Distributed Clustering Algorithms for Lossy Wireless Sensor Networks

A Thesis Presented

by

Zhexi Pan

to

The Graduate School

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Electrical Engineering

Stony Brook University

December 2009

Stony Brook University

The Graduate School

Zhexi Pan

We, the thesis committee for the above candidate for the Master of Science degree, hereby recommend acceptance of this thesis.

Yuanyuan Yang – Thesis Advisor
Professor, Department of Electrical and Computer Engineering

Sangjin Hong
Associate Professor, Department of Electrical and Computer Engineering

This thesis is accepted by the Graduate School.

Lawrence Martin
Dean of the Graduate School

Abstract of the Thesis

Distributed Clustering Algorithms for Lossy Wireless Sensor Networks

by

Zhexi Pan

Master of Science

in

Electrical Engineering

Stony Brook University

2009

Wireless sensor networks (WSNs) have emerged as a new information gathering paradigm in a wide-range of applications. An important class of such applications is continuous monitoring applications such as habitat monitoring and structural monitoring, where a large number of wireless sensor nodes are employed for continuous sensing in a field, and the sensing data from scattered sensor nodes are gathered and transmitted to a base station for processing. Among different approaches proposed to gather data in WSNs, clustering is generally considered as a promising approach for data gathering in large-scale WSNs due to its hierarchical nature.

Recent experimental studies have revealed that a large percentage of wireless links are lossy and unreliable for data delivery in WSNs. Such findings raise new challenges for the design of clustering algorithms in WSNs in terms of data reliability and energy efficiency. In this thesis, we propose distributed clustering algorithms for WSNs by taking into account of the lossy nature of wireless links. We first formulate the one-hop clustering problem that maintains reliability as well as saves energy into an integer program and prove its NP-hardness. We then propose a metric-based distributed clustering algorithm to solve the problem. We design a new metric called *selection weight* for each sensor node that can indicate both link qualities around the node and its capability of being a cluster head. We further extend the algorithm to multi-hop clustering to achieve better scalability. Extensive simulations have been conducted under a realistic link model and the results demonstrate that the proposed clustering algorithm reduces the total energy consumption in the network and prolongs network lifetime significantly compared to a typical distributed clustering algorithm, HEED, that does not consider lossy links.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to begin by thanking Professor Yuanyuan Yang, my thesis advisor and mentor for the past two years. Professor Yang has been an outstanding researcher with broad knowledge and grand vision, her enthusiasm in innovations, persistence and optimism towards research has deeply impressed me and encouraged me. She is also a wonderful advisor, providing me with support, encouragement, patience and freedom to explore research topics by myself. I thank her for the countless hours she has spent with me, reading my paper, and discussing everything from research to life experience. I am grateful to Professor Yang on both a professional and personal level.

I would like to thank Professor Esther M. Arkin for her initial feedback on the content of the integer program discussed in the thesis. Her suggestions were very helpful. I would also like to thank Professor Sangjin Hong for reading and commenting on my thesis.

I would like to thank my colleagues Min Yang, Miao Zhao, Ji Li, Xi Deng, Lin Liu, Dawei Gong, and Zhiyang Guo in High Performance Computing and Networking Research Lab for their friendship and all of their help during the past years. I also want to thank all my friends at Stony Brook who have helped me in various ways and have made my stay at Stony Brook a pleasant and memorable one.

I am indebted to my parents, Xianchun Pan and Liwa Hu , for everything that they have given to me. They have stood by me in everything I have done, continuously offering me constant support, patience, encouragement, understanding and endless love.

Finally, I would like to thank my fiance, Yu Zhang. We have known each other for more than 12 years and Yu has always been my best friend and soul mate. He teaches me how to enjoy life to the fullest and keeps my life filled with laughter and love.

Chapter 1

Introduction

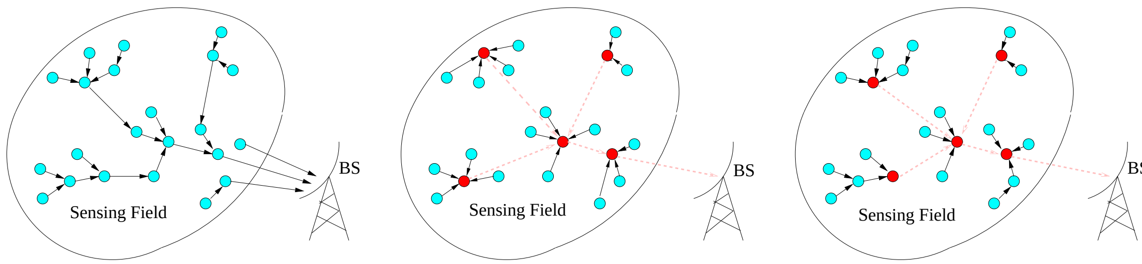
In this chapter, we introduce the clustering approach, its challenges and goals in the existence of unreliable links in wireless communications, and the contributions of the thesis.

1.1 The Clustering Approach

Wireless sensor networks (WSNs) have gained much attention recently for their potential use in a wide range of applications. An important class of such applications is continuous monitoring applications, such as habitat monitoring [1], structural monitoring [2], emergency response [3], etc. For these applications a large number of wireless sensor nodes are employed for continuous sensing in a field. The sensing data from scattered sensor nodes are then gathered and transmitted to a base station (BS) for processing.

Due to the tremendous practical interests, much research effort has already been devoted to efficient data gathering in WSNs and different approaches have been proposed, such as power aware routing [4, 5], mobile data gathering [6, 7] and clustering [8–12]. In a homogeneous network where sensors are organized into a flat topology, although power aware routing finds better routes to save energy, it shares one common feature with other routing schemes: sensor nodes close to BS are overloaded with the relay traffic from the nodes that cannot reach BS directly. Hence they consume more energy than others and become the bottlenecks of the network lifetime. Network lifetime is defined as the time until the first node depletes its energy. By introducing a mobile collector into the field, the mobile data gathering approach alleviates the burden of relay traffic for each node and thus prolongs the network lifetime. The mobile collector gathers data from sensors within its communication range while it traverses through the entire field. Although the benefits of mobile data gathering approach is remarkable, it may cause relatively long delay in data collection, since each sensor node has to wait for the collector before its data can be sent.

Clustering is generally considered as a promising approach for data gathering in large-scale WSNs due to its hierarchical nature. Compared to the aforementioned approaches, clustering alleviates the "hot spots" problem encountered in routing, and achieves a balance between the uniformity of energy consumption and the long data collection latency in mobile data gathering. Specifically, clustering is to group sensors into disjoint clusters such that sensors as cluster members form the lower layer of the network send data to their cluster heads, and cluster heads form the higher layer of the network and forward data to the BS. This hierarchical nature of clustering increases the scalability and is especially suitable for large-scale WSNs. Fig. 1.1 illustrates the routing and clustering approaches.



(a) Power-aware routing (b) One-hop clustering (c) Multi-hop clustering

Figure 1.1: Different ways to gather data in a WSN: routing and clustering.

1.2 Challenges and Design Goals

Due to the nature of wireless communications, WSNs may suffer from the unreliable wireless links. Recent empirical studies [13–15] have revealed the prevalence of lossy and asymmetric links in WSNs, which are unreliable for data delivery. It was reported in [14] that one third of the links in their test-bed composed of 60 Mica motes experienced more than 30% of the packet loss even under light traffic loads in an office building. The existence of such lossy links is problematic to support reliable data gathering over a long period of time in WSNs, as lossy links can result in not only failure of data delivery but also more energy consumption due to packet retransmissions.

Although a multitude of clustering algorithms [16] have been proposed for WSNs in the literature, the problem of lossy wireless links in WSNs has not been addressed by existing solutions. First, lossy links are unreliable for data delivery while reliability of packet transmission is critical in many applications. For example, in a sensor network for air pollution monitoring in a chemical plant or radiation level control in a nuclear plant, the reliability of packet transmission largely affects the quality of surveillance. A common approach of ensuring reliable packet delivery is to employ hop-by-hop retransmissions, where each individual link provides reliable forwarding to the next hop by retransmitting lost packets when necessary. Second, the energy consumption on transmission and reception is under-estimated by assuming all the wireless links are reliable, which fails to capture the actual energy consumed in realistic scenarios. The energy spent in a reliable packet delivery is proportional to the number of transmissions needed till the packet is successfully received. Therefore, it may be possible to reduce the total energy consumption in the network by selecting some "good" links other than using links randomly.

Based on the above discussions, taking into consideration of the lossy nature of wireless links, the design goals of our clustering approach in the thesis is to maintain reliability as well as save energy from unnecessary retransmissions. We also want to maintain favorable properties as a basic clustering approach, for example, a small amount of cluster heads generated in the network and uniform distribution of the cluster heads in a homogenous network.

1.3 Contributions

The thesis provides the main contribution in that the distributed clustering algorithms for WSNs that takes account of the lossy wireless links are proposed for the first time, and extensive simulations demonstrate a much better performance in terms of the packet delivery ratio, the total energy consumption in the network, and the network lifetime, compared to a typical clustering algorithm called HEED [11], that does not consider lossy links.

In more detail, we formulate the one-hop clustering problem under lossy links into an integer program and prove the NP-hardness of the problem. The integer program has the objective to minimize the total energy spent on transmissions and receptions in the entire network for a single round, in which each sensor node as a cluster member sends one data packet to its cluster head. It also has a constraint on the total number of cluster heads so that the goal of maintaining a small number of cluster heads can be satisfied. By solving the integer program we are able to obtain the optimal solution for the one-hop clustering problem. However, due to the NP-hardness of the clustering problem the brutal force search method of the optimal solution in a large network becomes infeasible. Therefore we proceed to propose a distributed clustering algorithm that works well for large-scale networks.

We propose a metric-based distributed clustering algorithm to solve the one-hop clustering problem. We design a new metric called *selection weight* for each sensor node that can indicate both link qualities around the node and its capability of being a cluster head. The distributed one-hop clustering algorithm is proven to be of low complexity in both time and messages per node in the thesis.

We further extend the algorithm to multi-hop clustering to achieve better scalability and discuss some implementation issues and an extension to deal with node failures, which is a common problem in applications involving a large-scale sensor networks.

Extensive simulations are conducted under a realistic link model and the results demonstrate that the proposed clustering algorithm can reduce the total energy consumption in the network and prolong network lifetime significantly compared to a typical distributed clustering algorithm, HEED, that does not consider lossy links.

1.4 Thesis Outline

The remainder of the thesis is organized as follows. Chapter 2 reviews the related work on clustering algorithms and lossy wireless link problems. Chapter 3 gives one-hop and multi-hop clustering algorithms, and discusses some implementation issues. Chapter 4 presents the simulation results and Chapter 5 concludes the thesis and discusses the future work.

Chapter 2

Related Work

In this chapter, we introduce the related work regarding clustering algorithms, empirical studies for wireless sensor networks and routing protocols that take the lossy link problem into consideration.

2.1 Clustering Algorithms

Clustering algorithms for wireless sensor networks (WSNs) have been extensively studied in the last few years, see, for example, [8, 10–12]. The max-min d-cluster algorithm proposed in [8] generates cluster heads that form a d-hop dominating set using two rounds of message flooding. Each node in the cluster is at most d hops away from its cluster head. Since this algorithm was designed for ad hoc networks, its clustering goal focuses on the stability of the cluster head other than considerations on energy efficiency. The LEACH protocol [10] designed for WSNs shows that clustering can prolong network lifetime significantly compared with routing. LEACH protocol selects cluster heads distributively at each node via probability-based self-election and forms a one-hop intra and inter clustering topology, where all the nodes are assumed within the communication range of each other and the base station. However, in practice this assumption may not always hold, since cluster heads usually are regular sensors and the base station is often not directly reachable from every node. The HEED algorithm proposed in [11] selects cluster heads using metric-based self-election and forms one-hop (intra) clusters. The metric takes into consideration of the residual energy of each node as well as the communication cost such as the neighbor proximity and the cluster density. Recently, a clustering protocol that uses a cluster-based cost metric was proposed in [12] for underwater sensor networks. The metric measures the communication cost for the entire cluster and takes into consideration of the characteristics of the underwater network, such as the relative location between the cluster head and the underwater base station, which requires the geographic information of all the nodes in the cluster.

2.2 Empirical Studies and Realistic Link Models

Due to the nature of wireless communications, WSNs may suffer from the unreliable wireless links. Several actual sensor network deployments[13–15, 17] have shown that a large fraction of wireless links are lossy and explored characteristics of these lossy links. In [13], authors presented empirical results from a simple flooding protocol and identified its complexities by examining separate effects at the various layers of the protocol stack. One important observation from their experiments is that the distribution of packet reception over distance is non-uniform. In [14], the authors reported measurements of packet delivery in three different environments, an indoor office building, a habitat and an open parking lot respectively. They pointed out the existence of a "gray area" that experienced large variability in packet reception. In [15], authors presented an empirical characterization of wireless links experienced on their platform, and then studied and evaluated their link estimator. They showed that using a simple time averaged EWMA estimator, frequency based table management, and cost-based routing with a minimum expected transmission metric was the most effective solution to the routing. From their empirical results they identified the existence of three distinct reception regions of a wireless link: connected, transitional and disconnected. The transitional region is quite significant in size and characterized by high-variance in reception rate and asymmetric connectivity. However, the link quality is stable when nodes are immobile. Similar result was also reported in a more recent paper in [18]. In [17], the authors studied on the causes of the transitional region and gave an analytical link model addressing its lossy characteristics.

2.3 Routing Protocols Considering Lossy Wireless Links

Although the aforementioned clustering algorithms could generate a set of cluster heads with good distributions and disjoint clusters, they did not consider the problem of lossy and asymmetric links in WSNs. Meanwhile, the effect of lossy links has received much attention in the design of routing protocols in multi-hop wireless networks, see, for example, [19–21]. In [19], an expected transmission count metric (ETX) was proposed for finding high throughput paths. Measurements from their test-bed demonstrated the effectiveness of ETX with much improved performance. In [20], a trade-off was identified in geographic routing between shorter high-quality links and longer lossy links, and the product of the packet reception rate and the distance was shown to be a good forwarding metric. In [21], the problem of finding a minimum energy reliable path in a hop-by-hop retransmission model was solved by using a similar link cost metric to ETX.

In summary, considering the lossy nature of the wireless links has demonstrated its advantage in routing, while it has not been explored in clustering. This motivates us to design a clustering algorithm that accounts for lossy links in WSNs.

Chapter 3

Distributed Clustering Algorithms for Lossy Wireless Sensor Networks

In this chapter, we present and analyze our distributed clustering algorithms for lossy wireless sensor networks (WSNs) in detail. We start with describing the network model that we used to capture the lossy nature of WSNs, formally formulate the clustering problem in the one-hop case and prove the NP-hardness of the clustering problem. We proceed to present distributed algorithms for one-hop and k-hop clustering problems one by one. We also have had discussions on some implementation issues and an extension to the algorithms to deal with node failures at the end of this chapter.

3.1 Network Model

We consider a set of sensor nodes randomly deployed onto a 2-D area. Sensor nodes are assumed to be static and equipped with omnidirectional antennas. Each sensor node has a fixed transmission power P_t and a fixed reception power P_r and the transmission range of each sensor is R.

Lossy wireless links are modeled by the probabilities of the packet reception ratio (PRR). We adopt a hop-by-hop retransmission model, for example, a simple automatic repeat request (ARQ) mechanism at the MAC layer. ARQ mechanism uses acknowledgments and timeouts to achieve reliable data transmissions. If a sender does not receive an acknowledgment before the timeout, it retransmits the packet until it receives an acknowledgment or exceeds a predefined number of transmissions, T. Each link (i, j) has a PRR of p_{ij} that is the probability of node j successfully receiving the packet transmitted by node i, and a non-negative weight $w_{ij} = \frac{1}{p_{ij}}$. If the number of retransmissions is not bounded, the weight of each link denotes the expected number of transmissions needed to send a packet successfully over the link. The value of p_{ij} in this thesis is generated using a realistic, empirically validated link model in [17]. The model agrees very well with previous empirical findings on the characteristics of wireless links. For example, the distribution of packet reception over distance is non-uniform [13], and there exists three distinct reception regions of a wireless link: connected, transitional and disconnected. While the transitional region is quite large in size and characterized by high-variance in PRR and asymmetry [15], the PRRs for a connected region and a disconnected region is almost always 1 and 0, respectively. Previous studies also pointed out that the link quality is stable when nodes are immobile [15, 18], which makes our probability model more reasonable. The model also takes the packet bit length l and transmission power P_t as its parameters. Therefore, a link from node i to node j has a higher p_{ij} when node i transmits a packet of smaller sizes or using a higher transmission power. The term "lossy" in this thesis refers to the intrinsic characteristic of the physical layer in wireless communications, and we do not consider other packet loss due to contention and interference in the MAC layer. This is reasonable because in a low-rate network, such as a sensor network, contention and interference from simultaneous transmissions can be avoided or effectively minimized by using TDMA-based MAC protocols.

Each node is initially provisioned with an amount of energy E_{max}, corresponding to a fully charged battery. The value of E_{max} of different sensor nodes can be different. Network lifetime is defined as the time until the first sensor node depletes its energy. Suppose node i transmits a packet of a fixed bit length l to its neighbor node j over link (i, j), the energy spent at node i until the packet is successfully received by node j is $w_{ij} \cdot l \cdot P_t$ and the energy spent at node j is $w_{ij} \cdot l \cdot P_r$. The total energy consumed over link (i, j) is $w_{ij} \cdot l \cdot (P_t + P_r)$, and we define a constant P_l equals $l \cdot (P_t + P_r)$. Apparently, the energy consumption of a reliable packet delivery over a link is proportional to its weight, and the energy consumption over a path is proportional to the sum of the weights of all the links on the path. Between any two nodes there always exists a *minimum energy cost path* that has the minimum sum of weights of all the links on the path.

In this thesis we mainly focus on the intra-cluster communications under lossy wireless links, instead of the inter-cluster communications. First of all, intra-cluster communications are directly related to the cluster head selection and cluster member association, which are the two main components of a clustering algorithm. On the other hand, inter-cluster communications may depend on network applications. For

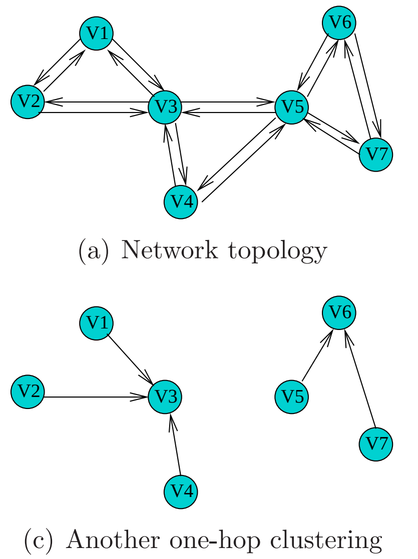


Figure 3.1: Different clustering schemes for a network.

example, cluster heads may use a multi-hop routing protocol to communicate with each other and the BS. It is also possible to introduce a mobile collector that visits each cluster head for data gathering to substitute the multi-hop routing.

3.2 Clustering Problem Formulation

Consider a connected network represented by a directed graph $G = (V, A)$, where V is the set of sensor nodes and A is the set of directed links. The graph is directed so as to account for the asymmetry of links. Each link (i, j) has a weight $w(i, j)$ as defined in the previous subsection. Clustering is to group sensors into disjoint clusters such that each sensor as a cluster member is associated with exactly one cluster head. We refer to the clustering as *one-hop clustering* in which each cluster member is within one hop of its cluster head, and the clustering as *k-hop clustering* in which each cluster member is at most k hops away from its cluster head. For any given network G, there exists multiple possible clustering schemes. For example, Fig. 3.1 shows three clustering schemes for the same network, where the first two are *one-hop clustering*, and the third one is *2-hop clustering*. Besides satisfying the reliability requirement by using retransmissions, our clustering goal is to save energy so that the network lifetime can be prolonged, and to keep the number of cluster heads small so that the network can have a good scalability. We start with the one-hop case. We first formulate it into an integer program to solve the problem optimally for small sizes and to gain some insights for designing a better algorithm. In the formulation we assume that the predefined number of transmissions, T, is not bounded, and that in a single round each node in the sensor network sends one packet to its associated cluster head until the packet is successfully received. Before we proceed, we explain some terms that will be used in the formulation. The notations are summarized in Table 3.1.

Each cluster head i has a cluster-centric communication cost C_i, which is the sum of energy cost on transmissions and receptions in a single round from each cluster member j to cluster head i, given by

$$C_i = \sum_{j \in V} P_l n_{ji} x_{j,\,i} \tag{3.1}$$

where P_l is a constant factor mentioned in the previous subsection that is introduced to calculate the energy cost on each link for every packet delivery. If there is a link (j, i) in the graph, $n_{ji} = w_{ji}$, which is the expected number of transmissions needed to send a packet successfully from node j to node i. If there is not a link (j, i), we define $n_{ji} = infinity$. We also define $n_{i,i} = 0$ since there is no need for each node to send a packet to itself. And $x_{j,\,i} = \{0, 1\}$ is a Boolean variable of node association. If

Table 3.1: Notations used in the formulation

Indices:	
$V = \{v_1, v_2, \ldots\}$	A set of sensor nodes in the network.
Constants:	
N_c	An integer system parameter that constrains the total number of cluster heads.
P_l	A constant factor introduced to calculate the energy cost on each link for every packet delivery.
$f_{j,i} = \{0, 1\}$	Location indicator. If node j has a direct link (j, i) to reach node i, $f_{j,i} = 1$, otherwise, $f_{j,i} = 0$.
n_{ji}	$n_{ji} = w_{ji}$, if there exists link (j, i), otherwise $n_{j,i} = infinity$, except $n_{ii} = 0$.
Variables:	
C_i	Cluster-centric communication cost, which is the sum of energy cost on transmissions and receptions in a single round from each cluster member to cluster head i.
$I_i = \{0, 1\}$	Indicator of selected cluster head. If node i is selected to be the cluster head, $I_i = 1$, otherwise, $I_i = 0$.
$x_{j,i} = \{0, 1\}$	Indicator of node association. If node j chooses node i as its cluster head, $x_{j,i} = 1$, otherwise, $x_{j,i} = 0$.

node j chooses node i as its cluster head, $x_{j,i} = 1$, otherwise, $x_{j,i} = 0$.

Then the one-hop clustering problem under lossy links can be formulated as follows.

$$\textbf{Minimize} \quad \sum_{i \in V} C_i = \sum_{i \in V} \sum_{j \in V} P_l n_{ji} x_{j,i} \tag{3.2}$$

$$\textbf{Subject to} \quad x_{j,i} \leq f_{j,i} \cdot I_i, \forall i, j \in V \tag{3.3}$$

$$\sum_{i \in V} x_{j,i} = 1, \forall j \in V \tag{3.4}$$

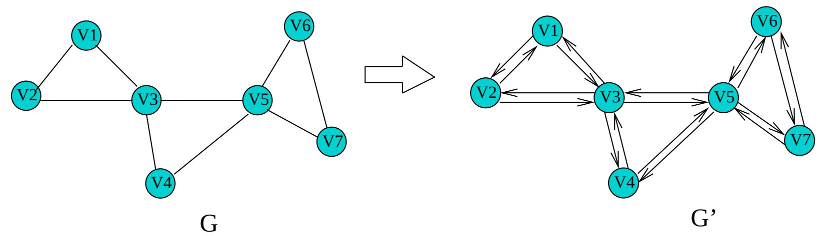
$$\sum_{i \in V} I_i \leq N_c \tag{3.5}$$

In the above formulation, objective function (3.2) minimizes the total energy consumption in the network by summing up energy cost C_i for each cluster i. Constraint (3.3) ensures that two conditions should be met before node j can be associated to node i. One condition is that there exists a link from node j to node i determined by the boolean location indicator $f_{j,i}$ and another condition is that node i is a cluster head. That is, if $x_{j,i} = 1$ then $f_{j,i} = 1$ and $I_i = 1$. Constraint (3.4) ensures that each node is associated to exactly one cluster head. Constraint (3.5) ensures that the number of selected cluster heads is small as long as a given value N_c is small. This constraint not only achieves our goal of having a small number of cluster head in order to increase the network scalability, but also it excludes the trivial case of generating "singleton" clusters which contain only one node (the cluster head itself) and have the cluster-centric communication cost $C_i = P_l n_{ii} x_{i,i}$ equal 0.

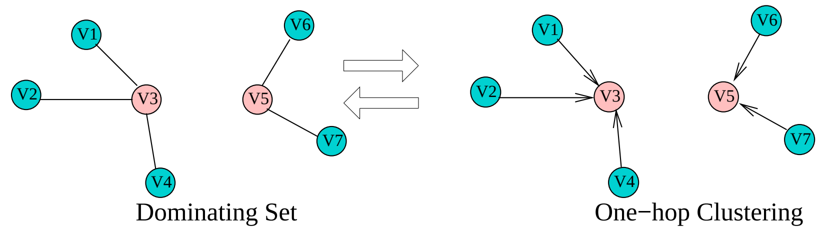
Note that the clustering problem, especially the one-hop clustering problem aforementioned, is very similar to the *dominating set (DS) problem* in the sense that a dominating set covers all other nodes in a graph while a set of cluster heads dominates all other sensor nodes in a WSN. Given an undirected graph G with a set of vertices V and a set of edges E, and a positive integer Z, a subset of vertices $D \subseteq V$ is called a dominating set if every vertex not in D is connected to at least one member of D by an edge. The DS problem is to determine whether there is a dominating set of size at most Z for G and it was proved to be NP-complete. Similarly, we have the following lemma concerning the NP-hardness of the general clustering problem.

Lemma 1. *The clustering problem under lossy links in WSNs is NP-hard.*

Proof. To show the problem is NP-hard, we reduce it from the NP-complete DS problem. The reduction algorithm takes an instance of the DS problem as input. Given a graph G and an integer Z, as shown in Fig. 3.2, it constructs an auxiliary



(a) Reducing dominating set problem to one-hop clustering problem.



(b) The relationship between the two problems

Figure 3.2: One-hop clustering problem is NP-hard.

graph $G^{'}$ by replacing every edge in G with two directed arcs pointing to each other in $G^{'}$. We set all the weights over arcs to 0, and the constraints on the number of cluster heads N_c to Z. Thus, $G^{'}$ becomes a special unweighted instance of one-hop clustering problem.

To complete the proof, we show that G has a dominating set D of size at most Z if and only if $G^{'}$ has a one-hop clustering with at most N_c cluster heads.

If we find a dominating set D of size at most Z in G, we use all the nodes in D as cluster heads in $G^{'}$, and assign each of the remaining nodes to exactly one node in D. By the definition of a dominating set, each node $j \in V - D$ is connected to some node $i \in D$ by an edge (i, j) in G. Since $G^{'}$ is constructed by replacing each edge in G with two directed arcs pointing to each other, we have arcs (i, j) and (j, i) in $G^{'}$. Thus we are able to assign cluster member j to cluster head $i \in D$ using arc (j, i). Clearly, the number of cluster heads in $G^{'}$ equals the size of dominating set D in G. This way we find the one-hop clustering with at most $Z = N_c$ cluster heads.

On the other hand, if we have the one-hop clustering with at most N_c cluster heads, we use these N_c cluster heads to form a set D in G. In the one-hop clustering problem, each node j is either a cluster member associated to some cluster head i via an arc (j, i) or a cluster head itself. Since both arcs (j, i) and (i, j) correspond to the same edge (j, i) in G, we have that each node j in G either has an edge (j, i) connecting to node $i \in D$ or in set D itself, which is a dominating set. The size Z of dominating set D equals the number N_c of cluster heads. Thus we find a dominating set D of size at most $N_c = Z$ in G.

We have shown in the above that the unweighted one-hop clustering problem is NP-hard. Since it is a special case of the clustering problem under lossy links (i.e. with weights), the general clustering problem is also NP-hard. $\square$

3.3 One-hop and k-hop Clustering Algorithms

In this section we present distributed algorithms for one-hop and k-hop clustering problems. The algorithms select a set of cluster heads first, then form clusters by associating each node to exactly one cluster head. Cluster heads are distributively self-elected based on a metric that accounts for lossy wireless links. Before we proceed to describe the algorithms in detail, we first introduce the metric.

3.3.1 Selection Weight Metric

Recall that our clustering goal is to minimize the total energy cost in the network including energy consumption on retransmissions, and to maintain a small number of cluster heads as well. As in other metric-based clustering algorithms, a better metric is to help select a better set of cluster heads that fulfill the clustering goal. We have following two observations that can help design a better metric to save energy.

- Cluster heads consume more energy than other nodes and thus may deplete their energy earlier.

- Energy consumption on a reliable packet delivery is proportional to the number of transmissions for the packet.

Based on the first observation, we should rotate the roles of cluster heads in the network, and select the nodes with relatively high residual energy E_{res} as cluster heads. Based on the second observation, we should select the nodes as cluster heads whose "potential clusters" have better link qualities and accordingly fewer retransmissions in the clusters. Combining these two aspects, we design a *selection weight metric* as follows.

Consider the largest possible one-hop cluster of node i, which consists itself and its one-hop neighboring nodes nbr_i. One-hop neighboring nodes are defined as those falls in the transmission range and have $PRR > 0$ over links connecting to node i. We use T_{avg} to denote the *average number of transmissions* for the cluster of node i when it is the cluster head

$$T_{avg}(i) = \frac{\sum_{j \in nbr_i} n_{ji}}{|nbr_i|} \tag{3.6}$$

where n_{ji} is as defined in the previous section. T_{avg} of a node provides a good estimate on the overall link qualities in the cluster if this node becomes the cluster head. Furthermore, since we use fixed transmission power and reception power for each link, it is not difficult to see that T_{avg} also provides an estimate on the average energy consumption on a successful delivery of a packet in the cluster, which is $T_{avg} \cdot P_l$. It captures energy consumed on retransmissions due to lossy links. Similarly, in the

k-hop clustering case, T_{avg} is defined for the largest possible k-hop cluster of node i which consists itself and its k-hop neighboring set nbr_i^k.

$$T_{avg}(i) = \frac{\sum_1^k \sum_{j \in nbr_i^k} n_{ji}^k}{|nbr_i^k|} \tag{3.7}$$

where n_{ji}^k denotes the sum of weights of all the links on the minimum energy cost path from node j to node i.

We define *residual energy ratio* E_{ratio} for each node to indicate its capability of being a cluster head as

$$E_{ratio} = \frac{E_{res}}{E_{max}} \tag{3.8}$$

where E_{max} is the maximum energy that a sensor node has when its battery is fully charged and E_{res} is the current residual energy in the node. The residual energy can be directly measured by the sensor node or estimated by calculation, as the energy consumed per bit for sensing, processing and communication is typically known and fixed.

We finally define the *selection weight metric* of node i as follows to model the link qualities in its neighborhood and its capability of being a cluster head.

$$W_{sel}(i) = \frac{T_{avg}(i)}{E_{ratio}} \tag{3.9}$$

3.3.2 One-hop Clustering Algorithm

We are now in the position to present our one-hop clustering algorithm which will be executed by each node periodically. The pseudocode for node u is given in Table 3.2.

We assume that each node knows its neighbors and weights over links in the

Table 3.2: One-hop clustering algorithm executed at node u

```
Input: receive(i, declaration), receive(i, acknowledge);
       /*receive a message from node i*/
Output: broadcast(u, declaration), broadcast(u, acknowledge);
       /*broadcast a message within one hop */

Pre-Process
calculate the selection weight;
broadcast weight to its one-hop neighbors;
isfinal (u) ← False;
create a list containing selection weights for its neighbors;
create a tentative cluster head set 𝒯 containing all the neighbors
and itself;
create an empty final cluster head set ℱ;

Cluster Head Selection Process
see Table 3.3

Cluster Formation Process
if isfinal (u) = False
   if receive(i, declaration)
       add node i into ℱ;
   end
   if ℱ ≠ ∅
       associate to a cluster head from ℱ using a link of the
       minimum weight;
   else
       broadcast(u, declaration);
      isfinal (u) ← True;
   end
end
```

neighborhood initially. We defer the implementation details on how to initialize the network to Section 3.4. During the execution of the algorithm, each node keeps a record of selection weights for all its neighbors, and maintains two lists for its tentative

Table 3.3: cluster head selection process executed at node u

```
Cluster Head Selection Process
if u = mincost(T)
    broadcast(u, declaration);
    isfinal (u) ← True;
    wait till cluster head selection process time out;
end
do {
   if receive(i, declaration)
       add node i into F;
       if this is the first time to hear the declaration message
           broadcast(u, acknowledge);
       end
   end
   if receive(i, acknowledge);
      remove node i from T;
   end
   if u has never received any declaration message and
    u = mincost(T)
      broadcast(u, declaration);
      isfinal (u) ← True;
      wait until time out;
   end
} until time out;
```

cluster heads $\mathcal{T}$ and final cluster heads $\mathcal{F}$. The length of a time slot in the algorithm is equal to the communication time of a message transmission in one hop. Note that the message used in the algorithm to exchange information is of a much smaller size than a data packet. According to the realistic link model, a smaller message size has a higher PRR when the transmission power is fixed. Thus, it is reasonable to assume that PRR for these small messages is close to 1. Hence, the transmission time of such a small message is bounded and well-defined. We also adopt the following two rules in the algorithm:

- *Rule 1: Head Selection*

  A node elects itself to be a cluster head, if it has the minimum selection weight among all the nodes in the tentative cluster head list, or wins the random selection when it has the same minimum weight as some other nodes in the list. In the second case, a node will randomly pick an ID number among nodes with the same minimum weight. If the ID number picked is itself, we say it wins the random selection.

- *Rule 2: Node Association*

  A node associates itself to a cluster head that has the minimum value among all the minimum energy cost paths from the node to possible cluster heads in the final cluster heads list $\mathcal{F}$. The minimum energy cost path is a path that has the minimum sum of weights of all the links (possibly contains only one link) on the path, which was defined in Section 3.1.

The algorithm can be described as follows. Initially, each node calculates its selection weight and locally broadcasts the weight to all the neighbors. In the meanwhile, each node puts all the neighbors and itself into the tentative cluster head list $\mathcal{T}$. After hearing from all the neighbors, each node enters the cluster head selection process and decides autonomously whether it will volunteer to be a cluster head. The duration of the head selection process is pre-defined and is three time slots long, in which we allow each node to run the head selection rule (each run lasts one time slot) at most twice. In the first time slot, each node runs the head selection rule for the first time. If a node satisfies the rule, it decides to be a cluster head and broadcasts a declaration message to its neighbors. We call such a cluster head *volunteer cluster* *head*. Volunteer cluster heads keep silent afterward and wait for the time-out of the head selection process. Otherwise, if a node does not satisfy the head selection rule, it does nothing in this time slot. In the second time slot, each node except for volunteer cluster heads responds in a message-driven fashion as follows. If a node hears a declaration message, it broadcasts an acknowledgment message to its neighbors, indicating that it has been covered by some cluster head. If the node hears more than one declaration messages from its neighbors, however, it only broadcasts the acknowledge message once, and puts all the nodes that have broadcast declaration messages into its final cluster head list $\mathcal{F}$. If a node has not heard any declaration messages but heard acknowledgment messages, which implies that there is no volunteer cluster head in its neighborhood, it keeps silent and removes the neighboring nodes that have sent the acknowledgment messages from its tentative cluster head list $\mathcal{T}$. If a node does not hear any message, it keeps silent and does nothing. In the third time slot, silent nodes in the second time slot run the head selection rule again to "break the silence" in the neighborhood.

After the head selection process ends, the cluster formation process starts. In this process, those nodes that have been covered by multiple cluster heads follow the node association rule to associate themselves to a cluster head. Those nodes that are uncovered declare themselves to be cluster heads, and we call such cluster heads *forced cluster heads*.

We have the following lemma concerning the correctness of the algorithm.

Lemma 2. *After the execution of one-hop clustering algorithm, a node is either a cluster head or a cluster member that belongs to exactly one cluster head.*

Proof. During the execution of the algorithm, each node maintains a binary variable

$isfinal$ to indicate its status of being a cluster head or not. If node i elects itself to be a cluster head, $isfinal$(i) is set to the value of "True" thereafter permanently. Initially, each node sets the value to "False" in the algorithm. Now assume a node at the end of the execution is neither selected as a cluster head nor associated to any of the cluster head. Since the node is not a cluster head, its value of $isfinal$ must be "False". Thus, the node satisfies the condition at the beginning of the cluster formation process, which is $isfinal(u) = False$. The node will end up either associating itself to exactly one cluster head in its final cluster head set $\mathcal{F}$ following the node association rule or being a cluster head, which contradicts the assumption. Thus, the lemma holds. □

The cluster heads generated by one-hop clustering algorithm have a good property that the probability that two cluster heads are one-hop away is small. Consider all the possible cases that two cluster heads selected are within one hop. It is not difficult to see that there are only two such cases. The first case is that two neighbors i and j both elect themselves to be cluster heads during the cluster head selection process, where two nodes have the same minimum selection weights and both win the random selection. Consider the simplest network consisting of two nodes i and j. We assume that they have the same selection weights with probability 1, which is very rare in practice. Even in this unrealistic worst scenario, the first case occurs with a probability of only $\frac{1}{4}$. The second case is that neighbors i and j both elect themselves to be cluster heads during the cluster formation process. Both nodes were not selected as cluster heads during the cluster head selection process, nor heard any declaration messages and have kept their final cluster head list $\mathcal{F}$ empty. They consider themselves uncovered and hence become forced cluster heads. Note that if node i announces to be a cluster head earlier than node j, node j will hear the declaration message of node i and join the cluster of node i, instead of becoming a cluster head itself. It follows that node i and node j become cluster heads within one hop only when they make their announcements simultaneously. Since our algorithm is executed distributively at each node and each node follows its own timing slots, it is very unlikely to achieve such exact synchronization in a network, therefore the chances that they both become cluster heads are small. Furthermore, when such accidental synchronization does occur, we can use a random back-off technique to solve the problem. That is, instead of executing the cluster formation process immediately after the head selection process times out, each node may wait for a random period of time to enter the next process.

We now analyze the message complexity of the algorithm.

Lemma 3. *The message complexity of one-hop clustering algorithm is $O(1)$ per node.*

Proof. During the pre-process, each node broadcasts one message of its selection metric to neighbors. During the clustering selection process, it is easy to see that each node broadcasts at most one message. Specifically, if a node becomes a cluster head, it broadcasts one declaration message and waits for time out thereafter. When a node hears a declaration message for the first time, it broadcasts one acknowledgment message. The rest of the nodes keep silent in the process. During the cluster formation process, nodes that have already been elected to be cluster heads keep silent and wait for others to associate to it. Each of the remaining nodes either sends one request message to associate itself to some cluster head or broadcasts one declaration message. Hence, during the execution of the algorithm each node sends at most three messages. □

3.3.3 k-hop Clustering Algorithm

In some monitoring applications in high-density sensor networks, especially those adopting a simple data aggregation scheme, it may be desirable to have multi-hop clusters to provide better scalability. In the following, we first describe the general k-hop clustering algorithm, and then focus on its differences from the one-hop clustering.

The basic idea of k-hop clustering algorithm remains the same as that of one-hop clustering: select a set of cluster heads based on the selection weight metric to form disjoint k-hop clusters. Similarly, k-hop clustering algorithm consists of three processes, is executed distributively by each sensor node, and uses the same rules. The pseudocode of the algorithm for node u is given in Table 3.4.

Compared to one-hop clustering, the main differences of k-hop clustering include: (1) the calculation of the metric and the identification of the minimum energy cost path; (2) the longer clustering duration, especially the duration of the head selection process; and (3) the higher message complexity per node. Based on the definition of the selection weight metric in Equation (3.9), we can calculate the metric when we have the knowledge of the minimum energy cost path from each cluster member to its cluster head. Fortunately, the minimum energy cost path for any two nodes in the network can be calculated by any distributed weighted shortest path algorithm. We use the distributed asynchronous Bellman-Ford algorithm [22] in this thesis. For k-hop clustering, we only need to know the weighted shortest path within k-hops, which largely reduces the problem complexity. It is worth mentioning that this minimum energy cost path problem is also equivalent to the problem of finding the minimum breadth-first search (BFS) tree of depth k rooted at each node. During each execution of k-hop clustering algorithm, the distributed Bellman-Ford algorithm is run only once

Table 3.4: k-hop clustering algorithm executed at node u

```
Input: receive(i, declaration), receive(i, acknowledge);
       /*receive a message from node i*/
Output: broadcast(u, acknowledge);
       /*broadcast a message within one hop */
       broadcast(u, declaration);
       /*broadcast a message within k hop */
```

Pre-Process

calculate the selection weight using distributed asynchronous Bellman-Ford algorithm;
broadcast weight to all its one-hop neighbors;
$isfinal$ (u) $\leftarrow$ False;
create a list containing selection weights for its neighbors;
create a tentative cluster head set $\mathcal{T}$ containing all the neighbors and itself;
create an empty final cluster head set $\mathcal{F}$;

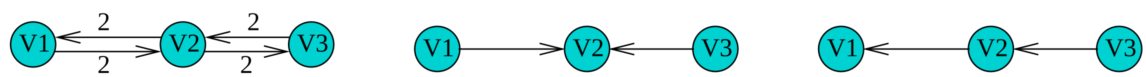
Cluster Head Selection Process

same as one-hop clustering algorithm, see Table 3.3

Cluster Formation Process

if $isfinal$ (u) $=$ False
  if receive(i, declaration)
    add node i into $\mathcal{F}$;
  end
  if $\mathcal{F} \neq \emptyset$
    associate to a cluster head from $\mathcal{F}$ using the minimum energy cost path;
  else
    broadcast(u, declaration);
    $isfinal$ (u) $\leftarrow$ True;
  end
end

at the beginning of the pre-process. The minimum energy cost paths calculated are not only used in the calculation of the selection weight metric, but also define the way



(a) Network configuration (b) One-hop clustering (c) 2-hop clustering

Figure 3.3: One-hop clustering vs. two-hop clustering.

how each node associates itself to its cluster head in the cluster formation process. The longer clustering time, especially the longer head selection process, lies in the fact that for k-hop clustering algorithm it needs k time slots to broadcast declaration messages. Therefore, the cluster head selection process needs at least $2k+1$ time slots to allow each node to run the head selection rule twice. The message complexity of k-hop clustering algorithm also increases as the declaration messages are broadcast up to k hops. Let's look at the message complexity of node i that broadcasts a declaration message. This declaration message can be generated by itself or by any other node in its k-hop neighboring set nbr_i^k. Thus the worst case for node i is to broadcast as many as $|nbr_i^k|$ messages. During the pre-process, each node broadcasts one message to its neighbors. During both the head selection process and the cluster formation process, it is possible for a node to broadcast declaration messages. Therefore, the worst case of the message complexity per node in k-hop clustering is $2|nbr_i^k|$.

It should be mentioned that although k-hop clustering has higher scalability, it may have lower energy efficiency compared to one-hop clustering. Let's look at a simple example in Fig. 3.3, where each link has a weight of 2 and each node has one packet to send. With the same throughput, one-hop clustering uses 4 units of energy while 2-hop clustering uses 6 units. This is because that in 2-hop clustering node v_2 not only sends its own packet but also relays a packet for node v_3. In general, in k-hop clustering, the larger the k is, the more scalability and the fewer cluster heads the network has. However, energy consumption increases due to relaying traffic, and thus it may shorten network lifetime. As will be seen in the next section, our simulation results reveal the performance deterioration in terms of network lifetime in 2-hop clustering compared to one-hop clustering.

3.4 Some Implementation Issues and Discussions

We have designed one-hop and k-hop clustering algorithms for WSNs under lossy links. In this subsection, we briefly discuss some implementing issues and an extension of our algorithms to deal with node failures.

The first issue is network initialization, especially how to explore neighboring nodes and how to know PRRs over links in the neighborhood. At the beginning, each node does not know its neighbors. Therefore they broadcast a "hello" message of a fixed size periodically, say, in a period of τ. Every node knows its neighbors when it receives the message. These hello messages are also link probes for each node to estimate PRRs over links in the one-hop neighborhood. Every node remembers the probes that it receives during a time window w. Then the average PRR at time t can be calculated by

$$\frac{\text{probes received in } [t-w,t]}{w/\tau}$$

where w/τ is the probes expected to be received in $[t-w,t]$. This method is called passive probing and was discussed in more detail in [19]. An alternative approach is that each node broadcasts its "hello" message once. Then every node knows its neighbors if it receives the message, uses the received signal strength as an indicator of the link quality [15], and estimates PRR accordingly. By initializing the network periodically, our clustering algorithms can be adaptive to the change of topologies and link qualities in the environment.

The second issue is re-clustering. Similar to most existing clustering algorithms, our clustering algorithms are triggered periodically to select a new set of cluster heads such that the load of cluster heads is balanced in the network. The length of the steady state phase after each execution of the clustering algorithm, in which sensors send packets to their cluster heads, is called *re-clustering period*. Although a short re-clustering period leads to more balanced load, it consumes more energy on the execution of the clustering algorithm, and may also make the network unstable. Thus re-clustering should be done periodically but infrequently.

Finally, our clustering algorithms can be extended to deal with node failures in the network, which is a common problem in applications involving a large scale deployment of sensors such as habitat monitoring [23]. By adopting the above technique to estimate link qualities periodically, we can compare a few most recent records of link qualities over the same link. If the deviation exceeds some threshold, we consider there is a node failure at the nodes connected by this link.

Chapter 4

Performance Evaluation

In this chapter, we evaluate the performance of the one-hop and 2-hop clustering algorithms through simulations. We have conducted two sets of simulations in wireless sensor networks (WSNs) with random topologies.

4.1 Simulation Settings

In both sets of simulations, we assume that each node is provisioned with an initial energy level of 1 Joule. Each node generates data at a rate of 2 packets/minute and each packet has a fixed size of 50 bytes.

We use the energy model introduced in [10] to model the energy consumption in transmissions and receptions. To deliver an l-bit packet in distance d, the transmission power is

$$E_{Tx} = \alpha + \beta \cdot d^2$$

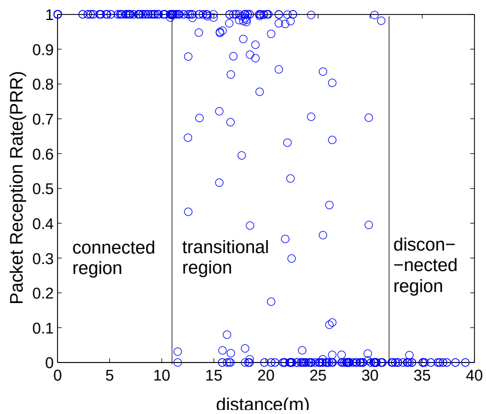
and the reception power is

$$E_{Rx} = \alpha$$

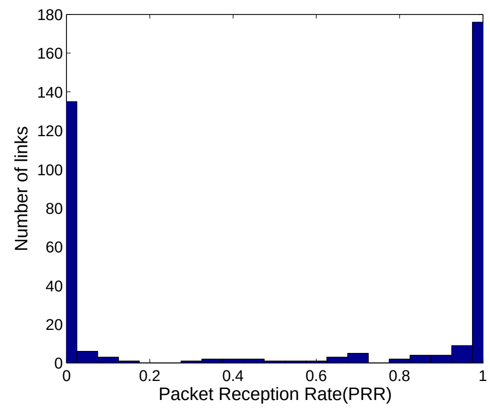
where α and β are two constants depending on the length of the packet and the electronics energy. In our settings with packet length of 50 bytes, $\alpha = 20000nJ$ and $\beta = 4000pJ/m.^2$. Since we use fixed transmission and reception powers and have transmission range $R = 40$, we set $d = R$ and obtain values for $E_{TX} = 26400nJ$ and $E_{Rx} = 20000nJ$ per packet.

We adopt a realistic link model described in [17] to calculate the packet reception ratio (PRR) p_{ij} and model lossy and asymmetric wireless links using these probabilities. We use non-return-to-zero (NRZ) encoding and non-coherent frequency-shift keying (FSK) modulation, which are commonly used in WSN test-beds. These are the parameters of the link model together with the packet length and transmission power. Fig. 4.1 shows a specific instance of statistics for all p_{ij} in a network of 20 sensors scattered in a 40m $\times$40m region with the same configuration of sensors as that in our simulations. It can be seen from Fig. 4.1(a), if two nodes are within a distance of $10m$, the links between them are reliable with $PRR = 1$. On the other hand, although some node pairs are within transmission range of $40m$, the links between them have $PRR = 0$. We do not consider these nodes as neighbors, and do not use these values for any p_{ij} in the algorithms. From Fig. 4.1(b), we can see that a majority of links in this network instance are reliable.

A simple automatic repeat request (ARQ) mechanism is used at the MAC layer to ensure reliability. We assume the maximum number of retransmissions before dropping a packet is 3.



(a) Packet reception ratio vs. distance



(b) Link distribution

Figure 4.1: A realistic link model in a wireless sensor network.

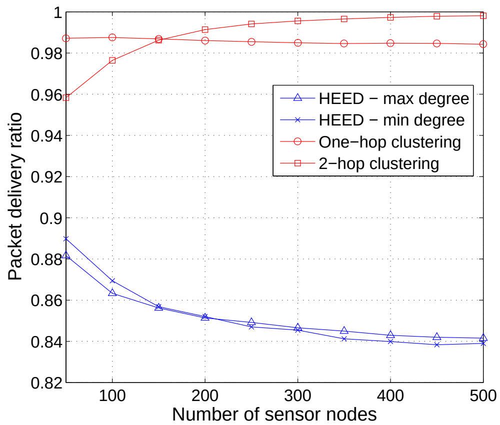
4.2 Performance of Large-scale Random Networks

We evaluate the clustering algorithms in large-scale random networks in a 150m $\times$ 150m field. All the simulation results are averaged over 100 runs, with each run using a different randomly generated topology with different link qualities.

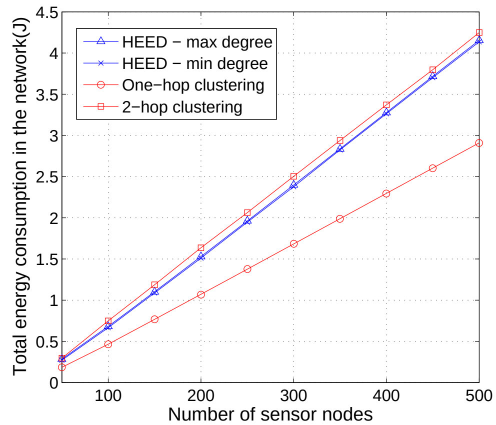
We choose a typical clustering algorithm called HEED for performance comparison. As shown in [11, 16], HEED outperformed previous clustering algorithms when it was proposed and has now become a well-accepted representative clustering algorithm. Depends on different clustering goals such as load distribution or dense clusters, HEED can be divided into HEED-max degree and HEED-min degree algorithms. We compare both of them with our algorithms.

Fig. 4.2 shows the network performance of different algorithms in a one-hour time period without re-clustering when the number of nodes varies from 50 to 500. Fig. 4.2(a) plots the data delivery ratio of each algorithm. The delivery ratio is calculated by the number of packets successfully delivered divided by the number of packets generated at the nodes. For all the algorithms compared, we use retransmissions with maximum retries of 3 before dropping a packet. We can see that both our one-hop and 2-hop clustering algorithms successfully deliver over 95% of total packets to corresponding cluster heads, while the two HEED algorithms yield lower delivery ratios due to lossy links. Furthermore, as the number of nodes increases, the performance of both HEED algorithms deteriorates. This is because that when the network becomes denser, without considering link quality, there are more nodes in the cluster using long lossy links in HEED algorithms, while the performance of one-hop clustering algorithm is quite stable in terms of delivery ratio and 2-hop clustering algorithm improves the performance by replacing long lossy links with two short reliable links. Fig. 4.2(b) indicates that one-hop clustering algorithm consumes the least amount of total energy in the network, and saves as much as 30% of the energy compared to HEED-max for all network sizes. Note that we impose a bound for the number of retransmissions, which in turn constrains that energy consumption for each packet transmission over an extremely lossy link is at most 3 times more than that over a reliable link. Fig. 4.2(c) shows the number of cluster heads generated by each algorithm. As expected, 2-hop clustering algorithm generates the least number of cluster heads. Other three algorithms have a similar number of cluster heads, which indicates that both our algorithms maintain a small number of cluster heads.

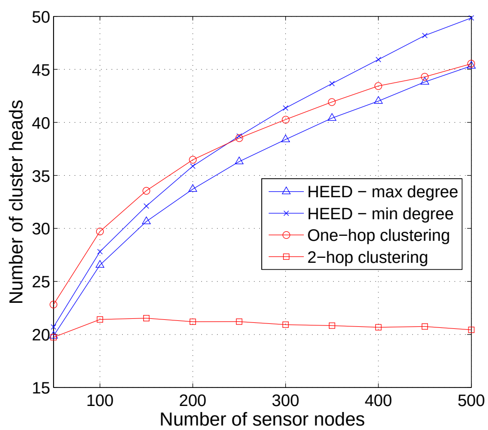
Fig. 4.3 illustrates the relationship between the network lifetime and the re-clustering period with 200 sensor nodes dispersed in the field. We can see that a short re-clustering period yields longer network lifetime due to frequent load balancing in the network. When the re-clustering period increases, the one-hop clustering algorithm is able to maintain good performance in terms of network life time compared to other algorithms, which indicates more uniform energy consumption in the one-hop clustering algorithm compared to other algorithms.



(a) Packet delivery ratio



(b) Total energy consumption in the network



(c) Number of cluster heads

Figure 4.2: Network performance in a one-hour period.

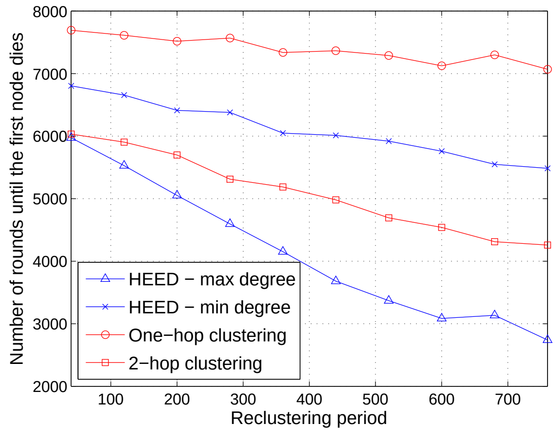


Figure 4.3: Network lifetime vs. re-clustering period.

Fig. 4.4 shows the network lifetime of each algorithm when the re-clustering period is set to be 1 hour and the number of nodes varies from 50 to 500. From the discussions in the previous section, we know that the message used for information exchange is of a small size and therefore consumes less energy in transmissions. We also proved that our clustering algorithms have a low message complexity that each node sends at most 3 messages for one-hop clustering during cluster creation. Thus, it is reasonable to assume that the energy consumption on cluster creation is negligible compared to the energy consumption on data packet transmissions after clusters are created. In this experiment, we did not calculate the energy consumption on cluster creation in the four algorithms compared. We can see that the one-hop clustering algorithm performs best and increases network lifetime by $17\% - 42\%$ compared to the HEED-min degree algorithm and by $28\% - 42\%$ compared to the HEED-max degree algorithm. These results clearly demonstrate the benefit of considering lossy links in the design and implementation of clustering algorithms.

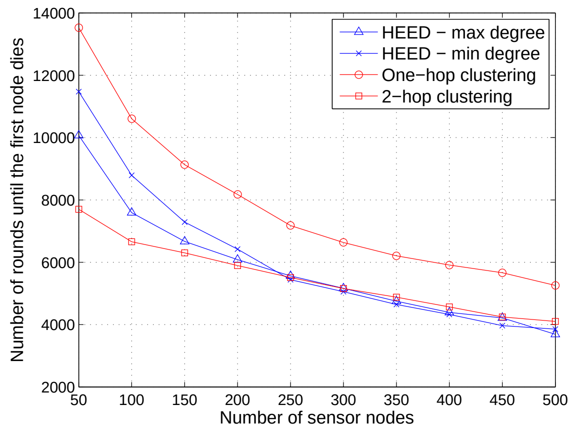


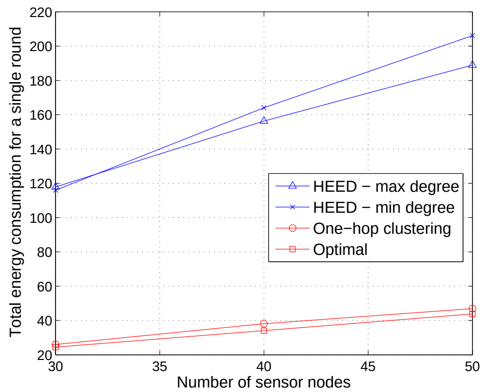
Figure 4.4: Network lifetime vs. number of nodes.

4.3 Performance Comparison with Optimal Results

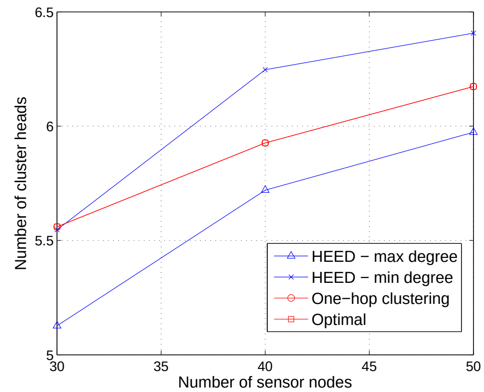
In this section, we further compare our one-hop clustering algorithm with the optimal solutions obtained by solving the integer program (IP) in Section 3.2 using ILOG OPL Studio software [24]. Due to the NP-hardness of the clustering problem, the brutal force search method of the optimal solution in a large network becomes infeasible. Therefore, we have managed to obtain optimal solutions for a few small networks to gain some insights on the optimal performance.

We mainly focus on the total energy consumption in the network for a single round, which is the objective function in the problem formulation.

We consider a bunch of sensor nodes scattered over a 50m $\times$ 50m square area. All the simulation results are averaged over 100 runs, with each run using a different randomly generated topology. In each run, we let N_c in the IP formulation take the same number of cluster heads generated in our clustering algorithm. Fig. 4.5 shows the results of different algorithms compared with optimal solutions when the number of sensor nodes varies from 30 to 50. We can see that the one-hop clustering



(a) Compared with the optimal cost of a single round



(b) Number of cluster heads in the network

Figure 4.5: Performance of one-hop clustering algorithm compared with optimal solutions.

algorithm is closer to the optimal solutions in small random networks compared to the two HEED algorithms.

Chapter 5

Conclusions

Wireless sensor networks (WSNs) have emerged as a new information gathering paradigm in a wide-range of applications. Clustering introduces a hierarchy to the network and is generally considered as a promising approach for data gathering in large-scale WSNs. Specifically, clustering is to group sensors into disjoint clusters such that sensors as cluster members form the lower layer of the network send data to their cluster heads, and cluster heads form the higher layer of the network and forward data to the base station.

While reliability in data delivery largely impacts the quality of surveillance in WSNs, empirical studies have revealed that a large percentage of wireless links are lossy and unreliable for data delivery. These lossy links may cause data delivery failure and energy wastage due to unnecessary packet retransmissions. Although a multitude of clustering algorithms have been proposed for WSNs in the literature, this lossy link problem was not taken into consideration in existing clustering algorithms.

5.1 Summary of Contributions

In this thesis, using retransmissions to guarantee reliability, we have tackled the clustering problem under lossy links with the objective to save energy in retransmissions as well as to maintain a small number of cluster heads. In particular, this thesis provides the following contributions.

We formulated the one-hop clustering problem under lossy links into an integer program and proved its NP-hardness. The integer program has the objective to minimize the total energy spent on transmissions and receptions in the entire network for a single round, in which each sensor node as a cluster member sends one data packet to its cluster head. The objective of saving energy in retransmissions in turn can prolong the network lifetime, which is defined earlier as the time until the first sensor node depletes its energy. By solving the integer program we are able to obtain the optimal solution for the one-hop clustering problem. However, due to the NP-hardness of the clustering problem the brutal force search method of the optimal solution in a large network becomes infeasible. Therefore we proceeded to propose a distributed clustering algorithm that would work well for large-scale networks.

We have proposed a metric-based distributed clustering algorithm to solve the one-hop clustering problem. We designed a new metric called *selection weight* for each sensor node that indicated both link qualities around the node and its capability of being a cluster head. The intuition comes from two observations in the following.

- Cluster heads consume more energy than other nodes and thus may deplete their energy earlier.
- Energy consumption on a reliable packet delivery is proportional to the number of transmissions for the packet.

Therefore, we introduced residual energy ratio E_{ratio} to indicate the capability of being a cluster head and used average number of transmissions $T_{avg}(i)$ for the cluster of node i when it was the cluster head, in order to indicate link qualities in the cluster and in turn to estimate the energy consumption in the cluster.

Our distributed one-hop clustering algorithm was proven to be of low complexity in both time and messages per node. The algorithm first selects a set of cluster heads based on selection weight metric in the cluster head selection process and then associates each sensor node to exactly one cluster head to form disjoint clusters in the cluster formation process. The low time complexity comes from the fact that we used the pre-defined duration for the cluster head selection process. For example, for one-hop clustering we defined three time slots long for the process. Under such a time-out setting, we also proved that we could always have a feasible solution. That is, after the execution of the one-hop clustering algorithm, a node is either a cluster head or a cluster member that belongs to exactly one cluster head. We analyzed the message complexity of one-hop clustering algorithm in detail in Chapter 3.3 and showed that during the execution of the algorithm each node sends at most three messages.

We further extended the algorithm to multi-hop clustering to achieve better scalability and discussed some implementation issues and an extension to deal with node failures, which is a common problem in applications involving a large-scale sensor networks.

Extensive simulations have been conducted under a realistic link model and the results demonstrate that the proposed clustering algorithm can reduce the total energy consumption in the network and prolong network lifetime significantly compared to a typical distributed clustering algorithm, HEED.

5.2 Future Work

For the clustering problem under lossy links, there is still much work to be done. In this thesis, reliability of packet delivery is required therefore retransmissions are needed. If we differentiate different links and use "good" links to construct clusters, we can save energy from unnecessary retransmissions. Making use of this intuition, the clustering algorithms with the objective of energy saving developed in this thesis try to minimize the total energy consumption in the entire network.

However, our algorithms may not work well for the cases that only partial reliability is required and retransmissions are not used. For example, the clustering goal is to have all the links between cluster members and their associated cluster heads had a reliable probability higher than 0.5 without using retransmissions, that is, with a packet reception ratio (PRR) higher than 0.5 over these links. This clustering problem with objective on having reliable-threshold links is apparently different from our problem described in the thesis, and can be formally stated in the following.

Reliable-threshold links clustering problem:

Let $G(V, A)$ be a directed graph representing a wireless sensor network, where V is the set of sensor nodes and A is the set of directed links. Each link (i, j) has associated a PRR of p_{ij}, and N_c be a positive integer that constraints the number of cluster heads in the network. For any cluster C_i with cluster head node i, define $p(j, i)$ to be the minimum PRR in the cluster C_i from its cluster member node j to the cluster head node i. The problem is to find a set of cluster heads $S \subseteq V$, with $|S| \leq N_c$, so as to $\max\{\min_C\{p(j, i)\}\}$.

Recall one earlier observation that cluster heads consume more energy than other nodes and thus may deplete their energy earlier and the definition of network lifetime as the time until the first sensor node depletes its energy. If rotation of cluster heads is not adopted, we conclude that network lifetime is determined by the time until the first cluster head depletes its energy. Let P_l denotes the energy spent on delivering one fixed data packet over a reliable link including both transmission and reception consumptions, and n_{ji} denotes the expected number of transmissions needed for a reliable data delivery over link (j, i). Now consider a cluster with node j as its cluster members and node i as its cluster head, who has maximum energy of E_{max} and a degree of d. We have the following lifetime of the cluster head L_i,

$$L_i = \frac{E_{max}}{\sum_d n_{ji} P_l} \tag{5.1}$$

We then obtain the network lifetime L using $L = \min_C L_i$. We can also see from the above that network lifetime mainly depends on two factors: number of transmissions over links and degrees of each cluster head.

Saving energy in retransmissions for each node as what we did in this thesis prolongs the network lifetime, which in nature is to reduce the number of transmissions incurred for each link from a cluster member to a cluster head. There is another important factor that we have not yet made fully use of: degrees. We formally state this maximal life clustering problem in the following.

Maximal lifetime clustering problem:

Let $G(V, A)$ be a directed graph representing a wireless sensor network, where V is the set of sensor nodes and A is the set of directed links. Each link (i, j) has associated a positive integer, denoting an expected number of transmissions needed for a reliable data delivery over the link, and N_c be a positive integer that constraints the number of cluster heads in the network. Each cluster head has lifetime L_i defined in Equation

(5.1). The problem is to find a set of cluster heads $S \subseteq V$, with $|S| \leq N_c$, so as to $\max\{\min_C\{L_i\}\}$.

Finally, it is also important to have experimental evaluation of these clustering algorithms in the future using real sensor network test-beds. The lossy link problem was pointed out and raised attention by previous empirical studies in the sensor field, therefore the best way to validate clustering algorithms under lossy links is to evaluate the performance in the real world.

Bibliography

[1] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications (WSNA '02)*, pages 88–97, Sep 2002.

[2] Ning Xu, Sumit Rangwala, Krishna Kant Chintalapudi, Deepak Ganesan, Alan Broad, Ramesh Govindan, and Deborah Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 2nd international conference on Embedded networked sensor systems(SenSys '04)*, pages 13–24, Nov 2004.

[3] K. Lorincz, D.J. Malan, T.R.F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton. Sensor networks for emergency response: Challenges and opportunities. *IEEE Pervasive Computing*, 3(4):16–23, Oct.-Dec. 2004.

[4] Suresh Singh, Mike Woo, and C. S. Raghavendra. Power-aware routing in mobile ad hoc networks. In *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking (MobiCom '98)*, pages 181–190, Oct 1998.

[5] S. Lindsey and C.S. Raghavendra. Pegasis: Power-efficient gathering in sensor information systems. In *IEEE Aerospace Conference Proceedings*, volume 3, pages 1125–1130, 2002.

[6] R.C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: Modeling a three-tier architecture for sparse sensor networks. In *IEEE First International Workshop on Sensor Network Protocols and Applications*, pages 30–41, May 2003.

[7] Ming Ma and Yuanyuan Yang. Sencar: An energy-efficient data gathering mechanism for large-scale multihop sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 18(10):1476–1488, Oct 2007.

[8] A.D. Amis, R. Prakash, T.H.P. Vuong, and D.T. Huynh. Max-min d-cluster formation in wireless ad hoc networks. In *Proceedings of IEEE INFOCOM*, volume 1, pages 32–41, 2000.

[9] S. Banerjee and S. Khuller. A clustering scheme for hierarchical control in multi-hop wireless networks. In *Proceedings of IEEE INFOCOM*, volume 2, pages 1028–1037 vol.2, 2001.

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

[11] Ossama Younis and Sonia Member-Fahmy. Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, 3(4):366–379, Oct - Dec 2004.

[12] Pu Wang, Cheng Li, and Jun Zheng. Distributed minimum-cost clustering protocol for underwater sensor networks (uwsns). In *IEEE International Conference on Communications (ICC '07)*, pages 3510–3515, June 2007.

[13] Alec Woo David Culler Deborah Estrin Deepak Ganesan, Bhaskar Krishnamachari and Stephen Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. 2002. UCLA Computer Science Technical Report UCLA/CSD-TR 02-0013.

[14] Jerry Zhao and Ramesh Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys '03)*, pages 1–13, 2003.

[15] Alec Woo, Terence Tong, and David Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys '03)*, pages 14–27, 2003.

[16] Ameer Ahmed Abbasi and Mohamed Younis. A survey on clustering algorithms for wireless sensor networks. *Computer Communnications*, 30(14-15):2826–2841, 2007.

[17] M. Zuniga and B. Krishnamachari. Analyzing the transitional region in low power wireless links. In *First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks*, pages 517–526, Oct 2004.

[18] Alexandra Meliou, David Chu, Joseph Hellerstein, Carlos Guestrin, and Wei Hong. Data gathering tours in sensor networks. In *Proceedings of the 5th international conference on Information processing in sensor networks(IPSN '06)*, pages 43–50, 2006.

[19] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th annual international conference on Mobile computing and networking (MobiCom '03)*, pages 134–146, 2003.

[20] Karim Seada, Marco Zuniga, Ahmed Helmy, and Bhaskar Krishnamachari. Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems (SenSys '04)*, pages 108–121, 2004.

[21] Suman Banerjee and Archan Misra. Minimum energy paths for reliable communication in multi-hop wireless networks. In *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc '02)*, pages 146–156, 2002.

[22] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, 1987.

[23] Robert Szewczyk, Joseph Polastre, Alan Mainwaring, and David Culler. Lessons from a sensor network expedition. In *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, pages 307–322, 2004.

[24] ILOG OPL Studio software. *www.ilog.com/products/oplstudio/*.