WIRELESS SENSOR NETWORK:
ENERGY EFFICIENCY, SECURITY, AND, FAULT TOLERANCE

PU WANG

Wireless Sensor Network: Energy Efficiency, Security, and, Fault Tolerance

by

© Pu Wang
Master of Engineering

A thesis submitted to the
School of Graduate Studies
in partial fulfillment of the
requirements for the degree of
Master of Engineering.

Department of Electrical and Computer Engineering
Memorial University of Newfoundland

May 4, 2008

ST. JOHN'S NEWFOUNDLAND

Abstract

Wireless sensor network (WSN) is an emerging networking paradigm that promises a wide range of potential applications in both civilian and military areas. WSN runs different kinds of applications in a variety of physical environments, which offers many challenges. The main design constraints include energy efficiency, fault tolerence, and security. In this thesis, we investigate the research problems involved in three types of sensor networks including the UnderWater Sensor Network (UWSN), the Wireless Terrestrial Sensor Network (WTSN), and the Wireless Multimedia Sensor Network (WMSN).

We first formulate the node clustering problem into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and prolong the network lifetime in UWSN. A novel distributed clustering protocol called minimum-cost clustering protocol (MCCP) is proposed, which selects a set of non-overlapping clusters from all potential clusters based on the cost metric assigned to each potential cluster and attempts to minimize the overall cost of the selected clusters. To provide a robust clustered architecture against cluster-head failures in UWSNs, a dependable clustering protocol is proposed in which two mechanisms are employed: fault prevention clustering and cluster head replication. Fault prevention clustering attempts to select those healthy nodes as cluster heads to prevent cluster head failures, and cluster head replication attempts to select a primary cluster head

and a backup cluster head for each cluster member so that the constructed cluster hierarchy can tolerate cluster-head failures.

The successful working of any fault recovery schemes heavily depends on a proper and efficient fault detection mechanism. Therefore, we propose a cooperative fault detection mechanism, which can accurately and quickly detect the failure of a cluster head through the independent fault status detection from each cluster member and a distributed agreement process for final decision. It runs concurrently with normal network operation at each cluster member and makes use of the data periodically sent by a cluster head as the heartbeats for fault detection. An agreement can be efficiently achieved within two TDM frames in each detection process.

To address the energy efficiency problem in WTSN, Slepian-Wolf coding is employed to remove the redundancy caused by the data correlation. We first consider the clustered Slepian-Wolf coding problem, which aims at selecting a set of disjoint potential clusters to cover the whole network such that the global compression gain of Slepian-Wolf coding is maximized, and then propose a distributed optimal-compression clustering protocol to solve the problem. Based on the resulting cluster hierarchy constructed, we study the optimal intra-cluster rate allocation problem to minimize the intra-cluster communication cost and further combine with the explicit entropy coding to minimize the inter-cluster communication cost.

Furthermore, based on inherent characteristic of Slepian-Wolf coding, we propose a combined data aggregation and encryption scheme, called spatially selective encryption, for efficient and secure data transmission in WTSNs. Using this mechanism, as long as the data of the cluster head (a.k.a. the visual key) is properly protected, the data from all cluster members are secure. This novel approach can significantly reduce the energy consumption for data encryption. An energy-efficient key establishment protocol is also proposed to securely and efficiently establish the key used

for encrypting the visual key.

Finally, we propose a clustered on-demand multi-channel MAC protocol (COM-MAC) to support energy-efficient, high-throughput, and reliable data transmission in WMSNs. A scheduled multi-channel medium access is used within each cluster so that cluster members can operate in a contention-free manner in both time and frequency domains to avoid collision, idle listening and overhearing. A traffic-adaptive and QoS-aware scheduling algorithm is executed to maximize the network throughput. A spectrum-aware ARQ is further incorporated to better exploit the unused spectrum for a balance between reliability and retransmission.

Acknowledgements

I would like to express my gratitude to my advisor Prof. Cheng Li for his continuous support, encouragement and guidance throughout my graduate study. His breadth of knowledge and his enthusiasm for research inspires me. My appreciation also goes to Dr. Jun Zheng from University of Ottawa, who shares invaluable expertise with me and help me improve my work.

While at MUN, I interacted with lots of wonderful and talented people. They taught me so much, and their precious friendship and generous support made my Master experience more fun and easier: Tianqi Wang, Shenqiu Zhang, Ling Wu, Liang Zhang, Chen Shi, Reza Shahidi, and all members of CERL group. I would specially like to thank Reza Shahidi for all his help with my research and personal life. I would like to thank my family for their unconditional love and support.

Contents

List of Tables

List of Figures

List of Symbols

The following symbols and definitions are used in this thesis.

- N_v denotes the neighbor set of candidate v
- $P(N_v)$ denotes the power set of N_v and constitutes all possible combinations of nodes in N_v
- A_v denodes the representative cluster of candidate v
- X_v denodes the set containing the cluster members of A_v
- avg(v) denodes the average cost of the representative cluster A_v
- *head* denodes a flag indicating a cluster head
- *cand* denodes a flag indicating a candidate
- *memb* denodes a flag indicating a cluster member
- G denodes a set containing the average costs sent by other cluster heads within 2-hop range of a candidate
- $INVITE(v, X_v)$ denodes a message inviting the nodes in set X_v to be the cluster members of candidate v

- *JOIN*$(v,\ u)$ denodes a message acknowledging that node v received the *INVITE* message sent by candidate u and joins the cluster as a cluster member of candidate u
- $E_{sym}(a, b)$ denodes performing symmetric key encryption using key a on data b
- $E_{pub}(a, b)$ denodes performing public key encryption using key a on data b
- $D_{pub}(a, b)$ denodes performing public key decryption using key a on data b

Chapter 1

Introduction

1.1 Wireless Sensor Networks

A wireless sensor network (WSN) typically consists of a large number of resource-constrained sensor nodes which have limited computation and communication capacities and communicate with each other through wireless communication channels, as shown in the Figure 1.1. WSN promises a wide range of potential applications in both civilian and military areas such as environmental monitoring, health applications, and battlefield surveillance [3].

Each node in a sensor network is typically equipped with a low-power radio transceiver, a small microcontroller, and an energy source, usually a battery. The constraints on sensor nodes such as low cost and small size lead to many challenging problems in sensor networking, such as constraints on energy, memory, computational speed and bandwidth. Some technical parameters of several typical sensor devices are shown in Table 1.1.

WSN can be used for different kinds of applications in a variety of physical environments, which offers many challenges. *Energy efficiency* is the primary concern in

Figure 1.1: Example of WSN [2].

wireless sensor networks. Sensor nodes are normally battery powered and may not be charged when they run out of energy. Therefore, the lifetime of a sensor network heavily depends on how the energy conservation mechanism is employed. Besides power loss, the sensor nodes may also fail due to aging, imperfections in manufacturing, and impairment caused by harsh environment. Thus, *fault tolerance* feature should be incorporated to fight against possible future failures. Since a WSN may be deployed in hostile areas where wireless communication might be monitored by an adversary. *Security* is also a critical issue, especially for those security-sensitive applications. Therefore, efficient and effective security mechanisms are required to protect the sensitive contents of the communications in the resource-constrained WSNs.

1.2 Sensor Network Challenges

In this thesis, we consider three types of sensor networks: underwater sensor network, wireless terrestrial sensor network, and wireless multimedia sensor network. Each type of sensor network promises a wide range of applications which presents many challenges for the design of the network.

Table 1.1: Sensor Platforms [1]

Platform	MICAZ	T-Mote Sky	XYZ	ECO	S-Mote
CPU	ATMEL	MSP430	ML-67Q500	nRF24el	CC2430
Clock	16 Mhz	8 Mhz	60 Mhz	20Mhz	32Mhz
Active Power	8 mA	2 mA	40 mA	3 mA	7 mA
Sleep Power	$20\mu A$	$27\mu A$	$20\mu A$	$2\mu A$	$0.9\mu A$
MCU + TX power	23.3 mA	21.8 mA	69.8 mA	10.5 mA	27 mA
MCU + RX power	21.0 mA	19.5 mA	57.5 mA	19.0 mA	24.7 mA
Price (MCU + RF)	$ 9.2	$ 9.5	$ 8.43	$ 3.29	$ 3.90

1.2.1 UnderWater Sensor Networks

1.2.1.1 Energy Efficiency

UnderWater Sensor Network (UWSN) is an emerging networking paradigm that promises a wide range of potential applications in both civilian and military areas [4, 5, 6]. A UWSN typically consists of several underwater sinks (called uw-sinks) located at the centers of different monitored areas, a number of ocean bottom sensor nodes surrounding each uw-sink, and a surface station providing a link to an on-shore control center, which collaborate to accomplish a common task, such as underwater environmental monitoring, mine reconnaissance, and military surveillance, as shown in Figure 1.2. Compared with traditional underwater monitoring or surveillance technologies, a UWSN has a number of advantages, such as unmanned underwater exploration, localized and precise information acquisition, large-scale underwater monitoring, reduced implementation cost, and more frequent operations [6], which have received much attention recently. UWSN has some unique characteristics, such as

highly limited bandwidth, long propagation delay, harsh geographical environment, and relatively small network scale [4]. More importantly, in the underwater, it is inconvenient to replace the battery of the sensor node located on the ocean bottom. Therefore, the lifetime of a UWSN is largely restricted by the energy constraint in sensor nodes. This means all aspects of sensor nodes, from hardware to protocols, must be designed to be extremely energy-efficient. Although some sensor nodes specially designed for UWSN can be recharged by current energy converter, an underwater sensor node must make efficient use of its limited energy capacity to ensure long-term and continuous underwater environmental monitoring.

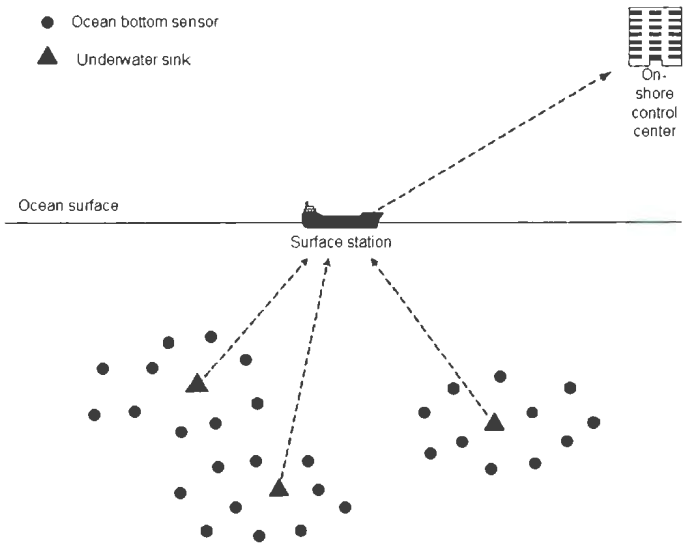


Figure 1.2: UWSN architecture.

To improve energy efficiency, node clustering has been widely considered in WSNs [7]. With clustering, each sensor node only needs to send data to its associated cluster head at a short distance while only the cluster heads need to relay the locally aggregated data to a data sink at a long distance, which can significantly reduce the

energy consumption of each sensor node and thus prolong the lifetime of the whole network. Moreover, node clustering leads to a hierarchical network architecture, which enables scalable medium access control, robust routing, and coordinate-free localization [8]. Due to the unique characteristics of UWSNs, however, the clustering algorithms proposed for terrestrial WSNs and mobile ad hoc networks (MANETs) cannot be applied to UWSNs directly without modification.

Node clustering has been widely studied for terrestrial WSNs and many clustering algorithms have been proposed in the literature [7, 9, 10, 11]. The most well-known examples are LEACH [7] and HEED [9]. In LEACH, clusters are generated based on the optimal number of cluster heads, which is calculated using the prior knowledge of uniform node distribution. Due to the effect of the ocean current during deployment and the irregularity of seafloor, however, the distribution of ocean bottom sensor nodes is usually non-uniform, which makes LEACH unsuitable for use in UWSNs. On the other hand, the cluster diameter in LEACH is assumed to be unlimited. This may result in the generated cluster members being located far away from the cluster head and each other. In this case, TDMA would become inefficient for UWSNs because of the high underwater propagation delay while otherwise TDMA is considered an efficient MAC protocol for cluster-based UWSNs [4]. In HEED, clusters are generated without any assumption about node distribution. The cluster diameter is limited and fixed, and a cluster head rotation scheme is employed for load balancing. Although HEED can achieve a good load balance in a small area, the traffic loads in different areas are still unbalanced, thus leading to unbalanced energy consumption in the whole network. It should be pointed out that both LEACH and HEED are cluster-head-centric algorithms, which first select cluster heads based on a selection policy, such as the node with the largest residual energy, and then adds each non-cluster-head node into the cluster of its nearest cluster head or the cluster head with some

predefined property, such as the largest node degree. In [8], a clustering protocol is proposed for an autonomous network of underwater vehicles, which employs a clustering algorithm called lowest-identifier clustering algorithm (LIDCA) proposed in [10] for terrestrial ad hoc mobile network. Although LIDCA is simple to implement, it is not suitable for UWSNs because it constructs the clusters based on node identifiers while energy efficiency is not a factor considered.

In this thesis, we formulate the node clustering problem into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and prolong the lifetime of the network. To solve the formulated problem, a novel distributed clustering protocol called minimum-cost clustering protocol (MCCP) is proposed, which selects a set of non-overlapping clusters from all potential clusters based on the cost metric assigned to each potential cluster and attempts to minimize the overall cost of the selected clusters. MCCP can adapt geographical cluster head distribution to the traffic pattern in the network and thus avoid the formation of hot spots around the uw-sink. It can also balance the traffic load between cluster heads and cluster members through periodical re-clustering the sensor nodes in the network.

1.2.1.2 Fault Tolerance

In addition to energy efficiency, fault tolerance is a great concern in underwater sensor networks [4, 5, 6]. To enable long-term and continuous underwater monitoring, underwater sensor nodes are usually complex and expensive systems and are normally equipped with various electronic and mechanical devices. However, the harsh underwater environment makes sensor nodes particularly vulnerable to failures or physical damages, which would largely affect the robustness of such networks. In particular, a single cluster-head failure can result in the loss of connectivity of all affected cluster members and thus disrupt the operation of the whole cluster. Thus, fault-tolerent

mechanisms are highly desired so as to ensure that the loss of a small faction of sensor nodes does not significantly affect the performance of the whole system.

To deal with cluster-head failures, most existing work employs reactive fault-recovery mechanisms to recover the connectivity of the cluster members in a failed cluster [7, 9, 12, 13, 14]. Two representative reactive fault-recovery mechanisms are re-clustering and backup cluster-head provisioning. For example, LEACH and HEED employ re-clustering to deal with cluster-head failures, i.e., periodically select cluster heads from the sensors in the network [7, 9]. However, they all suffer from the overhead caused by re-clustering. Frequent re-clustering can timely recover the failed clusters but would incur significant control overheads. Sporadic re-clustering can save energy, but would reduce the timeliness of the sensed data transimssion and the coverage of the whole network because the sensor nodes in the failed clusters can only be recovered until the next re-clustering is performed. In [13], a fault-tolerant clustering mechanism is proposed to dynamically perform re-clustering locally once most cluster heads reach a consensus about the presence of a failure. However, this mechanism is specifically designed for heterogeneous sensor networks, where cluster heads are powerful gateways which are responsible for fault detection and recovery. In [12], a robust energy-efficient distributed clustering (REED) is proposed for terrestrial WSNs. REED is a HEED-based protocol, which aims to construct a robust clustered architecture by selecting k independent sets of cluster heads. However, it is usually effective only for terrestrial WSNs, not for UWSNs, because its effectiveness is based on certain conditions on node density, which are usually satisfied in high-density terrestrial WSNs.

In this thesis, we propose a dependable clustering protocol to provide a robust clustered architecture against cluster-head failures in UWSNs. To achieve this objective, the proposed clustering protocol employs two mechanisms: fault prevention

clustering and cluster head replication. First, fault prevention clustering attempts to select those healthy nodes as cluster heads to prevent cluster head failures. Then, during clustering, cluster head replication attempts to select a primary cluster head and a backup cluster head for each cluster member so that the constructed cluster hierarchy can tolerate cluster-head failures.

1.2.1.3 Fault Detection

Fault detection is a prerequisite for recovering the disrupted cluster in the event of a cluster-head failure. In a clustered network, each cluster member can independently detect the failure of its cluster head by checking the heartbeats periodically sent by the cluster head [15]. Due to the channel uncertainty or signal interference in the harsh underwater environment, however, a sensor node may mistakenly detect a cluster-head failure that does not actually exist, which would unnecessarily trigger a fault recovery process and thus waste a considerable amount of energy in sensor nodes. To avoid such energy waste, it is important to accurately detect the failure of a cluster head. However, most previous work was focused on the recovery of a faulty cluster [12, 16, 14]. The accuracy problem in detecting a cluster-head failure has not been well addressed.

In [15] and [16], the heartbeat-based fault detection mechanisms were proposed, which detect the failure of a cluster head by checking the heat-beats periodically sent by a cluster head. This mechanism is simple to implement, but takes much time to achieve higher detection accuracy. Meanwhile, the gossiping-based fault detection [17], usually used in traditional ad hoc networks, is not suitable for underwater applications because a gossiping-based detection mechanism is usually based on a reliable and delay propagation negligible communication medium. It would incur severe contention and congestion, and thus lead to an unbounded delay.

In our research, we propose a cooperative fault detection mechanism for detecting cluster head failures in cluster-based UnderWater Sensor Networks (UWSNs). The proposed detection mechanism aims to accurately and quickly detect the failure of a cluster head in order to avoid unnecessary energy consumption caused by a mistaken detection. For this purpose, it allows each cluster member to independently detect the fault status of its cluster head and then employs a distributed agreement protocol to reach an agreement on the fault status of the cluster head among multiple cluster members.

1.2.2 Wireless Terrestrial Sensor Networks

1.2.2.1 Energy Efficiency

In a wireless terrestrial sensor network (WTSN) [3], a number of sensor nodes are densely deployed in a field of interest with one or more data sinks located either at the center or out of the field. The sensor nodes observe the phenomenon at different points of the field and send the measured data to the sink(s). The observed phenomenon is usually a spatially dependent continuous process, in which the observed data have a certain spatial correlation. In general, the degree of the spatial correlation in the data increases with the decrease of the distances between sensor nodes. Therefore, spatially proximal sensor observations are highly correlated, which leads to considerable data redundancy in the network [18]. To efficiently use network resources to increase energy efficiency in data transmission, it is highly desirable to remove such data redundancy through effective data aggregation techniques.

To remove data redundancy caused by the data correlation in WTSN, Slepian-Wolf coding [19, 20] can be employed. Slepian-Wolf coding is a data compression technique that can completely remove data redundancy without requiring inter-sensor commu-

nication and therefore is a promising technique for data aggregation in WSNs. This technique is based on the assumption that each sensor node has *a priori* knowledge of the correlation structure, which depends on the distances between sensor nodes and the characteristics of the observed phenomena [20]. However, applying Slepian-Wolf coding globally in the whole network is difficult and would incur significant additional cost because each sensor node needs the knowledge of the global correlation structure to encode its own data with. Moreover, Slepian-Wolf coding is not tolerant to relay and node failures because the data from one node may affect the decoding of the data from other nodes [21]. For these reasons, it is unsuitable to apply Slepian-Wolf coding globally in a large network. In a cluster-based network, however, each cluster covers a smaller number of sensor nodes within a small local range of the network, which makes it feasible to apply Slepian-Wolf coding locally within each cluster because in this case a sensor node only needs the knowledge of local correlation structure to perform coding. Meanwhile, it will not obviously compromise the compression performance because in the real world the spatial correlation usually decreases with distance [18, 22].

The practical implementation of Slepian-Wolf coding has been achieved in [23, 24, 25, 26], which paves the way for performing data aggregation based on this promising coding technique. In [27, 28], Cristescu *et al.* studied data aggregation using global Slepian-Wolf coding. It is assumed that each node uses multi-hop flat routing for sending data to the data sink, and the complete knowledge of the correlation between the readings produced by all nodes is available at each node. In this case, global Slepian-Wolf coding and shortest-path routing are jointly considered, aiming at minimizing the total cost for sending compressed data. Although it has been shown that applying Slepian-Wolf coding globally is difficult in these studies, optimally constructing a cluster hierarchy with Slepian-Wolf coding in a distributed manner is

not considered. In [21], it has been shown that applying Slepian-Wolf coding locally within each cluster is able to overcome the effect of node and relay failures on the data reconstruction at the remote sink. However, no clustering protocol has been proposed to construct a cluster hierarchy and no work has taken account into the intra-cluster transmission cost which depends on the rate allocation within each cluster. On the other hand, existing clustering protocols for WSNs [7, 9, 10, 11, 29, 30, 31, 32, 33] are generally correlation structure blind and are not designed to maximally exploit Slepian-Wolf coding with respect to global compression gain. In addition, little work has been conducted on the optimization of data compression in the context of node clustering. The effect of spatial correlation on MAC protocols and routing algorithms has been investigated in [34, 35].

In this thesis, we study the major problems in applying Slepian-Wolf coding for data aggregation in cluster-based WSNs with an objective to optimize data compression so that the total amount of data in the whole network is minimized. We first consider the clustered Slepian-Wolf coding problem, which aims to select a set of disjoint potential clusters to cover the whole network such that the global compression gain of Slepian-Wolf coding is maximized. To solve this problem, a distributed optimal-compression clustering protocol (DOC) is proposed. Under the optimal cluster hierarchy constructed by DOC, we then consider the optimal intra-cluster rate allocation problem and present an approximation algorithm that can find an optimal rate allocation within each cluster to minimize the intra-cluster communication cost. With the optimal intra-cluster rate allocation, the procedures to perform Slepian-Wolf coding within a cluster are also presented. Finally, we propose a low-complexity joint coding scheme that combines clustered Slepian-Wolf coding with inter-cluster explicit entropy coding to further strip the data redundancy caused by the possible spatial correlation between different clusters.

1.2.2.2 Security

When sensor networks are deployed in a hostile environment, security becomes extremely important. Security ensures that certain information is never disclosed to unauthorized entities. Transmission of sensitive information, such as strategic or tactical military information, requires security. Leakage of such information to enemies could have devastating consequences. However, security in WTSN is not easy to achieve. Compared with conventional computer systms, severe challenges exist in WSNs, in which sensor nodes have limited processing capacity, storage, bandwidth, and energy.

To protect the sensitive contents of the communications, encryption is usually used, where the original intelligible content is converted into apparently random nonsense. So far, network-wide encryption is widely employed for WSN, where all the sensor nodes in the network are required to perform encryption. Since a WSN typically involves hundreds or thousands of sensor nodes, network-wide encryption would cause considerable computation, communication, and storage overhead due to data encryption. Two types of commonly used encryption schemes include symmetric-key encryption and public-key encryption [36]. Compared with a public-key alternative (e.g., RSA), symmetric encryption (e.g., AES) is much more energy efficient and thus more suitable for WSNs. Since the symmetric encryption uses the same secret key for both encryption at the sender and decryption at the receiver, key distribution mechanisms are required to securely deliver a secret key to each pair of sensor nodes in the hostile environment. The constraints of WSN, however, will affect the implementation of key distribution mechanisms. On one hand, sensor nodes are normally battery powered and can not be recharged after the network deployment. This implies that the key distribution schemes should incur little communication and computational

cost. On the other hand, WSN is ad hoc in nature where the topology of the network is determined at the time of deployment. This limits the ability to pre-configure the sensor nodes for specific purposes. To combat these constraints, most existing solutions to the problem of distributing keys to sensor nodes of large-scale WSNs [37, 38, 39, 40] build on the random key pre-distribution scheme [37], which relies on probabilistic key sharing among the nodes in a WSN. Since this scheme only guarantees that after network deployment two neighboring sensor nodes will share one key with a certain probability, some information may need to be exchanged among multiple sensor nodes to establish a shared key for two neighboring nodes, thus leading to extra energy consumptions. To reduce this control overhead, the identity based hierarchical key distribution [41] can be applied and its energy efficiency actually relies on a specific network model, as shown in Figure 1.3. According to the model, network involves a base sink and several clusters of sensor nodes, each led by a gateway which has considerably more energy resources compared to the regular sensor nodes. These better equipped gateways can be used to perform key distribution while requiring no information exchange among the resource-constrained sensor nodes.

In this thesis, we propose a novel encryption mechanism, called spatially selective encryption, for data encryption within a single cluster. This encryption mechanism only requires the cluster head to encrypt its data while allowing all cluster members to send their data without performing any encryption. Using this mechanism, as long as the data of the cluster head, a.k.a., the virtual key, is properly protected, the data from all cluster members can be securely protected, which can significantly reduce the energy consumption for data encryption. Furthermore, an energy-efficient key establishment protocol is also proposed to securely and efficiently establish the keys used for encrypting the visual key.

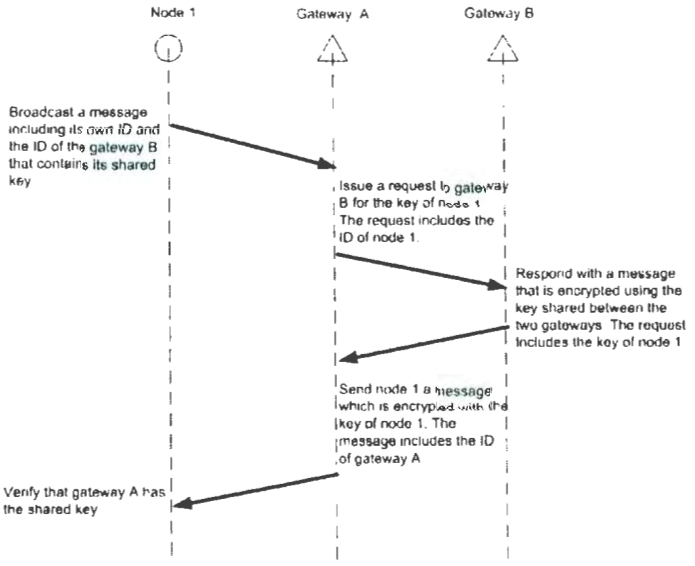


Figure 1.3: Illustration of the four-pass key distribution protocol model.

1.2.3 Wireless Multimedia Sensor Networks

1.2.3.1 High Throughput

Wireless Multimedia Sensor Networks (WMSNs) are an emerging networking paradigm that allows retrieving video streams, still images, as well as generic sensing data from the environment [42]. A WMSN promises a wide range of potential applications in both civilian and military areas which require visual and audio information, such as multimedia surveillance, advanced health care delivery, and industrial process control [42]. Different from conventional wireless sensor networks, a WMSN normally demands larger bandwidth and entails higher network throughput to transport large volume of data to the remote data sink rapidly and reliably. However, data rates provided by existing commercial sensor products, e.g., $250Kbps$ in MICAz [2], are not sufficient to support multimedia traffic.

On the other hand, current sensor nodes, such as MICAz and WINS [2, 42],

already support multiple channels for communication, for example, 40 channels in WINS [42]. Thus, by developing a multi-channel MAC protocol, which can effectively utilize the available channel capacity through the cooperative work from other sensor nodes, we can achieve a better support for multimedia applications which demand high data rates. The design of a highly efficient and reliable MAC protocol is thus critical. Conventionally, the goal is to provide sufficient transmission capacity at the minimum energy cost under a moderate network load condition. In order to support multimedia applications in wireless multimedia sensor networks, the design becomes the tradeoff between complexity/cost and the network throughput. Most MAC protocols in wireless sensor networks, such as S-MAC [43] and T-MAC [43], were proposed to support single-channel architecture. They are not suitable for multimedia applications: they are designed to be energy efficient, however, it is at the cost of increased latency and degraded network throughput. The Multi-frequency MAC protocol (MMSN) [44] is the first contention-based multi-channel protocol for wireless sensor networks. It consists of two parts: static channel assignment and contention-based data transmission. The channel assignment problem aims to statically allocate collision-free channels for nodes within two hops range. This problem can be reduced to a distance-2 coloring problem in graph theory. After channel assignment is completed, each sender switches to the receiver's channel for transmissions. Data transmission follows a contention-based approach on the per-packet basis, which will inevitably introduce significant control overhead. Besides their low energy efficiency, most contention-based multi-channel MAC protocols are particularly not suitable for delay-sensitive WMSN because every packet has to contend for medium access and the delay for data delivery could be potentially unbounded. The amount of time required to resolve collision is based on the load condition of the network, which makes it very difficult to guarantee a bounded delay.

In this thesis, we propose a clustered on-demand multi-channel MAC protocol (COM-MAC) to support energy-efficient, high-throughput, and reliable data transmission in WMSNs. The operation of proposed protocol consists three sessions: request session, scheduling session, and data transmission session. For COM-MAC to achieve high energy efficiency, first, a scheduled multi-channel medium access is used within each cluster so that cluster members can operate in a contention-free manner in both time and frequency domains to avoid collision, idle listening and overhearing. Second, to maximize the network throughput, a traffic-adaptive and QoS-aware scheduling algorithm is executed to dynamically allocate time slots and channels for sensor nodes based on the current data traffic information and QoS requirements. Finally, to enhance transmission reliability, a spectrum-aware ARQ is incorporated to better exploit the unused spectrum for a balance between the reliability and retransmission.

1.3 Main Contributions

The contributions of this thesis include a novel clustering protocol for underwater sensor network, a cooperative fault detection mechanism, a comprehensive study on the major problems in applying Slepian-Wolf coding for data aggregation to cluster-based WSNs, and a novel encryption mechanism, called spatially selective encryption.

First, we formulate the node clustering problem into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and prolong the lifetime of the network. A distributed minimum-cost clustering protocol is proposed to solve the formulated optimization problem. In the formulation, a cost metric called cluster cost is defined, which can reasonably measure the dynamic energy status of a potential cluster. To the best of our knowledge, this is the first time that the

node clustering problem is formulated into a cluster-centric cost-based optimization problem with an objective to improve energy efficiency and prolong network lifetime.

Second, we study the major problems in applying Slepian-Wolf coding for data aggregation to cluster-based WSNs, including the clustered Slepian-Wolf coding problem, the optimal rate allocation problem, and the joint Slepian-Wolf and explicit entropy coding problem. To the best of our knowledge, this is the first time that Slepian-Wolf coding is applied to cluster-based WSNs for data aggregation with specific effective solutions proposed to solve these problems, in particular, for the first time a distributed heuristic algorithm proposed to solve the clustered Slepian-Wolf coding problem.

Third, we propose a novel encryption mechanism, called spatially selective encryption, to achieve network wide security in WSNs. Unlike the conventional network-wide encryption,the proposed encryption mechanism requires that only a small portion of sensor nodes are selected to perform encryption of their data while allowing other sensors to send their data without performing any encryption. As long as the data of the selected nodes are protected, the data from other nodes can be properly protected. The proposed method can significantly reduce the energy consumption for both data encryption and key distribution.

Finally, we propose a cooperative fault detection mechanism with high accuracy and bounded delay for underwater sensor networks. To the best of our knowledge, this is the first time that the a failure detection scheme is proposed by exploiting the inherent characteristics of WSNs. Unlike the conventional heartbeat-based or gossiping-based fault detection mechanisms, the proposed detection mechanism is based a TDMA MAC protocol used in the network and runs concurrently with normal network operation by periodically performing a distributed detection process at each cluster member.

1.4 Thesis Organization

In chapter 2, we formulate the node clustering problem into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and prolong the lifetime of the network. To solve the formulated problem, a novel distributed clustering protocol called minimum-cost clustering protocol (MCCP) is proposed. In chapter 3, we present the proposed dependable clustering protocol to provide a robust clustered architecture against cluster-head failures in UWSNs. In chapter 4, we propose a cooperative fault detection mechanism for detecting cluster-head failures in cluster-based UnderWater Sensor Networks (UWSNs). In chapter 5, we study the major problems in applying Slepain-Wolf coding for data aggregation in a cluster-based WSN with an objective to optimize data compression so that the total amount of data generated in the whole network is minimized. In chapter 6, we present a combined data aggregation and encryption scheme using Slepain-Wolf coding for efficient and secured data transmission in wireless sensor networks (WSNs). In chapter 7, we propose a clustered on-demand multi-channel MAC protocol (COM-MAC) in order to maximize the network throughput with enhanced energy efficiency. In chapter 8, we conclude this thesis and introduce the future works.

Chapter 2

Distributed Minimum-Cost Clustering Protocol

2.1 Introduction

Node clustering has been widely studied in terrestrial wireless sensor networks [7, 9, 10, 11]. Due to the unique characteristics of underwater sensor networks, however, the clustering algorithms proposed for terrestrial WSNs and mobile ad hoc networks (MANETs) cannot be applied to UWSNs directly without modification.

In this chapter, we study the node clustering problem and consider energy efficiency in UWSNs. We formulate the problem into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and prolong the lifetime of the network. In the formulated problem, every node in the network is a cluster head candidate, which can potentially form a cluster together with some combination of its neighbors. The generation of a set of clusters is based on a cost metric (called *cluster cost*) defined for a potential cluster, which takes into account three important parameters that are relevant to the energy status of the cluster, including (1)

the total energy consumption of the cluster members for sending data to the cluster head; the residual energy of the cluster head and its cluster members; and the relative location between the cluster head and the uw-sink. To solve the formulated problem, we first propose a centralized minimum-cost clustering algorithm (MCCA) and then present a minimum-cost clustering protocol (MCCP) that implements MCCA in a distributed manner. Unlike most existing clustering algorithms, MCCA and MCCP select clusters, rather than cluster heads, based on the cost metric assigned to each potential cluster and attempts to minimize the overall cost of the selected clusters. Simulation results show that the proposed MCCP significantly improves the energy efficiency and prolong the network lifetime of a UWSN.

2.2 Problem Statement

2.2.1 Network architecture

A UWSN typically consists of several uw-sinks located at the centers of different monitored areas, a number of ocean bottom sensor nodes surrounding each uw-sink, and a surface station providing a link to an on-shore control center, as shown in Figure 2.1. A uw-sink has a sufficient power supply and is capable of handling multiple parallel communications with other sensor nodes. All sensor nodes are homogeneous and quasi-stationary. Each of them can adjust its transmission range with transmission power control. Unlike the terrestrial sensor network, a UWSN has some unique characteristics, such as highly limited bandwidth, long propagation delay, harsh geographical environment, and relatively small network scale [4]. Without loss of generality, we consider a network with one fixed uw-sink in this chapter.

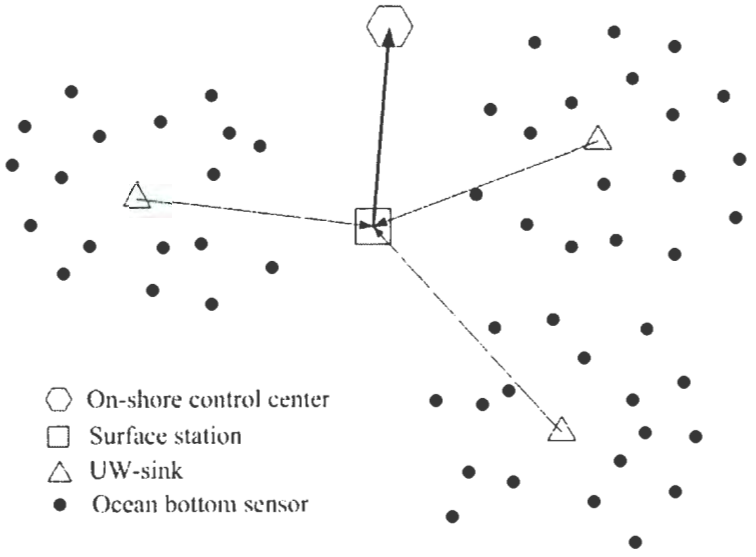


Figure 2.1: Network architecture.

2.2.2 Energy Model

We use the same energy model as used in [45], which was proposed for underwater acoustic networks. According to the model, to achieve a power level P_0 at a receiver at a distance d, the transmitter power $E_{tx}(d)$ must be

$$E_{tx}(d) = P_0 \cdot d^2 \cdot 10^{\frac{\alpha(f)}{10}}, \tag{2.1}$$

where $\alpha(f)$, measured in dB/m, is a medium absorption coefficient depending on the frequency range of interest under given water temperature and water salinity. $\alpha(f)$ is given by

$$\alpha(f) = 0.11\frac{10^{-3}f^2}{1+f^2} + 44\frac{10^{-3}f^2}{4100+f^2} + 2.75 \times 10^{-7}f^2 + 3 \times 10^{-6}, \tag{2.2}$$

where f is the carrier frequency for transmission measured in kHz.

2.2.3 Minimum-Cost Node Clustering

Given a network consisting of a finite set of sensor nodes V, every sensor node in the network is initially a cluster head candidate. We assume that the cluster diameter of each candidate is fixed, limited, and identical. The sensor nodes within the cluster diameter of a candidate v form a finite point set N_v with the cardinality of $|N_v|$, where N_v is called the neighbor set of candidate v. The power set of N_v, denoted by $P(N_v)$, is a set whose elements are the subsets of N_v and $P(N_v)$ constitutes all possible combinations of nodes in Nv. Thus, the cardinality of $P(N_v)$ is $2^{|N_v|}$. Each element of $P(N_v)$ is called a cluster member set of candidate v, which is denoted by N_v. Thus, a candidate v, combined with each cluster member set $B_v \in P(N_v)$, can form a potential cluster $A := B_v \bigcup \{v\}$, and the total number of potential clusters generated by candidate v is $2^{|N_v|}$. Obviously, there initially exist a total number of $\sum_{v \in V} 2^{|N_v|}$ potential clusters in the network, which are generated by all cluster head candidates in V. We use S to denote the cluster set which consists of all the potential clusters in the network.

Given the above assumptions, the node clustering problem can be formulated into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and thus prolong the lifetime of the network, i.e., to select a set of potential clusters C^* from the cluster set S to cover the whole network so that the overall *cost* of all selected clusters, is minimized, i.e.,

$$C^* = \arg\min_{C \subseteq S} \sum_{A \in C} cost(A), \tag{2.3}$$

where $\bigcup_{A \in C^*} A = V$ and $\bigcap_{A \in C^*} A = \phi$. $cost(A)$ is a cost metric defined for a cluster, which will be described in the next section. We also refer to this problem as minimum-cost node clustering (MCNC) problem.

2.2.4 Cost Metric

To consider energy efficiency, we define in the formulation a cost metric called *cluster cost* or simply *cost* for a potential cluster, which takes into account three important parameters that are relevant to the energy status of the cluster, namely, the total energy consumption of the cluster members for sending data to the cluster head; the residual energy of the cluster head and its cluster members; and the relative location between the cluster head and the uw-sink. For ease of exposition, we first consider the first and third parameters, and then incorporate the residual energy in the definition.

Given a potential cluster $A := B_v \bigcup \{v\}$, where v is a cluster head candidate and N_v is a cluster member set of v, the *cluster cost* of cluster A is defined as

$$cost(A) = \sum_{u \in B_v} cost(u, v) + cost(v, s), \tag{2.4}$$

where $\sum_{u \in B_v} cost(u, v)$ is the total energy consumed by all sensor nodes in N_v for sending one data packet to v, which is also called *intra-cluster cost*. $cost(v, s)$ is the energy consumed by v for sending one data packet to the uw-sink s directly, which is also called *relay cost*. The *relay cost* is proportional to the distance between v and s. In the above definition, the geographic distribution of a cluster head has a dominant impact on the *cost* of a potential cluster. A small change in the distance between a cluster head and the uw-sink would lead to a big change of the *relay cost* and this change will overwhelm the small fluctuation of the *intra-cluster* cost, which is bounded by the fixed cluster diameter and the limited variance of the number of cluster members in different clusters. Therefore, if a cluster head is closer to the area surrounding the uw-sink which has a higher traffic load, its cluster tends to have a smaller *cluster cost*. On the other hand, the effect of the *intra-cluster cost* would dominate the *cluster cost* if the cluster heads of different clusters have

very close spatial proximity relative to each other in a local area or have the same distance to the uw-sink but in different areas. In this case, a cluster with a smaller *intra-cluster cost* has a smaller *cluster cost*. The defined cost metric actually reflects the energy consumption of a potential cluster. To more comprehensively reflect the dynamic energy status of the cluster, the residual energy of the cluster head and its cluster members should also be considered. In this case, the cost metric should be defined such that when the cluster head candidates initially have plenty of residual energy, the energy consumption term dominates, while when the residual energy of the cluster head candidate becomes small, the residual energy term dominates. When two candidates have comparable residual energy levels, the one covering the cluster members with low residual energy should be given preference. Based on the above arguments, a complete definition of the cost metric is given as follows

$$\cos t(A) = \frac{\sum_{u \in E_v} cost(u, v) f_1(E_u) + cost(v, s) f_2(A)}{f_1(E_v)}, \tag{2.5}$$

where E_v(or E_u) is the residual energy of node v (or u) normalized by the initial energy of the node and

$$f_1(x) = \begin{cases} \frac{0.9}{1+(\frac{E_t}{x})^{\alpha}} + 0.1, x \geq E_t \\ \frac{0.9}{1+(\frac{E_t}{x})^{\beta}} + 0.1, x < E_t \end{cases} \tag{2.6}$$

$$f_2(X) = \frac{\sum_{x \in X} f_1(E_x)}{|X|}, \tag{2.7}$$

where E_t $(0 < E_t < 1)$ is a threshold for the residual energy. When the normalized residual energy of a node is larger than this threshold, a node is said to be in a high energy state. Otherwise, it is in a low energy state. E_t is used to control how long a node can act as a cluster head. For a selected cluster head, it can return to be a cluster member only when its normalized residue energy is below E_t. In this case,

$f_1(E_v)$ in the denominator would largely decrease, which leads to a big increase of the *cost* of the cluster head and thus makes the cluster head return to be a cluster member. Meanwhile, when a cluster head covers a cluster member in a low energy state, its cost metric will diminish because the numerator of the cost metric decreases with the number of the cluster members in a low energy state. α is a coefficient used to control the sensitivity of the cost metric to the residual energy when a node is in a high energy state while β is a coefficient used for the same purpose when a node is in a low energy state. Both α and β can be adjusted to accommodate less or more energy consideration in the cost metric. In this work, E_t is set to be 0.25 so that a cluster head can make sufficient contribution before it returns to be a cluster member. α is set to be 30 so that the cost metric changes dramatically over normalized residual energy in a high energy state because in this case $f_1(x)$ is approximately 1. β is set to be 2 so that the energy consumption term dominates the cost metric when a node is in a low energy state.

2.3 Distributed Minimum-Cost Clustering Protocol

In our research, we propose a distributed minimum-cost clustering algorithm (MCCP) to solve the MCNC problem in a distributed network environment. We first present a centralized minimum-cost clustering algorithm (MCCA) and then present MCCP that implements MCCA in a distributed manner.

2.3.1 Centralized Minimum-Cost Clustering Algorithm

In a centralized network, node clustering is performed at a central controller (e.g., a uw-sink), which has the full knowledge of the network.

If we consider $cost(A)$ as the weight associated with a cluster A, the MCNC problem is very similar to the minimum weight set cover (MWSC) problem, which is described as follows. Given a set of points (i.e., a set of sensor nodes V), a collection of potential point sets (i.e., the cluster set S), and a nonnegative weight assigned to each point set (i.e., $cost(A)$ assigned to a potential cluster $A \in S$), find a subset of the collection of potential point sets (i.e., a cluster set $C^* \subseteq S$) such that each element in the given set of points (i.e., each sensor node $u \in V$) belongs to at least one of the point sets in the subset (i.e., one cluster in the cluster set C^*) and the sum of the weights of the point sets in the subset (i.e., the overall cluster cost of all clusters in C^*) is minimized.

Note that the only difference between the two problems is that a point in the MWSC problem can be covered by more than one point set while a sensor node in the MCNC problem can be covered by one and only one cluster. Meanwhile, it is worth pointing out that if the MWSC problem has the same number of points as that in V, the total number of potential point sets in the MWSC problem is $2^{|V|}$, while in the MCNC problem the total number of potential clusters decreases to $\sum_{v \in V} 2^{|N_v|}$ because of the limited cluster diameter of each node in the network.

The MWSC problem is well-known to be NP hard. To solve this problem, an approximation algorithm was proposed by Chvátal [46], which is based on a sequential greedy method. If we use the same notations S to denote a collection of potential point sets and C^* to denote a collection of point sets we would like to find, Chvátal's algorithm starts with $C^* := \phi$ and each time greedily adds one "qualified" set to C^*

until the sets in C^* include all points. Figure 2.2 gives the pseudo-code of Chvátal's algorithm, where A is a set (called "qualified" set) with minimum $\frac{W(A)}{|A-Q|}$, W is a set containing all nodes which are covered by all "qualified" sets already added to C^*, and $A - W$ is a set of elements which are members of A, but not members of W. Hence, $|A - W|$ is the number of new nodes to be covered by C^* if a new "qualified" set A is added to C^*. This means that a selected "qualified" set A may have members in common with the sets already in C^*, which is not allowed in the MCNC problem.

```
CHVÁTAL(v)
1  C* ← φ and Q ← φ
2  while Q ≠ V
3        do
4           Select a set A ∈ S such that W(A)/|A−Q| is minimized
5           C* ← C* ⋃ {A} and Q ← Q ⋃ A
```

Figure 2.2: Chvátal's algorithm.

To solve the MCNC problem, we propose a heuristic algorithm called minimum-cost clustering algorithm (MCCA) based on Chvátal's algorithm to generate a pairwise disjoint C^*, in which any two distinct sets are disjoint. Figure 2.3 presents the pseudo-code of MCCA, where *cost* corresponds to *weight* in the MWSC problem, Z is a set containing all remaining cluster head candidates, H is a set containing the cluster heads of all clusters already selected, and M is a set containing all representative clusters already found. The main steps are described as follows.

1. Find out all nodes that have not been covered by the clusters already added into C^* and are still cluster head candidates (i.e., $Z = V - W$).

HEURISTIC ALGORITHM(v)

1 $C^* \leftarrow \phi$ and $Q \leftarrow \phi; H \leftarrow \phi$

2 **while** $Q \neq V$

3 **do** $Z \leftarrow V - Q$

4 $M \leftarrow \phi$

5 **for**

6 **do** $B_v \leftarrow \arg \min_{B \in P(N_v - Q)} \{\frac{cost(\{v\} \bigcup B)}{|\{v\} \bigcup B|}\}$; $R_v \leftarrow B_v \bigcup \{v\}$

7 $M \leftarrow M \bigcup \{R_v\}$

8 $A \leftarrow \arg \min_{R_u \in M} \{\frac{cost(R_u)}{|R_u|}\}$

9 $C^* \leftarrow C^* \bigcup \{A\}$ and $Q \leftarrow Q \bigcup A; H \leftarrow H \bigcup \{u\}$

Figure 2.3: Heuristic algorithm.

2. For each candidate v, construct its potential clusters by combining every possible combination of the nodes that are in its neighbor set N_v but have not been covered by the clusters already in C^*, i.e., $P(N_v - W)$. Then, calculate the cost of each of its potential clusters and find out its representative cluster R_v, which is the potential cluster with the minimum average cost $cost(R_v)/|R_v|$, where $R_v = \{v\} \bigcup B_v$.

3. Compare the average costs of the representative clusters of all candidates and select the representative cluster A with the minimum average cost as a "qualified" cluster to be added into C^*. The corresponding candidate u becomes a cluster head.

4. Return to step 1 and repeat the above steps until the clusters in C^* cover all nodes in the network.

2.3.2 Distributed Minimum-Cost Clustering Protocol

In the centralized MCCA, whether a candidate v can become a cluster head is determined by whether its corresponding representative cluster R_v has a minimum value of $cost(R_v)/|R_v|$ among the representative clusters of all candidates in the network. In a distributed network, however, each node makes a clustering decision independently. The value of $cost(R_v)/|R_v|$ of candidate v can only be changed if any of the candidates within a distance of at most 2 times the cluster diameter (2-hop range) becomes a cluster head, which may lead to the changes of $cost(R_v)$ and $|R_v|$because only the candidates within a 2-hop range may cover common nodes. Therefore, if the value of $cost(R_v)/|R_v|$ is smaller than that of any other candidate within a distance of at most 2 hops from v, MCCA will select v as a cluster head with its representative cluster before any of the candidates within a distance of at most 2 hops. Based on this observation, we present a distributed MCCP to implement the centralized MCCA in a distributed manner. With MCCP, all candidates, instead of a single central controller, need to find out their own representative clusters based on local information, and then exchange the average costs of their representative clusters within a 2-hop range to collaboratively select the "qualified" clusters.

MCCP consists of two stages: initialization stage and execution stage, which are described as follows.

- Initialization stage: When a network is initially deployed, all nodes are set to be candidates. Then each candidate constructs its neighbor set and its uncovered neighbor set, which consists of the nodes that are in the neighbor set but are still candidates.

- Execution stage: A candidate generates its potential clusters by searching every possible combination of elements in its uncovered neighbor set, selects a rep-

resentative cluster from these potential clusters, and sends the average cost of the representative cluster to all candidates within its 2-hop range. A candidate collects the average costs of the representative clusters of all candidates within its 2-hop range. If the candidate itself has a minimum average cost, it becomes a cluster head and advertises an *INVITE* message to all the nodes in its representative cluster to invite them become its cluster members. Otherwise, there are two cases for a candidate.

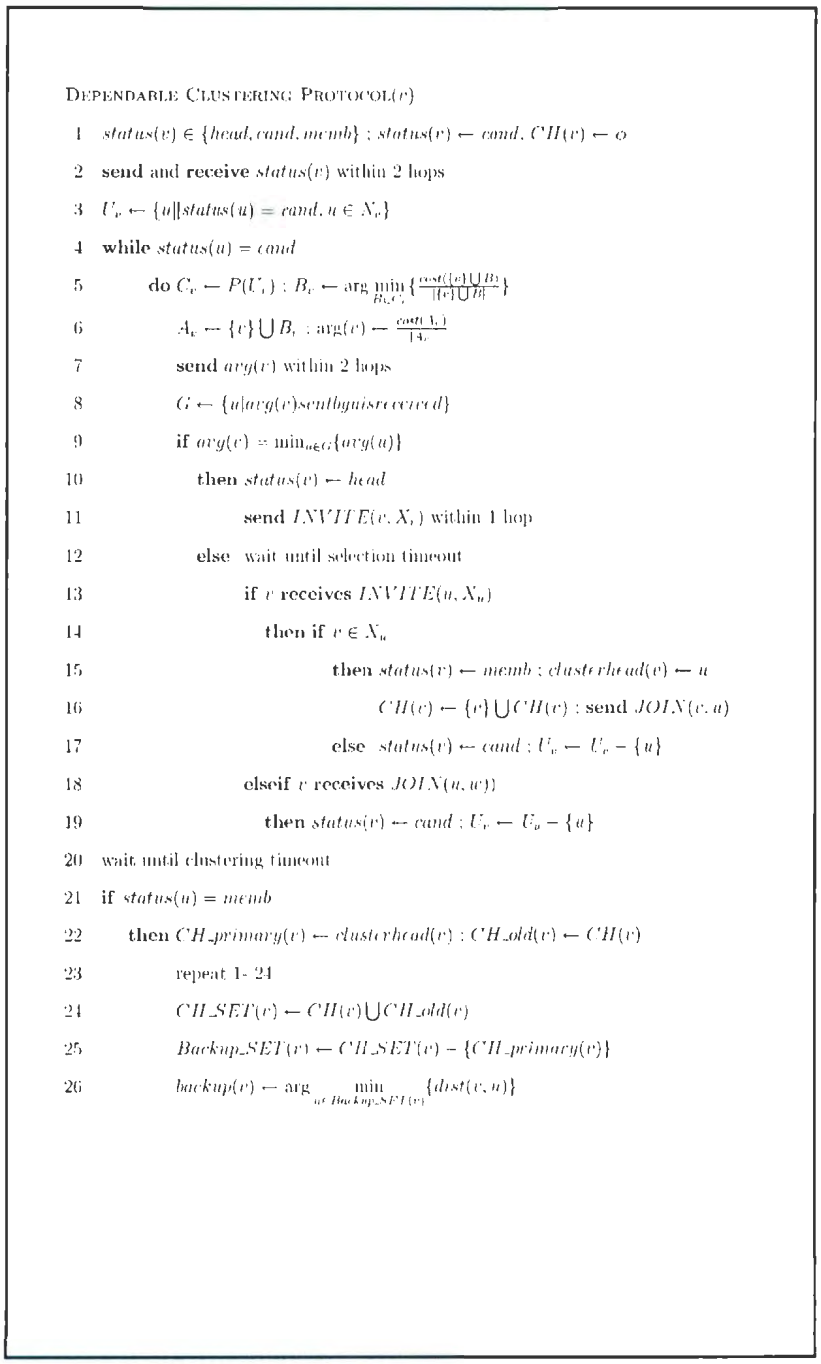
1. If an *INVITE* message is received and the destination of this message is the candidate, the candidate first changes its candidate status to a cluster member. Then it extracts the cluster head ID from the *INVITE* message and broadcasts a *JOIN* message to all the nodes within its cluster diameter. This *JOIN* message will acknowledge the receipt of the *INVITE* message to the candidate and at the same time notify the other candidates within the cluster diameter that the candidate has become a cluster member of some cluster head.
2. If no *INVITE* message is received or some *INVITE* messages for other nodes are received, the candidate stays in its candidate status and reselects its representative cluster because some elements in its uncovered neighbor set might have been covered by some cluster heads or have become cluster heads.

The above procedures are performed by all candidates until each of them becomes either a cluster head or a cluster member. At the end, no candidate remains in the network and every cluster member belongs to a cluster.

The pseudo-code of the above procedures are shown as in Figure 2.4 and the messages or variables used in the pseudo-code are defined in Table 2.1.

Table 2.1: Pesudocode Symbol

Symbols	Description
A_v	The representative cluster of candidate v
X_v	The set containing the cluster members of A_v
$avg(v)$	The average cost of the representative cluster A_v
head	A flag indicating a cluster head
cand	A flag indicating a candidate
memb	A flag indicating a cluster member
G	A set containing the average costs sent by other cluster heads within 2-hop range of a candidate
$INVITE(v, Xv)$	A message inviting the nodes in set Xv to be the cluster members of candidate v
$JOIN(v, u)$	A message acknowledging that node v received the *INVITE* message sent by candidate u and joins the cluster as a cluster member of candidate u



DEPENDABLE CLUSTERING PROTOCOL(v)

1 $status(v) \in \{head, cand, memb\}$; $status(v) \leftarrow cand$. $CH(v) \leftarrow \phi$
2 **send** and **receive** $status(v)$ within 2 hops
3 $U_v \leftarrow \{u \| status(u) = cand, u \in N_v\}$
4 **while** $status(u) = cand$
5 **do** $C_v \leftarrow P(U_v)$; $B_v \leftarrow \arg\min_{B_i \in C_v} \{\frac{cost(\{v\} \bigcup B_i)}{|\{v\} \bigcup B_i|}\}$
6 $A_v \leftarrow \{v\} \bigcup B_v$; $\arg(v) \leftarrow \frac{cost(A_v)}{|A_v|}$
7 **send** $arg(v)$ within 2 hops
8 $G \leftarrow \{u | arg(v) sent by u is received\}$
9 **if** $arg(v) = \min_{u \in G}\{arg(u)\}$
10 **then** $status(v) \leftarrow head$
11 **send** $INVITE(v, X_v)$ within 1 hop
12 **else** wait until selection timeout
13 **if** v **receives** $INVITE(u, X_u)$
14 **then if** $v \in X_u$
15 **then** $status(v) \leftarrow memb$; $clusterhead(v) \leftarrow u$
16 $CH(v) \leftarrow \{v\} \bigcup CH(v)$; **send** $JOIN(v, u)$
17 **else** $status(v) \leftarrow cand$; $U_v \leftarrow U_v - \{u\}$
18 **elseif** v **receives** $JOIN(u, w)$
19 **then** $status(v) \leftarrow cand$; $U_v \leftarrow U_v - \{u\}$
20 wait until clustering timeout
21 **if** $status(u) = memb$
22 **then** $CH\_primary(v) \leftarrow clusterhead(v)$; $CH\_old(v) \leftarrow CH(v)$
23 repeat 1- 24
24 $CH\_SET(v) \leftarrow CH(v) \bigcup CH\_old(v)$
25 $Backup\_SET(v) \leftarrow CH\_SET(v) - \{CH\_primary(v)\}$
26 $backup(v) \leftarrow \arg \min_{u \in Backup\_SET(v)} \{dist(v, u)\}$

Figure 2.4: Distributed minimum-cost clustering protocol.

2.3.3 Computational Complexity

Consider a network consisting of $|V|$sensors uniformly distributed over a region with a predefined node density ρ to guarantee the sensing coverage. The computational complexity of MCCA is $O(|V|)$. This is because in every iteration the algorithm selects at least one cluster with a cluster head v from the $2^{|U_v|}$ potential clusters generated by v, where Uv is the uncovered neighbor set of v. Since $|U_v| \leq |N_v|$, where N_v is the neighbor set of v, the algorithm terminates in $O(\sum_{v \in V} 2^{|N_v|})$ iterations. Furthermore, let r be the cluster diameter of v. The number of nodes within the limited cluster diameter of v is $|N_v| = \rho \pi r^2$, where ρ is the sensor density. Therefore, $2^{|N_v|}$ can be denoted by a constant D and the heuristic algorithm terminates in $O(\sum_{v \in V} D) = O(|V|)$ iterations. Similarly, MCCP also has a computational complexity of $(O|V|)$.

2.3.4 Properties

One important property of MCCP is its ability of adapting geographical cluster head distribution to the traffic pattern in the network and thus avoiding the formation of hot spots around the uw-sink. This is implemented by generating more cluster heads in the area closer to the uw-sink so that the high traffic load in that area is undertaken by all the cluster heads in a balanced manner. At the same time, the number of cluster members in a cluster depends on the distance between the cluster head and the uw-sink, which means that a cluster closer to the uw-sink consists of less cluster members.

With MCCP, candidates at different locations can construct clusters at the same time. MCCP only allows those candidates at a distance more than two times the cluster diameter to simultaneously become cluster heads in each iteration. For this reason, the effect of the formation of a cluster in an area far away from the uw-sink

on the formation of a cluster in the area closer to the uw-sink can be avoided. If two candidates are within a distance less than two times the cluster diameter, the one closer to the uw-sink is more likely to construct a cluster because its cluster tends to have a smaller average cost. The reason is that the cost metric is designed such that the distance between a cluster head and the uw-sink dominates the cost of a cluster. On the other hand, to become a cluster head, a candidate needs to select a proper set of nodes within its cluster diameter to build a cluster with the minimum average cost (representative cluster). If the candidate is around the uw-sink, only clustering the nodes closer to the candidate itself can lead to a representative with a smaller average cost. For a candidate far away from the uw-sink, its distance from the uw-sink dominates the cost metric. In this case, incorporating more nodes as cluster members without considering their proximity to the candidate can lead to a representative cluster with a smaller average cost.

This property can be illustrated by a simple example shown in Figure 2.5. Candidate A0 can generate four potential clusters: {A0}, {A0 A1}, {A0 A2}, and {A0 A1 A2}. Similarly, B0 has four potential clusters: {B0}, {B0 B1}, {B0 B2}, and {B0 B1 B2}. Obviously, B0 and A0 can become two cluster heads at the some time without affecting each other. The four potential clusters of A0 have average costs proportional to {52/1=25, (52+32)/2 = 17, (52+82)/2=44.5, and (52+82+32)/3= 32.66}. A0 associated with A1 constructs the cluster which has the minimum average cost because the cost metric is sensitive to the intra-cluster cost and adding A2 to the cluster will result in a larger average cost. The four potential clusters of B0 have average costs proportional to {402/1=1600, (402+32)/2 = 804.5, (402+82)/2 = 832, (402+82+32)/3= 557.6667}. For B0, the cluster with {B0 B1 B2} has the minimum average cost because the relay cost dominate the cost metric (in this case, cost metric approximately proportional to 402) and incorporating more nodes to construct the

clusters leads to representative clusters with a smaller average cost.

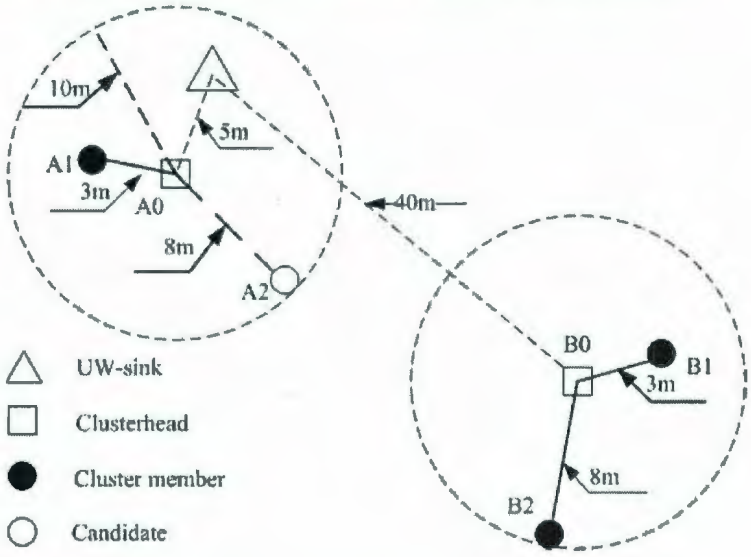


Figure 2.5: An illustration of MCCP property.

Another property of MCCP is the ability of balancing the traffic load between cluster heads and cluster members by periodically re-clustering the sensor nodes in the network. Since the cluster cost is recalculated each time clustering is performed, the residual energy term of the cluster cost can be periodically updated such that the roles of cluster heads and cluster members can be exchanged based on the current residual energy distribution. Normally, the period to re-cluster the network is in the range of days or months for underwater applications [4], where continuously transmitting sensed data is not required. Therefore, the network can be re-clustered by MCCP with a low frequency without obviously affecting network performance.

2.3.5 MAC Protocol

Similar to most existing clustering protocol, MCCP also employs TDMA for transmission scheduling within a cluster, where the cluster head generates TDMA schedule

and allocates a fixed slot to each cluster member in the cluster. Each cluster head communicates with other cluster head using direct-sequence spread spectrum (DSSS) to reduce the inter-cluster interface while each cluster member employs the pseudo-noise code assigned to its cluster head to spread the spectrum of its signal over the entire band before it transmits the signal to its cluster head. As a result, the effect of frequency fading caused by underwater multiple paths is mitigated. To enable TDMA, time synchronization under a high latency environment is required [17]. Meanwhile, to implement DSSS, CDMA code assignment (CA) algorithms are needed. The centralized CA algorithm in [7] uses the data sink to assign a unique orthogonal code to each cluster head while the distributed CA algorithms in [47] assigns different codes to adjacent clusters in order to spatially reuse spreading codes.

2.3.6 Multi-Hop Routing Protocol

After a cluster head collects data from its cluster members, it will send the data to the uw-sink. A cluster head far away from the UW-sink needs to find a multi-hop path to the sink. For this purpose, we use the multi-hop routing protocol proposed in [48], which was designed specifically for underwater sensor networks. Using this routing protocol, cluster head i selects cluster head j as its next hop if

$$j = \arg \min_{j \in N_i \bigcap H_i} c_{ij}, \tag{2.8}$$

where N_i is a set of nodes within the transmission range of cluster head i, and H_i is a set of cluster heads closer to the uw-sink s than cluster head i, i.e., $j \in H_i$ if and only if the d_{js}, the distance between node j and the uw-sink s, is less than d_{is}, the distance between node i and the uw-sink s. c_{ij} is the cost associated to the link between i and j, which is an estimate of the energy required to transmit a bit from cluster head i to

the uw-sink when cluster head j is selected. Since we do not consider retransmission in our simulation, c_{ij} is given by

$$c_{ij} = E_{tx}(d_{ij}) \cdot h_{is}, \tag{2.9}$$

where d_{ij} is the distance between i and j, and h_{is} is the estimated number of hops from node i to the uw-sink s, and is calculated as

$$h_{is} = \max(\frac{d_{is}}{< d_{ij} >_{is}}, 1), \tag{2.10}$$

where $< d_{ij} >_{is}$ is the projection of d_{ij} onto the line connecting node i with the uw-sink s.

2.4 Performance Evaluation

In this section, we evaluate the performance of MCCP through simulation experiments. We compare MCCP with HEED in terms of network lifetime. The network lifetime is measured in two different ways: temporal lifetime and capacity lifetime. The temporal lifetime is defined as the number of simulation rounds executed until a certain percentage of nodes die, and the capacity lifetime is defined as the total number of packets the uw-sink has received until a certain percentage of nodes die.

We used ns-2 to perform the simulation and extended it to include the underwater propagation loss and the physical layer characteristics of underwater transceivers. In the simulation, we considered 100 sensor nodes uniformly deployed in a 100m×100m sensing region unless otherwise specified, which is shown in Figure 2.6(a). The uw-sink is located at the center of the region. We assume that at the beginning of each round, each node has an initial battery power of *2J*. Each round of simulation

consists of an intra-cluster communication period consisting of six TDM frames and an inter-cluster communication period. After each round, the network is re-clustered by MCCP. The cluster diameter is *10 m*, and the transmission range of a cluster head for intra-cluster communication is equal to the distance between the cluster head and the uw-sink so that the cluster head can surely send its data to the uw-sink even if an intermediate node for the next hop cannot be found. In the energy model, we assume that $P0$=1.6×10-3 *J/bit* and f=10 *kHz*.

Table 2.2: Simulation Parameters

Type	Parameter	Value
System	Network grid	From (0,0) to (100,100)
	Sink	At (50,50)
	Initial energy	*2J*
Midfielders	Cluster radius	25 m
	Data packet size	100 bytes
	Packet header size	25 bytes
	Round	6 TDM
Radio model	$P0$	1.6×10^{-3} *J/bit*
	f	10 *kHz*

Figure 2.6(b) and Figure 2.6(c) show the cluster head distribution with MCCP and HEED. Both MCCP and HEED were simulated using the same topology shown in Figure 2.6(a). As expected, the cluster head distribution of MCCP is adaptive to the geographical traffic pattern of the network. In the area closer to the UW-sink, more cluster heads were generated to balance the higher traffic load in that area. In contrast, the cluster heads generated by HEED are evenly distributed in

the network because HEED does not have the adaptive property for the geographical traffic pattern of the network.

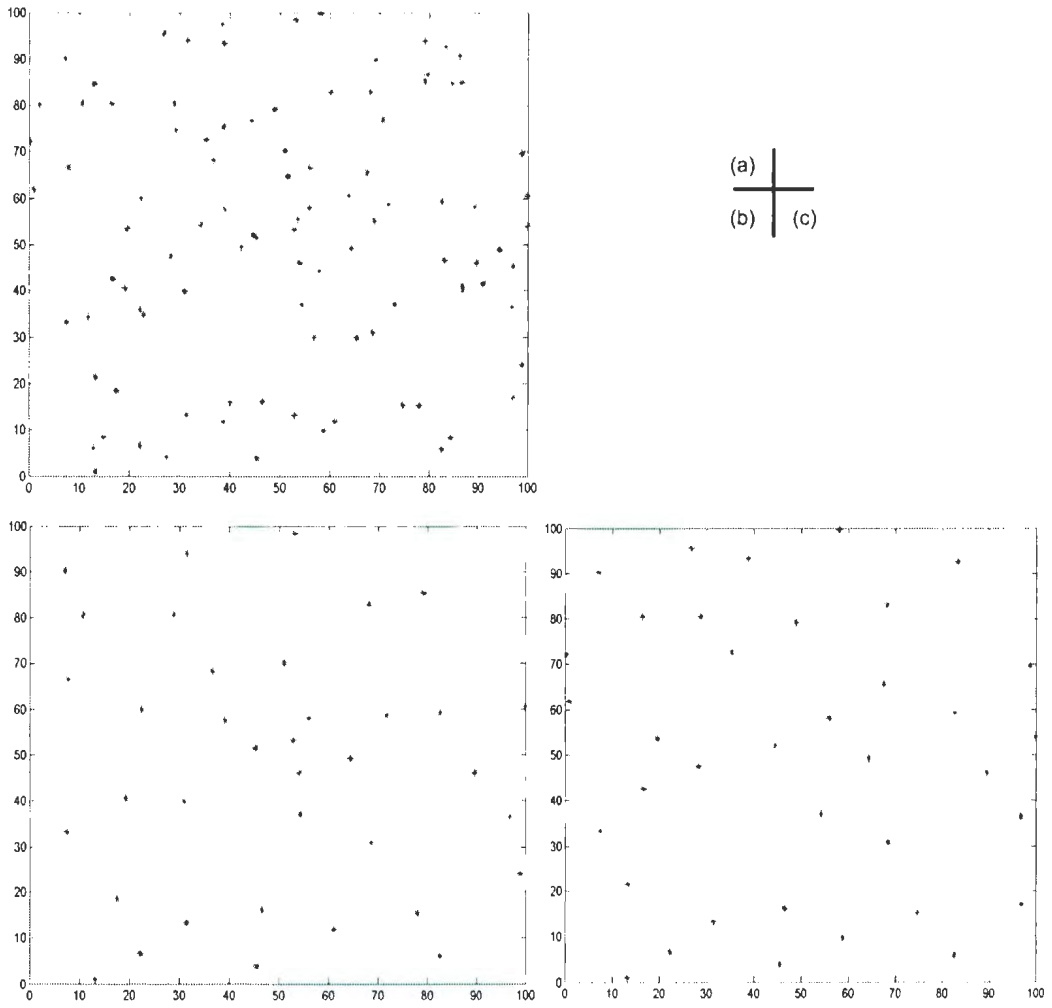


Figure 2.6: (a) Node distribution; (b) Cluster head distribution with MCCP; (c) Cluster head distribution with HEED.

Figure 2.7(a) shows the temporal lifetime with MCCP and HEED, respectively. Obviously, MCCP improves the temporal lifetime as compared with HEED, especially if the temporal lifetime is defined as the number of rounds when the first node

dies. Similarly, Figure 2.7(b) shows the capacity lifetime with MCCP and HEED, respectively.

Figure 2.8(a) show the effect of different number of nodes on temporal lifetime. It is seen that both MCCP and HEED have good network scalability because the temporal lifetime of both protocols does not deteriorate with the increment of the number of nodes. The performance enhancement of MCCP over HEED, in the term of temporal lifetime, is not affected by the number of nodes in the network. We can obtain the similar results for capacity lifetime, which are shown in Figure 2.8(b).

Figure 2.9(a) and Figure 2.9(b) show the effect of non-uniform node distribution on the improvement of temporal lifetime and capacity lifetime, respectively. Similar to the experiment in [9], we divide the network into four areas of equal sizes (A1, A2, A3, A4), and the probability of a node falling into each of the areas is 10 percent for A1, 20 percent for A2, 30 percent for A3, and 40 percent for A4. It is shown that the non-uniform node distribution does not affect the lifetime improvement of MCCP over HEED.

2.5 Summary

In this chapter, we have studied the node clustering problem in a UWSN and formulated the problem into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and prolong the lifetime of the network. To solve the formulated problem, a distributed minimum cost clustering protocol (MCCP) has been proposed, which can not only make geographical cluster head distribution adapt to the traffic pattern in the network and thus avoid the formation of hot spots around a uw-sink, but also balance the traffic load between cluster heads and cluster members through periodical re-clustering the sensor nodes in the network.

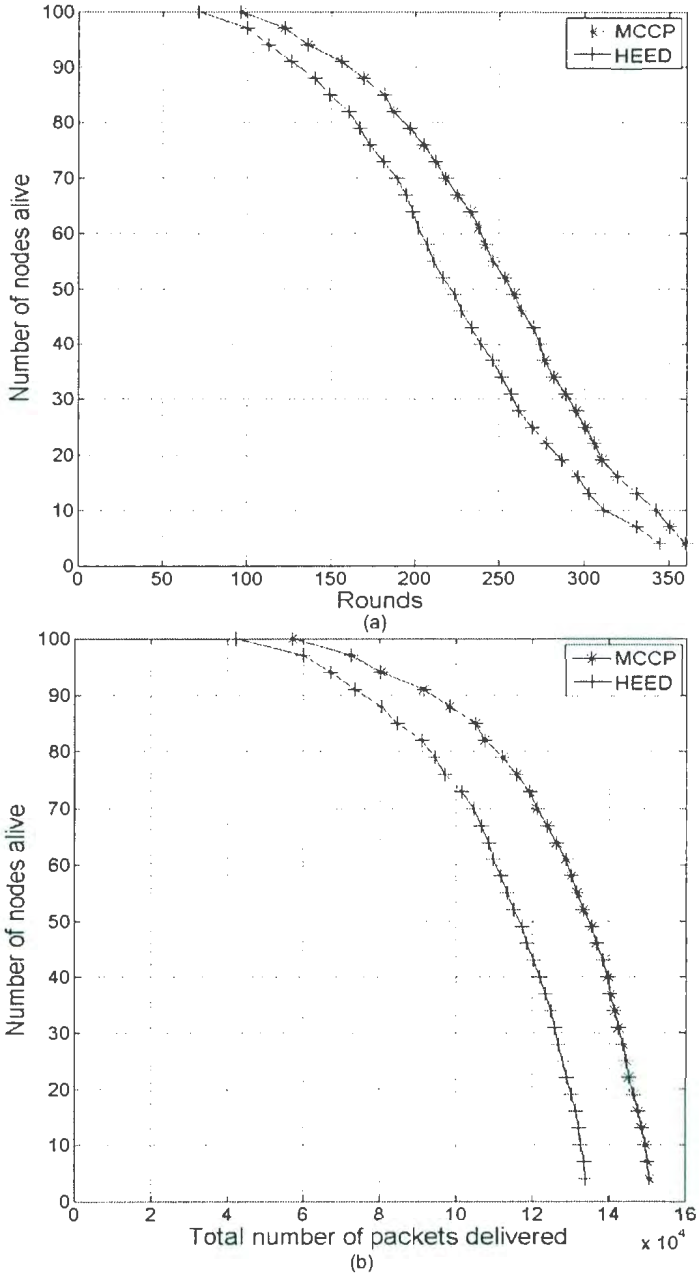


Figure 2.7: Network lifetime: (a) Temporal lifetime; (b) Capacity lifetime.

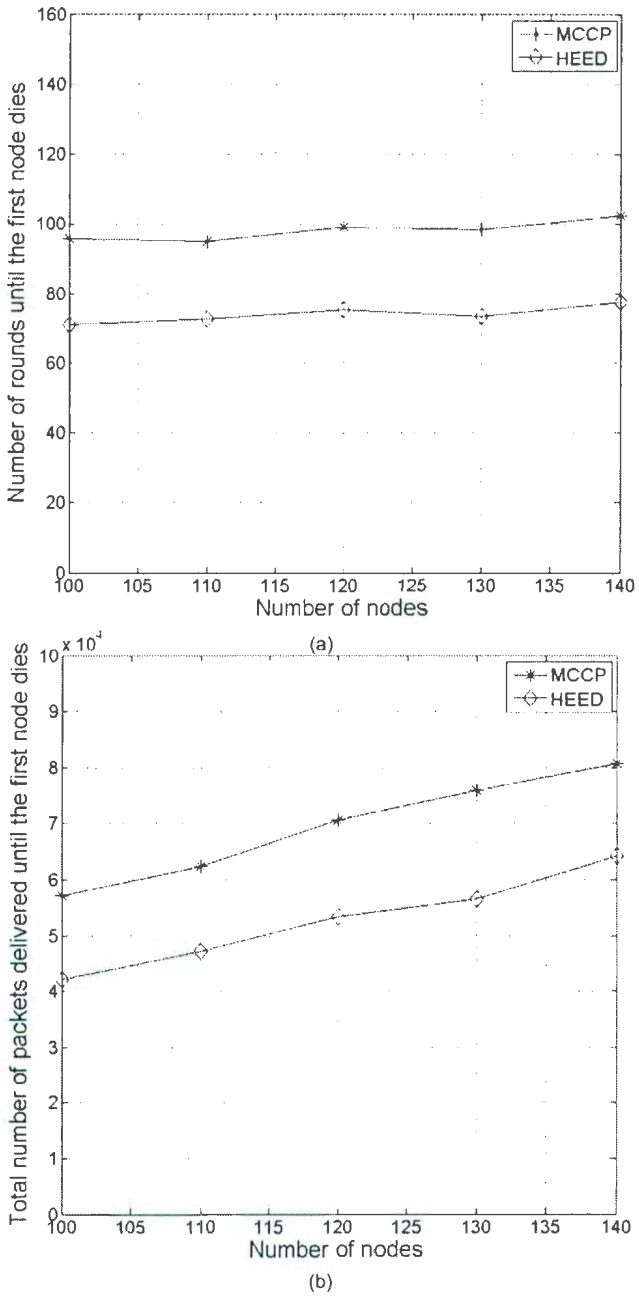


Figure 2.8: Impact of the network size on network time:(a) Temporal lifetime; (b) Capacity lifetime.

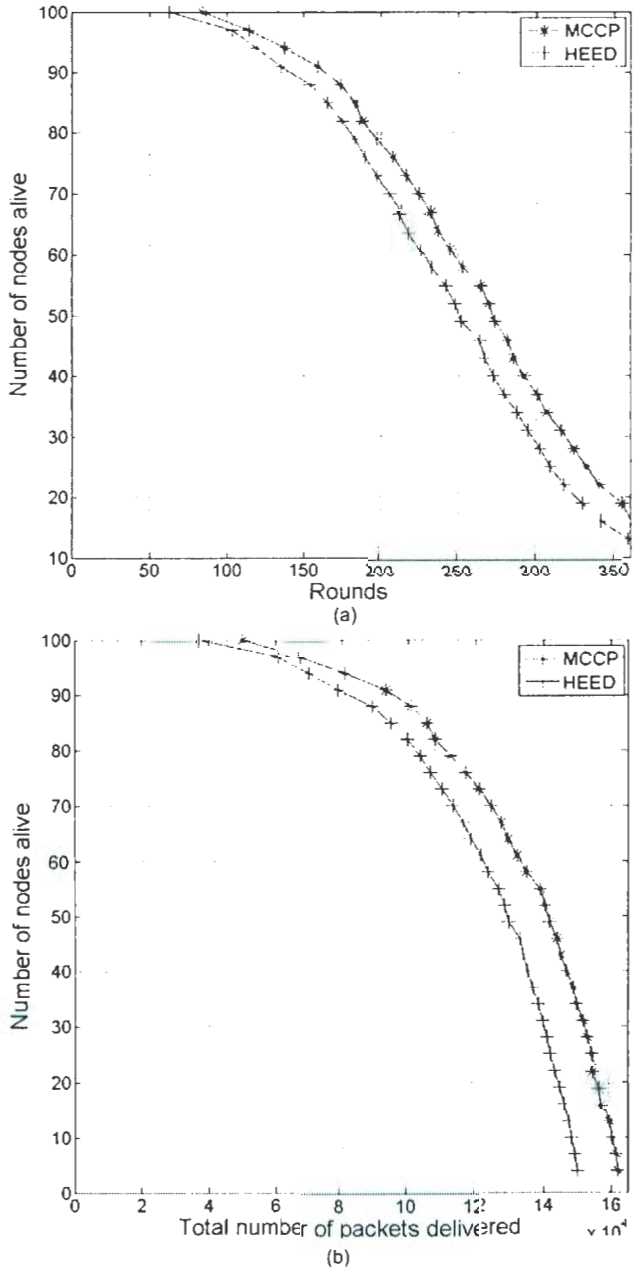


Figure 2.9: Impact of non-uniform node distribution on network lifetime:(a) Temporal lifetime; (b) Capacity lifetime.

The simulation results show that MCCP significantly improves the network lifetime as compared with the well-known HEED protocol.

Chapter 3

A Robust Architecture for Underwater Sensor Network

3.1 Introduction

Node clustering provides an effective approach to improve the energy efficiency and prolong the network lifetime of a WSN. Moreover, clustering leads to a hierarchical network architecture, which enables scalable medium access control, robust routing, and coordinate-free localization. In the harsh underwater environment, however, an underwater sensor node is vulnerable to failures or physical damages, which may affect normal network operation. In particular, a single cluster-head failure can result in the loss of connectivity of all affected cluster members and thus disrupt the operation of the whole cluster. To address this problem, re-clustering can be employed to recover the failed cluster. However, such a recovery mechanism is usually time-insensitive. Furthermore, in the case of a cluster-head failure, sensor nodes in the failed cluster will remain inactive until the next re-clustering is performed. Sensing coverage will be incomplete during the inactive period and thus affect normal network operation

inevitably. To shorten this inactive period, frequent re-clustering is required, which would result in significant control overheads and thus is not desired.

In this chapter, we propose a dependable clustering protocol to provide a robust clustered architecture against cluster head failures in UWSNs. To achieve the objective, the proposed clustering protocol employs two mechanisms: fault prevention clustering and cluster head replication. First, fault prevention clustering attempts to select those healthy nodes as cluster heads to prevent cluster head failures. Then, during clustering, cluster head replication attempts to select a primary cluster head and a backup cluster head for each cluster member so that the constructed cluster hierarchy can tolerate cluster head failures.

3.2 Fault Prevention Clustering

The proposed protocol consists of three phases: failure prediction, cost evaluation, and clustering optimization. The purpose of failure prediction is to predict the potential failure of an underwater sensor based on its lifetime distribution so that those unhealthy nodes are prevented from being selected as cluster heads. Cost evaluation is to evaluate the cost caused by the failure of a sensor node if the node is selected as a cluster head, taking into account both the reliability and residual energy status of the node. Clustering optimization aims at constructing a cluster hierarchy that minimizes the overall cost of all selected clusters based on the cost evaluation of each sensor node.

3.2.1 Failure Prediction

The purpose of failure prediction is to predict the potential failure of a sensor node based on its lifetime distribution so that those unhealthy nodes are prevented from

being selected as cluster heads during clustering. Unlike the sensor nodes used in terrestrial sensor networks, underwater sensor nodes are usually complex and expensive systems equipped with various electronic and mechanical devices. These components follow the progression of degradation of some parameter(s). Accordingly, parameter monitoring can be performed to predict the potential failure of a sensor node or the probability that a sensor node is going to fail within a certain period of time. This probability multiplied by the cost of such a failure provides us a criterion for selecting a cluster head.

The reliability of a sensor node can be represented by a lifetime-distribution function, which is defined as the probability that the actual lifetime of a sensor node is smaller than a given time, i.e.,

$$F(t) = \Pr(T < t), \tag{3.1}$$

where t is a given time, T is the actual lifetime of a sensor node. For the sake of simplicity and without loss of generality, we assume that the lifetime distribution of each sensor node is identical and follows the *Weibull* distribution [49], which is a popular statistical model used in reliability engineering and failure analysis. Its cumulative distribution function is given by

$$F(t; k, \lambda) = 1 - e^{-(t/\lambda)^k}, \tag{3.2}$$

where $k > 0$ is the shape parameter and $\lambda > 0$ is the scale parameter of the distribution. We choose $k > 1$ to indicate "wear out", i.e., a sensor node is more likely to fail as time progresses.

3.2.2 Cost Evaluation

The cost of a cluster-head failure depends on the cluster members associated with that failed cluster head. In the event of a cluster-head failure, if a cluster member in a failed cluster is located in the overlap area with an adjacent cluster, it can resume connectivity and short range communication by joining in the adjacent cluster. Otherwise, it has to communicate with the UW-sink directly, which would cause much more energy consumption or higher cost. For this reason, we consider the worst-case scenario as the cost of a cluster-head failure to stress the reliability of the whole network, which is defined as the sum of the energy consumption of all cluster members in a failed cluster for transmitting one data packet directly to the UW-sink. Since we use the worst-case cost as the cost of a cluster-head failure, we also call it *dependable cluster cost*. To achieve high network robustness, a cluster hierarchy with a minimum sum of *dependable cluster costs* of all clusters is preferred during the clustering process.

- *Dependable Cluster Cost*

Given a cluster $A := B_v \bigcup \{v\}$, where v is a cluster head candidate, N_v is a cluster member set of v, the *dependable cluster cost* of cluster A is defined as

$$\begin{aligned} cost(A,t) = [1 - F(t)] \textstyle\sum_{u \in B_v} [cost(u,v) + cost(v,s)] \\ + F(t) \textstyle\sum_{u \in B_v} cost(u,s), \end{aligned} \tag{3.3}$$

where $\sum_{u \in B_v} cost(u,v)$ is the total energy consumed by all sensor nodes in cluster A for sending one data packet to the cluster head v. $cost(v,s)$ (or $cost(u,s)$) is the energy consumed by v (or u) for sending one data packet to the UW-sink s directly, and E_v(or E_u) is the residual energy of candidate v (or cluster member u) normalized by the initial energy of the node. From Equation (3.3), it is easy to see that the

dependable cluster cost is actually the expectation value of the energy consumption of a cluster.

- *Modified Dependable Cluster Cost*

The *dependable cluster cost* defined in Equation (3.3) only considers the reliability of each sensor node. As indicated earlier, a clustering protocol should also consider the residual energy status of each sensor node in selecting cluster heads in order to support long-term seafloor monitoring. Therefore, the residual energy status of each sensor node should also be taken into account in the *dependable cluster cost*, which can be further modified as

$$\begin{aligned} cost(A,t) = [1 - F(t)]\frac{\sum_{u \in B_v} cost(u,v) f[E_u(t)] + cost(v,s)}{f[E_v(t)]} \\ + F(t) \sum_{u \in B_v} cost(u,s) \end{aligned} \tag{3.4}$$

and

$$f(x) = \begin{cases} \frac{0.9}{1+(E_t/x)^{30}} + 0.1, & x \geq E_{th} \\ \frac{0.9}{1+(E_t/x)^{2}} + 0.1, & x < E_{th}, \end{cases} \tag{3.5}$$

where E_{th} $(0 < E_{th} < 1)$ is a threshold for the residual energy. When the normalized residual energy of a node is larger than this threshold (e.g., $x \geq E_{th}$), a node is in a high energy state. Otherwise, it is in a low energy state (e.g., $x < E_{th}$). $f(E_u(t))$ and $f(E_v(t))$ are the energy functions of a node u and its corresponding cluster head candidate v, respectively.

- *Properties of Dependable Cluster Cost*

By using the modified *dependable cluster cost*, those low-energy nodes wil tend to become cluster members during the clustering process. As a result, they only need to

transmit data over a short distance to its associated cluster heads, thus consuming less energy. At the same time, the current energy converter equipped in each sensor node will recharge and make the node restore its high energy state and its qualification to become a cluster head in the future. In Equation (3.4), the energy function $f[E_v(t)]$ can make the cluster head candidate v have a higher cost and thus become less qualified to be selected as a cluster head. This is because the cluster-head candidate v in a low energy state (e.g., $E_v(t) < E_{th}$) will force $f[E_v(t)]$ to tend to be infinite small, thus gradually leading to an infinite large dependable cluster cost. On the other hand, $f[E_u(t)]$ can make a node u in a low energy state tend to become a cluster member. This is because a node in a low energy state (e.g., $E_u(t) < E_{th}$) will force its corresponding cluster head v to have a lower dependable cluster cost due to smaller $cost(u, v)f_1[E_u(t)]$, thus making v tend to include node u as a cluster member.

3.2.3 Optimal Clustering

The optimal clustering problem is to construct a cluster hierarchy that minimizes the overall *dependable cluster cost* in the whole network. Given a network consisting of a finite set of sensor nodes V, every sensor node in the network is initially a cluster head candidate. We assume that the cluster diameter of each candidate is fixed, limited, and identical. The sensor nodes within the cluster diameter of a candidate v form a finite point set N_v with the cardinality of $|N_v|$, where N_v is called the neighbor set of candidate v. The power set of N_v, denoted by $P(N_v)$, is a set whose elements are the subsets of N_v and $P(N_v)$ constitutes all possible combinations of nodes in Nv. Thus, the cardinality of $P(N_v)$ is $2^{|N_v|}$. Each element of $P(N_v)$ is called a cluster member set of candidate v, which is denoted by N_v. Thus, a candidate v, combined with each

cluster member set $B_v \in P(N_v)$, can form a potential cluster $A := B_v \bigcup v$, and the total number of potential clusters generated by candidate v is $2^{|N_v|}$. Obviously, there initially exist a total number of $\sum_{v \in V} 2^{|N_v|}$ potential clusters in the network, which are generated by all cluster head candidates in V. We use W to denote the cluster set which consists of all the potential clusters in the network.

Given the above assumptions, the optimal clustering at time t can be formulated into a optimization problem with an objective to select a set of potential clusters C from the cluster set W to cover the whole network so that the overall *dependable cluster cost* of all selected clusters is minimized, i.e.,

$$C^* = \arg \min_{C \in W} \sum_{A \in C} cost(A, t), \tag{3.6}$$

where $\bigcup_{A \in C} A = V$ and $\bigcap_{A \in C} A = \phi$. $cost(A, t)$ is the *dependable cluster cost.*

This optimal clustering problem is very similar to the minimum-cost node clustering (MCNC) problem introduced in Chapter 2. The difference between the two problems is the different cost metric applied. To solve the optimal clustering problem, we propose dependable clustering protocol based on the distributed minimum-cost clustering protocol (MCCP) proposed in Chapter 2 to minimize the overall *dependable cluster cost.* We also refer to this protocol as Fault Prevention Clustering.

3.3 Cluster Head Replication

3.3.1 Problem Statement

Fault Prevention Clustering is a proactive fault-tolerant mechanism which is aiming to prevent cluster heads from failing before failures really happen. However, the unpredictable factors in hash underwater environment may lead to unexpected death of

cluster heads. To solve this problem, we adopt the cluster head replication scheme, which is to construct a cluster hierarchy with each cluster member covered by two different cluster heads. This problem is similar to the *domatic* partition problem in graph theory [50]. A *domatic* partition is a partition of vertices in which each part is a dominating set. With the *domatic* partition, each vertex in the graph is either in the dominating set or has a neighbor in the set. In a clustered network, each sensor node is either a cluster head or a cluster member. Therefore, all cluster heads actually constructs *a dominating set.* The *domatic* partition can generate several different *dominating sets.* Correspondingly, a solution to the *domatic* partition problem can also partition the network such that each cluster member can be covered by several different sets of cluster heads. The *domatic* partition problem is a well-known NP-complete problem. A direct solution to this problem is to greedily select small dominating sets and iteratively remove the selected dominating sets from the graph until the remainder is no longer dominating [50]. Based on this greedy algorithm, we propose a Fault-tolerant protocol to construct a cluster hierarchy with each cluster member covered by two different cluster heads, which is described in the next section.

3.3.2 Fault-tolerant Protocol

The Fault-tolerant protocol is based on the greedy algorithm in [50] for solving the *domatic* partition problem. To select two cluster heads for each cluster member, it first selects a set of primary cluster heads (*i.e.*, a dominating set) to cover the whole network. Then it removes the selected cluster heads and selects another set of cluster heads from the remaining sensor nodes as backup cluster heads.

The proposed protocol has two major differences from the greedy algorithm. First, it does not need to select a small dominating set or a small number of cluster heads

to cover the whole network. Instead, a set of cluster heads with higher energy are selected. The purpose of selecting small dominating sets in the greedy algorithm is to guarantee that multiple different sets of cluster heads can be found so that each cluster member can be associated with multiple backup cluster heads. With this constraint, however, energy efficiency cannot be considered in clustering. For UWSNs, a sensor node is usually designed for extreme environment and has higher reliability. To reduce computational complexity and control overhead, one backup cluster head is usually acceptable. For this reason, it is unnecessary to select small dominating sets during clustering, which makes it possible to consider energy efficiency without affecting the selection of backup cluster heads.

Second, the greedy algorithm needs to find a new dominating set or a set of backup cluster heads which not only cover the remaining cluster members but also cover the primary cluster heads. In contrast, the proposed protocol only requires the selected backup cluster heads to cover the remaining cluster members because the backup cluster heads are only activated when the primary cluster heads fail.

3.4 Dependable Clustering Protocol

The dependable clustering protocol combines the fault prevention clustering protocol and 2-fault-tolerant protocol. The protocol procedures can be divided into three phases: initialization phase, clustering phase, and finalization phase.

3.4.1 Initialization Phase

Each node could be in one of the following three states: cluster head (denoted by *head*), cluster member (denoted by *memb*) and cluster head candidate (denoted by *cand*). Initially, every node is a cluster-head candidate, and thus is in the *cand* state.

To begin with, each node performs local topology discovery to find out its one-hop neighbors and maintains an uncovered neighbor set, which contains its one-hop neighbors still in the *cand* state. A candidate can potentially generate a number of different clusters by combining different nodes in the uncovered neighbor set. If a candidate v has a uncovered neighbor set U_v, it can generate $2^{|U_v|}$ potential clusters. Among all potential clusters, a candidate selects a cluster as its qualified cluster with minimum average cost. The cost assigned to a potential cluster is the dependable cluster cost, an indicator of robustness and energy efficiency of the whole cluster.

3.4.2 Clustering Phase

A candidate generates its potential clusters by searching every possible combination of elements in its uncovered neighbor set, selects a qualified cluster from these potential clusters, and sends the average cost of the representative cluster to all candidates within its 2-hop range. A candidate collects the average costs of the qualified clusters of all candidates within its 2-hop range. If the candidate itself has a minimum average cost, it becomes a cluster head and advertises an *INVITE* message to all the nodes in its qualified cluster to invite them become its cluster members. Otherwise, there are two cases for a candidate.

1. If an *INVITE* message is received and the destination of this message is the candidate, the candidate first changes its candidate status to a cluster member. Then it extracts the cluster head ID from the *INVITE* message, mark this node with the extracted ID as its primary cluster head, add the ID into its cluster head (CH) list, and broadcasts a *JOIN* message to all the nodes within its cluster diameter. This *JOIN* message will acknowledge the receipt of the *INVITE* message to the candidate and at the same time notify the other candidates

within the cluster diameter that the candidate has become a cluster member of some cluster head.

2. If no *INVITE* message is received or some *INVITE* messages for other nodes are received, the candidate stays in its candidate status and reselects its qualified cluster because some elements in its uncovered neighbor set might have been covered by some cluster heads or have become cluster heads. Meanwhile, after a node becomes a cluster member, it will keep monitoring the *INVITE* messages for other nodes and add the senders's IDs of these messages into its CH list.

The above procedures are performed by all candidates until each of them becomes either a cluster head or a cluster member. At the end, no candidate is left in the network and each cluster member is associated with a primary cluster head, which is contained in the CH list. Meanwhile, the CH list may also contain one or more cluster heads whose coverage areas intersect with that of the primary cluster head.

3.4.3 Finalization Phase

The CH list generated in the clustering phase may contain two or more cluster heads, which however is not guaranteed. In order to guarantee that the CH list of each cluster member contains two elements, all cluster members return to the *cand* state and the clustering protocol is performed one more time. As a result, each cluster member is covered by a new primary cluster head or becomes a cluster head itself. Thus, a CH list containing at least one element is generated for each cluster member and by combining the two CH lists the final CH list contains at least two cluster heads. For each cluster member, the cluster head in its backup CH list with the minimum distance to the cluster member is selected as the backup cluster head, where the backup CH list is a set excluding the primary cluster head from the CH list.

DEPENDABLE CLUSTERING PROTOCOL(v)

1 $status(v) \in \{head, cand, memb\}$; $status(v) \leftarrow cand$; $CH(v) \leftarrow \phi$
2 **send** and **receive** $status(v)$ within 2 hops
3 $U_v \leftarrow \{u \| status(u) = cand, u \in N_v\}$
4 **while** $status(u) = cand$
5 **do** $C_v \leftarrow P(U_v)$; $B_v \leftarrow \arg\min_{B \in C_v} \{\frac{cost(\{v\} \bigcup B)}{|\{v\} \bigcup B|}\}$
6 $A_v \leftarrow \{v\} \bigcup B_v$; $\arg(v) \leftarrow \frac{cost(A_v)}{|A_v|}$
7 **send** $avg(v)$ within 2 hops
8 $G \leftarrow \{u | avg(v) sentbyuisreceived\}$
9 **if** $avg(v) = \min_{u \in G}\{avg(u)\}$
10 **then** $status(v) \leftarrow head$
11 **send** $INVITE(v, X_v)$ within 1 hop
12 **else** wait until selection timeout
13 **if** v **receives** $INVITE(u, X_u)$
14 **then if** $v \in X_u$
15 **then** $status(v) \leftarrow memb$; $clusterhead(v) \leftarrow u$
16 $CH(v) \leftarrow \{v\} \bigcup CH(v)$; **send** $JOIN(v, u)$
17 **else** $status(v) \leftarrow cand$; $U_v \leftarrow U_v - \{u\}$
18 **elseif** v **receives** $JOIN(u, w)$)
19 **then** $status(v) \leftarrow cand$; $U_v \leftarrow U_v - \{u\}$
20 wait until clustering timeout
21 **if** $status(u) = memb$
22 **then** $CH\_primary(v) \leftarrow clusterhead(v)$; $CH\_old(v) \leftarrow CH(v)$
23 repeat 1- 24
24 $CH\_SET(v) \leftarrow CH(v) \bigcup CH\_old(v)$
25 $Backup\_SET(v) \leftarrow CH\_SET(v) - \{CH\_primary(v)\}$
26 $backup(v) \leftarrow \arg \min_{u \in Backup\_SET(v)} \{dist(v, u)\}$

Figure 3.1: Dependable clustering protocol.

The pseudo code of Dependable Clustering Protocol is given in Figure 3.1, $CH(v)$ is the CH list of candidate v. A_v is the qualified cluster of candidate v. U_v is the uncovered neighbor set of candidate v. $P(U_v)$, called power set of U_v, is a set whose elements are the subsets of U_v and constitute all possible combinations of nodes in U_v. $avg(v)$ is the average cost of A_v. X_v is a set containing the cluster members of A_v. *head* is a flag indicating a cluster head. *cand* is a flag indicating a candidate. *memb* is a flag indicating a cluster member. G is a set containing the average costs sent by other cluster heads within 2-hop range of a candidate, $INVITE(v, X_v)$ is a message inviting the nodes in set X_v to become the cluster members of candidate v, $JOIN(v,u)$ is a message acknowledging that node v has received the *INVITE* message sent by candidate u and joined the cluster as a cluster member of candidate u.

3.4.4 Computational Complexity

Consider a network consisting of $|V|$sensors uniformly distributed over a region with a predefined node density ρ to guarantee the sensing coverage. The computational complexity of dependable clustering protocol is $O(|V|)$. This is because the proposed protocol consists two consecutive steps. It first selects a set of primary cluster heads to cover the whole network. Then it removes the selected cluster heads and selects another set of cluster heads from the remaining sensor nodes as backup cluster heads. During each step, the clustering protocol MCCP is employed, which is introduced in Chapter 2. Since the computational complexity of MCCP is $O(|V|)$, which is discussed in Chapter 2.3.3, the computational complexity of dependable clustering protocol is also $O(|V|)$.

3.5 Performance Evaluation

In this section, we evaluate the performance of the proposed dependable clustering protocol using the network simulator ns-2.

The network robustness is defined as the ratio of the number of working sensor nodes to the total number of sensor nodes which still have battery energy. The working nodes do not include the failed cluster heads and the cluster members which lost the connectivity with the data sink due to the failed cluster heads. We use the reliability function of sensor nodes defined in Equation (3.2) to model the node failures over time. The shape parameter k in the reliability function is set to be 4 to suggest hardware degradation. The scale parameter λ is randomly selected within the range form 200 to 600 to simulate the different "wear out" processes of different sensor nodes. In investigating the network robustness, we only look at the first 35 clustering rounds. This allows for the isolation of the effect of node degradation failures from that of energy depletion.

Figure 3.2 shows the network robustness with the proposed protocol and HEED, respectively. As expected, the proposed protocol achieves better network robustness. This is because the proposed protocol considers not only energy efficiency of sensor nodes but also potential node failures due to non-battery factors while HEED is failure insensitive protocol which does not consider failure prevention during clustering.

Figure 3.3 shows the network capacity with the proposed protocol and HEED, respectively, which is defined as the total amount of data transmitted in the whole network within a certain period of time and is a performance metric affected by network robustness. It is seen that the proposed protocol can achieve better network capacity. This is because the proposed protocol can achieve better network robustness, which means a smaller number of sensor nodes are affected by the failed cluster

heads and lose connectivity with the data sink, thus leading to a larger amount of data transmitted.

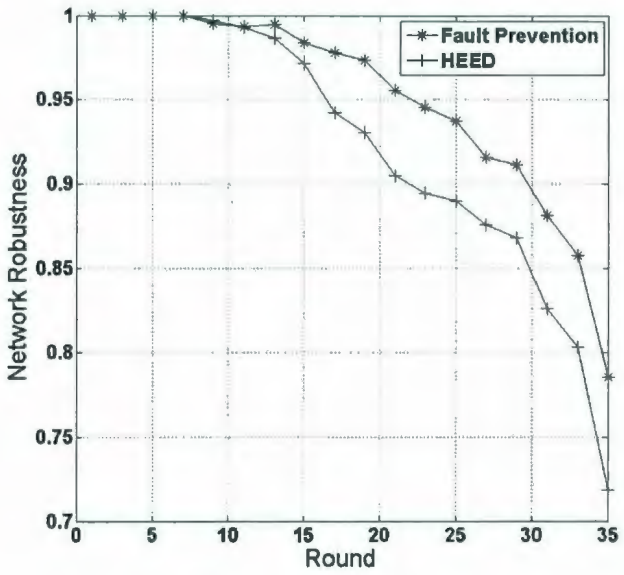


Figure 3.2: Network robustness.

Figure 3.4 shows the clustering properties by examining the maximum and average number of active cluster heads covering each cluster member. The active cluster heads include the primary cluster head, backup cluster head, and other cluster heads in each cluster member's CH list. As expected, the minimum number of active cluster heads of each cluster member is two. Meanwhile, some cluster members can be covered by multiple active cluster heads. Since the average number of active cluster heads exceeds three, the clustered architecture is expected to be survivable to support long-term underwater environmental monitoring.

Next, we compare the proposed protocol with MCCP in term of the total data received by the data sink during the first 5 simulation rounds. This allows for the isolation of the effect of cluster-head failures from that of energy depletion. We assume

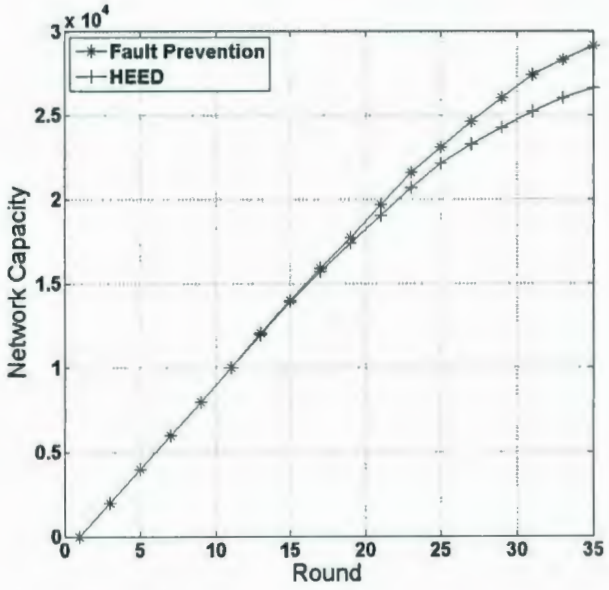


Figure 3.3: Network capability.

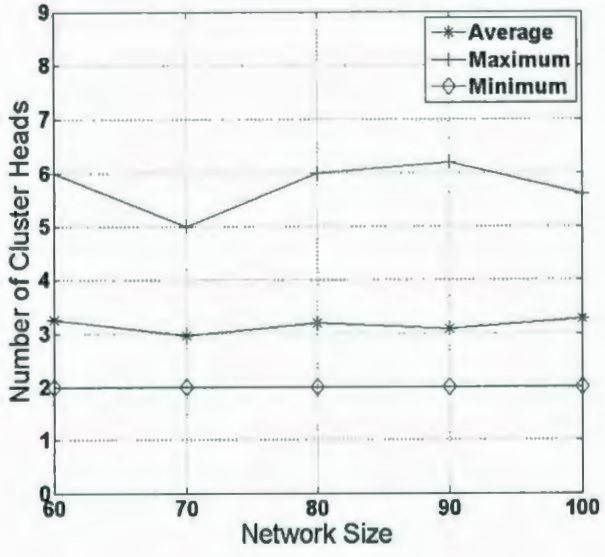


Figure 3.4: Number of cluster heads.

that at the beginning of every TDM frame interval, 4 cluster heads fail. The failed cluster heads are randomly selected from the currently primary cluster heads. Figure 3.5 shows the total data received by the sink with using the proposal protocol and MCCP, respectively. As expected, the proposed protocol outperforms MCCP. This is because using MCCP, the nodes associated with a failed cluster head have to wait until after re-clustering is performed to transmit their sensed data. In contrast, the proposed protocol allows the sensor nodes to quickly switch to the cluster with the backup cluster head. More importantly, since normally a cluster member is covered by more than two active cluster heads as seen in Figure 3.4, a sensor node will always find additional cluster heads even though the backup one fails.

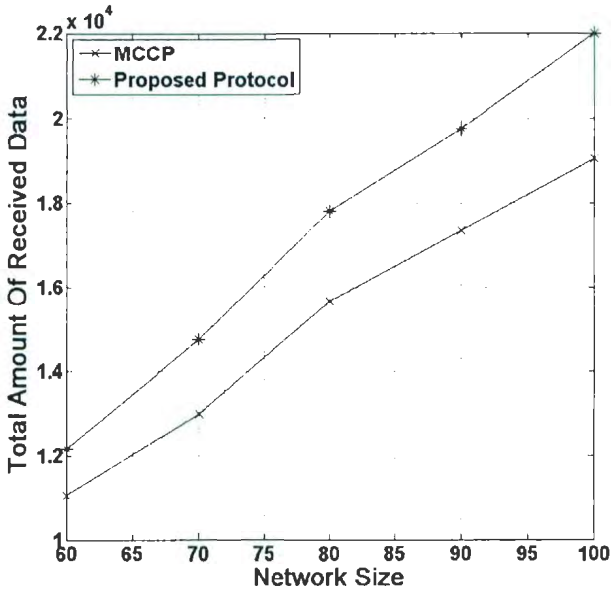


Figure 3.5: Network capacity

3.6 Summary

In this chapter, we have proposed a dependable clustering protocol to provide a robust cluster hierarchy against cluster-head failures in UWSNs. The proposed clustering protocol attempts to select those healthy nodes as cluster heads to prevent cluster head failures. Meanwhile, it attempts to select a primary cluster head and a backup cluster head during clustering so that the cluster members associated with the failed cluster head can quickly switch over to the backup cluster head in the event of a cluster-head failure. The simulation results have shown that the protocol can effectively enhance network robustness.

Chapter 4

Cooperative Fault Detection Mechanism

4.1 Introduction

Fault detection is a prerequisite for recovering a failed cluster in the event of a cluster-head failure. In a clustered network, each cluster member can independently detect the failure of its cluster head by checking the heartbeats periodically sent by the cluster head [15]. Due to the channel uncertainty or signal interference in the harsh underwater environment, however, a sensor node may mistakenly detect a cluster-head failure, which would unnecessarily trigger a fault recovery process and thus waste a considerable amount of energy in sensor nodes. To avoid such energy waste, it is important to accurately detect the failure of a cluster head. However, most previous work was focused on the recovery of a faulty cluster. The accuracy problem in detecting a cluster-head failure has not been well addressed [12, 14, 16]. Some proposed fault detection mechanisms such as the gossiping-based fault detection [17] are not suitable for underwater applications because a gossiping-based detection mechanism

is usually based on a reliable and propagation delay negligible medium, and would incur severe contention and congestion, and thus an unbounded delay.

In this chapter, we propose a cooperative fault detection mechanism for accurately and quickly detecting cluster-head failures in a clustered UWSN. The proposed detection mechanism runs concurrently with normal network operation by periodically performing a detection process at each cluster member. To increase detection accuracy, it allows each cluster member to independently detect the fault status of its cluster head and then employs a distributed agreement protocol to reach an agreement on the fault status of the cluster head among multiple cluster members. To reduce energy consumption, it uses a time division multiple access (TDMA) medium access control (MAC) protocol and makes use of the data periodically sent by a cluster head as the heartbeats for fault detection. A couple of forward and backward time-division-multiplexing (TDM) frames are specially structured for enabling multiple cluster members to reach an agreement within two frames in each detection process. Moreover, a schedule generation algorithm is also proposed for a cluster head to generate the transmission schedule in the forward and backward frames. Through simulation results, we show that the proposed detection mechanism can achieve high detection accuracy under high packet loss rates in the harsh underwater environment, and can detect a cluster-head failure faster than a traditional fault-detection mechanism with a delay bound of two TDM frames.

4.2 Network Architecture

A UWSN typically consists of several underwater sinks located at the centers of different monitored areas, a number of ocean bottom sensor nodes surrounding each uw-sink, and a surface station providing a link to an on-shore control center, as was

shown in Figure 1.2. A uw-sink usually has an adequate power supply and is capable of handling multiple parallel communications with the sensor nodes. All sensor nodes are homogeneous and quasi-stationary. Each of them can adjust its transmission range with transmission power control. Unlike terrestrial sensor networks, a UWSN has some unique characteristics, such as highly limited bandwidth, long propagation delay, harsh geographical environment, and relatively small network size [4]. Without loss of generality, we consider a clustered network with only one fixed uw-sink. We assume that the network is clustered into a set of clusters by performing a distributed clustering protocol, which always selects the healthiest sensors (*i.e.*, with the largest residual energy) as cluster heads and periodically re-clusters the network. Moreover, the network uses TDMA for medium access control, which is energy efficient and delay guaranteed. With the TDMA MAC protocol, time is divided into a series of TDM frames of equal size for each cluster. Each frame is further divided into a fixed number of timeslots. In each frame, the time slots are numbered from *0* to *n-1*, where n is the number of nodes in the cluster and is also called frame size. The nodes in each cluster are synchronized on a timeslot basis by using the time synchronization technique for an underwater environment proposed in [17] such that the nodes can transmit successively in their own timeslots in consecutive TDM frames. The cluster head of each cluster is responsible for allocating timeslots, generating the TDM schedule, and distributing the schedule to each cluster member. The cluster head reserves timeslot 0 for itself to distribute control information (*e.g.*, the TDM schedule) and transmit data packets.

4.3 Cooperative Fault Detection Mechanism

In a TDMA-based clustered network, each cluster member can detect the fault status of its cluster head by checking the heartbeats periodically sent by the cluster head. Due to the channel uncertainty or signal interference, however, the heartbeat signals may be corrupted during transmission, which would result in a sensor node to mistakenly detect the failure of the cluster head. In this case, a fault recovery process would be unnecessarily triggered, which would waste a significant amount of energy in sensor nodes. To address this problem, we can allow each cluster member to independently detect the fault status of the cluster head and then enable multiple cluster members to reach an agreement about the fault status of the cluster head through some control protocol. Only when the agreement is reached will a decision be made and a recovery process triggered. Based on this idea, we propose a cooperative fault detection mechanism for accurately detecting the failure of a cluster head in a cluster-based UWSN.

4.3.1 Fault Detection Mechanism

The proposed fault detection mechanism requires each cluster member in a cluster to maintain a status vector, in which each bit corresponds to a cluster member and is initialized to zero. Once a cluster member detects that the cluster head has failed, it will set its corresponding bit. Meanwhile, it will update the other bits in its vector as soon as it overhears a status vector from the other nodes. If multiple or a predefined number of bits in the status vector of a cluster member become "1", an agreement is considered being reached and the cluster member will conclude that its cluster head has failed. Note that in order to accommodate a cluster member failure, an agreement is supposed to be reached among multiple cluster members, rather than all cluster

members. Otherwise, an agreement may never be able to be reached because of a single cluster member failure.

The detection mechanism runs concurrently with normal network operation by periodically performing a detection process at each cluster member. The period of performing the detection process can range from one TDM frame to multiple TDM frames. In general, a shorter period can detect a failure faster but would cause more energy consumption, while a longer period can reduce energy consumption but would increase the delay in detecting a failure. The choice of the period depends on the network environment and empirical data. For example, for a military application such as underwater battlefield surveillance, a shorter period is more favored in order to timely detect and recover the vulnerable cluster heads. For a civilian application such as underwater environmental monitoring, a longer period is more favored in order to reduce energy consumption.

The detection process consists of two TDM frames and can be divided into two phases: the detection phase and the agreement phase.

In the first phase, each cluster member independently detects the fault status of the cluster head by checking the heartbeats periodically sent by the cluster head. Since the network uses TDMA for MAC, a cluster head is responsible for allocating timeslots for each cluster member and periodically transmits its data during its own timeslot in each frame. Accordingly, the data that are periodically sent by the cluster head can be used as the heartbeats for fault detection. This passive fault detection technique can avoid additional energy consumption caused by actively sending heartbeats. To check the heartbeats, each cluster member must keep awake during the timeslot of the cluster head.

In the second phase, multiple cluster members cooperate to reach an agreement on the fault status of the cluster head based on the independently detected status

by each cluster member. To reach an agreement, gossiping is a traditional technique to exchange the information that each cluster member maintains individually on the liveliness of the cluster head. However, this technique would cause severe contention and congestion, and may interrupt normal data transmission to the cluster head for an unbounded delay. In particular, the contention-based MAC protocols (*e.g.*, IEEE 802.11) that enable gossiping are impractical in an underwater environment [4]. To address these problems, we propose a distributed agreement protocol, which uses a contention-free transmission schedule based on the TDMA MAC protocol proposed for UWSNs [8, 51], thus guaranteeing a bounded detection delay while not interrupting normal data transmission between the cluster head and its cluster members.

4.3.2 Distributed Agreement Protocol

The distributed agreement protocol uses a couple of consecutive TDM frames to reach an agreement on the fault status of a cluster head among multiple cluster members. One of the frames is called forward frame and the other is called backward frame. Both forward and backward frames have the same frame size with the first timeslot reserved for the cluster head and the others allocated to cluster members. What makes them different is that the transmission order of cluster members in the backward frame is reversed with respect to that in the forward frame. Moreover, the transmission order of cluster members is scheduled in a local hierarchy so that after two consecutive forward and backward frames an agreement can be reached among multiple cluster members.

To better understand the above concepts, let us take a look at a simple example. Consider a cluster with eight cluster members denoted by $p1$, $p2$, ..., $p8$, as shown in Figure 4.1(a). For ease of exposition, we assume that a cluster head failure is detected

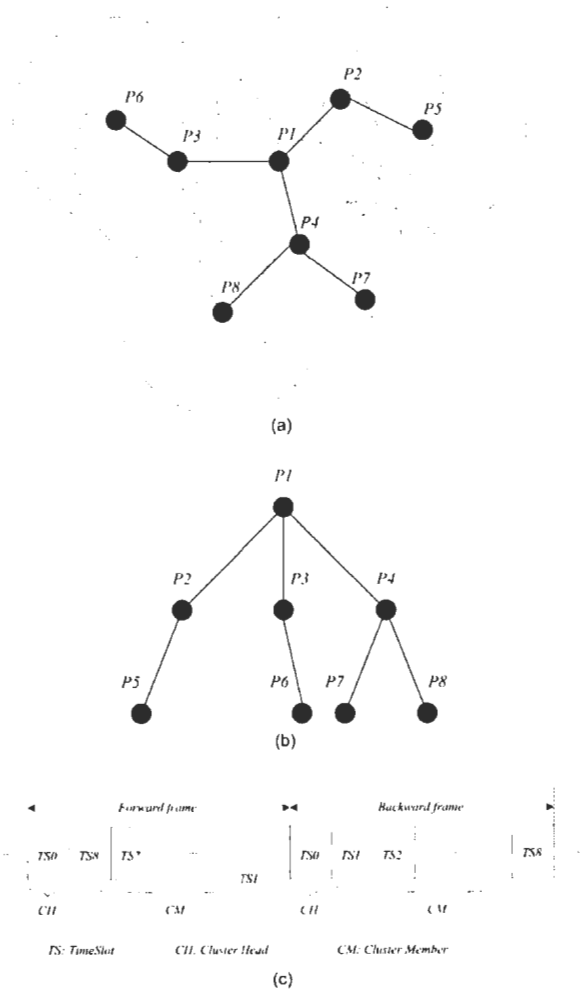


Figure 4.1: Example of distributed agreement protocol: (a) Cluster member distribution; (b) Spanning tree; (c) Frame structures.

only when an agreement among all the cluster members is reached. Suppose that cluster member *p1* is determined as the root and a broadcasting tree is constructed among these cluster members by the cluster head, as shown in Figure 4.1(b). In each detection process, the forward frame is scheduled before the backward frame. In the forward frame, the first timeslot is reserved for the cluster head for data transmission. During this timeslot, all cluster members must keep awake to check the data as a heartbeat. Accordingly, after the first timeslot, each cluster member can detect the current status of the cluster head. If a cluster member does not detect a fault status, it only needs to send its normal data with its status vector piggybacked on in its own timeslot, which will not cause additional energy consumption for fault detection. Otherwise, it needs to send a particular packet containing the status vector to the cluster head, which will cause additional energy cost.

On the other hand, a parent node (*e.g.*, *p4*) is always scheduled for transmission after its child nodes (*e.g.*, *p7* and *p8*). Since a parent node can overhear the data from its child nodes, it will extract and merge the status vectors with its own vector, piggyback the merged vector on its own data, and transmit the data in its own timeslot. Therefore, if the cluster members transmit in an order based on the tree structure {*p8, p7, ..., p1*} in the forward frame, the root node *p1* can overhear the status vectors of all the other nodes after node *p2* transmits its data in its timeslot (*i.e.*, *TS2*). Then it will extract and merge all received status vectors and transmit the merged vector together with its data in its own timeslot (*i.e.*, *TS1*). Up to this point, if the cluster only has *p2*, *p3*, and *p4*, all of them can overhear the status vector sent by *p1* and can thus judge whether the cluster head has failed as well. However, the cluster still has *p5*, *p6*, and *p7*, which are not within the transmission diameter of *p1* and thus are unable to overhear the merged status vector sent by *p1*. To enable these nodes to receive the merged status vector, the backward frame must be used,

in which the transmission order of cluster members is reversed with respect to the forward frame, *i.e.*, {*p1, p2, ..., p8*}, as shown in Figure 4.1(c). During the backward frame, if a child node receives the status vector from its parent, it will simply send it in its own timeslot and if the child node is a leaf node it will not send the status vector. As a result, after the backward frame, each cluster member can receive the merged status vector sent by the root node *p1* and thus know the fault statuses detected by all the other members. Therefore, an agreement is reached among all the cluster members on whether the cluster head has failed or not. It should be noted that to save energy *p1* does not have to transmit in the backward frame. Also, an agreement does not have to be reached among all the cluster members in order to accommodate cluster member failures.

Now the remaining problem is how to generate the transmission schedule of the cluster head and cluster members in each cluster. For this purpose, we propose a schedule generation algorithm, which will be described in next section.

4.3.3 Schedule Generation Algorithm

We first model a cluster as an undirected graph $Gc=(Vc, Ec)$, in which $Vc=\{p1, p2, \ldots, pn\}$ is the set of cluster members, excluding the cluster head, and an edge $\{pi, pj\} \in Ec$ if and only if pi is a one-hop neighbor of pj and vice versa. This graph can be represented by an adjacency list for future embedded computing. The adjacency list consists of $|Vc|$ lists, one for each cluster member pj, $0<i<|Vc|-1$, which gives the cluster members to which pj is adjacent. The cluster head in each cluster can easily obtain this list via topology discovery performed during clustering, when the lists of one-hop neighbors of each node are exchanged among adjacent nodes.

The schedule generation algorithm is performed by a cluster head to generate the

transmission schedule of all nodes in the cluster once the cluster is constructed. It consists of four consecutive different phases: cluster graph partition, component center determination, broadcasting tree construction, and transmission timeslot allocation.

In a cluster, it is not always possible to find a path between each pair of cluster members, which means that the graph of a cluster *Gc* could be an unconnected graph. An example of an unconnected cluster or graph is shown in Figure 4.2(a). This phenomenon is caused by the network deployment or the dynamical characteristic of the ocean currents, which makes uniform node distribution impossible. For an unconnected graph *Gc*, it must be first decomposed into a set of connected components (or simply components), each of which is a maximal connected subgraph of *Gc*. In Figure 4.2(a), for example, the unconnected graph is decomposed into two connected components *C1* and *C2*, where *C1* consists of nodes (*p1, p2, p3, p4, p5, p6*) and *C2* consists of nodes (*p7, p8, p9*). Once the cluster graph partition is completed, the component center determination, broadcasting tree construction, and transmission timeslot allocation are applied to each component.

In the second phase, the center of a subgraph is determined as the root node for constructing a broadcasting tree in each component. For example, the center of *C1* is *p1* and the center of *C2* is *p7*. The purpose is to increase the transmission reliability of the agreement protocol in the detection process. Due to the channel uncertainty and signal interference, the transmission of the status vector of each cluster member may not be reliable. To increase the reliability, it is desirable to minimize the distance (or the number of hops) from the root node to all other nodes. Since the center of a sub-graph is a vertex whose maximum distance to all other vertices is minimal, the shortest path tree can be constructed with the center of the subgraph as the root node in that component.

In the third phase, the shortest path tree directed from the root node of each

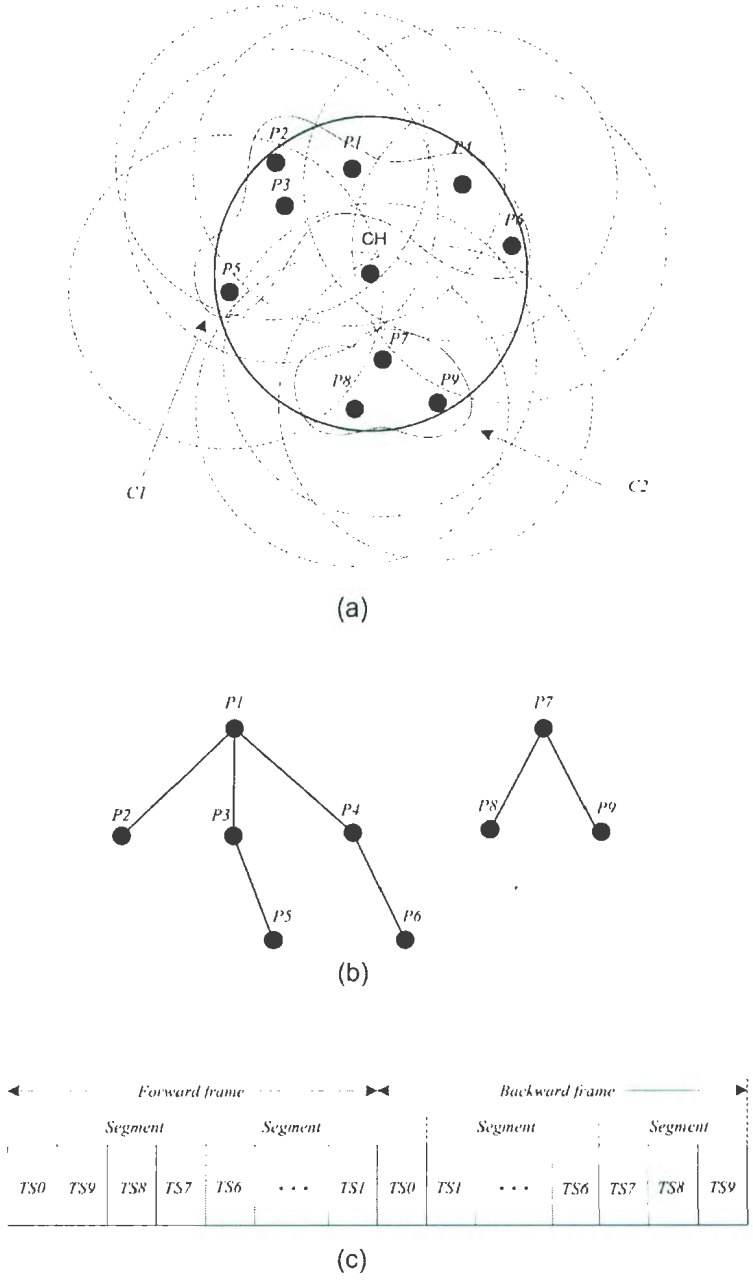


Figure 4.2: Illustration of schedule generation.

component is constructed, which has the minimum height among all the trees that could be constructed from that component, as shown in Figure 4.2(b). Consequently, the constructed broadcasting tree has the maximal lower bound of the transmission reliability.

In the fourth phase, timeslot allocation is performed for each member in a cluster based on the constructed broadcasting trees. In the allocation, both the forward frame and the backward frame can be divided into several segments. Each segment corresponds to a connected subgraph or component, and specifies the transmission order of the cluster members in that component. For each component, timeslot allocation is performed for the nodes at one level a time and at the same level in a specified order, *e.g.*, from right to left in Figure 4.2(b), where a level is defined in terms of the distance from the root of the tree. In each allocation, the smallest available timeslot number among all available timeslots is taken. Based on the above allocation policy, it is guaranteed that the set of nodes on each level transmit data only after receiving the data from all the nodes on an adjacent level. For a forward frame, timeslot allocation starts at the most bottom level of the tree and proceeds upwards while for a backward frame the order is reversed, as shown in Figure 4.2(c). This guarantees that two consecutive TDM frames, a forward frame followed by a backward frame, are sufficient to exchange the fault status of a cluster head among all the cluster members in the cluster. Note that the timeslot allocation for the forward frame is the same as that for the normal frame.

Next we further describe the schedule generation algorithm in more detail. The building block of this algorithm is the Breath First Search (BFS) algorithm [52] because of its following unique and useful characteristics:

1. BFS is a level-by-level traversal through a graph that visits all of the vertices

reachable from a particular source vertex, which means that BFS can return a component containing the given source vertex;

2. The traversal paths generated by BFS actually build a shortest-path tree (SPT) directed from the source vertex to all reachable nodes;

3. The level-by-level traversal of BFS is similar to the procedure of timeslot allocation.

Assume that the cluster head of a cluster initially already has the local cluster topology obtained via topology discovery during clustering, which is represented by an adjacency list. This local topology is then partitioned into a set of connected components by executing a cluster partitioning process based on the BFS algorithm. More specifically, if a cluster consists of a set of connected components C_1, C_2, . . . , C_k, the cluster partitioning process will return a tree corresponding to each C_i. BFS is a level-by-level traversal through a graph that visits all of the vertices reachable from a particular source vertex. In each execution, BFS actually returns a tree, which is a maximal connected subgraph (or a component) of Gc containing the selected source vertex. Given a cluster Gc, the cluster head first randomly selects a cluster member as the source vertex and makes a call to BFS, and then determines if there is any unvisited vertex left. Consequently, a set of components in Gc is obtained by making repeated calls to BFS on the unvisited vertices which have not been covered by a component yet.

For each component, the cluster head then determines the center of the component as the root node. Consider a component C with a set of vertices $\{v_1,v_2,\ldots,v_n\}$. The distance between any two vertices v_i and v_j in C, represented by $d(v_i, v_j)$, is the length of the shortest path in C between v_i and v_j. Hence, a point $x_0 \in \{v_1, v_2, \ldots, v_n\}$ is a center of C if

$$\max_{1\leq i\leq n} d(v_i, x_0) \leq \max_{1\leq i\leq n} d(v_i, x) \forall x \in \{v_1, v_2, \ldots, v_n\}. \quad (4.1)$$

Let Di=$\{d(v_1,v_i), \ldots, d(v_n,v_i)\}$ be a distance vector of v_i and d_i^m be the maximal entry in D_i. Obviously, if $d_c^m = \min(d_1^m, d_2^m, \cdots, d_n^m)$, v_c is the center of the component. To generate D_i for each vertex v_i, the cluster head first allocates a vector D_i=ϕ to each node v_i, and then make a call to BFS with v_i as the source vertex. BFS will first visit all vertices with a distance 1 to v_i, *e.g.*, $\{v_1,\ldots,v_a\}$, $a\leq n$, and set D_i=$\{d(v_1,v_i)$=1, $\ldots$, $d(v_a,v_i)$=1$\}$. Then it will visit all vertices with a distance 2 to v_i, *e.g.*, $\{v_a + 1,\ldots,v_b\}$, $a<b\leq n$, and set D_i=$\{d(v_1,v_i)$=1,$\ldots,d(v_a,v_i)$=1, $d(v_a + 1,v_i)$=2,$\ldots$, $d(v_b,v_i)$=2$\}$. In the same manner, BSF will visit and process the remaining nodes until all nodes in $\{v_1,v_2,\ldots,v_n\}$ have been visited. As a result, a vector D_i=$\{d(v_1,v_i), \ldots, d(v_n,v_i)\}$ is obtained. Since it is well known that BFS actually returns a shortest-path tree with vi as the root, D_i is surely the distance vector of v_i, which consists of the length of the shortest path between the source vertex v_i and each node $\{v_1,v_2,\ldots,v_n\}$.

Meanwhile, timeslot allocation is performed concurrently with the above center determination process. Before the process, a FIFO queue Q_i is allocated for each node $v_i\in\{v_1,v_2,\ldots,v_n\}$ in the component. During the process, BSF is called for each source vertex v_i (a candidate center). Each time BSF visits a node $q_i\in\{ v_1,v_2,\ldots,v_n\}$, it puts q_i into the queue Q_i. As a result, we can obtain an order $Q_i\in\{q_i,\ldots,q_n\}$ upon the completion of the current BSF. After completing the whole center determination process, a queue vector $\{Q_1,\ldots,Q_n\}$ is obtained. If v_c is the center of the component, Q_c actually stores the timeslot order of the nodes $\{v_1,v_2,\ldots,v_n\}$ for the backward frame. By reversing this order, the order for the forward frame can be accordingly obtained. BFS is level-by-level traversal which always visits the nodes at a distance l

to the source node before the nodes at a distance $l+1$. Therefore, the order obtained by running BFS is the order needed for the backward frame. The pseudo code of the algorithm is given in Figure 4.3 and Figure 4.4. Q is a FIFO queue, $level(v)$ is the level of node v, *comp* is a set consisting of the nodes in a component, SCH is a queue indicating the transmission orders of the nodes in a component, *comp_set* is a set containing the components in a cluster, and $center(C)$ is a set containing the components in a cluster.

4.4 Performance Evaluation

In this section, we evaluate the performance of the proposed fault detection mechanism through simulation experiments. We first investigate the performance in terms of detection accuracy under different packet loss rates and cluster sizes. The detection accuracy is described by the probability of false positive, which is defined as the probability that an operational cluster head is mistakenly detected as a faulty one. Also, we compare the proposed detection mechanism with a traditional fault detection mechanism in terms of detection time taken to achieve comparable probability of false positive, where the detection time is defined as the number of TDM frames.

In the simulation, we consider a single cluster and assume that all nodes including the cluster head have the same transmission range. The sensor nodes are uniformly distributed in the region of the cluster. Due to the unreliable underwater communication channel, packet loss exists during transmission, which is described by the packet loss rate. Moreover, we use the same energy model used in [45], which was proposed for underwater acoustic networks.

In the first simulation experiment, we investigate the effects of the packet loss probability and the cluster size on detection accuracy. Due to the complex and

SCHEDULE GENERATION ALGORITHM (V)

1 ▷ Begin Initialization
2 **for** $v \in V$
3 **do** $level(v) \leftarrow \infty$
4 ▷ End Initialization
5 ▷ Begin Cluster Graph Partition
6 $comp\_set \leftarrow \phi$
7 **for** $v \in V$
8 **do if** $level(u) = \infty$
9 **then** $comp\_set \leftarrow comp\_set \cup BSF(u, getcomp)$
10 ▷ End Cluster Graph Partition
11 ▷ Begin Component Center Determination
12 **for** $C \in comp\_set$
13 **do** $center\_candidate \leftarrow \phi$; $candidate\_level \leftarrow \phi$
14 **for** $level(u) = \infty$
15 **do** $levelarray \leftarrow BSF(u, getlevel)$; $v^* \leftarrow \arg \max_{v,\ levelarray} levelarray(v)$
16 $level\_m \leftarrow \max_{v\_levelarray} levelarray(v)$
17 $center\_candidate \leftarrow center\_candidate \cup v^*$
18 $candidatelevel(v^*) \leftarrow level\_m$
19 $center(C) \leftarrow \arg \max_{v,\ center\_candidate} candidate\_level(v)$
20 ▷ End Component center determination
21 ▷ Begin Broadcasting Tree Construction and Transmission Timeslot Allocation
22 $forwardframe \leftarrow \phi$; $backwardframe \leftarrow \phi$
23 **for** $C \in comp\_set$
24 **do** $schedule \leftarrow BSF(center(C), getschedule)$
25 $backwardframe \leftarrow append(backwardframe, schedule)$
26 $forwardframe \leftarrow reverse(backwardframe)$
27 ▷ End Broadcasting Tree Construction and Transmission Timeslot Allocation

Figure 4.3: Schedule Generation Algorithm.

```
BFS(s, option)
 1  Q ← ∅; level(s) ← 0
 2  if option = getcomponent
 3     then COMP ← ∅ ; COMP ← COMP ⋃ {s}
 4  elseif option = getschedule
 5     then SCH ← ∅; enqueue(SCH, s)
 6  enqueue(SCH, s)
 7  if Q ≠ ∅
 8     then u ← dequeue(Q)
 9          for v ∈ adj(u)
10               do if level(v) = ∞
11                     then enqueue(Q, v)
12                          level(v) ← level(u) + 1
13                          if option = getcomponent
14                             then COMP ← COMP ⋃ {v}
15                          elseif option = getschedule
16                             then enqueue(SCH, v)
17  if option = getcomponent
18     then return(COMP)
19  elseif option = getschedule
20     then return(SCH)
21  elseif option = getslevel
22     then return(level)
```

Figure 4.4: BSF.

expensive transceivers equipped on each sensor node for the extreme underwater environment, sensor nodes are usually sparsely deployed in a UWSN. For this reason, we assume that in a cluster there are only a small number of sensor nodes (*e.g.*, 10-20) in the cluster, which are available to cooperatively detect the cluster-head failure. On the other hand, since the proposed detection mechanism aims at achieving accurate fault detection in the hostile underwater environment, we only consider the network scenario where the packet loss possibility is relatively high (*e.g.*, 0.4-0.6), mimicking the unreliable underwater communication.

Figure 4.5 shows the simulation results of the first experiment. It is seen that with the increase of the packet loss rate the probability of false positive increases, which leads to lower detection accuracy. Also, a larger number of sensor nodes lead to a smaller probability of false positive, i.e., higher detection accuracy. Under a high packet loss rate of 0.5, the probability of false positive is below 10^{-4} with a cluster of 15 nodes, which is quite low. As expected, the proposed detection mechanism can achieve high detection accuracy even under high packet loss rates in the harsh and sparse underwater environment.

In the second simulation experiment, we compare the proposed detection mechanism with a traditional detection mechanism in term of detection time under the same detection accuracy or probability of false positive. In the traditional mechanism, a sensor node can independently detect the status of the cluster head by checking the heartbeats periodically sent by the cluster head. Each sensor node can individually make a decision on a failure if it misses several consecutive heartbeats.

Figure 4.6 shows the simulation results of the second experiment with a cluster of 15 nodes. It is seen that the detection time with the traditional mechanism is much larger than that with the proposed mechanism and it decreases non-linearly with the increase of the packet loss rate. The detection time with the proposed

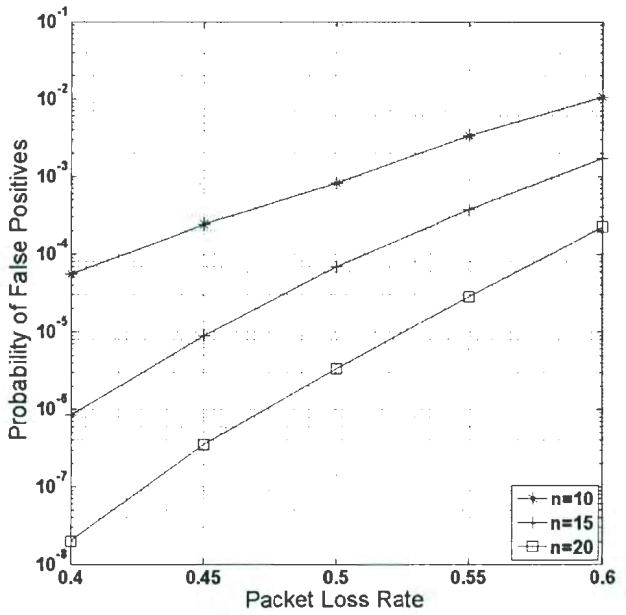


Figure 4.5: Detection accuracy.

mechanism keeps constant at 2 TDM frames, which is much faster than that with the traditional mechanism. Note that the detection time with the traditional mechanism is obtained under the same detection accuracy. The specific values of the probability of false positive are 8×10^{-007}, 8×10^{-006}, 6×10^{-005}, 3×10^{-004}, and 1×10^{-003}, corresponding to the packet loss rates 0.40, 0.45, 0.50, 0.55, and 0.6, respectively.

In the last simulation experiment, we investigate the energy cost of the proposed fault detection mechanism under different cluster sizes and packet loss rates. The energy cost is defined as the average energy consumption of the sensor nodes within a cluster for cooperatively performing a fault detection process during two consecutive TDMA frames.

Figure 4.7 shows the simulation results of the experiment. It is seen that with the decrease of the packet loss rate the energy cost decreases, which is straightforward.

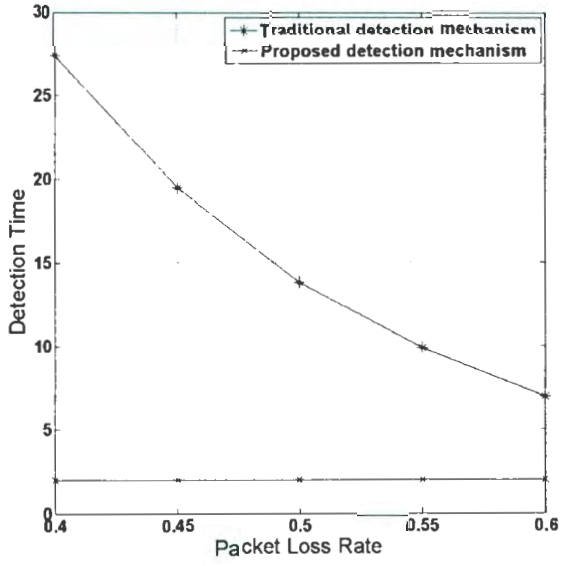


Figure 4.6: Comparison of detection time

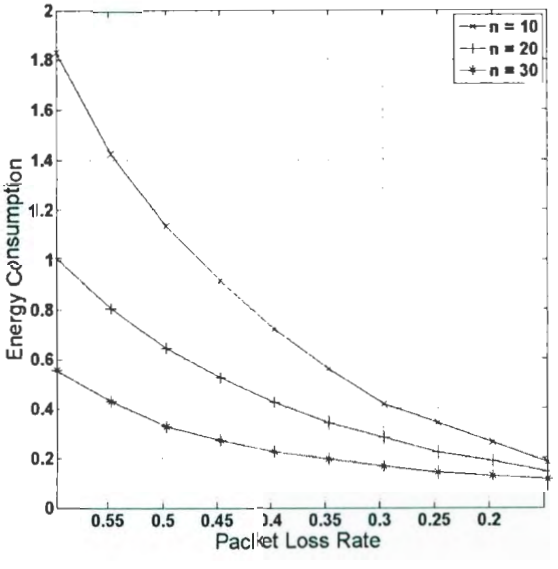


Figure 4.7: Energy consumption.

Also, a larger number of sensor nodes lead to a lower energy cost. This is because during the detection phase if a cluster member detects the failure of the cluster head and does not overhear a different status sent by the other nodes, it needs to send a particular packet containing its vector status to the cluster head, which would consume additional energy. Otherwise, if a cluster member does not detect a fault status, it only needs to send the normal data with its status vector piggybacked on, which would not cause additional energy consumption. The more the cluster members, the lower is the probability that a cluster member sends a particular packet containing its vector status.

4.5 Summary

In this chapter, we have proposed a cooperative fault detection mechanism for accurately and fast detecting cluster-head failures in TDMA-based clustered UWSNs. The proposed mechanism allows each cluster member to independently detect the fault status of its cluster head and then employs a distributed agreement protocol to reach an agreement on the fault status of the cluster head among multiple cluster members. A couple of forward and backward TDM frames are specially structured for enabling multiple cluster members to reach an agreement within two frames in a fault detection process. A schedule generation algorithm has also been proposed for a cluster head to generate the transmission schedule in the forward and backward frame. Simulation results show that the proposed mechanism can achieve high detection accuracy under high packet loss rates in the harsh underwater environment, and can also detect a cluster-head failure faster than the traditional fault detection mechanism with a delay bound of two TDM frames. Moreover, it makes use of the data periodically sent by a cluster head as the heartbeats for fault detection and

uses a couple of specially-structured forward and backward frames in the agreement process, which are energy efficient and do not affect normal network operation.

Chapter 5

Distributed Data Aggregation Using Slepian-Wolf Coding

5.1 Introduction

Wireless sensor networks have many applications which require a dense deployment of a large number of sensor nodes in a field of interest with one or more data sinks located either at the center or out of the field [3]. The sensor nodes observe the phenomenon at different points in the field and send the observed data to the sink(s). The observed phenomenon is usually a spatially dependent continuous process, in which the observed data have a certain spatial correlation. In general, the degree of the spatial correlation in the data increases with the decrease of the separation between sensor nodes. Therefore, spatially proximal sensor observations are usually highly correlated, which leads to considerable data redundancy in the network [18]. To efficiently use network resources to increase energy efficiency in data transmission, it is highly desirable to remove such data redundancy through effective data aggregation techniques.

Slepian-Wolf coding [19, 20] is a distributed source coding technique that can completely remove data redundancy with no need for inter-sensor communication and therefore is a promising technique for data aggregation in a WSN. This technique is based on the assumption that each sensor node has *a priori* knowledge of the correlation structure of the network, which depends on the distances between sensor nodes and the characteristics of the observed phenomenon [20]. However, applying Slepian-Wolf coding globally in the whole network is difficult because each sensor node needs to know the global correlation structure to encode its own data, which would incur significant additional costs. Moreover, Slepian-Wolf coding is not tolerant to communication failures because the data from one node may affect the decoding of the data from other nodes [21]. For these reasons, Slepian-Wolf coding is usually not suitable for being applied globally in a large network.

In a cluster-based network, however, each cluster covers a smaller number of sensor nodes within a smaller local range of the network. This makes it more feasible to apply Slepian-Wolf coding locally within each cluster as in this case a sensor node only needs the knowledge of local correlation structure to perform encoding. Meanwhile, it will not obviously compromise the compression performance because the spatial correlation usually decreases with distance [18, 22]. Despite the promising perspective, many technical issues remain to be studied and resolved in order to put this technique into practical use.

In this chapter, we study the major problems in applying Slepain-Wolf coding for data aggregation in a cluster-based WSN with an objective to optimize data compression so that the total amount of data generated in the whole network is minimized. The first problem is how to cluster the sensor nodes, given the spatial correlation structure of the network, such that the global compression gain of Slepian-Wolf coding is maximized, or in another word, the total rate (bits) of the encoded

data from all clusters is minimized. To solve this problem, we propose a distributed optimal-compression clustering protocol (DOC) based on an approximation algorithm for solving the minimum weight set cover problem in graph theory.

With an optimal cluster hierarchy constructed by DOC, the next problem is how to optimally allocate a rate to each node within a cluster such that the intra-cluster communication cost, which is defined as the total energy consumed by all the sensor nodes in the cluster for sending data encoded with the allocated rates, is minimized. To address this problem, we first prove that there exists an approximation algorithm that can find an optimal rate allocation within each cluster, and then present the procedures to perform Slepian-Wolf coding locally within each cluster with the optimal rate allocation. In addition, we propose a joint clustered Slepian-Wolf coding and explicit entropy coding scheme to further reduce possible correlation in the data generated between different clusters.

5.2 Problem Statements

In this section, we introduce the concept of Slepian-Wolf coding, describe the clustered Slepian-Wolf Coding problem and the optimal intra-cluster rate allocation problem.

5.2.1 Slepian-Wolf Coding

Consider a network consisting of N sensor nodes uniformly distributed in a region of interest, where each node i produces reading X_i and all the readings constitute a set of jointly ergodic sources denoted by $X = (X_1, X_2, \ldots, X_N)$ with distribution $p(x_1, x_2, \ldots, x_N)$, which corresponds to the spatial correlation structure known by each node *a priori*. According to Slepian-Wolf Theorem [19], the nodes can jointly encode their data without inter-node communication, with a rate (in *bits*),$R(U)$,

lower-bounded by their joint entropy $H(X_1, X_2, \ldots, X_N)$ as long as their respective rates are under the constraints given by

$$R(U) \geq H(X(U)|X(U^c)), \tag{5.1}$$

for all $U \subseteq \{1, 2, \cdots, N\}$, where $\{1, 2, \cdots, N\}$ is a set of the indices of sensor nodes in the network, U^c is the complementary set of U, $H(X)$ is the entropy of X, and

$$R(U) = \sum_{i \in U} R_i, \tag{5.2}$$

$$X(U) = \{X_j | j \in U\}. \tag{5.3}$$

For example, consider a simple case of two sensor nodes producing readings X_1 and X_2. Their individual rates should be subject to

$$R_1 \geq H(X_1|X_2), \tag{5.4}$$

$$R_2 \geq H(X_2|X_1), \tag{5.5}$$

$$R_1 + R_2 \geq H(X_1,\ X_2). \tag{5.6}$$

According to chain theory [20], under the above constraints, it is always possible to find a rate allocation for the two nodes, which makes the total rate (bits) of two nodes equal to their joint entropy, i.e.,

$$R_1 + R_2 = H(X_1) + H(X_2|X_1) = H(X_1, X_2). \tag{5.7}$$

Correspondingly, for an arbitrary ordering of N nodes (e.g., in the ascending or descending order of nodes' ID numbers), there exits a rate allocation (vector) $\{R_i\}_{i=1}^{N}$ such that the number of generated bits from all nodes can achieve the value of their joint entropy, e.g.,

$$\sum_{i=1}^{N} R_i = H(X_1, X_2, \cdots, X_N), \tag{5.8}$$

where $R_1 = H(X_1), R_i = H(X_i|X_{i-1}, X_{i-2}, \cdots, X_1), 2 \leq i \leq N.$,

Therefore, a cluster of nodes A can be encoded with $H(X_1, X_2, \cdots, X_{|A|})$ bits using Slepian-Wolf coding without communicating with each other, where $|A|$ is the number of nodes in cluster A, and there always exists an optimal rate allocation to achieve this local maximum compression performance.

5.2.2 Clustered Slepian-Wolf Coding Problem

Consider a network consisting of a finite set of sensor nodes V. Every sensor node in the network is initially a cluster head candidate. We assume that each candidate has an identical cluster diameter within which all other nodes may become its cluster members. The nodes within the cluster diameter of a candidate v form a finite point set Nv with the cardinality of $|N_v|$, which is called the neighbor set of candidate v. The power set of Nv, denoted by $P(N_v)$, is a set whose elements are the subsets of N_v. $P(N_v)$ contains all possible combinations of nodes in $N)_v$. Thus, the cardinality of $P(Nv)$ is $2^{|N_v|}$. Since a candidate v, associated with each combination of nodes (cluster members) within its cluster diameter (e.g., a set of nodes B_v, where $B_v \in P(N_v)$), can form a unique potential cluster (e.g., $A := B_v \bigcup \{v\}$), a candidate v can generate up to $2^{|N_v|}$ potential clusters. Recall that initially every node in the network is a candidate, thus there are a total number of $|V|$candidates. Therefore, there exists

a cluster set S consisting of $\sum_{v \in V} 2^{|N_v|}$ potential clusters in the whole network. Meanwhile, each potential cluster A can be encoded with $H(X_1, X_2, \cdots, X_{|A|})$ bits using Slepian-Wolf coding in that cluster.

With the above assumptions, the clustered Slepian-Wolf coding problem is to select a set of disjoint potential clusters C^* from the cluster set S to cover the whole network such that the global compression gain of Slepian-Wolf coding is maximized, or more specifically, the total rate (bits) of the encoded data generated by all the clusters (or all the nodes) in the network, is minimized, i.e.,

$$C^* = \arg\min_{C \subseteq S} \sum_{A \in C} H(X(A)), \tag{5.9}$$

where $\bigcup_{A \in C^*} A = V$, $\bigcap_{A \in C^*} A = \phi$, and $X(A) = \{X_j | j \in A\}$.

5.2.3 Optimal Intra-Cluster Rate Allocation Problem

Suppose that a cluster hierarchy has already been constructed in the clustered Slepian-Wolf coding problem. Consider a cluster A with $|A|$sensor nodes and let $\{Ri;\ i=1,\ 2,\ \ldots,\ |A|\}$ be a rate vector allocated to the nodes in the cluster. Also, let $d(i,\ 1)$ be the distance between node i and the cluster head v, which is used to estimate the energy consumed by node i for sending one bit of data to the cluster head v because normally transmission energy dissipation is proportional to signal propagation distance. Then the objective of the intra-cluster rate allocation problem is to find a rate vector for the nodes in the cluster under the constraints given by Equation (5.1) such that the total energy consumed by all nodes for sending the data encoded with their individual rates to the cluster head is minimized, i.e.,

$$\{R_i^*\}_{i=1}^{|A|} = \arg \min_{\{R_i\}_{i=1}^{|A|}} \sum_{i=1}^{|A|} d(i,1) R_i \tag{5.10}$$

subject to

$$\sum_{i \in Y} R_i \geq H(X(Y)|X(Y^c)), \forall Y \subseteq \{1, 2, ..., |A|\}, \tag{5.11}$$

where $\{1, 2, \ldots, |A|\}$ is a set of the indices of the sensor nodes in the cluster A. Note that the clustered Slepian-Wolf coding problem assumes that there exists a rate allocation such that the total rate of encoded data in a cluster is equal to the joint entropy of readings or observations. However, a solution to the intra-cluster rate allocation problem considered may generate a rate allocation which is not subject to that assumption. We will prove in Section 5.4 that the presented optimal solution conforms to the assumption in the clustered Slepian-Wolf coding problem.

5.3 Clustering Using Slepian-Wolf Coding

In this section, we present a distributed optimal-compression clustering protocol (DOC) to solve the clustered Slepian-Wolf coding (CSWC) problem. The CSWC problem is very similar to the minimum-cost node clustering (MCNC) problem introduced in Chapter 2. The difference between the two problems is the different cost metric applied. To solve the optimal clustering problem, we propose dependable clustering protocol based on the distributed minimum-cost clustering protocol (MCCP) proposed in Chapter 2 to maximize global compression gain of Slepian-Wolf coding.

In DOC, a candidate generates its potential clusters by searching every possible combination of elements in its uncovered neighbor set, calculates each potential cluster's entropy which only depends on the distances between the nodes in the cluster, selects a representative cluster, and sends the average entropy of the representative cluster to all candidates within its 2-hop range. A candidate collects the average entropies sent by all candidates within its 2-hop range. If the candidate itself has minimum average entropy, it becomes a cluster head and advertises an *INVITE* mes-

sage to all the nodes in its *representative* cluster to invite them become its cluster members. Otherwise, if an *INVITE* message is received by a candidate and the destination of this message is the candidate, the candidate first changes its candidate status to a cluster member. Then it extracts the cluster head ID from the *INVITE* message and broadcasts a *JOIN* message to all the nodes within its cluster diameter. This *JOIN* message will acknowledge the receipt of the *INVITE* message and at the same time notify the other candidates within the cluster diameter that the candidate has become a cluster member of some cluster head. If no *INVITE* message is received or some *INVITE* messages for other nodes are received, the candidate stays in its candidate status and reselects its representative cluster because some elements in its uncovered neighbor set might have been covered by some cluster heads or have become cluster heads. The above procedures are performed by all candidates until each of them becomes either a cluster head or a cluster member. The pseudo-code of the above procedures is described in the Figure 5.1. Where A_v is the *representative* cluster of candidate v, $arg(v)$ is the average cost of A_v, X_v is the set containing the cluster members of A_v, *head* is a flag indicating a cluster head, *cand* is a flag indicating a candidate, *memb* is a flag indicating a cluster member, G is a set containing the average costs sent by other cluster heads within 2-hop range of a candidate, $INVITE(v, X_v)$ is a message inviting the nodes in set X_v to become the cluster members of candidate v, $JOIN(v, u)$ is a message acknowledging that node v received the *INVITE* message sent by candidate u and joins the cluster as a cluster member of candidate u.

5.4 Intra-Cluster Rate Allocation

With an optimal cluster hierarchy constructed by DOC, we now consider the optimal intra-cluster rate allocation problem described in Section 5.2.

```
DOC(v)
1  status(v) ∈ {head, cand, memb}
2  status(v) ← cand
3  send and receive status(v) within 2 hops
4  U_v ← {u || status(u) = cand, u ∈ N_v}
5  while status(u) = cand
6        do
7           C_v ← P(U_v)
8           B_v ← arg min_{B∈C_v} { H(X({v} ⋃ B)) / |{v} ⋃ B| }
9           A_v ← {v} ⋃ B_v
10          arg(v) ← H(X(A_v)) / |A_v|
11          send arg(v) within 2 hops
12          G ← {u | arg(v) sent by u is received}
13          if arg(v) = min_{u∈G}{arg(u)}
14            then status(v) ← head
15                 send INVITE(v, X_v) within 1 hop
16            else
17                 wait until selection timeout
18                 if v receives INVITE(u, X_u)
19                   then if v ∈ X_u
20                          then status(v) ← memb
21                               send JOIN(v, u)
22                          else status(v) ← cand
23                               U_v ← U_v − {u}
24                 elseif v receives JOIN(u, w))
25                   then status(v) ← cand
26                        U_v ← U_v − {u}
```

Figure 5.1: Distributed optimal-compression clustering protocol.

5.4.1 Optimal Intra-Cluster Rate Allocation

The intra-cluster rate allocation problem can be analogized to the global rate allocation problem discussed in [27]. Given a multi-hop sensor network consisting of N nodes, where each node i produces reading Xi and uses the shortest path with weight $e(i,s)$ to reach a common sink s, the global rate allocation problem is to find an optimal rate vector $\{R_i^*\}_{i=1}^N$ for all N nodes so that the total flow cost $\sum_{i=1}^N e(i,s)R_i^*$ under the constraints given by Equation (5.1) is minimized. According to [27], the optimal rate vector is given by

$$\begin{aligned} R_1^* &= H(X_1) \\ R_i^* &= H(X_i|X_{i-1}, X_{i-2}, \cdots, X_1), 2 \leq i \leq N \end{aligned} \tag{5.12}$$

where the nodes (i=1, 2, ..., N) with the observations of $(X_1, X_2, \cdots, X_N)$ are organized in the descending order of the weights of the shortest paths, i.e.,

$$e(1,s) \leq e(2,s) \leq \cdots \leq e(N,s). \tag{5.13}$$

The analogies between these two problems are given as follows:

1. In the intra-cluster rate allocation problem, each cluster is analogous to the whole network in the global rate allocation problem because it performs coding independently of all other clusters.

2. In the intra-cluster rate allocation problem, the cluster head of each cluster can be considered as a local virtual data sink. Thus, the cluster head is analogous to the data sink and each cluster member is analogous to a sensor node in the global rate allocation problem.

3. In the intra-cluster rate allocation problem, the shortest path between a cluster member i and the cluster head v is a single-hop path with a distance $d(i,1)$, which is analogous to the weight $e(i,s)$ of the shortest path between a sensor node i and the data sink s in the global rate allocation problem.

Due to the above analogies, the intra-cluster rate allocation problem can be solved by using the same approximation algorithm for solving the global rate allocation problem. The optimal intra-cluster rate allocation has the same form as Equation (5.12).

Although we can obtain an optimal rate allocation, we still need to prove that this solution is valid, i.e., the optimal rate allocation obtained by Equation (5.12) does not contradict the assumption in the clustered Slepian-Wolf coding problem, where the rate allocation for each cluster must satisfy the condition that the total rate of the coded sensor readings in a cluster (e.g., a whole network in the extreme case) is equal to their joint entropy. According to chain theory, we can easily prove the validity of the solution given by Equation (5.12) if the whole network is considered as a single cluster, i.e.,

$$\sum_{i=1}^{N} R_i^* = H(X_1) + \sum_{i=2}^{N} H(X_i|X_{i-1}, X_{i-2}, \cdots, X_1) = H(X_1, X_2, \cdots, X_N). \quad (5.14)$$

Therefore, let $\{R_i^*\}_{i=1}^{|A|}$ be a rate vector to be allocated to the nodes in a given cluster A consisting of $|A|$ sensor nodes and the observation at node i in the cluster is X_i. Let $d(i,1)$ be the distance between node i and the cluster head v. Note that $d(1,1)$=0 denotes the distance between the cluster head v and itself. The optimal intra-cluster rate allocation is given by

$$R_1^* = H(X_1), \quad (5.15)$$

$$R_i^* = H(X_i|\{X_j|d(j,1) \leq d(i,1), j \in A\}), 2 \leq i \leq |A|. \tag{5.16}$$

Here the cluster head with zero distance to itself is encoded with a rate equal to its unconditional entropy and each of the cluster members in the cluster is encoded with a rate equal to its respective entropy conditioned on all the other nodes in the cluster which are closer to the cluster head than itself. According to chain theory, we have

$$\sum_{i=1}^{|A|} R_i^* = H(X_1, X_2, \cdots, X_{|A|}). \tag{5.17}$$

Therefore, $\{R_i^*\}_{i=1}^{|A|}$ is a valid rate vector for the optimal clustered Slepian-Wolf coding problem.

5.4.2 Clustered Slepian-Wolf Coding with Optimal Intra-Cluster Rate Allocation

With an optimal intra-cluster rate allocation, we now discuss how to perform Slepian-Wolf coding within a single cluster. Consider a cluster A with $|A|$ sensor nodes shown in Figure 5.2, where the node in black represents the cluster head and the nodes in white represent cluster members. The cluster head produces reading *X1*. From left to right, the first cluster member is the closest one to the cluster head and produces reading *X2*, the next closest one produces reading *X3*, and so on. Thus, the clustered Slepian-Wolf coding within this cluster is described as follows:

1. The cluster head schedules the cluster members in the descending order of their distances to the cluster head itself, as shown in Figure 5.2.

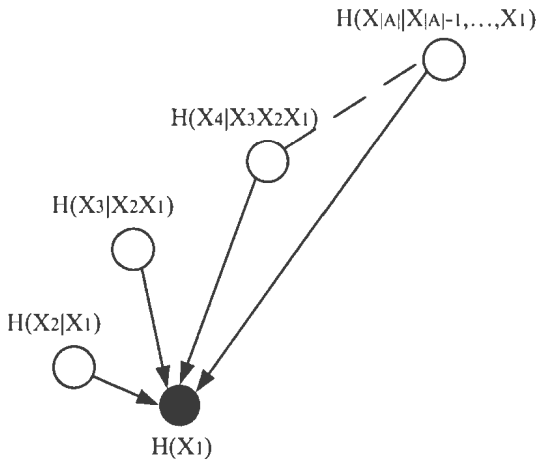


Figure 5.2: Slepian-Wolf coding within a cluster.

2. The cluster head generates a list for each cluster member i, which contains the indices (or IDs) of all the other nodes that are closer to the cluster head than cluster member i. For example, the list for cluster member 3 contains (2, 1).

3. The cluster head distributes the generated lists within the cluster. After receiving the list, a cluster member i encodes its reading with a rate equal to the its respective entropy conditioned on all the nodes in the received list, i.e., cluster member 3 encodes its data with a rate equal to $H(X_3|X_2X_1) = H(X_3X_2X_1) - H(X_2X_1)$. Note that in this case only distances among (X_3, X_2, X_1) is needed to calculate the rate and perform encoding with *a priori* knowledge of the correlation structure.

4. After the cluster head receives the compressed data from all its cluster members, it relays the data to the data sink, where conditional decoding is performed on the collected data. The sink decodes the cluster head's reading X_1 encoded with a rate equal to $H(X_1)$ without using any side information, while all other readings are decoded with the knowledge of the readings of the nodes that are

closer to the cluster head. For example, reading X_2 which is encoded with a rate of $H(X_2|X_1)$ is decoded with the knowledge of X_1, and reading X_i which is encoded with a rate equal to $H(X_i|X_{i-1}, \cdots, X_1)$ can be decoded with the knowledge of $(X_1, X_2, \cdots, X_{i-1})$. Thus, the loss of the data from one sensor node may affect the reconstruction of the sensor values from other nodes within the same cluster, but does not affect the decoding of the data from other clusters.

5.5 Joint Coding Mechanism

Slepain-Wolf coding can completely remove the data redundancy within each cluster with *a priori* knowledge of the correlation structure. However, the encoded data from two physically separated clusters may still have a certain amount of information in common or redundancy even though the correlation degree generally decreases quickly with the spatial separation between clusters. Explicit entropy coding is a low-complexity in-network data aggregation technique, where each sensor node encodes/decodes its reading only conditioned on the readings (explicit side information) it has already received from other nodes with no need to know the correlation structure *a priori* [35, 53]. Since in a cluster-based network, a cluster head uses other cluster heads as a relay, data (side information) from one cluster are available at the relay cluster heads. In this case, explicit entropy coding can be used to further reduce the potential correlation in the data from different clusters without significantly increasing coding complexity.

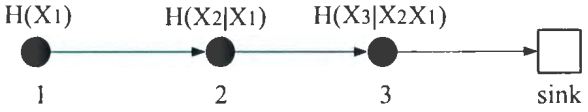


Figure 5.3: Joint clustered S-W coding and explicit entropy coding.

Based on the above observation, we propose a joint coding scheme, in which the Slepian-Wolf coding is first applied within each cluster. If a relay cluster head receives data from other cluster heads, it performs explicit entropy encoding only on its own sensed data, which can not be compressed via clustered Slepian-Wolf encoding because the optimal intra-cluster rate allocation requires the cluster head to encode its own data with a rate equal to the unconditional entropy. The coding scheme is briefly illustrated in Figure 5.3, where node 1, 2, and 3 are three cluster heads, and cluster head i produces reading X_i (i=*1, 2, 3*). Initially, a cluster head i encodes its reading X_i with a rate equal to $H(X_i)$ due to the requirement of clustered Slepian-Wolf coding, and when cluster head i receives data (side information) from other cluster heads, it re-encodes X_i with a new rate equal to its respective entropy conditioned on all other cluster heads which the side information has been received from and forwarded to. In summary, when explicit entropy coding is applied jointly with clustered Slepian-Wolf coding, the data sent by a given cluster head depends not only on the received data from cluster members in its own cluster but also on the data from other clusters whose cluster heads use that cluster head as a relay to the data sink. Therefore, the additional compression gain obtained by explicit entropy coding actually depends on the routing structure, as shown in Figure 5.3. We can see that a cluster head closer to the data sink encode its own data with a smaller rate, which can distribute more evenly the traffic load throughout the network, helping to avoid the formation of hot spots around the data sink.

5.6 Data Aggregation Using Distributed Lossy Coding

5.6.1 Distributed Lossy Coding

Consider a network with N sensor nodes distributed in a region of interest, where each node i produces reading Xi and all the readings constitute a set of jointly ergodic sources denoted by $X=(X_1,X_2,\ldots,X_N)$ with distribution $p(x_1,x_2,\ldots,x_N)$, which describes the spatial correlation structure and is known by each node *a priori*. The distributed lossy coding allows a distortion level of D_i in the reconstruction of source reading X_i. According to the rate-distortion region for coding correlated sources with high-resolution quantization [54], the nodes in the network can jointly encode their data without inter-node communication, with a total rate (in *bits*) lower-bounded by

$$h(X_1, X_2, \ldots, X_N) - 1/2 \log(2\pi e)^N \prod_{1 \le i \le N} D_i. \tag{5.18}$$

As long as their respective rates are under the constraints given by

$$R(U) \geq h(X(U)|X(U^c)) - 1/2 \log(2\pi e)^{|U|} \prod_{i \in U} D_i, \tag{5.19}$$

for all U$\subseteq\{1,2,\ldots,N\}$, where $\{1,2,\ldots,N\}$ is a set of the indices of sensor nodes in the network, Uc is the complementary set of U, $h(X)$ is the differential entropy of X, and

$$R(U) = \sum_{i \in U} R_i, \tag{5.20}$$

$$X(U) = \{X_j | j \in U\}. \tag{5.21}$$

For example, consider a simple case of two sensor nodes producing readings *X1* and *X2* with reconstruction distortion *D1* and *D2*. Their individual rates should be subject to

$$R_1 \geq h(X_1|X_2) - \frac{1}{2}\log(2\pi e)D_1, \tag{5.22}$$

$$R_2 \geq h(X_2|X_1) - \frac{1}{2}\log(2\pi e)D_2, \tag{5.23}$$

$$R_1 + R_2 \geq h(X_1,\ X_2) - \frac{1}{2}\log(2\pi e)^2 D_1 D_2. \tag{5.24}$$

According to chain theory [20], under the above constraints, it is always possible to find a rate allocation for the two nodes, which makes the total rate of the two nodes equal to their joint entropy subtracted by the distortion factor, *e.g.*,

$$\begin{aligned} R_1 + R_2 &= h(X_1) + h(X_2|X_1) - \tfrac{1}{2}\log(2\pi e)D_1 - \tfrac{1}{2}\log(2\pi e)D_2 \\ &= h(X_1, X_2) - \tfrac{1}{2}\log(2\pi e)^2 D_1 D_2. \end{aligned} \tag{5.25}$$

Correspondingly, for an arbitrary ordering of N nodes (*e.g.*, in the ascending or descending order of nodes' ID numbers) and a given distortion vector $\{D_i\}_{i=1}^N$, there always exists an optimal rate allocation vector $\{R_i\}_{i=1}^N$ such that the total rate of encoded data generated by all the nodes is equal to their joint entropy subtracted by the distortion factor, *i.e.*,

$$\sum_{i=1}^{N} R_i = h(X_1, X_2, \cdots, X_N) - \frac{1}{2}\log(2\pi e)^N \prod_{1 \leq i \leq N} D_i, \tag{5.26}$$

where

$$R_1 = h(X_1) - \frac{1}{2}\log(2\pi e)D_1 \tag{5.27}$$

and

$$R_i = h(X_i|X_{i-1}, X_{i-2}, \cdots, X_1) - \frac{1}{2}\log(2\pi e)D_i \tag{5.28}$$

where

$$2 \leq i \leq N$$

.

Therefore, a cluster of nodes A can be encoded with

$$h(X_1, X_2, \cdots, X_{|A|}) - \frac{1}{2}\log(2\pi e)^{|A|} \prod_{1 \leq i \leq |A|} D_i, \tag{5.29}$$

using distributed lossy coding without inter-node communication, and there always exists an optimal rate allocation to achieve this local maximum compression performance.

5.6.2 Clustered Lossy Coding (CLC) Problem

Given a network with a finite set of sensor nodes V, every sensor node in the network is initially a cluster head candidate. We assume that each candidate has an identical cluster diameter within which all other nodes may become its cluster members. The nodes within the cluster diameter of a candidate v form a finite point set N_v with the cardinality of $|N_v|$, which is called the neighbor set of candidate v. The power set of N_v, denoted by $P(N_v)$, is a set whose elements are the subsets of N_v and $P(N_v)$ constitutes all possible combinations of nodes in N_v. Thus, the cardinality of $P(N_v)$

is $2^{|N_v|}$. Since a candidate v, associated with each combination of nodes (or cluster members) within its cluster diameter e.g., a set of nodes B_v, where $B_v \in \mathrm{P}(N_v)$, can form a unique potential cluster (e.g., $A{:=}B_v \cup \{v\}$), a candidate v can generate up to $2^{|N_v|}$ potential clusters. Since initially every node in the network is a candidate, there are a total number of $|V|$ candidates. Therefore, there exists a cluster set S consisting of $\sum_{v \in V} 2^{|N_v|}$ potential clusters in the whole network. Meanwhile, each potential cluster A can be encoded with $h(X_1, X_2, \cdots, X_{|A|}) - \frac{1}{2}\log(2\pi e)^{|A|} \prod_{1 \le i \le |A|} D_i$ bits using lossy coding in that cluster.

Given the above assumptions, the CLC problem is to select a set of disjoint clusters C^*, with an allocated distortion vector, $\{\{D_i\}_{i \in A^*}\}_{A^* \in C^*}$, from the cluster set S to cover the whole network such that under given total and individual distortion bounds, the total rate (in *bits*) of encoded data generated by all the clusters (or all the nodes) in the network is minimized, *i.e.*,

$$\{C^*, \{\{D_i^*\}_{i \in A^*}\}_{A^* \in C^*}\}$$

$$= \arg \min_{\substack{C \subseteq S \\ \{\{D_i\}_{i \in A}\}_{A \in C}}} \sum_{A \in C} (h(X(A)) - \frac{1}{2}\log(2\pi e)^{|A|} \prod_{\{D_i\}_{i \in A}} D_i) \tag{5.30}$$

subject to

$$\sum_{i \in V} D_i \le D_{total} \tag{5.31}$$

$$D_i \le D_{\max} 1 \le i \le |V| \tag{5.32}$$

where $\bigcup_{A \in C^*} A = V$, $\bigcap_{A \in C^*} A = \phi$,$X(A)$=$\{Xj|j \in A\}$, D_{total} is the maximum total distortion bound and D_{max} is the maximum individual bound.

5.6.3 Problem Decoupling

Proposition: The CLC problem can be decoupled into two independent optimization problems: the optimal clustering problem and the optimal distortion allocation problem.

1) Optimal clustering problem

The optimal clustering problem is to select a set of disjoint potential clusters C^* from the cluster set S to cover the whole network such that the global network entropy, or the sum of the entropies generated by all the clusters (or all the nodes) in the network is maximized, *i.e.*,

$$C^* = \arg\min_{C \subseteq S} \sum_{A \in C} h(X(A)), \tag{5.33}$$

where $\bigcup_{A \in C^*} A = V$, $\bigcap_{A \in C^*} A = \phi$, and $X(A) = \{X_j | j \in A\}$.

2) Optimal distortion allocation problem

Given the maximum total distortion bound D_{total} and maximum individual distortion bound D_{max}, the optimal distortion allocation problem is to find a distortion vector $\{Di^*\}i \in V$ such that the product of the distortion allocated for each sensor node is maximized, *i.e.*,

$$\{D_i^*\}_{i \in V} = \arg\max_{\{D_i\}_{i \in V}} \frac{1}{2} \log(2\pi e)^{|V|} \prod_{1 \le i \le |V|} D_i \tag{5.34}$$

subject to

$$\sum_{i \in V} D_i \le D_{total}, D_i \le D_{\max}, 1 \le i \le |V|. \tag{5.35}$$

Proof: The CLC problem described by Equation (5.30) can be equivalently written as

$$\{C^*, \{\{D_i^*\}_{i\in A^*}\}_{A^*\in C^*}\} = \arg \min_{\substack{C\subseteq S \\ \{\{D_i\}_{i\in A}\}_{A\in C}}} \sum_{A\in C} h(X(A)) - \arg \max_{\substack{C\subseteq S \\ \{\{D_i\}_{i\in A}\}_{A\in C}}} \sum_{A\in C} \frac{1}{2}\log(2\pi e)^{|A|} \prod_{\{D_i\}_{i\in A}} D_i \quad (5.36)$$

In the first part of Equation (5.36), the objective function $\sum_{A\in C} h(X(A))$ does not contain any factor related to the distortion vector $\{Di\}i\in A$. Hence, the argument $\{\{Di\}i\in A\}A\in C$ related to distortion can be omitted. Then the first part becomes

$$\arg \min_{C\subseteq S;\{\{D_i\}_{i\in A}\}_{A\in C}} \sum_{A\in C} h(X(A)) = \arg \min_{C\subseteq S} \sum_{A\in C} h(X(A)). \quad (5.37)$$

According to the logarithmic identity, we have

$$\log a + \log b = \log ab. \quad (5.38)$$

Hence, the second part of Equation (5.36) can be written as

$$\arg \max_{\substack{C\subseteq S \\ \{\{D_i\}_{i\in A}\}_{A\in C}}} \sum_{A\in C} \frac{1}{2}\log(2\pi e)^{|A|} \prod_{\{D_i\}_{i\in A}} D_i$$

$$= \arg \max_{\substack{C\subseteq S \\ \{\{D_i\}_{i\in A}\}_{A\in C}}} \frac{1}{2}\log(2\pi e)^{|\bigcup_{A\in C} A|} \prod_{\{D_i\}_{i\in(\bigcup_{A\in C} A)}} D_i. \quad (5.39)$$

Since C is a set consisting of a combination of potential clusters to cover the whole network, for any $C \subseteq S$, $\cup A \in C A = V$, where S is the cluster set defined in Section 2.2. Hence, the second part of Equation (5.36) can be further written as

$$\begin{aligned} &\arg \max_{\substack{C \subseteq S \\ \{\{D_i\}_{i \in A}\}_{A \in C}}} \frac{1}{2} \log(2\pi e)^{|\cup_{A \in C} A|} \prod_{\{D_i\}_{i \in (\cup_{A \in C} A)}} D_i \\ &= \arg \max_{C \subseteq S; \{D_i\}_{i \in V}} \frac{1}{2} \log(2\pi e)^{|V|} \prod_{\{D_i\}_{i \in V}} D_i. \end{aligned} \tag{5.40}$$

Since the argument $C \subseteq S$ in Equation (5.40) is not relevant to the objective function $\frac{1}{2} \log(2\pi e)^{|V|} \prod_{\{D_i\}_{i \in V}} D_i$, Equation (5.40) can be finally written as

$$\begin{aligned} &\arg \max_{\substack{C \subseteq S \\ \{\{D_i\}_{i \in V}}} \frac{1}{2} \log(2\pi e)^{|V|} \prod_{\{D_i\}_{i \in V}} D_i \\ &= \arg \max_{\{D_i\}_{i \in V}} \frac{1}{2} \log(2\pi e)^{|V|} \prod_{1 \le i \le |V|} D_i. \end{aligned} \tag{5.41}$$

Hence, Equation (5.36) can be written as

$$\{C^*, \{D_i^*\}_{i \in V}\} = \arg \min_{C \subseteq S} \sum_{A \in C} h(X(A)) - \arg \max_{\{D_i\}_{i \in V}} \frac{1}{2} \log(2\pi e)^{|V|} \prod_{\{D_i\}_{i \in V}} D_i, \tag{5.42}$$

subject to

$$\sum_{i \in V} D_i \le D_{total}, D_i \le D_{\max}, 1 \le i \le |V| \tag{5.43}$$

Obviously, the first part of Equation (5.42) only depends on argument C while the second part is only relevant to argument $\{Di\} i \in V$. Therefore, Equation (5.36) can be decoupled into two independent parts: Equation (5.33) and Equation (5.34).

That is, the overall CLC problem can be decoupled two subproblems: (1) to optimally cluster the network to minimize global network entropy without considering distortion allocation; Equation (2) to optimally allocate a distortion to each node without considering node clustering.

5.7 Performance Evaluation

In this section, we evaluate the effects of the spatial correlation degree and the network size on the compression performance of DOC through simulations based on NS-2. Also, we investigate the performance of optimal intra-cluster rate allocation with respect to the intra-cluster communication cost under the cluster hierarchy constructed by DOC.

Unless otherwise specified, we consider a network with 100 sensor nodes uniformly deployed in a 100m×100m sensing region and a sink located at the center of the region. The simulation results are based on the average of 30 experiments and each experiment uses a different randomly-generated topology.

For the correlation structure, we assume that the observations $X_1, X_2, \cdots, X_N$ at N sensor nodes are modeled as an N-dimensional random vector $X = [X_1, X_2, \cdots, X_N]^T$, which has a multivariate normal distribution with mean (0, 0, ..., 0) and covariance matrix K, i.e., the density of X is

$$f(X) = \frac{1}{(\sqrt{2\pi})^N |K|^{1/2}} e^{-\frac{1}{2} X^T K^{-1} X} \tag{5.44}$$

and the differential entropy of $(X_1, X_2, \cdots, X_N)$ is

$$h(X_1, X_2, \cdots, X_N) = \frac{1}{2} \log(2\pi e)^N |K| \tag{5.45}$$

where $|K|$denotes determinant of the matrix K [20]. In the simulation, we use an exponential model of the covariance $k_{ij} = \exp(-d_{ij}^2 \cdot \theta)$ to model the observed physical event such as electromagnetic waves [9], where d_{ij} denotes the distance between the nodes measuring X_i and X_j, respectively. The parameter θ controls the relation between the distance d_{ij} and the covariance k_{ij}, and it can be set to different values to indicate different levels of correlation within a given distance. For the sake of simplicity and without loss of generality, we use differential entropy instead of discrete entropy because we assume that the sensor readings from different nodes are quantized with an identical and sufficient small quantization step, in which case the differential entropy differs from discrete entropy by only a constant [20, 55].

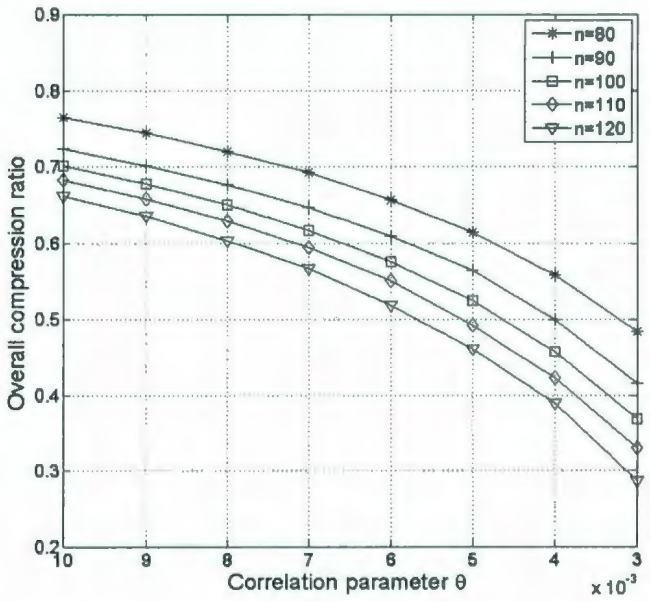


Figure 5.4: Impacts of the degree of correlation and the network size on overall compression ratio.

Figure 5.4 shows the effects of the correlation degree and the network size on the

compression performance. The compression performance is measured in an overall compression ratio, which is the total amount of data produced in the whole network after clustered Slepian-Wolf coding is applied over the total number of bits generated by all nodes without using this distributed source coding scheme. The network size or the total number of sensor nodes n is set to be {80, 90, 100, 110, 120}. The parameter θ in the covariance model is set to be {0.01, 0.009, ..., 0.0003}, where θ=*0.01* indicates low correlation and θ=*0.03* indicates high correlation. From the figure, it is seen that in the case of higher correlation a better compression performance is achieved because the Slepian-Wolf coding can remove more redundancy caused by the higher spatial correlation among the readings of different sensor nodes. In addition, the compression performance is improved as the network size or the density of sensor nodes increases. This behavior is due to the fact that the denser sensor deployment results in more sensor nodes residing within each cluster while clustered Slepian-Wolf coding can completely get rid of the highly redundant data generated by these sensor nodes in closer proximity to each other.

Figure 5.5 shows the intra-cluster communication cost with the optimal rate allocation and an ID-based rate allocation, respectively. As described in Section 2.1, the ID-based scheme first schedules nodes in a cluster A in the ascending (or descending) order of nodes' ID numbers. Thus, the rate assigned to the node i with ID number $ID(i)$ is given by $R_i = H(X_i|\{X_j|ID(j) \leq ID(i), j \in A\})$. The intra-cluster communication cost is given by Equation (5.26) and we use parameter θ=*0.006* to model moderate spatial correlation. The result shown is an average of the intra-cluster communication costs of all clusters in the network, when varying the network size. As expected, the optimal intra-cluster rate allocation results in less communication cost compared with the one only based on node's ID because the former scheme jointly considers rate assignments and transmission distances between the cluster members

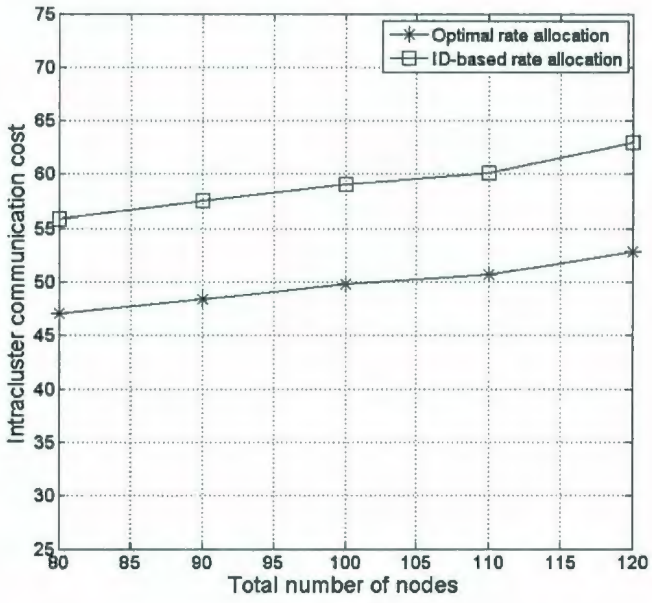


Figure 5.5: Intra-cluster communication cost with optimal rate allocation and ID-based rata allocation.

and the cluster head.

Figure 5.6 compares the intra-cluster communication cost of the distributed data aggregation technique using the clustered Slepian-Wolf coding with that of a widely-used centralized data aggregation technique [7, 9] under different cluster sizes (n) and a moderate correlation degree ($\theta = 0.006$). With the centralized aggregation technique, each cluster member periodically sends its original data to the cluster head and the data from all cluster members are aggregated at the cluster head. It is observed that the distributed aggregation technique using Slepian-Wolf coding leads to lower intra-cluster transmission cost than that with the centralized aggregation technique. This is because the distributed aggregation technique allows each cluster member to individually remove the redundancy existing in its data prior to sending

the data to the cluster head, thus significantly reducing the intra-cluster transmission cost.

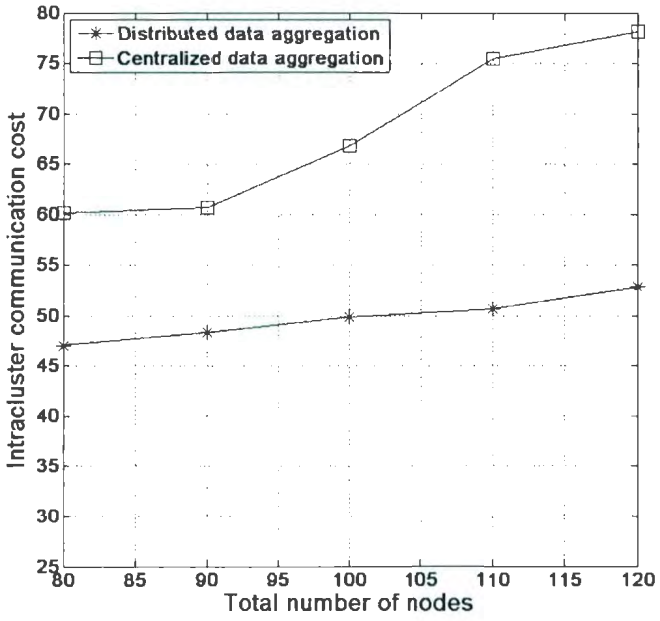


Figure 5.6: Total intra-cluster communication cost with distributed data aggregation and centralized data aggregation.

Figure 5.7 compares the total amount of data generated in the network using the clustered Slepian-Wolf coding and the joint coding, respectively. As expected, the joint coding results in obviously less amount of data, thus leading to better compression performance. This is because the joint coding employs clustered Slepian-Wolf coding combined with inter-cluster explicit entropy coding, which can further strip the data redundancy caused by the possible spatial correlation between different clusters. In addition, it is observed that in the case of high correlation (i.e., small value of the correlation parameter), the joint coding can achieve better performance in terms of the total amount of generated data because higher correlation leads to

more data redundancy between spatially separated clusters, which can be further removed by the joint coding. Meanwhile, more data redundancy removed from the network infers less energy consumed for data transmission. Figure 5.8 shows the total inter-cluster communication cost with Slepian-Wolf coding and joint coding, respectively. The total inter-cluster communication cost is defined as the sum of the communication costs of all cluster heads for relaying data to the remote sink, where the communication cost is represented by [data volume $\times$ transmission distance]. As expected, less communication cost is incurred with the joint coding. In addition, it is observed that under moderate correlation (e.g., θ =0.006), the joint coding leads to 9 % less communication cost than that with only the clustered Slepian-Wolf coding, but only 4 % less data. This is because each cluster head employs multi-hop routing for relaying data from other cluster heads and removing data redundancy locally at each cluster head by joint coding leads to further energy saving for each cluster head along the multi-hop routing path.

Figure 5.9 shows the approximate ratio of the total amount of data transmitted with the clustered Slepian-Wolf coding to that transmitted with the optimal coding in a network of 50 nodes uniformly deployed in the same region. With the optimal coding, Slepian-Wolf coding is applied globally in the whole network with the assumption that each node has the full knowledge of the correlation structure of the network, which can remove all data redundancy in the network and thus achieve the maximal compression gain. However, this is costly and usually impossible in a real-world large network. We investigated the total amount of data transmitted in the network with the cluster diameter ranging from 10 to 20 and the correlation parameter ranging from 0.005 to 0.01. In Figure 5.9, it is seen that with the increase of the cluster range the total amount of data transmitted with clustered Slepian-Wolf coding becomes closer to the optimal result because increasing the cluster range means that more

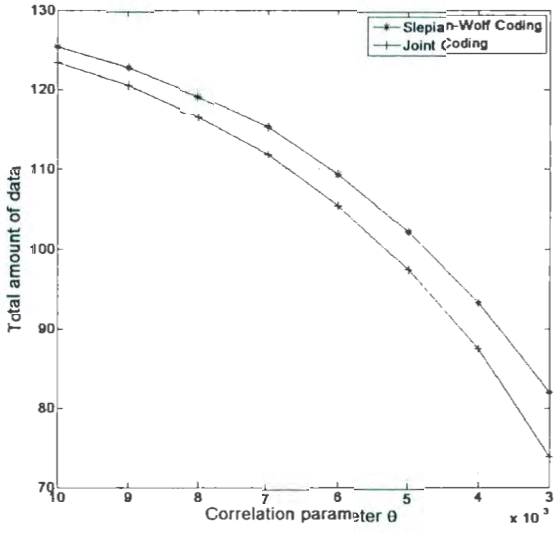


Figure 5.7: Total amount of data generated with Slepian-Wolf coding and joint coding.

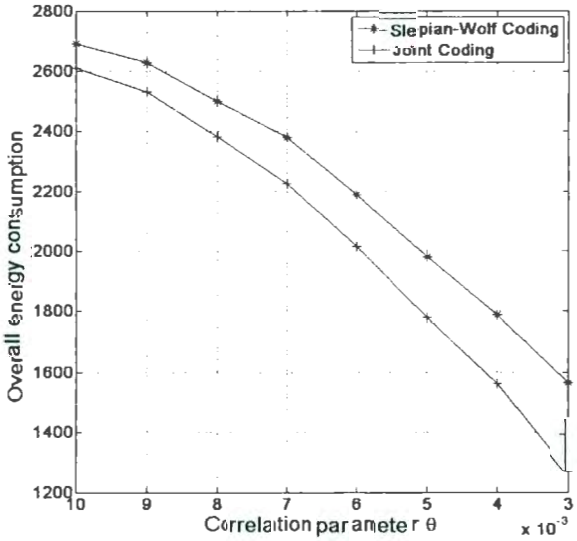


Figure 5.8: Total inter-cluster communication cost with Slepian-Wolf coding and joint coding schemes.

nodes are included in each cluster, thus further reducing the data redundancy caused by the possible spatial correlation between different clusters.

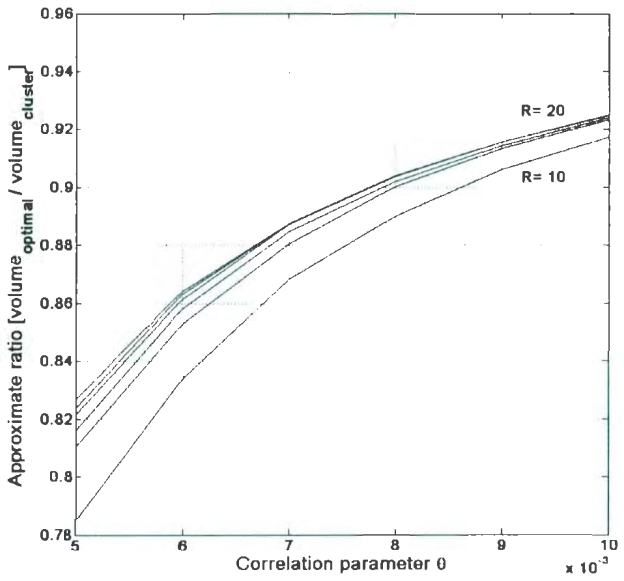


Figure 5.9: Approximate ratio of the total amount of data transmitted with the clustered Slepian-Wolf coding to that transmitted with the optimal coding

Figure 5.10 shows the relation between the total rate of encoded data and the distortion allocated to each sensor node. The parameter θ is set to be *0.01*. The total distortion bound *Dtotal* changes from $8e^{-4}$ to $8e^{-2}$ and the individual bound *Dmax* is set to be *0.1Dtotal*. According to the optimal distortion allocation, each sensor node is allocated an identical distortion equal to *Dtotal/80*. It is seen that the total rate of encoded data decreases monotonically as the distortion allocated to each node increases. This result is expected because a larger distortion allowed at each sensor node leads to a smaller total rate of encoded data in the network.

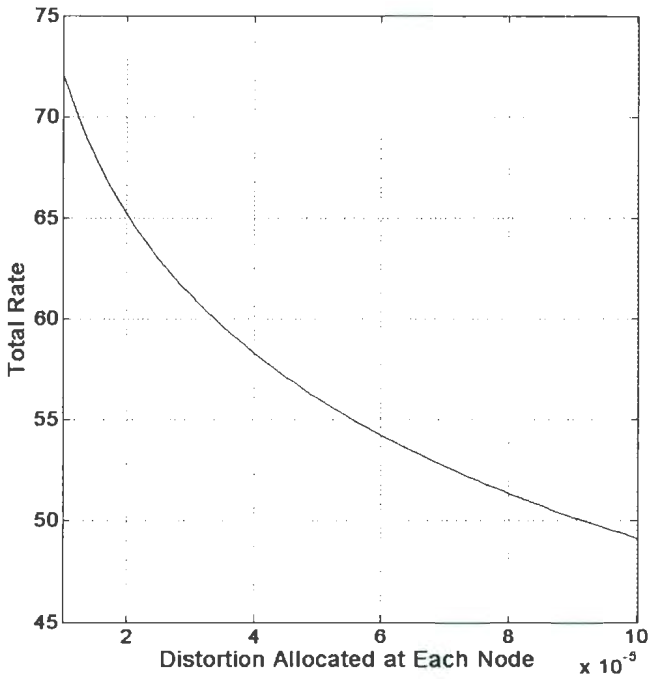


Figure 5.10: Relation between the total rate and the allocated distortion.

5.8 Summary

In this chapter, we have studied the major problems in applying Slepian-Wolf coding for data aggregation in cluster-based WSNs, including the clustered Slepian-Wolf coding problem, the optimal intra-cluster rate allocation problem, and the joint intra-clustered Slepian-Wolf coding and inter-cluster explicit entropy coding problem. A distributed optimal-compression clustering protocol (DOC) was proposed, which can select a set of disjoint potential clusters that maximize the global compression gain of Slepian-Wolf coding. Under the optimal cluster hierarchy constructed by DOC, we then presented an approximation algorithm that can find an optimal rate allocation within each cluster and described the procedures to perform Slepian-Wolf coding

with an optimal intra-cluster rate allocation. Finally, we present a low-complexity joint coding scheme that combines clustered Slepian-Wolf coding with inter-cluster explicit entropy coding to further reduce the data redundancy caused by the possible spatial correlation between different clusters. The simulation results demonstrate that the clustered Slepian-Wolf coding enabled by DOC can significantly reduce the total amount of data in the whole network while the transmission cost within each cluster can be remarkably reduced by performing the optimal intra-cluster rate allocation.

Chapter 6

Combined Data Aggregation and Encryption

6.1 Introduction

In the previous chapter, we have shown that to increase energy efficiency in wireless sensor networks, it is desirable to remove data redundancy and for this purpose data aggregation has been widely used. On the other hand, data security is an important issue for many WSN applications. To provide data security, the conventional way is to encrypt the sensed data at each sensor node and then decrypt the data at the data sink(s). However, network-wide encryption would cause considerable computational, communication, and storage overhead due to data encryption and key management operations.

Slepian-Wolf coding [19, 20] is a distributed source coding technique that can completely remove data redundancy without requiring inter-sensor communication and is therefore a promising technique for data aggregation in a WSN. To perform Slepain-Wolf coding, each sensor node must know *a priori* the correlation structure of

the whole network, which depends on the distances between the sensor nodes and the characteristics of the observed phenomena [20]. For this reason, Slepian-Wolf coding is not suitable or practical for being applied globally in a large network. In a cluster-based network, however, each cluster covers a smaller number of sensor nodes within a smaller local range of the network. To perform Slepain-Wolf coding, each sensor node only needs to know the local correlation structure of the cluster, which is practically easy to obtain. In addition to its capacity of removing data redundancy, Slepain-Wolf coding has the inherent characteristic of joint decoding, which we find could be used to achieve the effect of encryption within a single cluster. This observation has motivated us to apply Slepain-Wolf coding for both data aggregation and encryption in cluster-based WSNs in order to achieve efficient and secured data transmission.

In this chapter, we propose a combined data aggregation and encryption scheme using Slepain-Wolf coding for efficient and secured data transmission in WSNs. We first study the optimal intra-cluster rate allocation problem in using Slepain-Wolf coding for data aggregation, which aims to find a rate allocation subject to Slepian-Wolf theorem such that the total energy consumed by all sensor nodes in a cluster for sending encoded data is minimized. Based on the properties of Slepain-Wolf coding with optimal intra-cluster rate allocation, we then propose a novel encryption mechanism, called spatially selective encryption, for data encryption within a single cluster. This encryption mechanism only requires the cluster head to encrypt its data while allowing all cluster members to send their data without performing any encryption. Using this mechanism, as long as the data of the cluster head (or the *virtual key*) is protected, the data from all cluster members can also be protected, which can significantly reduce the energy consumption for data encryption. Furthermore, an energy-efficient key establishment protocol is also proposed to securely and efficiently establish the key used for encrypting the *virtual key*.

Data aggregation and encryption have been widely studied in the context of WSNs [9, 56, 57]. However, no research work has been found on combined data aggregation and encryption using Slepain-Wolf coding. To the best of our knowledge, this is the first attempt to apply Slepian-Wolf coding in cluster-based WSNs for both data aggregation and encryption purposes.

6.2 Data Aggregation Using Slepian-Wolf Coding

6.2.1 Optimal Rate Allocation for Slepian-Wolf Coding

Since we are considering cluster-based WSNs, we first suppose that a cluster hierarchy has already been constructed in the network by using a clustering protocol such as HEED [9] or any other clustering protocol. Consider a cluster A with $|A|$sensor nodes and let $\{Ri;\ i=1,\ 2,\ \ldots,\ |A|\}$ be a rate vector allocated to the nodes in the cluster. Also, let $d(i,\ 1)$ be the distance between node i and the cluster head v, which is used to estimate the energy consumed by node i for sending one bit data to the cluster head v because normally transmission energy dissipation is proportional to signal propagation distance. Then the objective of the intra-cluster rate allocation problem is to find a rate vector for the nodes in the cluster under the constraints given by Equation (6.2) such that the total energy consumed by all nodes for sending the data encoded with their individual rates to the cluster head is minimized, i.e.,

$$\{R_i^*\}_{i=1}^{|A|} = \arg \min_{\{R_i\}_{i=1}^{|A|}} \sum_{i=1}^{|A|} d(i,1)R_i, \tag{6.1}$$

subject to

$$\sum_{i \in Y} R_i \geq H(X(Y)|X(Y^c)), \forall Y \subseteq \{1,2,...,|A|\}, \tag{6.2}$$

where $\{1,\ 2,\ \ldots,\ |A|\}$ is a set of the indices of the sensor nodes in the cluster A.

As discussed in Chaper 5, the intra-cluster rate allocation problem can be solved by using the same approximation algorithm [27] for solving the global rate allocation problem. Using the approximation algorithm, the optimal intra-cluster rate allocation is obtained as follows.

Let $\{R_i^*\}_{i=1}^{|A|}$ be a rate vector to be allocated to the nodes in a given cluster A consisting of $|A|$ sensor nodes and the observation at node i in the cluster is X_i. Let $d(i, 1)$ be the distance between node i and the cluster head v. Note that $d(1, 1)=0$ denotes the distance between the cluster head v and itself. Therefore, the optimal intra-cluster rate allocation is given by

$$R_1^* = H(X_1), \tag{6.3}$$

$$R_i^* = H(X_i|\{X_j|d(j,1) \leq d(i,1), j \in A\}), 2 \leq i \leq |A|. \tag{6.4}$$

Here the cluster head with zero distance to itself is encoded with a rate equal to its unconditional entropy and each of the cluster members in the cluster is encoded with a rate equal to its respective entropy conditioned on all the other nodes in the cluster which are closer to the cluster head than itself. According to chain theory [20] and Equation (6.4), the sum of the rates allocated to the sensor nodes within the cluster is equal to their joint entropy, *i.e.*,

$$\sum_{i=1}^{|A|} R_i^* = H(X_1, X_2, \cdots, X_{|A|}). \tag{6.5}$$

Therefore, $\{R_i^*\}_{i=1}^{|A|}$ is a valid rate vector for achieving the entropy limit or the maximal compression gain of Slepian-Wolf coding.

6.2.2 Properties

With the optimal rate allocation, Slepain-Wolf coding has the following important properties.

Property 1*: *Joint Decoding

In Slepain-Wolf coding, decoding of the data from a sensor node is jointly or conditionally performed with the knowledge of the data from the other sensor nodes. According to the optimal rate allocation in Equation (6.4), reading X_1 from the cluster head, which is encoded with a rate equal to $H(X_1)$, can be decoded without using any other information. However, reading X_2 from cluster member 2 encoded with a rate equal to $H(X_2 \,|X_1)$ can be decoded only with the knowledge of reading X_1. Similarly, reading X_3 encoded with a rate equal to $H(X_3|X_2, X_1)$, can be decoded only with the knowledge of readings (X_1, X_2). In this manner, reading X_i from cluster member i encoded with a rate equal to H(X_i $|X_i - 1, \ldots, X_1$) cannot be decoded without the knowledge of readings (X_1, X_2, $\ldots$, $X_i - 1$).

Property 2: Data Independency

The data encoded with the optimal allocated rates at each sensor node is independent of that of any other sensor node, which implies that the encoded data at each sensor node are not spatially correlated and thus have no redundancy.

Note that this property can be verified by using a proof-by-contradiction method: if the argument were not true, a better rate allocation is supposed to be found to further remove the remaining data redundancy, which is contradictory to the fact that the optimal rate allocation already achieves the maximal compression gain of Slepian-Wolf coding described in Equation (6.5).

The two properties make it possible to use Slepain-Wolf coding to achieve the effect of data encryption within a cluster. Based on this observation, we propose a spatially selective encryption mechanism, which will be described in the next section.

6.3 Spatially Selective Encryption

6.3.1 Spatially Selective Encryption

The basic concept of the spatially selective encryption is to select a subset of sensor nodes from a cluster, and encrypt their data using symmetric-key or public-key cryptographic ciphers such as AES or RSA [36]. The sensor nodes *not* selected will send their data without any encryption. Since the data from the selected sensors are protected by cryptographic ciphers, it is practically impossible for attackers to reconstruct any data from these sensor nodes. If the data from the sensor nodes not selected cannot be reconstructed either, we can achieve a security level for the cluster which is equivalent to that with the data of all sensor nodes encrypted.

Given the properties discussed in Section 6.2, this concept can be implemented by using Slepian-Wolf coding with the optimal intra-cluster rate allocation in Equation (6.4). Specifically, in the proposed encryption mechanism, we only select the cluster head of a cluster as the node that encrypts its data using a cryptographic algorithm while allowing all cluster members to send their data without any encryption. According to *Property 1*, the data from each cluster member cannot be reconstructed without the knowledge of the data from the cluster head. This implies that Slepain-Wolf coding not only has the data compression function but also can achieve the effect of data encryption. Since the data of the cluster head actually acts as a *key* for reconstructing the data of each cluster member, we refer to it as the *virtual key* hereafter to differentiate it from the secret key in the symmetric-key cryptography and public-private key pairs in the public-key cryptography [36]. Intuitively, if the *virtual key* is protected, the data of all cluster members are also protected with no need to perform encryption. However, this observation is true only if the following

conditions are satisfied.

1. An attacker or intruder cannot reveal the *virtual key* by analyzing the data of the cluster members with no encryption.

2. An attacker or intruder cannot reveal the *virtual key* by performing the Brute-Force Attack (or Exhaustive Searching) [36] on the key space.

According to *Property 2*, the optimal rate allocation in Equation (6.4) can lead to the maximal data compression gain, which ensures that the data from the cluster head is not revealable even if the unencrypted data from the cluster members are available. On the other hand, the observed phenomenon at each sensor node is usually a random process. This means that the sensed data at the cluster head is also of randomness and vary constantly, thus leading to an enormous key space which prohibits the Brute-Force Exhaustive Searching from practically predicting the virtual key. Therefore, both conditions are actually satisfied, which ensures that as long as the virtual key is protected, the data of all cluster members are also protected without need to perform any encryption. This can considerably reduce the energy and resource consumptions for achieving data security in the network. On one hand, the amount of energy consumed for computing the cryptographic algorithms on the sensor nodes is significantly reduced because only a limited number of cluster heads need to perform encryption on their own sensed data. On the other hand, the amount of energy consumed for establishing and distributing the cryptographic keys is considerably reduced because the cluster members do not perform any encryption operations, thus eliminating the need to exchange key management information (e.g., secret keys) within each cluster.

6.3.2 Combined Data Aggregation and Spatially Selective Encryption

Now we discuss how to perform the combined data aggregation and encryption scheme within a single cluster.

Consider a cluster A with $|A|$ sensor nodes shown in Figure 6.1, where the node in black represents the cluster head and the nodes in white represent cluster members. The cluster head produces reading *X1*. From left to right, the first cluster member is closest to the cluster head and produces reading *X2*, and so on. Thus, the process of the combined data aggregation and encryption using Slepian-Wolf coding within a cluster is described as follows:

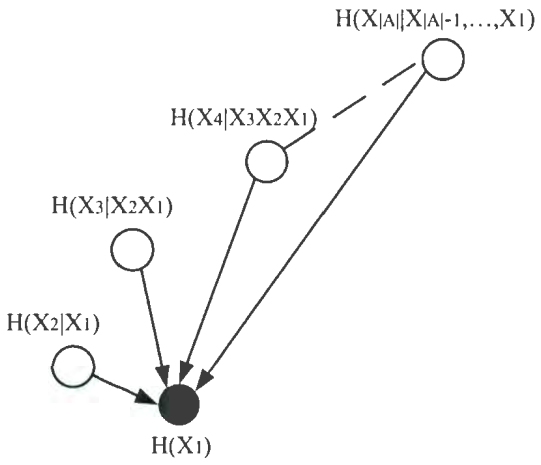


Figure 6.1: Spatially selective encryption within a cluster.

1. The cluster head arranges the cluster members in the descending order of their distances to the cluster head itself, as shown in Figure 6.1.

2. The cluster head generates a node list for each cluster member i, which contains the indices (or IDs) of all the other members that are closer to the cluster head

than cluster member i. For example, the list for cluster member 3 contains (2,1).

3. The cluster head distributes the generated node lists within the cluster. Once receiving the list, each cluster member encodes its reading with a rate equal to the its respective entropy conditioned on all the nodes in the received list, *e.g.*, cluster member 3 encodes its data with a rate equal to $H(X_3 \mid X_2X_1)$=H($X_3X_2X_1$)-H(X_2X_1). After encoding the sensed data, each cluster member $i \in A$ sends its encoded data to the cluster head without performing encryption.

4. After the cluster head receives the encoded data from all its cluster members, it only encrypts its own encoded data or the *visual key*, and sends the encrypted data as well as the encoded data from its cluster members to the data sink, where decryption is first performed on the data from the cluster head, and then conditional decoding is performed on the unencrypted data. In Figure 1, the sink will first decrypt the data of the cluster head, and then decodes reading X_1 encoded with a rate equal to $H(X_1)$ without using any other information. However, reading X_2 encoded with a rate equal to $H(X_2|X_1)$ is reconstructed with the knowledge of X_1. In this manner, the data from cluster member i encoded with a rate equal to $H(X_i|X_i - 1, \ldots, X_1)$ can be reconstructed with the knowledge of $(X_1, X_2, \ldots, X_i - 1)$.

6.3.3 Energy-Efficient Key Establishment Protocol

Since the spatially selective encryption only performs encryption on the *visual key*, now the problem is how to properly select an encryption algorithm to encrypt the *virtual key*. As mentioned, the *virtual key* is actually the sensed data frequently sent by a cluster head. Since a cluster head is also a resource-constrained sensor node, it

is important to select a low-cost encryption algorithm in order to alleviate the burden of each cluster head. Compared with a public-key cryptography alternative [36], a symmetric-key cryptography algorithm [36], which uses a smaller key size and less complex arithmetic, is much more energy efficient and thus more suitable for WSNs. However, the symmetric-key cryptography uses the same secret key for both encryption at the cluster head and decryption at the data sink. How to securely establish the secret key between the cluster head and the data sink in a hostile communication environment is a critical problem that must be well addressed.

To address the key establishment problem, conventional key establishment protocols usually use a key pre-distribution scheme or a master-key based scheme [36]. For the key pre-distribution scheme, the data sink pre-distributes a unique secret key to each sensor node. After clustering is performed, the selected cluster head uses the pre-distributed secret key to perform encryption on its visual key. For the master-key based scheme, the secret keys will be periodically updated. A secret key is only used for a certain fixed period and then will be discarded. The data sink will generate a new secret key and distribute it to the corresponding cluster head with encryption using a pre-distributed master key shared by the data sink and the cluster head. Both the schemes use static keys throughout the whole lifespan of the network, which will largely increase the chance for crypto-analytic attackers to reveal the secret or master keys. Therefore, both the schemes are unable to securely establish a secret key between the data sink and each cluster head.

To address the key security problem, public-key based key establishment protocols [36] are often used, which securely distribute and dynamically update the secret keys shared by the data sink and each cluster head. However, public-key cryptographic operations are computationally intensive, which is usually not suitable for being used in energy-constrained sensor nodes. Unlike sensor nodes, the data sink is

equipped with more resources, and thus has more powerful computational capability. To take advantage of this capability, it is more desirable to put computationally-intensive operations on the data sink while allowing the cluster heads to perform computationally-light operations. Based on this idea, we propose an energy-efficient key establishment protocol based on Rabin's public key algorithm.

Rabin's algorithm is a variant of the well-known RSA algorithm [36]. It has the characteristic of computational asymmetry in encryption and decryption. It requires the encryption party to perform a single modular squaring operation, which is computationally light, and the decryption party to perform a modular exponentiation operation, which is computationally intensive. Meanwhile, Rabin's algorithm is provable secure based on the intractability of the RSA problem (RSAP) [36], which is reducible to a hard problem of factoring large integers [36].

The energy-efficient key establishment protocol consists of two main phases: 1) key generation and pre-distribution and 2) key establishment, which are described as follows.

Phase 1: Key Generation and Pre-Distribution

Before the sensor nodes are deployed, the data sink generates and pre-distributes a unique public key to each sensor node. For this purpose, the data sink chooses two large primes p_i and q_i, and then calculates the public key n_i=$p_i \times q_i$, pre-distributes the public key n_i to sensor node i, and keeps the private key (p_i, q_i) itself. Due to the special characteristic of public-key cryptography, data encrypted by a sensor node with its public key can be decrypted only with the corresponding private key, which in our case is kept securely at the data sink.

Phase 2: Key Establishment

1. After clustering is performed, each cluster head i generates a random value as

the secret key K_s, and encrypts K_s by performing public key encryption (or a single modular squaring operation), i.e.,$E_{pub}(n_i, K_s)=K_s{}^2 \bmod n_i=y_i$, where n_i is a public key of cluster head i. The encrypted value y_i is then sent to the data sink.

2. After the data sink receives y_i, it performs public key decryption using the private key (p_i, q_i), *i.e.*, $D_{pub}((p_i, q_i), y_i)$, to recover the secret key K_s. Since only the data sink can decrypt y_i, only the data sink and cluster head i know the secret key K_s.

3. Each cluster head uses the secret key K_s to perform symmetric key encryption on its own encoded data or the *visual key* K_v, i.e., $E_{sym}(K_s, K_v)$, and then send the encrypted *virtual key* to the data sink.

Usually, a secret key is used for a certain fixed period. Once the secret key expires, the key establishment phase is performed again.

6.4 Performance Evaluation

In this section, we evaluate the performance of the combined data aggregation and encryption scheme through simulations based on NS-2. We first investigate the energy saving with the proposed data aggregation scheme. Then we compare the computational energy consumption of the spatially selective encryption with that of the network-wide encryption. Finally, we investigate the energy consumption for key establishment using the proposed key establishment protocol.

For the correlation structure, we assume that the observations $X_1, X_2, \ldots, X_N$ at N sensor nodes are modeled as an N-dimensional random vector $X=[X_1, X_2, \ldots, X_N]T$, which has a multivariate normal distribution with mean (0,0,...,0) and covariance

matrix K, *i.e.*, the density of X is $f(X) = \frac{1}{(\sqrt{2\pi})^N |K|^{1/2}} e^{-\frac{1}{2}X^T K^{-1} X}$and the differential entropy of $(X_1, X_2, \ldots, X_N)$ is $h(X_1, X_2, \cdots, X_N) = \frac{1}{2}\log(2\pi e)^N |K|$ *bits*, where $|K|$denotes determinant of the matrix X [19]. We use an exponential model of the covariance $k_{ij} = \sigma^2 exp[-(d_{ij})2\theta]$ to model the observed physical event such as electromagnetic waves [18], where d_{ij} denotes the distance between the nodes measuring X_i and X_j respectively. The parameter θ controls the relation between the distance d_{ij} and the covariance k_{ij}, and it can be set to be different values to indicate different correlation degrees within a given distance. For the sake of simplicity and without loss of generality, we use differential entropy instead of discrete entropy because we assume that the sensor readings from different nodes are quantized with an identical and sufficient small quantization step, in which case the differential entropy differs from discrete entropy by only a constant [19, 20].

Figure 6.2 shows the ratio of the intra-cluster communication cost of the proposed data aggregation scheme to that of a widely-used centralized data aggregation scheme, under different correlation degrees (θ) and different cluster sizes (n). In the centralized aggregation scheme, each cluster member periodically sends its original data to the cluster head and the data from all cluster members are aggregated at the cluster head. It is seen that the proposed scheme leads to less intra-cluster transmission cost than the centralized scheme and much more energy saving is obtained with the proposed scheme for a higher correlation degree and a larger cluster size. This is because the proposed scheme allows each cluster member to individually remove the redundancy existing in its data prior to sending the data to the cluster head, thus significantly reducing the intra-cluster transmission cost. On the other hand, a higher correlation degree among the readings of different sensor nodes leads to more data redundancy, and a larger cluster size means a higher density of sensor nodes deployed within each cluster and accordingly more data redundancy, which can be completely

striped by the clustered Slepian-Wolf coding, thus resulting in more transmission energy consumption reduced.

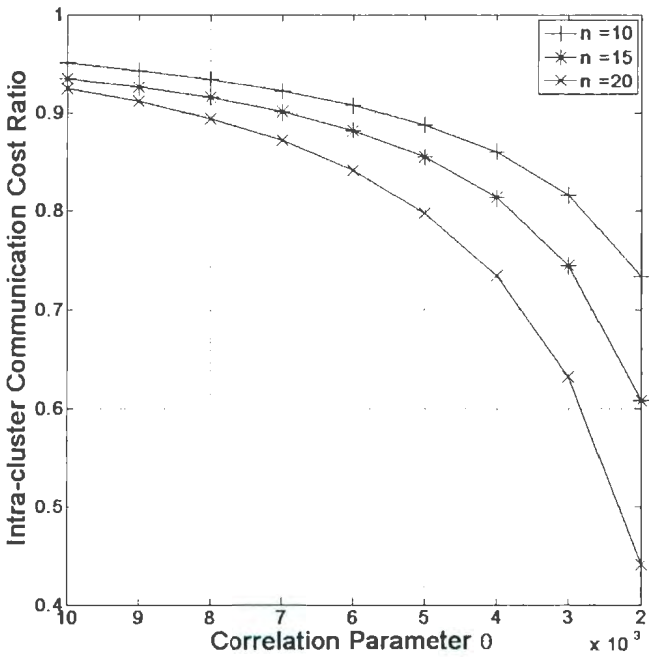


Figure 6.2: Intra-cluster communication cost ratio.

Figure 6.3 shows the computational energy consumption with the spatially selective encryption and the network-wide encryption, respectively, under different network sizes. The computational energy consumption is measured in the total amount of data required for encryption within a cluster. For the network-wide encryption, we considered two scenarios: (1) It is combined with the centralized data aggregation with infinite aggregation gain, where a cluster head aggregates the packets from all cluster members into a single packet; (2) It is combined with the proposed data aggregation. It is sent that the spatially selective encryption significantly reduces the computational energy consumption compared with the network-wide encryption.

This is because the spatially selective encryption only requires a cluster head to encrypt its data, while the network-wide encryption requires each node in a cluster to perform encryption on its data. Moreover, the network-wide encryption with the proposed data aggregation scheme incurs less computational cost than the one with the centralized data aggregation scheme. This is because the proposed data aggregation scheme can help to reduce the amount of data that need to be encrypted by completely stripping data redundancy within a cluster through Slepian-Wolf coding.

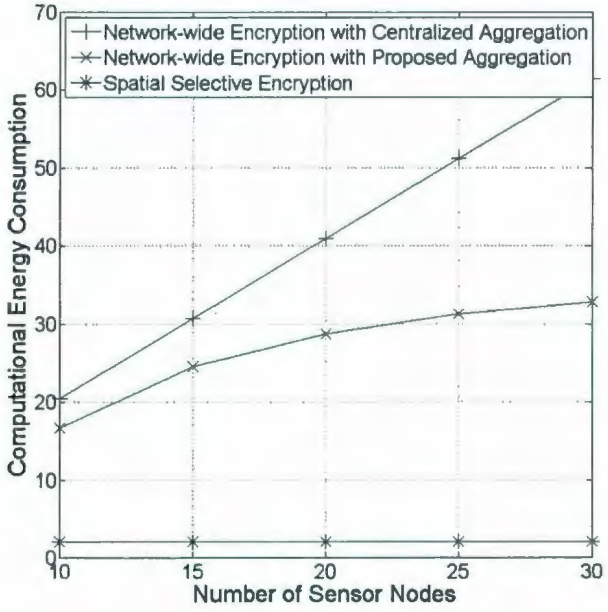


Figure 6.3: Computational energy consumption for data encryption (θ=*0.007*).

Figure 6.4 shows the energy consumption of the proposed key establishment protocol. The energy consumption includes the computational energy consumption for running Rabin's algorithm and the communication energy consumption for key establishment. To model the communication cost, we apply HEED [9], a well-known clustering protocol, to construct the clustered hierarchy, and adopt the relevant pa-

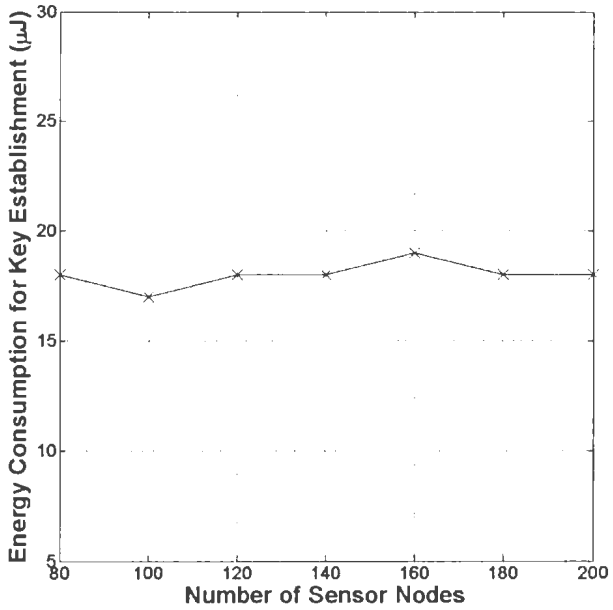


Figure 6.4: Energy consumption for key establishment.

rameters, radio transmission model, and routing protocol used in [9], which are typical settings for sensor networks. To model the computational cost, we use real experimental data [58] of the energy consumption for MIPS4000, a low-power microprocessor for sensor nodes, in computing a single modular squaring required for Rabin's scheme/algorithm. Since only the cluster heads are required to perform the key establishment protocol, we use the average energy consumption of the cluster heads to evaluate the energy consumption of the protocol. It is seen from Figure 6.4 that the average energy consumption of the cluster heads is less than 20 μJ under different network sizes. Therefore, the proposed key establishment protocol is energy efficient in key establishment.

6.5 Summary

In this chapter, we have proposed a combined data aggregation and encryption scheme using Slepain-Wolf coding for efficient and secured data transmission in wireless sensor networks. We first studied the optimal intra-cluster rate allocation problem for Slepain-Wolf coding and then proposed the spatially selective encryption mechanism for data encryption within a single cluster based on the properties of Slepain-Wolf coding with optimal intra-cluster rate allocation. The proposed encryption mechanism only requires the cluster head to encrypt its data while allowing all cluster members to send their data without performing any encryption. Furthermore, an energy efficient key establishment protocol is also proposed to securely and efficiently establish the key used for encrypting the data of the cluster head. Through simulation results, we showed that the combined data aggregation and encryption scheme can significantly improve energy efficiency in data transmission while providing a high level of data security.

Chapter 7

Multi-Channel Medium Access Control Protocol

7.1 Introduction

Wireless Multimedia Sensor Networks (WMSNs) are an emerging networking paradigm that allows retrieving video streams, still images, as well as generic sensing data from the environment [42]. A WMSN promises a wide range of potential applications in both civilian and military areas which require visual and audio information, such as multimedia surveillance, advanced health care delivery, and industrial process control [42]. Different from conventional wireless sensor networks, a WMSN normally demands larger bandwidth and entails higher network throughput to transport large volume of data to remote data sink rapidly and reliably. However, data rates provided by existing commercial sensor products, e.g., $250Kbps$ in MICAz [2], are not sufficient to support multimedia traffic. On the other hand, current sensor nodes, such as MICAz and WINS, already support multiple channels for communication, for example, 40 channels in WINS [42]. Thus, by developing a multi-channel MAC protocol, which

can effectively utilize the available channel capacity through the cooperative work from other sensor nodes, we can achieve a better support for multimedia applications which demand for high data rates.

Multi-channel MAC design has been studied for mobile ad hoc networks (MANETs) research [44, 59, 60, 61]. However, the proposed protocols in the literature are not immediately suitable for the energy-constrained wireless sensor networks. These protocols normally require that each node perform dynamic channel sharing using handshaking, e.g., *RTS* and *CTS* in *IEEE* 802.11, to complete a negotiation and to avoid multi-channel hidden terminals. Such a mechanism usually will incur a considerable control overhead. However, in sensor networks, the relatively static nature makes it possible to dispense with the contention overhead by better coordinated channel scheduling. Besides the low energy efficiency, those protocols also assume a flat, homogeneous architecture in which nodes have identical capacities. This underlying flat architecture, however, is not suitable for multimedia applications, where each node transmits not only its own data but also the traffic generated by other nodes. The large volume of data resulting from multi-media applications will then quickly drain the battery of the sensor nodes, thus, significantly reducing the network lifetime.

In this Chapter, we propose a clustered on-demand multi-channel MAC protocol (COM-MAC) in order to maximize the network throughput with enhanced energy efficiency. The network under investigation follows three-tier architecture: a data sink at the top, a set of sensor nodes at the bottom, and a relatively small number of aggregators in the middle. The data sink communicates with the aggregators using an out-of-band channel. Each aggregator has multiple transceivers and is able to operate on a set of available channels simultaneously. Abundant power supply is assumed for those nodes in our investigation. A sensor node has one transceiver but is able to switch among the channels managed by its associated aggregator. In COM-

MAC, a clustering protocol is first applied so that each sensor is associated with one aggregator. In this case, an aggregator is called a cluster head and the sensor nodes associated with it are called its cluster members. A cluster head, together with its associated members are called a cluster collectively. Within each cluster, a scheduled multi-channel medium access protocol is considered, where the cluster head coordinates the communication among its members in a contention-free manner within both the time and frequency domains, so as to avoid collision, idle listening and overhearing. Secondly, to maximize the network throughput, a traffic-adaptive and QoS-aware (Quality of Service) scheduling algorithm is performed where the cluster head dynamically allocates time slots and channels for its members according to the current QoS requirements and network traffic status. Finally, to enhance transmission reliability, a spectrum-aware ARQ is used to opportunistically exploit the unused spectrum for a balance between the reliability and retransmission. Through simulation results, we show that the proposed COM-MAC can improve network throughput significantly, while by introducing very small control overhead.

7.2 Design of the COM-MAC Protocol

7.2.1 Network Architecture and Assumptions

As shown in Figure 7.1, a WMSN consists of several more powerful nodes (cluster heads) located at the center of different monitoring area, a number of regular multimedia sensor nodes surrounding each cluster head, and a remote data sink which will store the multimedia content locally for later retrieval. In addition, we make the following assumptions regarding the configuration of the network:

1. There are N different channels available for use and all channels have the same

bandwidth.

2. All multimedia sensor nodes are identical and quasi-stationary. Each sensor node is equipped with a single half-duplex transceiver, which means a sensor node will not be able to transmit and receive simultaneously.

3. A multimedia sensor node can only transmit or receive on one channel at a time. But it is able to switch among channels dynamically. The channel switching time is less than $224\mu s$ according to [62].

4. Each cluster head is equipped with N half-duplex transceivers, which means that a cluster head can transmit or receive on N channels simultaneously. In addition, each cluster head will have sufficient power supply and better processing capacity.

5. The working of a cluster of sensor nodes is synchronized to the cluster head and each sensor node can communicate directly with its cluster head.

6. A cluster head can usually communicate directly with the data sink using an out-of-band channel. However, if direct communication is unavailable, multi-hop routing is also employed.

7.2.2 Overview of the COM-MAC Protocol

We assume that the clustering process has been completed by performing some distributed clustering protocol and each sensor node bas been associated with a nearest cluster head. Within a cluster, the operation is organized in time intervals. Each interval consists of three consecutive sessions: request session, scheduling session and data transmission session, as shown in Figure 7.2.

During the request session, each sensor node sends a REQ message to the cluster head. QoS requirements, such as the amount of multimedia data to be transmitted, delivery deadline, and priority information, are included in the REQ message. Based on the information, during the scheduling session, the cluster head schedules the transmission for each sensor nodes using certain optimal scheduling algorithm, and then distributes the resulting schedule to all sensor nodes in the cluster. The schedule gives the information of time slots and channels assigned to each sensor node during the following data transmission session. Sensor nodes will then switch to the assigned channel and start to send their data in the scheduled slot without further contention. Next, we examine the details of the three sessions in the COM-MAC protocol.

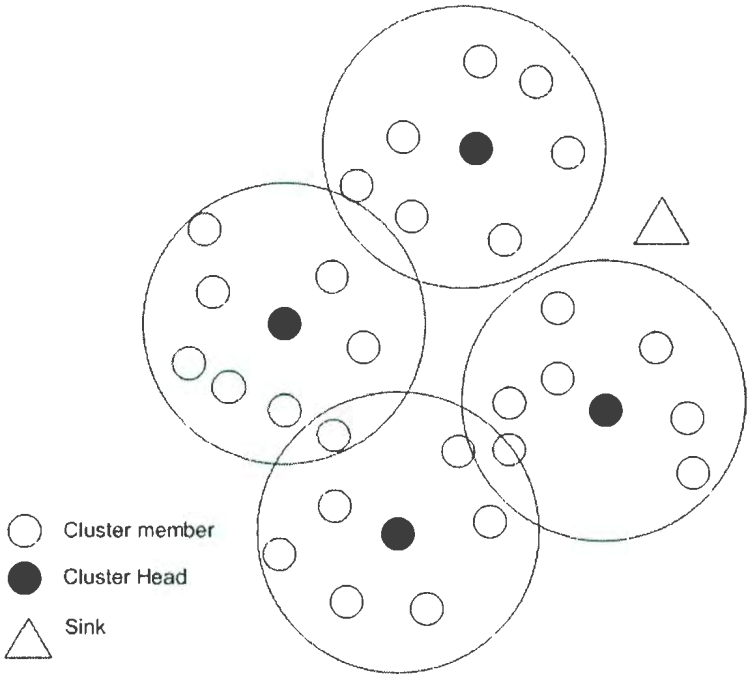


Figure 7.1: WMSN network architecture.

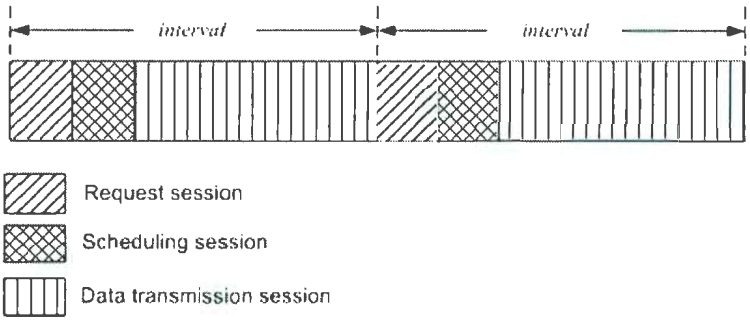


Figure 7.2: Frame structure.

7.2.3 Request Session

During the request session, two protocols are designed for each sensor node to send its request to the cluster head. One is a contention-based protocol and the other is a contention-free TDMA/FDMA based protocol.

The operation of the contention-based protocol consists of two steps: the control channel assignment phase and the request transmission phase. During the control channel assignment phase, a channel is allocated for each node to transmit the request message (REQ). To improve channel utilization, all available channels can be used as control channels during the request session. Because the number of available channels is usually limited, it is likely that a channel be assigned to multiple nodes. To avoid possible congestion, sensor nodes will be evenly distributed to the available channels. Furthermore, different channels will be assigned to geographically adjacent sensor nodes because nodes in close proximity tend to discover the same event and then request for transmission simultaneously, which tend to introduce potential contention without proper channel spatial reuse schemes. After control channel is assigned, the request transmission phase starts. If a node has data to send, it will notify the cluster head by sending a REQ message using the assigned control channel. Upon receiving the REQ, the cluster head replies with an ACK. To further avoid possible

collisions, a random backoff scheme is employed in each sensor node before sending the *REQ* message. The backoff interval is randomly chosen in the range of [0, *Treq* – *Treq_trans*], where *Treq* is the duration of the request session and *Treq_trans* is the duration to deliver the *REQ* message.

The contention-free TDMA/FDMA protocol consists of two phases: control slot assignment and request transmission. Control slot assignment, which is only performed when the network is initially deployed, is used to statically allocate a time slot of a channel to each sensor node to send request message. Similar to the mechanism incorporated in the contention-based protocol, all available channels can be used as control channels. Each channel is further divided a number of time slots. The duration of each slot is equal to the time to transmit a *REQ* message. The total number of slots for each channel is calculated by X/Y, where X denotes the total number of sensor nodes in a cluster and Y denotes the total number of channels available. Then, each slot on each channel is assigned to a unique node so that any request can be sent without interfering with the transmission of other nodes.

The two proposed protocols can be used for different application scenarios. When network traffic load is not very intensive and the channel condition is relatively unreliable, the contention-based protocol is favored because the probability of potential collision and congestion to send the request message is low. Although the request message may be lost due to collision, a simple retransmission scheme can be used to avoid the transmission delay of multimedia data. In contrast, the contention-free TDMA/FDMA protocol is more appropriate for applications with reliable channel conditions and heavier traffic load.

7.2.4 Scheduling Session

Based on the information obtained in the request session, during the scheduling session, cluster heads then generate a schedule to coordinate the data transmissions of each sensor node and broadcast the message through control channels, where all sensor nodes are tuned to and listen on. The schedule to be broadcast on each control channel will only include the information for sensor nodes which are assigned to that channel. In order to enhance the transmission reliability, the schedule broadcast will be repeated. However, for better energy efficiency, a sensor node will turn off its transceiver once a full schedule has been received, until its time slot for transmission approaches.

For each sensor node, the generated schedule includes both time slot and radio channel for data transmission. For example, considering a cluster with 16 sensor nodes, let p_i denote the request sent by node i, where i = 1,2,...,16, which also describes the amount of data to be sent by a sensor node. Assuming that there are 5 non-overlapping channels available, there can be a large number of different potential schedules. One possible schedule is shown in Figure 7.3. Obviously, this schedule is not optimal with respect to network throughput because a big portion of unused spectrum, which is indicated by *H1, H2, H3* and *H4*, is not utilized for the transmission for node 6. To enhance network throughput, we first show that the maximal throughput problem can be converted to an optimal multi-channel scheduling problem. Then we present a heuristic algorithm to solve the problem.

The throughput is a function of the channel capacity used for data transmission. Given that the packet size is P bits, time used to deliver a packet is T, and capacity of the channel is C *bps*, then the throughput is given by

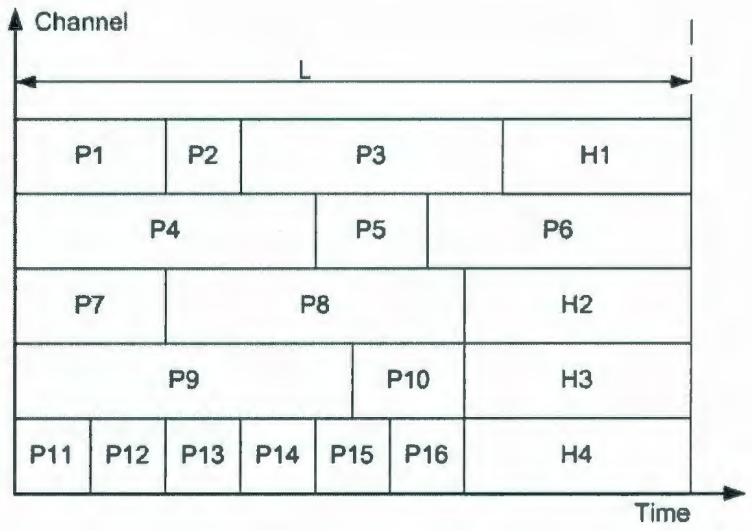


Figure 7.3: An example of request schedule.

$$\eta = P/TC. \tag{7.1}$$

In WMSNs, P is equal to the sum of requests from all sensor nodes, i.e., $P = \sum_{i=1}^{M} P_i$, where M is number of the requests. The channel capacity is given by $N \times C$. Because T is the total amount of time to transmit all requests, in other words, the maximal finish time among all channels (for example, $T = L$ in Figure 7.3), the network throughput in the multi-channel scenario is given by

$$\eta = \sum_{i=1}^{M} P_i/TNC. \tag{7.2}$$

Because $P = \sum_{i=1}^{M} P_i$ is fixed after the request session, for example, $P = \sum_{i=1}^{16} P_i$ in Figure 7.3, and so does N and C after network is deployed, maximizing throughput η is equivalent to minimizing T.

If each request is associated with a priority to indicate the QoS requirement, such as the maximum allowed delay, the optimal multi-channel scheduling problem can

be described as: Given a set of requests $\{p_1, p_2, \ldots, p_m\}$, which is associated with a corresponding set of priorities $\{r_1, r_2, \ldots, r_m\}$ and N available channels, find an assignment schedule that puts all requests to the N channels based on the order of their priorities so that the total amount of time to transmit all requests, in other words, the maximum transmission time, on any channel is minimized.

To solve this problem, a request scheduling heuristic is provided. First, all requests of the same priority are grouped together. Then, all groups are sorted based on the descending order of the priorities. We then schedule the requests for each group according to their priority level, beginning with the group with of the highest priority. Within each group, all the requests are also sorted based on the descending order of the transmission time and a request with the minimum amount of time to transmit will be assigned to a channel first until all channel capacity has been fully utilized. This process is repeated until a complete schedule is generated.

7.2.5 Data Transmission Session

After receiving the schedule, each sensor node will transmit its data during the assigned time slots and channel. Each time slot is further divided into two sections: data transmission section and *ACK* section. The *ACK* section is used to support link-layer error control, which is critical for WMSN. Normally, for high-quality video perception, a frame loss rate of lower than 10^{-2} is required [42]. However, the inherent unreliable nature of wireless medium poses a significant challenge for the applications of WMSN. To address this problem, an implicit selective repeat *ARQ* technique is employed, which is briefly described below.

After receiving the packets from a sensor node, the cluster head acknowledges every properly received packet by sending an *ACK* message. Packets which are not

acknowledged are simply assumed to be lost in the network, which will be marked by the sensor nodes and retransmitted during the next interval. Although such a retransmission mechanism improves the transmission reliability, extra delay will be introduced, which is not favored for critical applications such as real-time video or audio transmission. To mitigate this problem, a hybrid MAC protocol is used to better exploit the unused spectrum during the data transmission session, for example, *H1*, *H2*, *H3*, and *H4* as shown in Figure 7.3.

The working of the Hybrid MAC is briefly described here: First, in the scheduling section, the cluster head piggybacks the completion time of each channel to its broadcast schedule, for example, the completion time of $p3$ for channel 1 in Figure 7.3. When receiving the schedule, each node obtains the explicit knowledge of the potential available spectrum. Then, after the scheduled contention-free data transmission, the sensor nodes can employ an energy-efficient MAC, such as S-MAC, to retransmit the lost packets. In this way, better balance between the required reliability and the sustainable delay can be achieved at the application layer.

7.3 Performance Evaluation

In this section, we evaluate the performance of the proposed COM-MAC through simulation experiments using ns-2. We investigate the performance in terms of network throughput and transmission delay. The performance of COM-MAC is compared with that for a baseline protocol, the multi-channel TDMA (M-TDMA) protocol. For M-TDMA, the cluster head first evenly distribute the cluster members on the available channels. Then, the cluster head generates a TDMA schedule on each channel and allocates a fixed slot to each cluster members.

In our experiments, the capacity for each channel is $250kbps$. The transmission

range of each node is approximately $10m$. Each source node generates and transmits a constant-bit rate (CBR) data stream. Each node is randomly selected to have a packet arrival rate between 0 and 10 to reflect the network traffic dynamics. Each simulation run is performed for the duration of 30 seconds. Each data point in the performance figures is the average of 20 runs. Unless otherwise specified, we assume 3 channels and the packet size is 525 bytes,

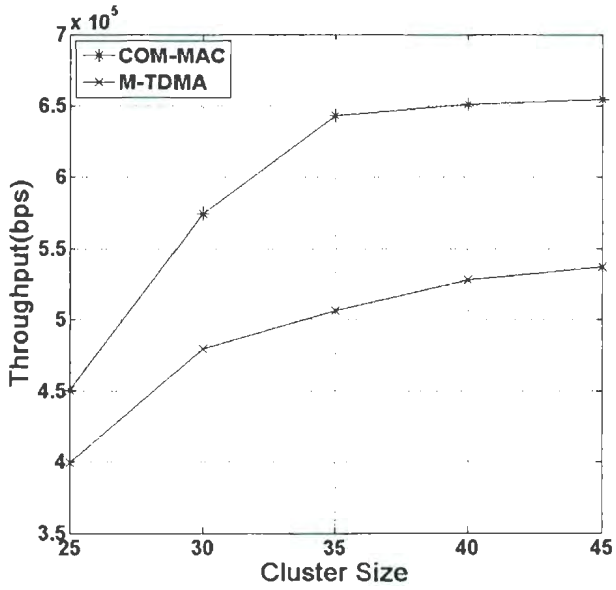


Figure 7.4: Throughput performance for various cluster sizes.

Figure 7.4 compares the network throughput performance of COM-MAC and M-TDMA for different cluster sizes, which is the total number of sensor nodes in a cluster. It is clear that the throughput of both protocols increase as the cluster size increases. This is because more sensor nodes need to access channels to transmit data to the cluster head. It is also noticed from the figure that the increase of the throughput of COM-MAC becomes slower when the cluster size exceeds 45. This is because the channel tends to be saturated when more nodes are trying to utilize the channel. As

expected, COM-MAC outperforms M-MAC for different conditions. This is because COM-MAC is designed to maximize the network throughput by applying traffic-adaptive scheduling algorithm where the time slots and channels are dynamically allocated for its members according to data traffic. However, M-TDMA assigns each node a fixed time slot without considering the traffic load condition on other nodes.

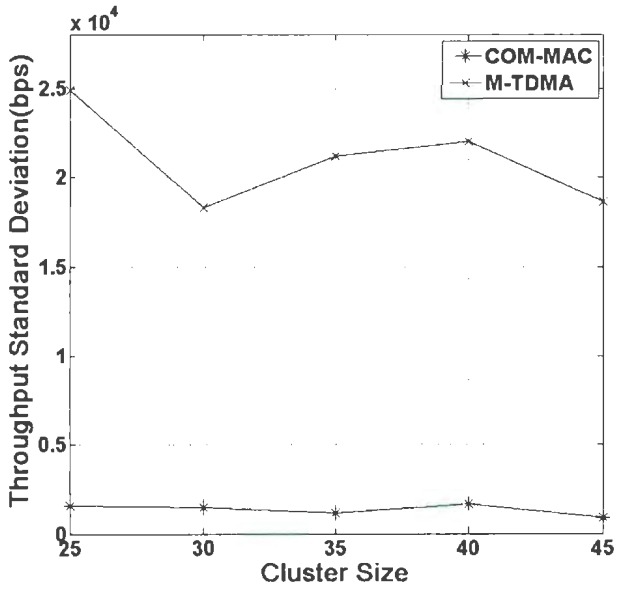


Figure 7.5: Throughput standard deviation for various cluster sizes.

Figure 7.5 shows the standard deviation of the throughout performance for the three available channels over different cluster size. It is clear that the curve for COM-MAC is much smoother than that from the M-TDMA protocol. This indicates that COM-MAC can achieve better and more balanced utilization of each available channel so that the network throughput can be maximized. In contrast, the standard deviation of M-TDMA is much larger and possesses significant fluctuation, which indicates that by using M-TDMA, some channels have already been over-utilized while some channels still remain under-utilized, even if some nodes have data to be

transmitted.

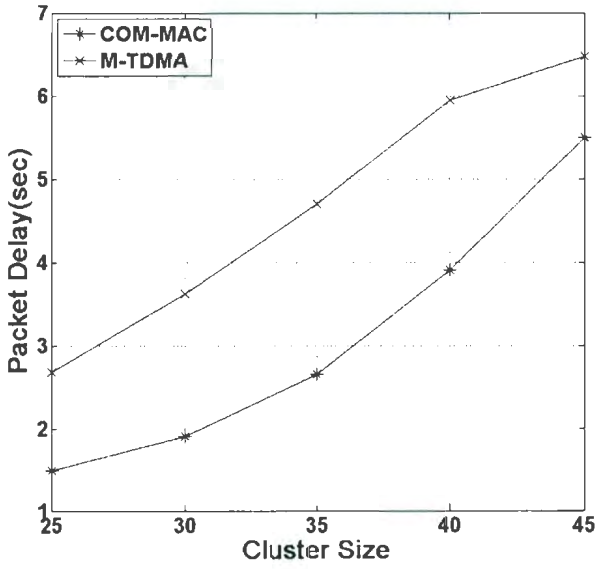


Figure 7.6: Packet delay performance for various cluster sizes.

Figure 7.6 shows the delay performance compassion of COM-MAC and M-TDMA protocols as cluster size increases. It is obvious that COM-MAC incurs lower delay when compared to that from M-TDMA. This is due to the fact that with M-TDMA, a sensor node has to wait until its time slot comes to send out its data, even if there are time slots on its assigned channel or on other channels available for use. COM-MAC can achieve a better utilization of the channel resources, thus leading to a better delay performance. We also notice that the delay performance increases as cluster size increases for both protocols. This is because that larger cluster size will lead to heavier network load so that a packet has to wait longer to be transmitted.

7.4 Summary

In this chapter, we have studied a cluster based on-demand multi-channel MAC protocol, which is proposed to provide better energy-efficiency, high-throughput, and data reliability support in wireless multimedia sensor networks. We have proposed a scheduled multi-channel medium access within each cluster for contention-free intra-cluster communication. We also proposed a scheduling algorithm to achieve better channel utilization under different traffic conditions and QoS requirements. We incorporated a spectrum-aware *ARQ* scheme to enhance the transmission reliability. Our simulation results demonstrate that COM-MAC can achieve better network throughput and lower delay when compared with the baseline M-TDMA protocol at the cost of a small control overhead.

Chapter 8

Conclusions and Future Work

8.1 Summary of Contributions

In this thesis, we have investigated several key issues in sensor networks with the purpose to improve energy efficiency, fault tolerance, and security. The topics include clustering, fault recovery and detection, data aggregation, encryption and key distribution, and multi-channel medium access control. We have analyzed the characteristics of three types of sensor networks, including underwater sensor networks, wireless terrestrial sensor networks, and wireless multimedia sensor networks. Different characteristics of each type of sensor network will enable a wide range of applications which present many challenges for the design of such network including energy efficiency, fault tolerance, and security. To address the challenges, we have proposed several novel and effective mechanisms: (1) distributed minimum-cost clustering protocol, (2) robust architecture for underwater sensor network, (3) cooperative fault detection mechanism, (4) distributed data aggregation using Slepian-Wolf coding, (5) spatially selective encryption, and (6) clustered on-demand multi-channel MAC protocol.

- **Distributed Minimum-cost Clustering Protocol**

  We have studied the node clustering problem in a UWSN and formulated the problem into a cluster-centric cost-based optimization problem with an objective to improve the energy efficiency and prolong the lifetime of the network. To solve the formulated problem, a distributed minimum cost clustering protocol (MCCP) has been proposed, which can not only adapt geographical cluster head distribution to the traffic pattern in the network, and thus avoid the formation of hot spots around a uw-sink, but also balance the traffic load between cluster heads and cluster members through periodical re-clustering the sensor nodes in the network. The simulation results show that MCCP significantly improves network lifetime as compared with the well-known HEED protocol.

- **Robust Architecture for Underwater Sensor Network**

  We have proposed a dependable clustering protocol to provide a robust cluster hierarchy against cluster-head failures in UWSNs. We have proposed a dependable clustering protocol to provide a robust clustered architecture against cluster-head failures in UWSNs. The proposed clustering protocol takes into account both the reliability and residual energy status of each sensor node, and introduces failure prediction, cost evaluation, and clustering optimization during clustering to construct an efficient and robust cluster hierarchy. The proposed clustering protocol attempts to select those healthy nodes as cluster heads to prevent cluster head failures. Meanwhile, it attempts to select a primary cluster head and a backup cluster head during clustering so that the cluster members associated with the failed cluster head can quickly switch over to the backup cluster head in the event of a cluster-head failure. The simulation results have

shown that the protocol can effectively enhance network robustness.

- **Cooperative Fault Detection Mechanism**

  We have proposed a cooperative fault detection mechanism for accurately and quickly detecting cluster-head failures in TDMA-based clustered UWSNs. The proposed fault detection mechanism allows each cluster member to independently detect the fault status of its cluster head and then employs a distributed agreement protocol to reach an agreement on the fault status of the cluster head among multiple cluster members. A couple of forward and backward TDM frames are specially structured for enabling multiple cluster members to reach an agreement within two frames in a fault detection process. A schedule generation algorithm is also proposed for a cluster head to generate the transmission schedule in the forward and backward frames. Our simulation results have shown that the proposed detection mechanism can achieve high detection accuracy under high packet loss rates which is typical in the harsh underwater environment, and can detect a cluster-head failure faster than the traditional fault detection mechanism with a delay bound of two TDM frames. Moreover, it makes use of the data packets periodically sent by a cluster head as the heartbeats for fault detection and uses a couple of specially-structured forward and backward frames in the agreement process, which are energy efficient and do not affect normal network operation.

- **Distributed Data Aggregation using Slepian-Wolf Coding**

  We have studied the major problems in applying Slepian-Wolf coding for data aggregation in cluster-based WSNs, including the clustered Slepian-Wolf coding problem, the optimal intra-cluster rate allocation problem, and the joint intra-clustered Slepian-Wolf coding and inter-cluster explicit entropy coding prob-

lem. A distributed optimal-compression clustering protocol (DOC) has been proposed, which can select a set of disjoint potential clusters that maximize the global compression gain of Slepian-Wolf coding. Under the optimal cluster hierarchy constructed by DOC, we then presented an approximation algorithm that can find an optimal rate allocation within each cluster and described the procedures to perform Slepian-Wolf coding with an optimal intra-cluster rate allocation. Finally, we present a low-complexity joint coding scheme that combines clustered Slepian-Wolf coding with inter-cluster explicit entropy coding to further reduce the data redundancy caused by the possible spatial correlation between different clusters. The simulation results have demonstrated that the clustered Slepian-Wolf coding enabled by DOC can significantly reduce the total amount of data exchange in the whole network, and the transmission cost within each cluster can be remarkably reduced by performing the optimal intra-cluster rate allocation.

- **Spatially Selective Encryption**

  Based on our study on distributed data aggregation using Slepian-Wolf coding, we further proposed a combined data aggregation and encryption scheme using Slepain-Wolf coding for efficient and secured data transmission in wireless sensor networks. We have first studied the optimal intra-cluster rate allocation problem for Slepain-Wolf coding and then proposed the spatially selective encryption mechanism for data encryption within a single cluster based on the properties of Slepain-Wolf coding with optimal intra-cluster rate allocation. The proposed encryption mechanism only requires the cluster head to encrypt its data while allowing all cluster members to send their data without performing any encryption. Furthermore, an energy efficient key establishment protocol

is also proposed to securely and efficiently establish the key used for encrypting the data of the cluster head. Through simulation results, we have showed that the combined data aggregation and encryption scheme can significantly improve energy efficiency in data transmission while providing a high level of data security.

- **Clustered On-demand Multi-channel MAC Protocol**

  We have studied a cluster based on-demand multi-channel MAC protocol, which is proposed to provide better support for energy-efficiency, high-throughput, and data reliability in wireless multimedia sensor networks. We have proposed a scheduled multi-channel medium access within each cluster for contention-free intra-cluster communication. We have also proposed a scheduling algorithm to achieve better channel utilization under different traffic conditions and QoS requirements. We have incorporated a spectrum-aware ARQ scheme to enhance the transmission reliability. Our simulation results demonstrate that COM-MAC can achieve better network throughput and lower delay performance when compared with the baseline M-TDMA protocol at the cost of a small control overhead.

8.2 Future Work

- **Parameter choices in MCCP**

  For MCCP, we have appropriately set the values of the parameters α , β , and E_t in the cost metric. These parameters may have an impact on the performance of MCCP. Moreover, the re-clustering period of MCCP may also have an impact on the network performance. It is worthwhile to investigate on how different

parameter choices will affect the network performance.

- **Enhanced Robust Architecture for Underwater Sensor Networks**

  For the proposed robust architecture for underwater sensor networks, we adopted the *Weibull* distribution to model the life distribution of sensor nodes. Because a more precise prediction model will lead to a better knowledge of the robustness of the network architecture, it is important to construct and apply a statistical model based on experimental data derived from the dedicated sensing area so that the established architecture could fit well in the real underwater applications. Meanwhile, introducing redundant sensor nodes is also an effective scheme to increase the network robustness. This scheme, however, will increase the system cost. Therefore, combining our proposed scheme with the redundancy approach may yield interesting robust architectures for future applications.

- **Combined Clustered Slepian-Wolf coding and Network Coding**

  We have proposed a novel data aggregation mechanism based on Slepian-Wolf coding. In this research we have demonstrated that this scheme can significantly reduce the data redundancy, thus prolonging network lifetime. On the other hand, network coding is another promising approach to reduce data redundancy, which allows the transmission of mixed data over intermediate network nodes. The original messages can be properly recovered at the receiver end. Jointly considering these two coding schemes may lead to more advantages for enhancing energy efficiency for future sensor network applications.

- **Cluster Head Protection in Spatial Selective Encryption**

  We have addressed the security problem by proposing a novel encryption approach, called the spatial selective encryption. Using the proposed scheme, the

security of the whole network mainly depends on that of the cluster heads. To secure the data from cluster heads, Rabin's algorithm is adopted, which is a variant of the well-known RSA algorithm. Since Rabin's algorithm only aims at protecting data of the cluster heads from being revealed, how to avoid cluster heads being compromised by enemies should be investigated in the future so that the capture of a cluster head will not jeopardize the security of the whole cluster. There are two interesting schemes which can be considered: dynamically re-clustering and anonymous routing. In the first scheme, dynamically re-clustering could make the compromised cluster heads replaced by new ones, whereas in the latter one, anonymous routing could make the routing IDs concealed from enemies.

- **Multi-channel MAC with Time-varying Channels**

  A multi-channel MAC protocol has been proposed to support energy-efficient, high-throughput, and reliable data transmission. To maximize the network throughput, a traffic-adaptive scheduling algorithm has been proposed to dynamically allocate time slots and channels for sensor nodes based on the current traffic condition. The channels considered are homogeneous, time-invariant and have equivalent bandwidth. It is highly desired that a new optimal scheduling algorithm will be developed under more realistic channel conditions, such as time-varying heterogeneous channels.

References

[1] H. C. S. Choi and S. Cho, "A soc-based sensor node: Evaluation of retos-enabled cc2430," in *Proceedings of IEEE Communications Society Conference on Sensor, Mesh, and Ad-Hoc Commnucations and Networks(SECON 2007)*, San Diego, USA, June 2007, pp. 132–141.

[2] X. Corporation, *XBOW MICA2 Mote Specifications*, also available as *http : //www.xbow.com.*.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "wireless sensor networks: A survey," *Computer Networks (Elsevier) Journal*, vol. 4, no. 12, pp. 393–422, Mar. 2002.

[4] I. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: Research challenges," *Elsevier's Journal of Ad Hoc Networks*, vol. 3, no. 3, pp. 257–279, Feb. 2005.

[5] J. Heidemann, W. Ye, J. Wills, A. Syed, and Y. Lil, "Research challenges and applications for underwater sensor networking," in *Proceeding of IEEE Wireless Communications and Networking Conference (WCNC'06)*, Las Vegas, NV, Apr. 2006, pp. 63–72.

[6] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou, "Challenges: Building scalable mobile underwater wireless sensor networks for aquatic applications," *IEEE Network*, vol. 20, no. 3, pp. 12–18, May 2006.

[7] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, Oct. 2002.

[8] F. Salva-Garau and M. Stojanovic, "Multi-cluster protocol for ad hoc mobile underwater acoustic networks," in *Proceeding of IEEE OCEANS'03*, vol. 1, San Francisco, CA, Oct. 2002, pp. 660–670.

[9] O. Younis and S. Fahm, "Heed: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, Oct. 2004.

[10] S. Bandyopadhyay and E. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *Proceeding of IEEE INFOCOM'03*, vol. 3, San Francisco, CA, Apr. 2003, pp. 1713–1723.

[11] V. Mhatre and C. Rosenberg, "Homogeneous vs heterogeneous clustered networks: A comparative study," in *Proceeding of IEEE ICC'04*, vol. 6, Paris, France, June 2004, pp. 3646–3651.

[12] O. Younis, S. Fahmy, and P. Santi, "An architecture for robust sensor network communications," *International Journal of Distributed Sensor Networks*, vol. 1, no. 3, pp. 305–327, 2005.

[13] G.Gupta. and M.Younis, "Fault-tolerant clustering of wireless sensor networks," in *Proceeding of IEEE WCNC'03*, vol. 3, New Orleans, Louisiana, Mar. 2003, pp. 1579 – 1584.

[14] F.Kuhn, T.Moscibroda, and R.Wattenhofe, "Fault-tolerant clustering in ad hoc and sensor networks," in *Proceeding of IEEE International Conference on Distributed Computing Systems*, Sheraton Society Hill, Philadelphia, July 2006, pp. 68 – 78.

[15] O.Younis, S.Fahmy, and P.Santi, "An architecture for robust sensor network communications," *International Journal of Distributed Sensor Networks*, vol. 1, no. 3-4, pp. 305–327, 2005.

[16] Szu-Chi.Wang and Sy-Yen.Kuo, "Communication strategies for heartbeat-style failure detectors in wireless ad hoc networkss," in *Proceeding of international Conference on Dependable Systems and Network*, San Francisco, CA, June 2003, pp. 361 – 370.

[17] A. Syed and J. Heidemann, "Time synchronization for high latency acoustic networks," in *Proceedings of the IEEE INFOCOM'06*, Barcelona, Spain, Apr. 2006, pp. 1–12.

[18] M. C. Vuran, O. B. Akan, and I. F. Akyildiz, "Spatio-temporal correlation: Theory and applications for wireless sensor networks," *Computer Networks (Elsevier) Journal*, vol. 45, no. 3, pp. 245–261, 2006.

[19] D. Slepian and J. Wolf, "Noiseless coding of correlated information sourcess," *IEEE Transactions on Information Theory*, vol. IT-19, pp. 471–480, July 1973.

[20] T. M. Cover and J. A. Thoma, *Elements of Information Theory*, 1st ed. New York, NY, USA: John Wiley and Sons, Inc., 1991.

[21] D. Marco and D. L. Neuhoff, "Reliability vs. efficiency in distributed source coding for field-gathering sensor networks," in *Proceeding of the Third International Symposium on Information Processing in Sensor Networks (IPSN'04)*, Berkeley, CA, Apr. 2004, pp. 161–168.

[22] S. S. Pradhan, J. Kusuma, and K. Ramchandran, "Distributed compression in a dense microsensor network," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 51–60, Mar. 2002.

[23] A. Aaron and B. Girod, "Compression with side information using turbo codes," in *Proceeding of IEEE Data Compression Conference (DCC'02)*, Snowbird, UT, Apr. 2002, pp. 252–261.

[24] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (discus): Design and construction," *IEEE Transactions on Information Theory*, vol. 49, pp. 626–643, Mar. 2003.

[25] C. Lan, A. Liveris, K. Narayanan, Z. Xiong, and C. Georghiades, "Slepian-wolf coding of multiple m-ary sources using ldpc codes," in *Proceeding of IEEE Data Compression Conference (DCC'04)*, Snowbird, UT, 2002, p. 549.

[26] M. Sartipi and F. Fekri, "Distributed source coding in wireless sensor networks using ldpc coding: The entire slepian-wolf rate region," in *Proceeding of IEEE Wireless Communications and Networking Conference (WCNC'05)*, New Orleans, Louisiana, Mar. 2005, pp. 1939–1944.

[27] R. Cristescu, B. Beferull-Lozano, and M. Vetterli, "Networked slepian-wolf: theory, algorithms, and scaling laws," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4057–4073, 2005.

[28] ——, "On network correlated data gathering," in *Proceeding of IEEE INFOCOM'04*, Hong Kong, China, Mar. 2004, pp. 2571–2582.

[29] B. Han and W. Jia, "Wsn19-4: Efficient construction of weakly-connected dominating set for clustering wireless ad hoc networks," in *Proceeding of GLOBECOM'06*, San Francisco, CA, Nov. 2006, pp. 1–5.

[30] A. Youssef, M. Younis, M. Youssef, and A. Agrawala, "Wsn16-5: Distributed formation of overlapping multi-hop clusters in wireless sensor networks," in *Proceeding of GLOBECOM'06*, San Francisco, CA, Nov. 2006, pp. 1–6.

[31] Y. P. Chen and A. L. Liestman, "A zonal algorithm for clustering ad hoc networks," *International Journal of Foundations of Computer Science*, vol. 14, no. 2, pp. 305–322, Apr. 2003.

[32] T. Kanugo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "A local search approximation algorithm for k-means clustering," in *Proceeding of the 18th Annual ACM Symp. on Computational Geometry*, Barcelona, Spain, June 2002, pp. 10–18.

[33] A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huyn, "Max-min d-cluster formation in wireless ad hoc networks," in *Proceeding of IEEE INFOCOM'00*, Tel-Aviv, Israel, Mar. 2000, pp. 31–41.

[34] M. C. Vuran and I. F. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 316–329, 2006.

[35] B. Krishnamachari, S. Pattem, and R. Govindan, "The impact of spatial correlation on routing with compression in wireless sensor networks," in *Proceeding of International Symposium on Information Processing in Sensor Networks (IPSN'04)*, Berkeley, CA, Apr. 2004, pp. 28–35.

[36] D.R.Stinson, *Cryptography: Theory and Practice*, 2nd ed. United States of America: CRC Press, 2002.

[37] Eschenauer and V. Gligor, "A key management scheme for distributed sensor networks," in *Proceeding of 9th ACM Conf. Comp. and Commun*, Washington, DC, Nov. 2002, pp. 41–47.

[38] H. Chan, A. Perrig, , and D. Song, "Random key pre-distribution schemes for sensor networks," in *Proceeding of IEEE 2003 Symposium on Security and Privacy*, Oakland, California, May 2003, pp. 197– 213.

[39] M.F.Younis, K.Ghumman, and M.Eltoweissy, "Improving key pre-distribution with deployment knowledge in static sensor networks," *ACM Transactions on Sensor Networks*, vol. 1, no. 2, pp. 204–239, 2005.

[40] W. D. et al, "A pairwise key pre-distribution scheme for wireless sensor networks," in *Proceeding of 10th ACM Conf. Comp. and Commun. Security*, Washington, D.C., Oct. 2003, pp. 42– 51.

[41] G. J. et al., "A low-energy key management protocol for wireless sensor networks," in *Proceeding of IEEE 2003 Symposium on Security and Privacy*, Oakland, California, 2003, p. 335.

[42] I.F.Akyildiz, T.Melodia, and K.R.Chowdry, "A survey on wireless multimedia sensor networks," *Elsevier Computer Networks Journal*, vol. 51, no. 4, pp. 493 506, 2004.

[43] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated, adaptive sleeping for wireless sensor networks," *IEEE/ACM Transaction on Networking*, vol. 12, no. 3, pp. 493–506, 2004.

[44] G. Zhou, C. Huang, T. Yan, J. A. Stankovic, and T. F. Abdelzaher, "Mmsn: Multi-frequency media access control for wireless sensor networks," in *Proceeding of IEEE INFOCOM'06*, Barcelona, Spain, 2006, pp. 1–13.

[45] J. Sozer, M. Stojanovic, and J. Proakis, "Underwater acoustic networks," *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 72–83, Jan. 2000.

[46] Chvatal, "A greedy heuristic for the set covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233–235, 1979.

[47] L. Hu, "Distributed code assignments for cdma packet radio networks," *IEEE/ACM Transactions on Networking*, vol. 1, no. 6, pp. 668–677, Dec. 1993.

[48] D. Pompili, T. Melodia, and I. F. Akyildiz, "Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks," in *Proceedings of the 12th annual ACM international conference on Mobile computing and networking (MobiCom'06)*, New York, NY, USA, 2006, pp. 298 -309.

[49] W. Weibull, "A statistical distribution function of wide applicability," *J. Appl. Mech.-Trans. ASME*, vol. 18, no. 3, pp. 293–297, 1951.

[50] U. Feige, M. Halldrsson, G. Kortsarz, and A. Srinivasan, "Approximating the domatic number," *IAM Journal on Computing*, vol. 32, no. 1, pp. 172–195, 2003.

[51] P. Wang, C. Li, , and J. Zheng, "Distributed minimum-cost clustering protocol for under-water sensor networks (uwsns)," in *Proceeding of IEEE International Conference on Communications (ICC2007)*, Glasgow, Scotland, June 2007, pp. 3510–3515.

[52] T. Cormen, C. Leiserson, R. Rivest, and C.Stein, *Introduction to Algorithms*, 2nd ed. United States of America: MIT Press and McGraw-Hill, 2001.

[53] A. Scaglione and S. D. Servetto, "On the interdependence of routing and data compression in multi-hop sensor networks," in *Proceeding of the 8th Annual International Conference on Mobile Computing and Networking (Mobicom'02)*, Atlanta, Georgia, Sept. 2002, pp. 140–147.

[54] R.Zamir and T.Berger, "Multiterminal source coding with high resolution," *IEEE Transactions on Information Theory*, vol. 45, no. 2, pp. 106–117, 1999.

[55] X. Xiong, A. D. Liveris, and S.Cheng, "Distributed source coding for sensor networks," *IEEE Signal Processing Magazine*, vol. 21, no. 5, pp. 80–84, Sept 2004.

[56] J. P. Walters, Z. Liang, W. Shi, and V. Chaudhary, *Security in Distributed, Grid, and Pervasive Computing*, 1st ed. United States of America: CRC Press, 2006.

[57] M.F.Younis, K.Ghumman, and M.Eltoweissy, "'location-aware combinatorial key management scheme for clustered sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 8, pp. 865–882, 2006.

[58] D. Carman, P. Kruus, and B. Matt, "Constraints and approaches for distributed sensor network security," Tech. Rep., Sept. 2000.

[59] S.-L.Wu, C.-Y. Lin, Y.-C. Tseng, and J.-P. Sheu, "A new multichannel mac protocol with on-demand channel assignment for multi-hop mobile ad hoc networks," in *Proceeding of 2000 International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN '00)*, Dallas, TX, Dec. 2000, p. 232.

[60] J. So and N. Vaidya, "Multi-channel mac for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver," in *Proceeding of ACM MobiHoc*, Roppongi, Japan, May 2004, pp. 222–233.

[61] P. Bahl, R. Chandra, , and J. Dunagan, "Ssch: slotted seeded channel hopping for capacity improvement in ieee 802.11 ad-hoc wireless networks," in *Proceeding of the 10th Annual International Conference on Mobile Computing and Networking (MOBICOM '04)*, Philadelphia, PA, Sept. 2004, pp. 216–230.

[62] IEEE, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, 1997.