. Rabaey, "Energy aware routing for low energy ad hoc sensor networks," in *Wireless Communications and Networking Conference, 2002. WCNC2002. 2002 IEEE*, vol. 1, pp. 350–355 vol.1, Mar 2002.

[75] L. Lovsz, "Random walks on graphs: A survey," 1993.

[76] S. D. Servetto and G. Barrenechea, "Constrained random walks on random graphs: Routing algorithms for large scale wireless sensor networks," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA '02, (New York, NY, USA), pp. 12–21, ACM, 2002.

[77] I. Mabrouki, X. Lagrange, and G. Froc, "Random walk based routing protocol for wireless sensor networks," in *Proceedings of the 2nd international conference on Performance evaluation methodologies and tools*, ValueTools '07, (ICST, Brussels, Belgium, Belgium), pp. 71:1–71:10, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[78] C. Avin and G. Ercal, "On the cover time and mixing time of random geometric graphs," *Theor. Comput. Sci*, 2007.

[79] C. M. Angelopoulos, S. Nikoletseas, D. Patroump, and C. Rapropoulos, "A new random walk for efficient data collection in sensor networks," in *Proceedings of the 9th ACM international symposium on Mobility management and wireless access*, MobiWac '11, (New York, NY, USA), pp. 53–60, ACM, 2011.

[80] V. Rodoplu and T. Meng, "Minimum energy mobile wireless networks," in *Communications, 1998. ICC 98. Conference Record. 1998 IEEE International Conference on*, vol. 3, pp. 1633–1639 vol.3, Jun 1998.

[81] B. Parkinson and J. J. Spiker, *Global Positioning System: Theory and Applications, Volume 1*, vol. 163. xx 1996.

[82] L. Li and J. Halpern, "Minimum-energy mobile wireless networks revisited," in *Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 1, pp. 278–283 vol.1, Jun 2001.

[83] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom '01, (New York, NY, USA), pp. 70–84, ACM, 2001.

[84] Y. Yu, R. Govindan, and D. Estrin, "Geographical and energy aware routing: a recursive data dissemination protocol for wireless sensor networks," tech. rep., 2001.

[85] B. Karp and H. T. Kung, "Gpsr: Greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, (New York, NY, USA), pp. 243–254, ACM, 2000.

[86] I. Stojmenovic and X. Lin, "Gedir: Loop-free location based routing in wireless networks," in *Proc. IASTED Int. Conf. on Parallel and Distributed Computing and Systems*, pp. 1025–1028, 1999.

[87] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-case optimal and average-case efficient geometric ad-hoc routing," in *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*, MobiHoc '03, (New York, NY, USA), pp. 267–278, ACM, 2003.

[88] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *Wirel. Netw.*, vol. 8, pp. 481–494, Sept. 2002.

[89] H. Nyquist, "Certain topics in telegraph transmission theory," *American Institute of Electrical Engineers, Transactions of the*, vol. 47, pp. 617–644, April 1928.

[90] D. L. Donoho and M. Elad, "Optimally sparse representation in general (non-orthogonal) dictionaries via 1 minimization," in *Proc. Natl Acad. Sci. USA 100 2197202*, 2003.

[91] E. Candes and T. Tao, "Decoding by linear programming," *Information Theory, IEEE Transactions on*, vol. 51, pp. 4203–4215, Dec 2005.

[92] M. Elad, "Optimized projections for compressed sensing," 2006.

[93] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *Information Theory, IEEE Transactions on*, vol. 53, pp. 4655–4666, Dec 2007.

[94] D. Needell and R. Vershynin, "Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, pp. 310–316, April 2010.

[95] D. Needell and J. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301 – 321, 2009.

[96] X. Wang, Z. Zhao, Y. Xia, and H. Zhang, "Compressed sensing based random routing for multi-hop wireless sensor networks," in *Communications and Information Technologies (ISCIT), 2010 International Symposium on*, pp. 220–225, 2010.

[97] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing markov chain on a graph," *SIAM REVIEW*, vol. 46, pp. 667–689, 2003.

[98] M. Lin, C. Luo, F. Liu, and F. Wu, "Compressive data persistence in large-scale wireless sensor networks.," in *GLOBECOM*, pp. 1–5, IEEE, 2010.

[99] M. Sartipi and R. Fletcher, "Energy-efficient data acquisition in wireless sensor networks using compressed sensing," in *Data Compression Conference (DCC), 2011*, pp. 223 –232, March 2011.

[100] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, 1953.

[101] Y. Lin, B. Liang, and B. Li, "Data persistence in large-scale sensor networks with decentralized fountain codes," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 1658–1666, 2007.

[102] D. Vukobratovic, C. Stefanovic, V. Crnojevic, F. Chiti, and R. Fantacci, "A packet-centric approach to distributed rateless coding in wireless sensor networks," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON '09. 6th Annual IEEE Communications Society Conference on*, pp. 1–8, 2009.

[103] H. Zheng, f. yang, X. Tian, X. Gan, X. Wang, and S. Xiao, "Data gathering with compressive sensing in wireless sensor networks: A random walk based approach," *Parallel and Distributed Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.

[104] J. Luo, L. Xiang, and C. Rosenberg, "Does compressed sensing improve the throughput of wireless sensor networks?," in *Communications (ICC), 2010 IEEE International Conference on*, pp. 1–6, May 2010.

[105] L. Xiang, J. Luo, and A. Vasilakos, "Compressed data aggregation for energy efficient wireless sensor networks," in *Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2011 8th Annual IEEE Communications Society Conference on*, pp. 46–54, June 2011.

[106] L. Xiang, J. Luo, and C. Rosenberg, "Compressed data aggregation: Energy-efficient and high-fidelity data collection," *Networking, IEEE/ACM Transactions on*, vol. 21, pp. 1722–1735, Dec 2013.

[107] R. Xie and X. Jia, "Transmission-efficient clustering method for wireless sensor networks using compressive sensing," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 25, pp. 806–815, March 2014.

[108] C. Luo, F. Wu, J. Sun, and C. W. Chen, "Efficient measurement generation and pervasive sparsity for compressive data gathering," 2010.

[109] J. Luo, L. Xiang, and C. Rosenberg, "Does compressed sensing improve the throughput of wireless sensor networks?," in *Communications (ICC), 2010 IEEE International Conference on*, pp. 1–6, 2010.

[110] W. Wang, M. Garofalakis, and K. Ramchandran, "Distributed sparse random projections for refinable approximation," in *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pp. 331 –339, april 2007.

[111] R. Xie and X. Jia, "Minimum transmission data gathering trees for compressive sensing in wireless sensor networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pp. 1–5, 2011.

[112] M. Elad, "Optimized projections for compressed sensing," *Signal Processing, IEEE Transactions on*, vol. 55, pp. 5695–5702, Dec 2007.

[113] S. Lee, S. Pattem, M. Sathiamoorthy, B. Krishnamachari, and A. Ortega, "Compressed sensing and routing in multi-hop networks," *University of Southern California CENG Technical Report*, 2009.

[114] S. Lee and A. Ortega, "Joint optimization of transport cost and reconstruction for spatially-localized compressed sensing in multi-hop sensor networks," 2010.

[115] M. Mahmudimanesh, A. Khelil, and N. Yazdani, "Map-based compressive sensing model for wireless sensor network architecture, A starting point," in *Mobile Wireless Middleware, Operating Systems, and Applications - Workshops, Mobilware 2009 Workshops, Berlin, Germany, April 2009, Revised Selected Papers*, pp. 75–84, 2009.

[116] H. Hu and Z. Yang, "Spatial correlation-based distributed compressed sensing in wireless sensor networks," in *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*, pp. 1–4, Sept 2010.

[117] M. Leinonen, M. Codreanu, and M. Juntti, "Distributed correlated data gathering in wireless sensor networks via compressed sensing," in *Signals, Systems and Computers, 2013 Asilomar Conference on*, pp. 418–422, Nov 2013.

[118] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393 – 422, 2002.

[119] I. Mabrouki, X. Lagrange, and G. Froc, "Random walk based routing protocol for wireless sensor networks," in *Proceedings of the 2nd international conference on Performance evaluation methodologies and tools*, ValueTools '07, (ICST, Brussels, Belgium, Belgium), pp. 71:1–71:10, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[120] C. M. Angelopoulos, S. Nikoletseas, D. Patroump, and C. Rapropoulos, "A new random walk for efficient data collection in sensor networks," in *Proceedings of the 9th ACM international symposium on Mobility management and wireless access*, MobiWac '11, (New York, NY, USA), pp. 53–60, ACM, 2011.

[121] T. Do, L. Gan, N. Nguyen, and T. Tran, "Fast and efficient compressive sensing using structurally random matrices," *Signal Processing, IEEE Transactions on*, vol. 60, pp. 139–154, Jan 2012.

[122] M. Rabbat, J. Haupt, A. Singh, and R. Nowak, "Decentralized compression and predistribution via randomized gossiping," in *Information Processing in Sensor Networks, 2006. IPSN 2006. The Fifth International Conference on*, pp. 51–59, 2006.

[123] C. Luo, F. Wu, J. Sun, and C. W. Chen, "Efficient measurement generation and pervasive sparsity for compressive data gathering," *Wireless Communications, IEEE Transactions on*, vol. 9, pp. 3728–3738, December 2010.

[124] J. Wang, S. Tang, B. Yin, and X.-Y. Li, "Data gathering in wireless sensor networks through intelligent compressive sensing," in *INFOCOM, 2012 Proceedings IEEE*, pp. 603–611, March 2012.

[125] R. Berinde and P. Indyk, "Sparse recovery using sparse random matrices," MIT-CSAIL Technical Report, 2008.

[126] R. Bellman, "On a routing problem," *Quart. Appl. Math.*, vol. 16, pp. 87–90, 1958.

[127] S. D. Servetto and C. Univerisity, "Constrained random walks on random graphs: Routing algorithms for large scale wireless sensor networks," pp. 12–21, 2002.

[128] L. Lima and J. Barros, "Random walks on sensor networks," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, pp. 1–5, 2007.

[129] H. Tian, H. Shen, and T. Matsuzawa, "Randomwalk routing for wireless sensor networks," in *Parallel and Distributed Computing, Applications and Technologies, 2005. PDCAT 2005. Sixth International Conference on*, pp. 196–200, 2005.

[130] R. Berinde, A. Gilbert, P. Indyk, H. Karloff, and M. Strauss, "Combining geometry and combinatorics: A unified approach to sparse signal recovery," in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*, pp. 798–805, Sept 2008.

[131] C. Avin and G. Ercal, "On the cover time and mixing time of random geometric graphs," *Theoretical Computer Science*, vol. 380, no. 12, pp. 2 – 22, 2007. Automata, Languages and Programming.

[132] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Mixing times for random walks on geometric random graphs," in *the proceedings of SIAM ANALCO*, pp. 240–249, SIAM, 2005.

[133] D. Vukobratovic, C. Stefanovic, V. Crnojevic, F. Chiti, and R. Fantacci, "A packet-centric approach to distributed rateless coding in wireless sensor networks," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON '09. 6th Annual IEEE Communications Society Conference on*, pp. 1 –8, June 2009.

[134] F. Xue and P. R. Kumar, "The number of neighbors needed for connectivity of wireless networks," *Wirel. Netw.*, vol. 10, pp. 169–181, Mar. 2004.

[135] T. S. Rappaport, *Wireless Communications: Principles and Practice (2nd Edition)*. Prentice Hall, 2 ed., Jan. 2002.

[136] http://lcav.epfl.ch/op/edit/sensorscope en.

[137] J. M. Steele, "Minimal spanning trees for graphs with random edge lengths," in *Mathematics and computer science, II (Versailles, 2002)*, Trends Math., pp. 223–245, Basel: Birkhäuser, 2002.

[138] R. Bellman, "On a routing problem," *Quart. Appl. Math.*, vol. 16, pp. 87–90, 1958.

[139] S. Chandler, "Calculation of number of relay hops required in randomly located radio network," *Electronics Letters*, vol. 25, no. 24, pp. 1669–1671, 1989.

[140] M. Davenport, J. Laska, J. Treichler, and R. Baraniuk, "The pros and cons of compressive sensing for wideband signal acquisition: Noise folding versus dynamic range," *Signal Processing, IEEE Transactions on*, vol. 60, pp. 4628–4642, Sept 2012.

[141] M. Handy, M. Haase, and D. Timmermann, "Low energy adaptive clustering hierarchy with deterministic cluster-head selection," in *Mobile and Wireless Communications Network, 2002. 4th International Workshop on*, pp. 368 – 372, 2002.

[142] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 1, pp. 4 vol. (xxxv+2866), march 2004.

[143] R. Xie and X. Jia, "Minimum transmission data gathering trees for compressive sensing in wireless sensor networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pp. 1–5, Dec.

[144] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "Energy-efficient broadcast and multicast trees in wireless networks," *Mobile Networks and Applications*, vol. 7, pp. 481–492, 2002.

[145] H. L. Yap, A. Eftekhari, M. Wakin, and C. Rozell, "The restricted isometry property for block diagonal matrices," in *Information Sciences and Systems (CISS), 2011 45th Annual Conference on*, pp. 1 –6, March 2011.

[146] M. Wakin, J. Y. Park, H. L. Yap, and C. Rozell, "Concentration of measure for block diagonal measurement matrices," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 3614 –3617, march 2010.

[147] T. Do, L. Gan, N. Nguyen, and T. Tran, "Fast and efficient compressive sensing using structurally random matrices," *Signal Processing, IEEE Transactions on*, vol. 60, pp. 139–154, Jan 2012.

[148] T. Dang, N. Bulusu, and W. chi Feng, "Rida: A robust information-driven data compression architecture for irregular wireless sensor networks," 2007.

[149] G. Gupta and M. Younis, "Fault-tolerant clustering of wireless sensor networks.," in *WCNC*, pp. 1579–1584, IEEE, 2003.

[150] L. Chitnis, A. Dobra, and S. Ranka, "Analyzing the techniques that improve fault tolerance of aggregation trees in sensor networks," *J. Parallel Distrib. Comput.*, vol. 69, pp. 950–960, Dec. 2009.

[151] W. Wang, "Distributed sparse random projections for refinable approximation," in *In IEEE/ACM Int. Symposium on Information Processing in Sensor Networks (IPSN*, 2007.

APPENDIX A

RANDOM WALK BASED DATA GATHERING IN WIRELESS SENSOR NETWORKS

A.1 Additional Analysis for D-CSR to calculate E_{dtoBS} in order to compare with M-CSR

In our previous results in Section 3.4.2, E_{toBS} is calculated in a square area network based on the mean square distance to send directly CS measurements from RWs to the BS. To be compared with M-CSR, we provide an additional analysis to calculate E_{dtoBS} for this circular area network with the BS at the center as shown in Figure A.1.

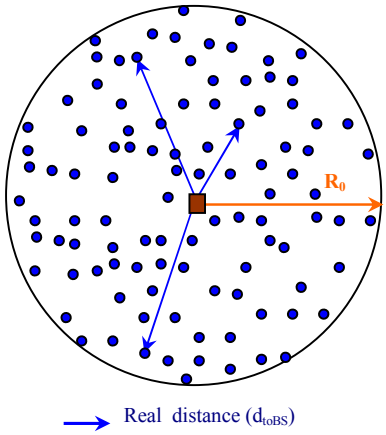


Figure A.1: Real distances from any random node to the BS in a circle shape area arbitrary network

Since sensors are uniformly randomly distributed in the model and the sensors are also chosen randomly, d_{toBS} can be considered as a random variable. The expectation

of the square distance $\mathrm{E}[d_{toBS}^2]$ can be calculated following the idea in [13] as

$$\mathrm{E}[d_{toBS}^2] = \int\int r'^2 \rho(r', \theta)\, r'\, dr'\, d\theta, \tag{A.1}$$

where random variable r presents a real transmitting distance from any random node to the BS. As assumed, the sensors are uniformly distributed in the area with the radius of R_0, and $\rho(r', \theta) = 1/(\pi R_0^2)$ is the joint probability function (pdf). Finally, we obtain

$$\mathrm{E}[d_{toBS}^2] = \frac{1}{\pi R_0^2} \int_{\theta=0}^{2\pi} \int_{r=0}^{R_0} r'^3 dr'\, d\theta \tag{A.2}$$

$$= \frac{R_0^2}{2}. \tag{A.3}$$

VITA

Minh Tuan Nguyen

Candidate for the Degree of

Doctor of Philosophy

Dissertation: DATA COLLECTION ALGORITHMS IN WIRELESS SENSOR NETWORKS EMPLOYING COMPRESSIVE SENSING

Major Field: Electrical Engineering

Biographical:

Personal Data: Born in Thai Nguyen City, Thai Nguyen Province, Vietnam on April 05, 1978.

Education:

Received the B.S. degree from Hanoi University of Communications and transport, Hanoi, Vietnam, 2001, in Electrical Engineering
Received the M.S. degree from Military Technical Academy (Le Quy Don Technical University), Hanoi, Vietnam, 2007, in Electrical Engineering
Completed the requirements for the degree of Doctor of Philosophy with a major in Electrical Engineering Oklahoma State University in December, 2015.

Experience:

Working in industry as technical consultant (2001-2003).
7 years working as lecturer and researcher at Thai Nguyen University of Technology (TNUT), Viet Nam (2003-2010).
Teaching Assistant, Research Associate at Oklahoma State University (2010-2015).
Serve as reviewer for several prestigious international conferences and journals.
Serve as Section chair at SoSE2015
Author of 13 conference papers published (12 IEEE, 01 other) and 1 IEEE conference accepted, 1 journal paper accepted and some journal papers submitted or in progress