# Wireless Sensor Network as a Distribute Database

By

**Weiwei He**

**A Doctoral Thesis**

Submitted in partial fulfilment

of the requirements for the award of

**Doctor of Philosophy**

**of**

**Loughborough University**

May 2016

# Acknowledgement

I would like to express my gratitude to all those who gave me guidance and helped to complete this thesis.

I want to thank my supervisor, Professor Shuang-hua Yang from the Computer Science Department, for giving me supervision and support to do the necessary research work and to clarify my confusion about both detailed research work and research directions. I have furthermore to thank my supervisor, Dr Lili Yang from the Business school, who gave me useful suggestions and guidance, as well as flexibility to choose what I want to do, and provided great help whenever it is needed.

Our group supported me in my research work. I want to thank them for all their help, support, interest and valuable hints.

Especially, I would like to give my special thanks to my parents, my husband and my little daughter to support my studies and encouragement that enabled me to complete this work.

**Thank you all!**

# Abstract

Wireless sensor networks (WSN) have played a role in various fields. In-network data processing is one of the most important and challenging techniques as it affects the key features of WSNs, which are energy consumption, nodes' life circles and network performance. In the form of in-network processing, an intermediate node or aggregator will fuse or aggregate sensor data, which are collected from a group of sensors before transferring to the base station. The advantage of this approach is to minimize the amount of information transferred due to lack of computational resources.

This thesis introduces the development of a hybrid in-network data processing for WSNs to fulfil the WSNs constraints. An architecture for in-network data processing were proposed in clustering level, data compression level and data mining level. The Neighbour-aware Multipath Cluster Aggregation (NMCA) is designed in the clustering level, which combines cluster-based and multipath approaches to process different packet loss rates. The data compression schemes and Optimal Dynamic Huffman (ODH) algorithm compressed data in the cluster head for the compressed level. A semantic data mining for fire detection was designed for extracting information from the raw data by the semantic data-mining model is developed to improve data accuracy and extract the fire event in the simulation. A demo in-door location system with in-network data processing approach is built to test the performance of the energy reduction of our designed strategy. In conclusion, the added benefits that the technical work can provide for in-network data processing is discussed and specific contributions and future work are highlighted.

**Keywords: sensor data processing, data aggregation, in-network, data compression, data mining, wireless sensor network.**

# Publications

W. He, S. Yang and L. Yang, “Real-time data mining methodology and emergency knowledge discovery in wireless sensor networks”, PGNet, 2010.

W. He, S. Yang and L. Yang, “NMCA: Neighbour-aware multi-path clustering aggregation in wireless sensor networks”, In Networking, Sensing and Control (ICNSC)’ 10th IEEE International Conference, 2013. (Best student paper)

W. He, S. Yang and L. Yang, “In-Network Data Processing Architecture for Energy Efficient Wireless Sensor Networks”, WF-IoT, 2015.

# Abbreviations

| | | |
|---|---|---|
| BBN | : | Bayesian Belief Network |
| CAG | : | Cluster Aggregation |
| CDP | : | Compression Data Packet |
| CH | : | Cluster Head |
| CM | : | Cluster Member |
| DB | : | Database |
| DC | : | Distributed Compression |
| DCA | : | Double-path Cluster Aggregation |
| DCS | : | Data Centric Storage |
| DD | : | Directed Diffusion |
| DP | : | Double-path |
| ECR | : | Event Condition Rule |
| ED | : | Episode Discovery |
| ERS | : | Emergency Response System |
| GN | : | Gateway Node |

| | | |
|---|---|---|
| GPS | : | Global Positioning System |
| GUI | : | Graphical User Interface |
| IEEE | : | Institute of Electrical and Electronic Engineers |
| KDD | : | Knowledge Discovery in Database |
| LCVC | : | Low-Complexity Video Compression |
| LEACH | : | Low-Energy Adaptive Clustering Hierarchy |
| LZO | : | Lempel-Ziv-Oberhumer |
| MAX-P | : | Max-path |
| NCS | : | Neighbour-aware Data Compression |
| NMCA | : | Neighbour-aware Multi-path Cluster Aggregation |
| NoA | : | No-aggregation Algorithm |
| ODH | : | Optimal Dynamic Huffman |
| PCA | : | Principle Component Analysis |
| PINCO | : | Pipelined In-network Compression |
| RDR | : | Regular Data Report |
| RF | : | Radio Frequency |
| RFID | : | Radio Frequency Identification |
| RSSI | : | Received Signal Strength |
| RTLS | : | Real Time Location System |

| | | |
|---|---|---|
| SC | : | Sampling Compression |
| SP | : | Single-path |
| SQL | : | Structured Query Language |
| TDoA | : | Time Difference of Arrive |
| TN | : | Total Nitrogen |
| ToA | : | Time of Arrive |
| TOF | : | Time of Fight |
| UEN | : | Urgent event notification |
| UI | : | User Interface |
| UWB | : | Ultra Wide Band |
| WPAN | : | Wireless Personal Area Network |
| WSN | : | Wireless Sensor Network |

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1. Introduction

## 1.1 Background to the Research

Wireless sensor network (Lewis, 2004) (WSN) is the collections of inexpensive, battery powered and wirelessly networked electrical devices with sensing hardware for measuring various environmental elements such as light, temperature, gas density, pressure and vibration etc. WSNs are an emerging technology for providing intelligent ubiquitous computing environments. WSNs consist of an amount of autonomous nodes that operate without human interaction (e.g. configuration of network routes, recharging of batteries, tuning of parameters) for weeks or months. It can gather or process data over prolonged periods in potentially inhospitable environments. The benefit of a WSN is that it allows to remotely observing the physical information of the environment.

WSNs consists of low cost, resource limited sensor nodes that can be randomly distributed across areas of varying sizes to autonomously form wireless networks (Tanenbaum, 2003). There are many applications monitoring the physical world by collected and processed sensor data: supervising items in a factory warehouse, organizing vehicle traffic in a large city, monitoring earthquakes in shake-test sites, and monitoring habitat: petrels (birds) on Great Duck Island (Mainwaring et al., 2002). WSN nodes collect data for the surrounding environment and share information collected with other nodes in the same network. The whole WSN can be viewed as a sensor network database. The sensor network database is a distribution database that collects physical measurements about the environment, indexes them, and serves queries from users and other applications external to or from within the network. Each sensor in WSN takes time-stamped and location (store data) measurements of physically phenomena (sensor data), e.g. temperature, light etc.

In most applications of WSNs, the end-users are interested in events. The scope for potential WSN based applications is enormous, existing applications include, patient monitoring in hospitals, tracking of military ordinance, location tracking, environmental monitoring in extreme or remote locations such as near volcanoes or in the tropical rainforest respectively, and home automation (Akyildiz et al. 2002).

The work outlined in the thesis primarily focuses on data processing for WSNs. The features of WSNs are suitable for a wide range of application areas. Depending on the applications, nodes of a sensor network may generate huge volumes of data and queries that may prove difficult to process for any traditional database. The distributed sensor network, and the new types of queries that need to be formulated in a sensor network information processing system, brings new in-network data processing and manages issues which cannot be solved directly by existing database techniques.

## 1.2 Motivation

WSN has great potential of providing information about the environment. How to deal with sensor data has become a new research topic, which is receiving gradually increased attention, as its benefits have been demonstrated in diverse applications such as environment monitoring, industrial sensing, military practice, emergency response, etc.

There are two approaches to implement a sensor network database. First, traditional management of data and context has been carried out using a central approach, i.e. a warehouse approach. In this approach, data is extracted from the sensor network in a predefined way and is stored in a database located on a unique front-event server. The data is then processed centrally to extract the required information. The disadvantages are that the nodes near the access point become traffic hot spots, central points of failure and may be depleted of energy prematurely. Due to these disadvantages, the central approach may not be suitable for a wireless embedded distributed system. The only viable alternative is to migrate from off-line processing to processing the data within the network to get as close to the data source as possible. Hence, the second approach is an in-network (distribute) approach (Govindan et al., 2002), which stores the data within the network itself and allows queries to inject anywhere in the network. The approach only extracts relevant data from the sensor network and allows data to aggregate before it is sent to an external query. In-network processing has been proven as one of the most energy efficient query processing paradigms for WSNs, where the processing is conducted inside the sensor network and close to the sources of data generation.

This research was driven by the motivation to contribute for in-network data processing. It aims to design and implement an architecture based on sensor data for widely range applications from the WSN, to provide reliable, quality and effective information to incident commanders or other related users. Depending on the application, nodes of a sensor network may generate huge volumes of data and queries that may difficultly be processed in any traditional database. The distributed sensor network and the new types of queries that need to be formulated in a sensor network information processing system brings out new in-network data processing and manage issues which cannot be solved directly with the existing database technique.

The technique to view WSN as a distributed database assumes that only a minimal amount of data is transferred from and within the network. It includes an intermediate node or aggregator fuses or aggregates sensor collected from a group of sensor nodes before sending the processed data to the base station. In recent researches, the most data processing techniques only can be used in one or several similar specific applications. For different applications, we have to propose a data processing approaches for both environment and human behaviour monitoring. In this thesis, we propose a suitable data processing architecture for emergency response system (environment monitoring) and in-door location tracking system (human behaviour monitoring).

## 1.3 Research Objectives and Questions

The research aims focus on the design and implementation of sensor data processing approaches that are suitable for widely range of different application and fulfil the WSNs constraints.

The idea is to embed data processing capabilities in the sensor network in order to reduce the amount of data the needs to be transmitted, and guarantee the data accuracy of processing. This thesis proposes an in-network data processing architecture to perform data aggregation, data compression and data mining in the network for maximum energy and quality benefits. The major applications requirements are network lifetime, event detection, information quality, data timeless and self- organization. Those are effected by energy consumption, heterogeneity, data accuracy, real-time and unpredictable. The data processing architecture can suitable for both environment and human behaviour monitoring applications

The list of research questions considered, and the specific research objectives associated with each research question to obtain the research aim, are described as follow:

RQ1: What are the factors that need to be considered for WSN constraints?

Objective associated with RQ1: Investigate the existing literatures to understand the restrictions of wireless communication, i.e., energy consumption, heterogeneity, data accuracy.

RQ2: How can the technology capabilities of WSNs be implement in different applications (emergency response system and in-door location system) and mitigate the WSN constraints?

Objective associated with RQ3:

- Understand the goals, tasks and information requirements from the chosen user group raised form the WSNs constraints.
- Research the existing literature available on data processing for sensor data within the sensor network.
- Design a suitable in-network data processing architecture for both environment and human monitoring applications

RQ3: What constitutes a suitable data aggregation to mitigate WSN constraints within the network?

Objective associated with RQ4:

- Propose a suitable data aggregation technique for sensor data to limit the communication of unnecessary events for energy consumption, improve the data accuracy and deploy the flexible structure for network self-organization
- Evaluate the efficiency of the proposed data aggregation and its cost-effectiveness

RQ4: What constitutes a suitable data compression to mitigate WSN constraints within the network?

Objective associated with RQ5:

- Analyse and propose a suitable data compression for data mining for both alert and regular sensor data to reduce energy consumption by minimizing data volume and represent sensor data
- Evaluate the effectiveness of the proposed data compression approach in comparison to the existing approaches

RQ5: How to extract information from sensor data to mitigate WSN constraints within the network?

Objective associated with RQ6:

- Analyse and specify what “meaning” can be in the context of WSN-based onsite emergency response system, and propose a data mining approach.
- Evaluate the performance of the proposed meaning extraction approach and discuss its application

## 1.4 Contributions of the research

**Contribution 1:** The design of in-network data processing architecture that addresses three levels of in-network data processing: data clustering level, data compression level and data mining level. The proposed architecture can mitigate the WSN constraints. Clustering level can be done to establish the cluster for the network and only aggregate and transmit the interested data in the cluster head, which is to limit the communication of unnecessary events for energy consumption, improve the data accuracy and deploy the flexible structure for network self-organization. Data compression level can reduce energy consumption by minimizing data volume and represent data for data mining. Due to the sensor data must be semantically annotated, data mining level is used to process data form heterogeneous sources to get the acceptable knowledge for the specify users.

**Contribution 2:** A Proposal Neighbour-aware Multi-path Cluster Aggregation

(NMCA) presented for clustering level, which combines cluster-based and multipath approaches for processing different packet loss rates. It overcome the problems of tree-based, cluster-based and multipath schemes and improves the energy efficiency and data accuracy of the network. Simulation results demonstrated that the approach could improve data accuracy and reduce energy efficiency and data throughput.

**Contribution 3:** The proposal of communication compression schemes for the event and the regular packets and Optimal Dynamic Huffman algorithm for data compression. The most significant advantages of designing WSN are how to improve the energy efficiency and make the compression algorithm lightweight for the intermediate sensor nodes within the network. The in-network data compression techniques can significantly decrease the whole network energy consumption. Neighbour-aware scheme is high performance for several events at the same time. It is the communication compression technique, which reduces the energy consumption and response quickly in the event detection. Optimal Dynamic Huffman algorithm can be easily employed in practical and any wireless sensor network for in-network data compression.

**Contribution 4:** A semantic real-time data-mining model for emergency event detection. This utilises the semantic annotator, knowledge base and data mining rule engines to detect fire and improve the detection accuracy. The semantic process is applied the data format structure and outlier detection to pre-process data for data mining rules. Event detection process is used the threshold-base rules from the knowledge base to extract the event by data mining rules engines. Simulation demonstrated that applying this mode could improve efficiency and accuracy of the existing event detection algorithms.

**Contribution 5:** A case study for in-door location tracking system embed in-network data processing. The designed in-network data processing architecture has been tested and successfully implemented in the human behaviour monitoring and helps to reduce the energy consumption for the entire system with 13% maximum per hour compared with a plain system without the strategy.

All of the contributions aim to create suitable in-network data processing approaches to help improve the energy efficiency and other constraints of WSNs for both environment and human behaviour monitoring application.

## 1.5 Organization of the Thesis

The thesis was organized as follows.

Chapter 1 (this chapter) introduces the background to the research and puts the focus of the thesis into context. It also identifies the potential challenges to and the motivation behind the undertaking of the research. The scope of the thesis is outlined, and the aims and objectives of the research are identified. In addition, the research questions of the thesis demonstrating the link between the chapter and the contributions of the thesis are summarized.

Chapter 2 is a literature review of wireless sensor network. The two main purposes of the literature review are to understand the background and overview of WSNs, and present the constraints of WSN and the research method for data processing. The in-network data processing is aggregated by clustering and collaborated with the surrounding sensor nodes, compressed data in cluster head and performed mining and integration of events on the gathered knowledge in the gateway node.

Chapter 3 investigates the existing literature on in-network data processing in WSNs, as it has been identified that it remains a challenge to make sense of the large amount of data collected from WSNs. The three different in-network data processing techniques for mitigate the constraints of WSN are reviewed in three sections. Arguments on what constitutes a suitable sensor data processing approach are summarized, and general recommendations of suitable sensor data processing approach derived.

Chapter 4 proposes the architecture for in-network data processing. It includes three levels: clustering level, compression level and data mining level. The data clustering level was further developed and applied hybrid approaches: multi-path aggregation and data clustering. Data compression level is to reduce the size of transmission packet for energy efficiency. The data mining level is to integrate semantic data for gathering acceptable information for event detection. It addresses the design requirements and initial ideas for different levels. Those three levels are presented in the following three chapters.

Chapter 5 presented NMCA (neighbour-aware multi-path cluster aggregation) which combines cluster-based and multipath approaches for processing different packet loss rates. Simulation results demonstrated that the approach could improve data accuracy and reduce energy efficiency and data throughput.

Chapter 6 presents a communication compression scheme for both event and regular packets in the network and proposed Optimal Dynamic Huffman algorithm to compress data in the cluster head, then decoding in the gateway node. It reduces the energy consumption and has less compression ratio compared with other algorithms in the area.

Chapter 7 proposes semantic real-time data mining model for emergency response system, which includes receiver and transmitter, knowledge base, semantic annotation and data mining rule engine. In the annotation process, we have pre-processed the raw data form the network and reduced the outliers. In the event detection process, we design an event-rule model for extracting information from the data.

Chapter 8 introduces the scenario of in-door localisation application, followed by the algorithms used in a formal location and tracking system. This chapter combines the research of in-network data processing that has been taken through this thesis with a lab test. It helps to optimise the performance of the project system, and proves the value of this research.

Chapter 9 concludes the thesis. This states what are felt to be the most important contributions of the thesis, and identifies future research questions that have emerged from the research contained in this thesis.

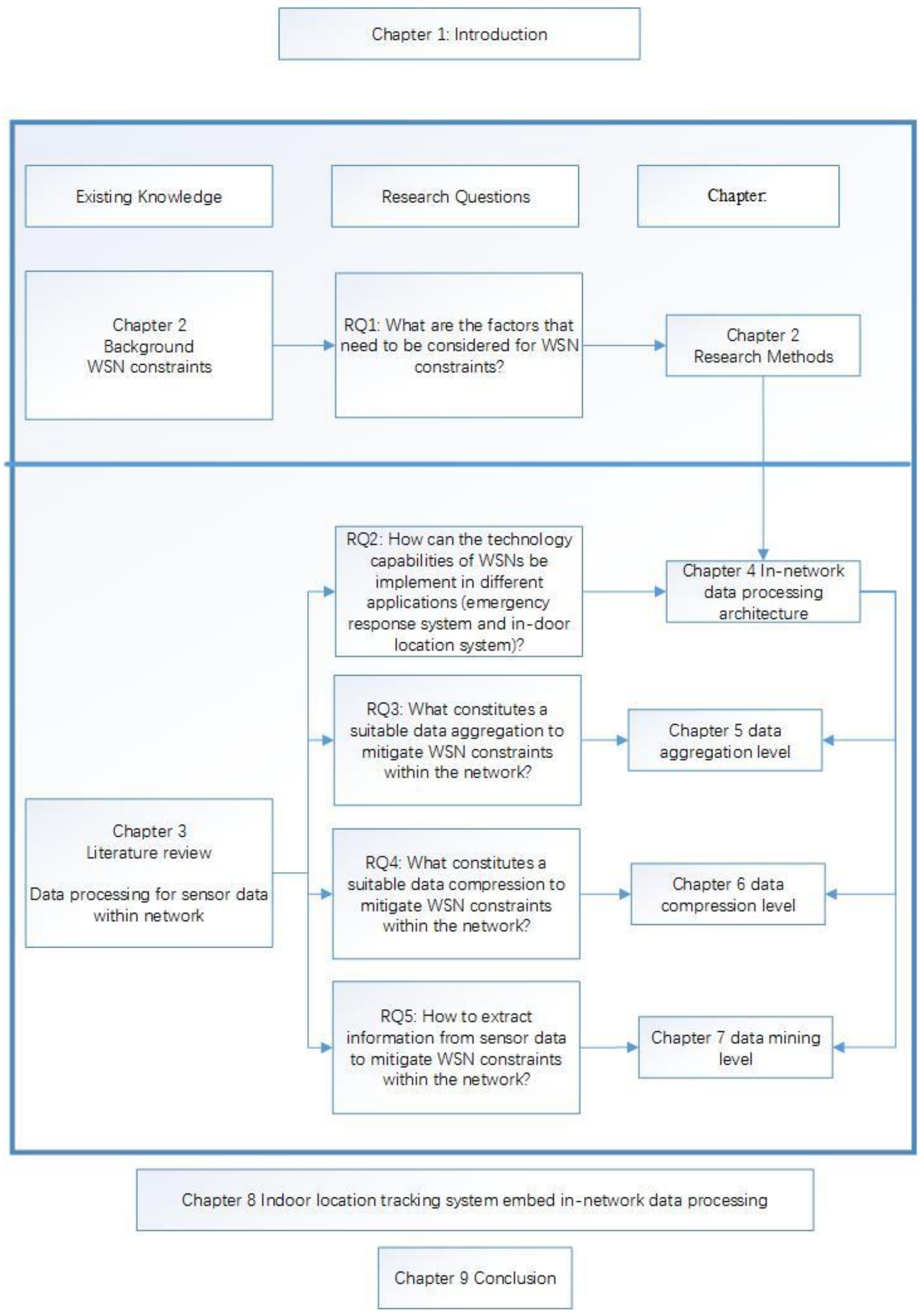

Chapter 1: Introduction
Existing Knowledge
Research Questions
Chapter
Chapter 2 Background WSN constraints
RQ1: What are the factors that need to be considered for WSN constraints?
Chapter 2 Research Methods
RQ2: How can the technology capabilities of WSNs be implement in different applications (emergency response system and in-door location system)?
Chapter 4 In-network data processing architecture
RQ3: What constitutes a suitable data aggregation to mitigate WSN constraints within the network?
Chapter 5 data aggregation level
Chapter 3 Literature review
Data processing for sensor data within network
RQ4: What constitutes a suitable data compression to mitigate WSN constraints within the network?
Chapter 6 data compression level
RQ5: How to extract information from sensor data to mitigate WSN constraints within the network?
Chapter 7 data mining level
Chapter 8 Indoor location tracking system embed in-network data processing
Chapter 9 Conclusion

*Figure 1-1: Structure of thesis*

# Chapter 2. Background

This chapter provides a comprehensive review current state of WSNs and application requirements for data processing. Then we present adopted research methods.

This chapter is structured as follows: Section 2.1 reviews an overview of wireless sensor network and its application respectively. Section 2.2 provides the necessary requirements to be an efficient WSN in different applications. Then Section 2.3 explain initial research solutions for which are identified in previous section. The functional requirements for an energy efficient and data corrected heterogeneous WSN are presented in Section 2.4. The chapter concludes with a summary in Section 2.5.

## 2.1 Wireless sensor network (WSN)

A WSN is composed of low cost, low power, multifunctional sensor nodes that are small and communicate wirelessly over short distances (Akyildiz and Su, 2002). The sensor nodes collaborate to sense and collate information about the environment, through a set of transducers and radio receiver (Barontib et al. 2007), and to forward information towards a central gateway node. Furthermore, information from the sensor nodes is accessed via a gateway sensor node that is depicted in Figure 2-1. It also includes sensor node, sink node, a connection to the Internet or satellite and a task manager node. Data are collected at the wireless sensor node, compressed and transmitted to the gateway. Individual sensor nodes are connected to other nodes in their neighbour and servers or other terminal unit through a wireless network. They use a multi-hop routing protocol to communicate with spatially distant nodes. After primary processing, the data gathered from the sensor field are sent to a base station (gateway node/sink), which is responsible for transferring data to another network. This function makes a sink similar to a gateway in a traditional network. (Akyildiz and Su, 2002)

This is a greater deal of flexibility when it comes to deployment choices for WSNs compared to other networks. Sensor nodes do not have a fixed location and most of them are randomly deployed to monitor a sensor field.

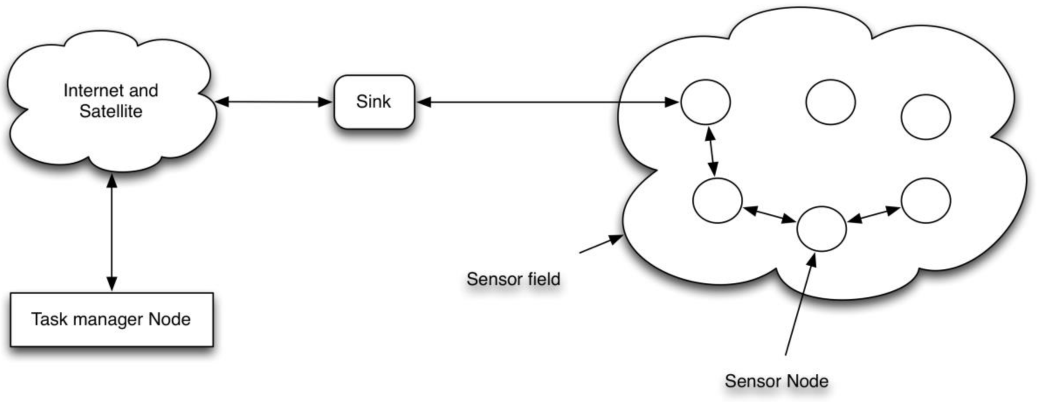


*Figure 2-1A generic WSN architecture (Akyildiz and Su, 2002)*

### 2.1.1 WSN Nodes

As depicted in Figure 2-2, a WSN node consists of five main components, a processing unit, memory, transceiver, sensors and power supply. The processor unit is responsible for making the WSN node to communicate with other sensor nodes, and execute application code. The memory unit can store the node’s programs, including the network stack and application programs. The transceiver allows the node to communicate with the neighbour nodes. One of the most important components of a WSN node is power supply, which provides the node with life, and is normally limited so that once the node's power is exhausted the node can no longer operate. The sensor component consists of two parts: first, there is the analogue sensing component that physically measures environmental characteristics such as temperature; second, there is the analogue to digital converter that transfers the environmental readings into a digital representation that can be handled by the node processor.

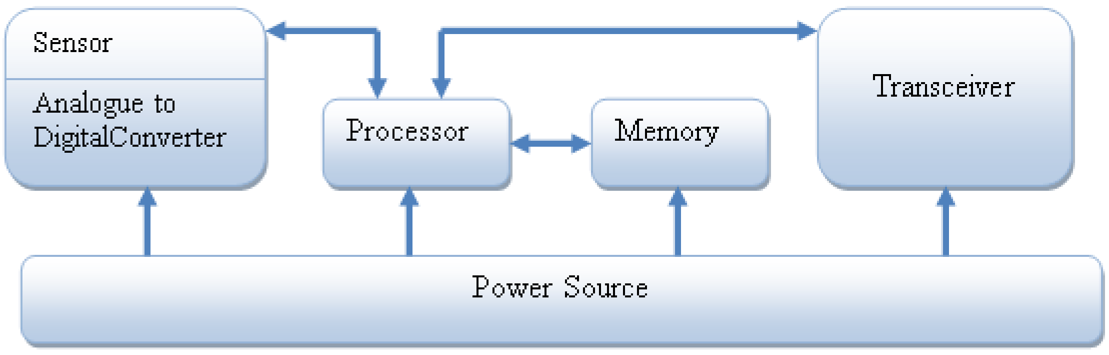


*Figure 2-2 The structure of a WSN node (Benini et al., 2006)*

A wireless node may consist of one or more sensors. There are numerous different types of sensors produced by different manufacturers. However, the primary objective of sensors is to monitor their immediate environment for a wide variety of ambient conditions. Here is a short list of the ambient conditions monitored by existing nodes that have been identified by (Akyildiz and Su, 2002) and (Lewis 2004):temperature, humidity, vehicular movement, lightning conditions, pressure, soil component, noise levels, the presence or absence of certain kinds of objects, mechanical stress levels on attached objects, and so on.

### 2.1.2 Data Communication in WSNs

Typical communication distance for low power wireless radios ranges from a few feet to around 100 feet, depending on transmission power and environmental conditions. The short ranges mean that almost all real deployments must make use of multi-hop communication, where intermediate nodes relay information for their peers.

Radio is a broadcast medium. It can be used in the operation system to transmit message in a particular node. Nodes receive per-message, link-level acknowledgments indicating whether the intended neighbour node received a message. No end-to-end acknowledgments are provided.

The requirements that sensor networks must be low maintenance and easy to deploy, mean that communication topologies must be automatically discovered by the devices rather than fixed at the time of network deployment. Typically, devices keep a short list of neighbours that they have heard transmit recently, as well as some routing information about the connectivity of those neighbours to the rest of the network. To assist in making intelligent routing decisions, nodes associate a link quality with each of their neighbours. We describe a simple ad-hoc routing protocol called tree-based routing in the following.

A basic primitive in many data dissemination and collection protocols is a routing tree, which is a spanning tree rooted at a particular node over the radio-connectivity graph of the network. A routing tree allows a base station at the root of the network to disseminate a query and collect query results. This routing tree is formed by forwarding a routing request from every node in the network: the root sends a request, all child nodes that hear this request process it and forward it on to their children, and so on, until the entire network has heard the request. Each request contains a hop-count, or level indicating the distance from the broadcaster to the root. To determine their own level, nodes pick a parent node that is (by definition) one level closer to the root than they are. This parent will be responsible for forwarding the node's query results, as well as those of its children, to the base station. We note that this type of tree-based communication topology is common within the sensor network community. Finally, note that it is possible to have several routing trees if nodes keep track of multiple parents. This technique can be used to support several simultaneous queries with different roots.

### 2.1.3 WSN Applications

The integration of sensing, processing and communication components into a small, battery-powered sensor node opens the door to a wide range of real-world application ( , 2007). There is no single set of design system to fulfil the different range of WSN applications. Therefore, there is a range of diverse applications that lend themselves to WSNs. (Akyildiz and Su, 2002) summarized these under two main application areas: environment monitoring (e.g. military, home and other commercial areas), and human behaviour monitoring (e.g. health, military, location tracking).

- Environmental monitoring

WSN applications for environmental monitoring and control have numerous benefits for scientific communication and for society as a whole (Cardell-Oliver, 2005 and Antoine-Santoni, 2006). It should be a very important part of the Internet of Things (IoT). Environmental applications of WSNs are deployed in monitoring the movement of wildlife such as birds, monitoring factors that affect the growing of crops, detecting and tracking fire in our project, and measuring levels of pollution across distributed locations. These applications can involve both indoor and outdoor environments, which may consider huge monitoring areas (i.e. hundreds or thousands square kilometres) and may remain long periods (i.e. months or years) of monitoring and the give a long-term result or trends for users. Hundreds or thousands of sensors may deploy in different areas in order to collect the whole information and knowledge about the environments. An example of a WSN environmental monitoring is indoor living monitoring. (Chanin, 2008) designed a detect intruder motion by using TMote Sky Platform as the sensor nodes.

The environmental monitoring often requires the system to sense and respond to the environment information. The sensor nodes collect data in the continuous periods, and then transmit back to gateway nodes for further analysis. Typically, these applications require low sampling rates. Because the common factors such as temperature, humidity and light data do not change quickly in regular periods. However, the most important requirement of environmental monitoring is how to extend the network life due to these applications often operating in unattended areas for long periods. In

Chapter 4, the fire detection case study is the category of environmental monitoring applications.

- Human behaviour monitoring

WSN has the ability to track and detect patterns in their surroundings, which makes it to use in surveillance applications. Military applications (Durisic et al., 2012) are one of the major areas in battlefield surveillance, which require fast deployment, and self-organizing capabilities of WSNs. On the other hand, WSNs can be deployed in the buildings, airports, shops or homes for monitoring the human habits or useful information by sensing and transmitting the detected data to the gateway nodes or station. Surveillance application (Seo, 2014) must immediately report the sensed data back to the gateway in a reliable and timely fashion, which is different with the environmental monitoring applications. It is also crucial for this category of applications to locate and track selected moving objects within WSNs. We will address a case study of an Ultra-Wide Band based indoor tracking system in Chapter 9.

Health care (Alemdar and Ersoy, 2010) is one of the most promising applications in WSNs that provides the potential to offer these vulnerable groups a greater level of independence in their lives. Potential applications include integrated control of household devices from a single controller, remote patient monitoring and better real time monitoring of patients in hospital. The integration of WSN with health-care applications is highly convenient and beneficial not only for doctors, but also for patients and disabled people. WSNs can be used to monitor people’s movement, habits or health information with the tracking and sensing devices, then reporting this information back to the medical centre or the relevant authorities.

Similar to surveillance applications, health care applications also require high accuracy to track the location of patients and medical personal, and have to return the data in a timely fashion. However, a longer lifetime is not as critical as in environmental applications. Because the wearable sensor node batteries can be replaced easily by patients or staff.

The work in this thesis focuses on the challenges facing the applications for both environmental and human behaviour monitoring. The next section explores the constraint of WSN, which is the challenges for data processing in WSN.

## 2.2 Constraints of WSN

WSN (Lewis, 2004) is the collection of inexpensive, battery powered and wirelessly networked electrical devices with sensing hardware for measuring various environmental elements such as light, temperature, gas density, sounds. They promise to measure the world at remarkably fine granularity. WSNs consist of amount of autonomous nodes that operate without human interaction (e.g. configuration of network routes, recharging of batteries, tuning of parameters) for weeks or months. A sensor network database is a distributed database that collects physical measurements about the environment, indexes them, and serves queries from users and other applications external to or from within the network. Each sensor in WSN takes time-stamped and location (stored data) measurements of physically phenomena (sensor data), e.g. temperature, light etc. The sensor network database is the combination of sensor data and stored data in its data model and query languages. There are many applications monitoring the physical world by querying and analysing sensor data: supervising items in a factory warehouse, organizing vehicle traffic in a large city, monitoring earthquakes in shake-test sites, and monitoring habitat: petrels (birds) on Great Duck Island (Mainwaring et al., 2002).

The features of WSNs are suitable for a wide range of application areas. In military applications, WSNs are ideal for the task of battlefield surveillance, because they provide a low risk level for personal. For environmental monitoring applications, WSNs are often used for building environment monitoring, home automation, industrial control, and assets management in logistic industry, etc. By receiving the characteristics of WSNs and the corresponding application areas, the challenges for developing. This section discusses the limitations the complicate the design of WSNs specifically for different applications. We review recent literatures and give the summary of WSNs applications and their requirements as followed.

### 2.2.1 Energy Constraints

Energy consumption in WSNs is a significant problem, which is due to the power restrictions and limited radio range of the sensor nodes. Since deployment of WSNs is supposed to be random and requires little or no infrastructure involvement, the power supply for driving wireless sensor nodes is mainly provided by a battery (Qi et al., 2002). This is a most important factor which seriously limits the use of WSNs. WSNs are designed to work in unattended area or work along over a considerable long period of time as frequent battery replacement might not be easily achieved. For environmental monitoring applications, it requires more efficient energy consumption then other applications, which is shown in Table 2.1.

Sensor nodes consume power for various operations: running sensor board, receiving and transmitting information or data, processing data and regular remaining. In Table 2.2, it shows communication between sensor nodes consumes most energy compared to sensing and computing. In fact, the cost of transmitting one bit is equal to the energy of 800-1000 data processing instructions (Hill, 2000). A battery normally powers the transmitter and receiver used by the wireless sensor node. Among the typical components composing wireless sensor node, the radio transmitter consumes the most energy.

Since current technology cannot provide a long-term power supply without replacing the battery, WSNs often limit the transmission power as an effective way to save energy use on wireless sensor node (Cardei and Wu, 2006). Consequently, the effective transmission range of the WSN nodes is restricted. The sensor nodes do not have enough power or communication range to send messages directly to the gateway nodes.

*Table 2-1* Power model for Mica2 (Shnayder, 2004)

| Mode | Current (mA) |
|---|---|
| **Receive** | 7.00 |
| **Transmit with max power** | 21.50 |
| **CPU(active)** | 8.00 |
| **CPU (idle)** | 3.20 |
| **Sensor board** | 0.70 |
| **LEDs** | 2.20 |

### 2.2.2 Heterogeneity

The types of sensors contained in the WSNs determine whether a network is heterogeneous or homogeneous. We call the network homogeneous when it deploys all sensors to measure the same phenomenon. But in most applications, the network is capable measuring different phenomena, which we call heterogeneous (Mhatre and Rosenberg, 2004).

The distributed WSNs have given rise to monitor various facilities and to extract knowledge form disparate sensor nodes in a meaningful manner.

Heterogeneity can occur at different levels: system, structure, syntax and semantics (Sheth, 1999). System heterogeneity is caused by various hardware and software components, which includes different types of sensor nodes with different capabilities and functions. Structure heterogeneity refers to different storage structures and data model in the WSNs. Syntactic heterogeneity corresponds to various data representation and formats. Semantic heterogeneity is related to the same concept have different meanings in different WSNs.

### 2.2.3 Real-Time Operation

Real-time operation describes how timely the data and provide to be useful to applications (Blasch and Plano, 2005). This attribute is especially important for monitoring applications, which require either continuous real-time monitoring or periodical-based and conational-based online analytical results. Most of these applications, such as battlefield surveillance, disaster and emergency response, deal with various kinds of real-time constraints in response to the physical world.

To compare with the traditional distributed systems, the real-time operation may have many challenges as follows. First, sensor networks directly interact with the real physical properties, which is unpredictable and hard to characterize. Second, it should not be efficient to active all sensor nodes all the time for the network fast response.

### 2.2.4 Data Accuracy

Each sensor node will generate sensory data and transfer to a specified task manager node for further processing. Because of the characteristics of wide deployment and limited wireless communication range, the implementation of data acquisition and transfer requires the involvement of dedicated communication protocols. To improve distributed data and query accuracy is very important to sensor networks (Elnahrawy and Nath, 2003).

### 2.2.5 Unpredictability

WSN may be very unpredictable in their operations. A wireless sensor network often consists of a large number of sensor nodes in order to provide an effective sensor field. They can easily cover a relatively wide area. This characteristic makes it impossible for users to maintain the whole network manually. Comprehensive management architecture is required to monitor the WSNs, configure network parameters and implement system updating (Wagenknecht et al. 2008).  Moreover, the topology of WSNs may not be static in the network area. Sensor nodes can easily die and new sensor nodes may be randomly added to the network. All of these require that the sensor network should have the ability to adjust itself when the topology of network has changed (Bharathidasan and Pondurn, 2003). Thus, the network needs to adapt to changing conditions and requirements in order to remain its operations.

The constraints of WSN due to the limited resources of the sensor nodes address the challenges faced by the WSN for data processing. In the following section, we present the potential existing data processing approaches to mitigate WSN constraints.

## 2.3 Techniques to Mitigate WSN Constraints

There are various studies on information processing in sensor network (Brooke and Burrell, 2003; Cerpa et al. 2001, Juange, 2002), the authors approached a sensor network as a distributed database and proposed pushing of declarative queries into the sensor network, instead of executing them in a central location, with the aim of decreasing energy consumption during the processing. In WSN recent research, it is addressed as in-network data processing, which can reduce energy consumption by minimizing data volume locally and transmit the processed data in real-time.

This section briefly describes the research techniques for in-network data processing used to solve the problems faced by WSNs in term of energy consumption, real-time operation, data accuracy and heterogeneity.

### 2.3.1 Data aggregation

In-network data aggregation (Silberstein, 2007) is deployed at intermediate nodes for merging data from multiple homogeneous sensors and then transmitting the aggregated data to the gateway. It leads to substantial increases energy efficiency in the network.   In the aggregation operator, summarization functions such as minimum, maximum, or average are applied to reduce the volume of data (Madden, 2005). It helps to remove redundant data and improve the reliability of the information gathered from multiple sensors. Figure 2-3 is the structure of in-network data aggregation. Depending on application requirements, it can be categorised into three major tasks: flirtation, aggregation and integration of data, to reduce transmission within the sensor network (Silberstein, 2007). Hence, first, data aggregation can improve the energy consumption for WSNs.

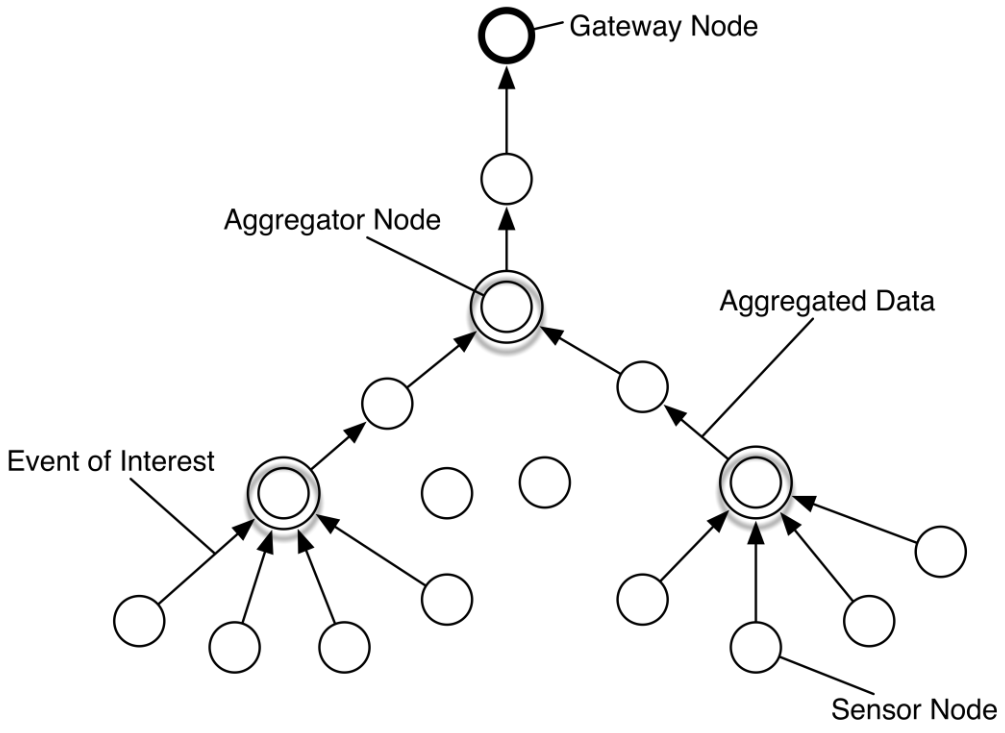


*Figure 2-3 The structure of in-network data aggregation*

For most applications, the users are not interested in the raw data. They are only interested in a specific event. In-network filtering reduces transmission by restricting the source transmitting a message, only sending the information when specific events occur (Madden, 2002). Therefore, data aggregation can improve the information quality. Data aggregation is also performed to detect the specific events by fusing data from multiple sensor nodes or networks, which can realize WSN to be homogeneous. It is similar to data filtering, but it detects high-level events of interest. It also can reduce overall communication by avoiding the transmission of irrelevant sensor data.

Many data aggregation approaches (Sasirekha and Swamynathan, 2015) organize the WSN into a “structural” architecture such as cluster, tree or chain, so that they can select a subset of sensors to serve as the aggregation nodes to deal with data aggregation. A subset of sensors are selected as aggregation nodes to “merge” multiple sensing data by, for instance, taking their average, maximum or minimum values. Hence, it can help to improve the data accuracy by aggregating the correlation data.

### 2.3.2 Data Compression

Data compression is used to reduce the number of bits to be transmitted and has the potential to drastically reduce the energy consumption and so increase network lifetime. There are sampling-level (Cande`s and Wakin, 2008; Haupt et al. 2008) and communication-level (Lu et al., 2010; Tulone and Madden, 2006) compression. Because WSNs have limited power supply, bandwidth for communication, processing speed and memory space, researchers focus on how to achieve the maximum utilization of limited sensor resource. Data compression is one effective method to utilize limited resources of WSNs. It will reduce the energy consumption during processing and transmitting data in each node for longer lifetime of sensor network. It also can reduce the data size for the effective bandwidth. In Figure 2-4, the data compression includes node layer and client layer. The node layer is to compress data by energy efficient compression algorithm. The compressed data is sent to the client layer, which is to reconstruct data by decoding algorithm.

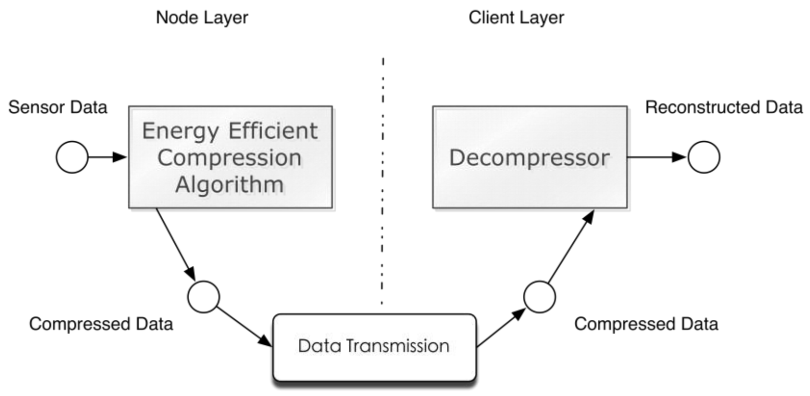


*Figure 2-4 The structure of data compression*

### 2.3.3 Data Mining

Data mining is to extract information from heterogeneity networks (Mao et al., 2012). Data mining is suitable for both environment and human monitoring applications. The major purpose of WSNs is to detect the specific applications in which they are interested. When the specific events are detected, they report them to the user. In most applications, the interesting events may consist of complex properties, which require integrating data form multiple heterogeneous data sources. However, the user requires the sensor nodes data to be able to understand for the complex events. Knowledge discovery in database (KDD) is considered to be a more encompassing process that includes understanding application domain, data integration and selection, data mining, pattern evaluation and finally consolidation and use of extracted "knowledge" (Fayyad, 1996). Semantic Data integration is a particulate step in large process of knowledge discovery in database (KDD). Data mining is one of techniques in the data integration. The Gartner Group stated that "data mining is the process of discovering meaningful new correlations, patterns and trends by sifting through large amounts of data stored in repositories, using pattern recognition technology as well as statistical and mathematical techniques" (Kantardzic,2003). Applying this definition, to the context of emergency response systems, suggests a need to extract useful and previously unknown or unexpected emergency knowledge from a large amount of real-time environmental data in order to detect a fire incident and support emergency services to enable them to provide a more effective response. Real-time data mining combines modern data mining techniques with time series analysis techniques. The traditional data mining method focuses on a limited perspective, which is inappropriate for analysing data from an assumed model. Therefore, it cannot process time series data and newly emerged type in real time. Time series can deal with temporal sequences of datasets and forecasting data, but it cannot handle large amounts of data.

Figure 2-5 shows a general architecture using a layered approach for data mining from heterogeneous sensor nodes. The sensor data may be from a single sensor network or different WSNs. The gateway nodes collect the data from different WSNs in the bottom level in Figure 2-5. The gateway nodes perform translation, filtering, aggregation and integrated data, and then send results to mediators/brokers. Data mining operations are deployed in the centralized location in the middleware processing layer, and processed information is finally transmitted to the application layer for different users.

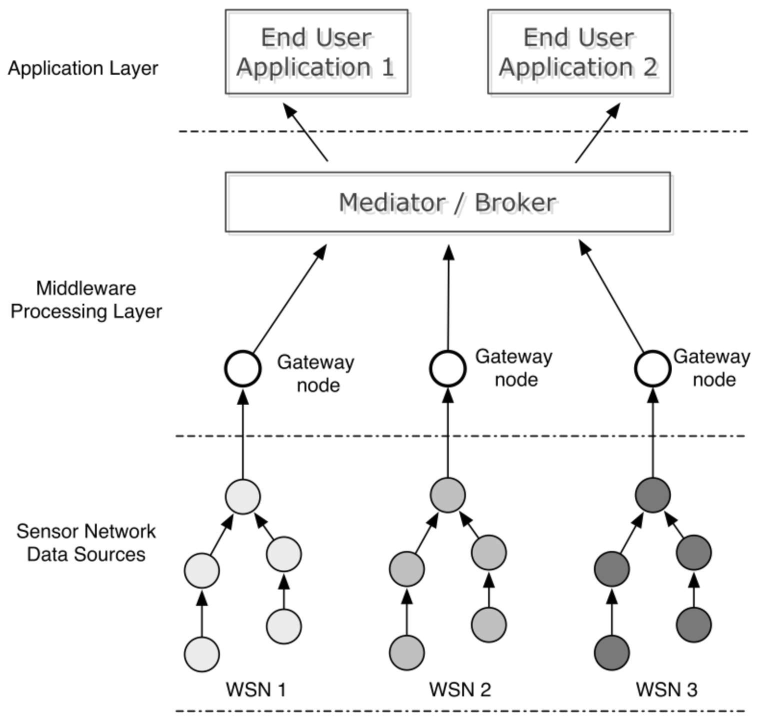


*Figure 2-5 Data Mining from heterogeneous data Source*

## 2.4 Adopted Research Method

This section provide an overview of the research method adopted for the research and the rationale behind the adopted methodology, followed by a detailed description of the research undertaken at different stages of the research methodology. The research project involves the analysis and evaluation of the existing data processing approaches on resource limited WSNs. It is as a benchmark to gauge the relative effectiveness of the new proposed approaches. There are more than twenty different systems development methodologies (Avison & Fitzgerald 2003), a review of these methodologies is outside the scope of this thesis. The objective of this research require the use of qualitative approaches. The quantitative approaches help to provide statistics such as, the percentage of energy saving by an existing data processing, for providing a benchmark to contract the improvement of the proposed approaches. All of these methodologies recognized as three stages: concept development stage, system building stage and system evaluation stage. The concept development stage indicates the relationship between stakeholders and requirements. System building stage is to propose and design the architecture for the requirements. System evaluation is to implement the system in operation to verified the result.  A brief summary of these stages is provided in Table 2-3, followed by a through discussion of the research undertaken at each stage.

*Table 2-2 System development methodology stages and research undertaken*

| Stage | Research Undertaken |
|---|---|
| Concept Development | Investigation and Domain Analysis<br>• Literature review<br>• Identification of potential challenges in WSNs and the existing data processing approaches |
| System Building | Design a suitable In-network data processing architecture<br>• Suitable for both environment and human monitoring applications<br>• Mitigate WSN Constraints<br>Development of the existing approaches as a benchmark for evaluating each approaches.<br>• Design of a hybrid data aggregation approach for the network structure and the improvement of data |

| | |
|---|---|
| | accuracy<br>• Design of a data compression approaches to enable further energy saving at the aggregation points.<br>• Design of a data mining models to extract the information for heterogeneity network. |
| System Evaluation | Evaluation<br>• Evaluation of Indoor location tracking system<br>• Evaluation of each proposed approaches with existing approaches |

### 2.4.1 Concept Development Stage

The concept development stage consisted of an extensive literature review, as part of the literature review, the existing data WSN constraints were investigated and analysed by the application requirements. The major application requirements are network lifetime (Heinzelman et al., 2000), event detection (Alsheikh et al., 2015), information quality (Martinez, 2007), data timeless (Paradis,2008) and self-organization (Walpola, 2009) in Table 2-3. As we mentioned in section 2.2, they make the WSN able to consider about energy consumption, heterogeneity, data accuracy, real-time and unpredictability. They are the factors that are necessary for sensor data processing within a heterogeneous WSN.

*Table 2-3 The relationship between application requirements and WSN constraints*

| Application Requirements | WSN Constraints |
|---|---|
| Network lifetime | Energy constraint |
| Information Quality | Data accuracy |
| Self-organization | Unpredictability |
| Data timeliness | Real-time |

| | |
|---|---|
| Event detection | Heterogeneity |

Moreover, a thorough review of existing data processing techniques to fulfil the WSN constraints was undertaken. In-network data aggregation can be done in the sensor node is to organize the WSN into a structural architecture such as tree, chain or cluster (Sasirekha and Swamynathan, 2015). It can limit the communication of unnecessary events for energy consumption, improve the data accuracy and deploy the flexible structure for network self-organization. The existing data aggregation approaches is merge multiple sensor data by maximum, average or minimum. It is suitable for environment monitoring for instance, the average temperature degree in the room, or the highest concentration of carbon monoxide in the building. However, for human behaviour monitoring or the alter information, the sensor data have to be processed by lossless processing approaches. Data compressing is another energy efficiency approaches (Haupt et al. 2008). Further lossless compression can be not only achieved by take advantage of the correlation of sensing data to condense their size, but also generated at approximately the same time, reach the controller is unimportant to the application. Both data aggregation approaches and data compression approaches is for the spatial or temporal related sensor data. In heterogeneity WSN, there are several different types of sensors in the same nodes. Data mining is to process sensor data for mining the acceptable information for users from heterogeneous sources (Mao et al., 2012). Hence, the research solutions are to design a process of mitigating the constrains of energy, heterogeneity, unpredictability, real-time and data accuracy by using in-network data aggregation, data compression and data mining at the sensor nodes in the WSN.

### 2.4.2 System Building Stage

The system building stage is to design a suitable in-network data processing architecture to fulfil the WSN constraints, which is a hybrid architecture with three levels: data aggregation, data compression and data mining. In the cluster aggregation level component, we divide the network to several clusters and aggregate the sensor data. Data aggregation clustering is to computer intermediate results at the node level such as the sum or average of the sensor readings by clusters. This reduces the need for communication considerably. In the compression level, we compress those aggregated data in the packet to reduce the communication packet size. Data Compression is to compress several smaller records into a few larger ones. Thus, it reduces the number of packets and thereby the communication cost. After data compression, all of the processed data are transmitted from to the Gateway Node (GN). In the mining level, GN is deployed data mining to extract information from the packets to get the acceptable knowledge for the specific users. It applies data mining model for detecting the specific event and given the result of current states. Data aggregation is to limit the communication of unnecessary events for energy consumption, improve the data accuracy and deploy the flexible structure for network self-organization. Data compressing can reduce energy consumption by minimizing data volume and represent sensor data for data mining. Data mining process data for mining the acceptable information for users from heterogeneous sources. These three methods for viewing the network to mitigate the five WSN constraints.

We propose three new approaches for those three levels. For data aggregation, the existing data aggregation identified from the literature review, see Chapter 3.

For data compression, the existing data compression identified from the literature review, see Chapter 3. The initial existing approaches to combine the cluster aggregation and multipath aggregation to overcome the disadvantages for them (see Chapter 5). The propose neighbour-aware multipath cluster aggregation explored in-network aggregation clustering as a power-efficient and accurate mechanism for processing and collecting data in wireless sensor network. Our focus was on applications where a large number of nodes produce data periodically, which is consumed by fewer sink nodes. Multi-path data transmission can overcome the robustness problems. But it requires more energy compare with the single path. Hence, we propose the neighbour-aware phase, which is used to discard the redundant packages, only send the satisfied important data to aggregated in cluster head for further transmission. NMCA presents the network structure to employ in-network data processing. By classifying the network into clusters by the region and exploited neighbour-aware and multi-path phase to improve data accuracy of the event. After the clustering level, the aggregated data is prepared in the cluster head for further processing by compression level. It also can be used independently for a small-area data processing.

For data compression, development of effective compression algorithm is a key to improve utilization of the limited resources of WSNs (energy, bandwidth, computation power). It also suffers in dynamic environments and network. But it requires more memory storage and computation power. Therefore, our proposal methods should be lightweight and dynamic for the environment. Therefore, we propose the communication compression scheme for UEN packets to reduce the redundant packages and a lightweight Optimal Dynamic Huffman (ODH) algorithm for data compression level. Neighbour-aware compression scheme for UEN is high performance for several events at the same time. It reduces the energy consumption and data redundancy in the event detection for communication compression. Optimal Dynamic Huffman algorithm can be easily employed in practice and any wireless sensor network for in-network data compression. In order to evaluate the algorithm, we computed the compression ratio obtained in several real datasets containing temperature collected at different locations and during distinct periods. The datasets also have the different correlation rates. Both of Adaptive Huffman algorithm and ODH are high performance for different correlated and repeated data. However, ODH is more suitable for in-network data processing.

For data mining, the existing data mining identified from the literature review, see Chapter 3. Existing research has revealed the data mining that is an emerging field that identifies elements of information contained in datasets that imply meaning in the context of application and can be interpreted by the users to facilitate their tasks. The existing meaning extraction research is highly domain-specific. A key disadvantage is the fact that the entire action history must be stored and processed off line, which is not practical for large prediction tasks over a long period. Therefore, there is a need for the propose data mining, which is used to detect the event in the network and improve the data accuracy by a lightweight computation. Data clustering and data compression techniques have a significant benefit for energy efficiency and communication latency. However, for data mining in the gateway node, it must be designed to satisfy for not only energy efficiency and communication latency, but also the accuracy of the conveyed information. Semantic data mining includes the semantic annotator, knowledge base and data mining rule engines to detect fire and improve the detection accuracy. The semantic process is applied to the data format structure and outlier detection to pre-process data for data mining rules. Event detection process uses the threshold-base rules from the knowledge base to extract the event by data mining rules engines. The simulation demonstrated that applying this mode could improve efficiency and accuracy of the existing event detection algorithms.

### 2.4.3 System Evaluation Stage

During the system evaluation stage, the three proposed approaches are tested individually with the existing approaches. It is also evaluated in the case study about in-door location system as a whole. For data aggregation level, we choose to use the work of LEACH (Govindan et al, 2002) as the benchmark for comparison. LEACH is the most popular clustering aggregation for WSN (Md and Koo, 2012). The performance is evaluated mainly, according to network lifetime, data accuracy and data throughput. For data compression level, Adaptive Huffman algorithm (Tharni and Ranjan, 2009) as the benchmark to evaluate the compression ratio. We also evaluate the performace of energy consumption by comparing the proposed aggregation algorithms (NMCA) and data compression algorithm (ODH). For data mining, we establish the emergency response system to evaluate the performance of the data mining model.

## 2.5 Summary

This chapter introduces the background of WSN, sensor nodes and various applications, which are environmental monitoring and human behaviour monitoring According to the requirements of applications, we identify the major WSN constraints and provided the initial research solutions for them. It provides the research solution for the proposed work. This chapter address the adopted research methodology.

The next chapter analyses the existing systems and approaches for these three solutions to mitigate the WSN constraints and its related areas.

# Chapter 3. Related Work

This chapter is examined how other researchers have addressed the issues faced by WSNs constraints. It identifies the advantages and weaknesses in the recent existing work and provides a justification for the present study.
This chapter is structured as follows. Section 3.1 introduces the focus of this chapter, the scope of the analysis and the existing approaches. Section 3.2, 3.3 and 3.4 then review the existing approaches for in-network database in their respective area: in-network data processing, data compression and data mining. It discusses the implications of the research summarized in this chapter in general and their weaknesses in Section 3.5. This chapter concludes with a summary in Section 3.6

## 3.1 Introduction

This chapter analyses the related approaches which is to view the whole network as a database for data processing and events detection in an energy efficient way. Those approaches we called in-network database for WSN. This section gives more details about the scope for evaluation of related work in respective area.

In-network data aggregation, data mining and data compression have been introduced in the previous chapter. This chapter reviews the specific related work pertaining to the mentioned techniques to understand the overview of other research done in these areas. These three factors are very important for an energy efficient heterogeneous WSN. This chapter will present wherever includes all these techniques. These will be the initial approaches for our further research.

The technique to view WSN as a distributed database assumes that only a minimal amount of data is transferred from the network. It exploits that an intermediate node is used to process or aggregate sensory data collected from a group of sensor nodes before sending the processed data to the base station. For example, in a sensor group comprised of temperature sensors, the aggregator can collect temperature readings from different sensor nodes in a given time interval, locally process these readings, and then forward an average of these readings (Zhao, 2003). The benefits of in-network processing for WSNs include improved scalability, prolonged lifetime, and increased versatility. Since aggregation reduces the volume of data communicated throughout the sensor network, then the benefits of in-network database include prolonging the lifetime of WSN, which is one of the most critical factors in the design and deployment of WSNs.

We reviewed the current related literatures and classified those approaches for in-network into three categories. As shown in Figure 3-1, they can be classified in three categories. These categories are those methods, which are network structure based approaches, which are used for applications, which are oriented for data. These can be further classified into different sub-categories.  In terms of network structure based approaches, there are two sub-categories: flat and hierarchical network structure. In terms of operating based approaches, there are two main subcategories in recent research topics: model-driven and event-driven in WSNs. Data Oriented approaches, which are the most important part in our further research, can be classified into two areas: Outlier detection and Data uncertainty. The classifications indicate three key areas of in-network database for WSN: structure, data and application.

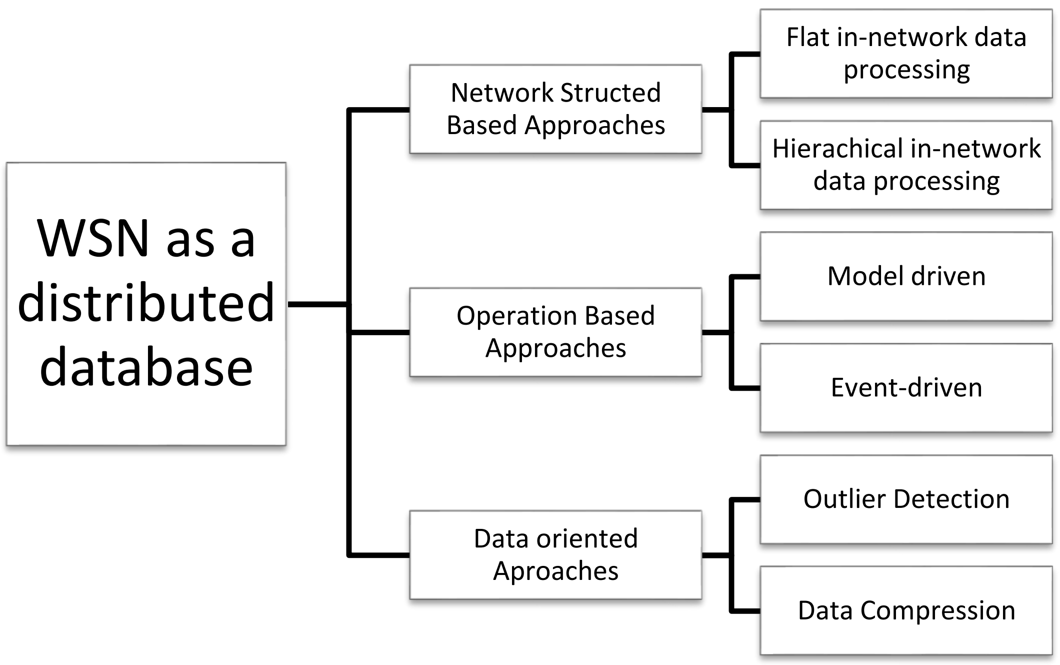

WSN as a distributed database
Network Structed Based Approaches
Flat in-network data processing
Hierachical in-network data processing
Operation Based Approaches
Model driven
Event-driven
Data oriented Aproaches
Outlier Detection
Data Compression

*Figure 3-1Classification of the existing data processing techniques within WSN*

The research method described concept development stage in Section 2.4 are used as criteria for the analysis of work in related area. We analyse the classification and list three approaches as follows.

- In-network data aggregation: Processing of data such as filtering, aggregation or integration can be performed at different levels of WSNs. Existing work is built in network structure for processing data. The proposal of the approach is to establish the network structure of data processing.

- Data compression: To compress data in the sensor node to make energy efficient or adept limited bandwidth in the WSN. It is a data oriented approach. The existing work analysis described in this chapter will present the initial ideas and their weaknesses for further improvement in our research.

- Data mining: To process the data from multiple sensor nodes, a mechanism is required to represent the information for all the participants. Existing work is examined the particular operations for different applications. It is an operating based approach.

## 3.2 In-network Data Aggregation

Due to the different structures of WSNs, the in-network data aggregation techniques can be classified into three main categories, which are flat, hierarchical and location-based structures. In the flat in-network data aggregation, all the sensor nodes have the same level of function and responsibly. All the wireless sensor nodes forward the data to all the neighbour nodes without any attention to the network topology. Hierarchical in-network data aggregation has different roles of functionality distributed among the wireless sensor nodes. Different strategies have been proposed for efficient in-network data aggregation such as Directed Diffusion and Cougar, which are flat in-network data aggregation, or reign based techniques, which are hierarchical approaches.

1) Flat structure

This section will discuss the most popular flat in-network database systems, which deal with data centric storage techniques in flat architecture. Flat data processing uses the data centric storage (DCS) (Ratnasamy et al., 2003) for processing data, where there is a base station responsible for sending requests and queries to other nodes and waiting for their responses. The DSC algorithms are proposed to overcome the shortcoming of the flooding algorithms. To compare with the flooding based algorithms, the DCS algorithms organize the storage of event information in an efficient way so that a query can be directly answered without flooding it to the whole network. In a DCS algorithm, a special node called a hash node is selected to store all the events of a particular type and answer queries asking for this type of event. In DCS, relevant data are stored by name at nodes within the sensor network; all data with the same general name will be stored at the same sensor node (not necessarily the one that originally gathered the data). Queries for data with a particular name can then be sent directly to the node storing those named data, without the flooding required in some data-centric routing proposals.

### (a) Directed Diffusion

Directed diffusion (Intanagonwiwat et al., 2000) is a data-centric, data dissemination paradigm for sensor networks. Directed diffusion has some novel features: data-centric dissemination, reinforcement-based adaptation to the empirically best path, and in-network data aggregation and caching. These features can enable highly energy-efficient and robust dissemination in dynamic sensor networks, while at the same time minimizing the per-node configuration that is characteristic of today's networks. Directed diffusion supports data-centric routing and application-specific data processing inside the network. In this thesis, we do not discuss routing protocol in details. Directed diffusion can achieve significant energy savings with in-network data aggregation. The instantiation of directed diffusion described in the earlier work establishes low-latency paths between sources (sensor nodes that detect phenomena) and sinks (user nodes) using localized algorithms. Paths from different sources to a sink from aggregation tree rooted at sink. Data from different sources is opportunistically aggregated.

It is organized in three phases (see Figure 3-2, originally shown in (Intanagonwiwat et al., 2000),

- Interest dissemination
- Gradient setup
- Data forwarding along the reinforced paths (path reinforcement and forwarding)

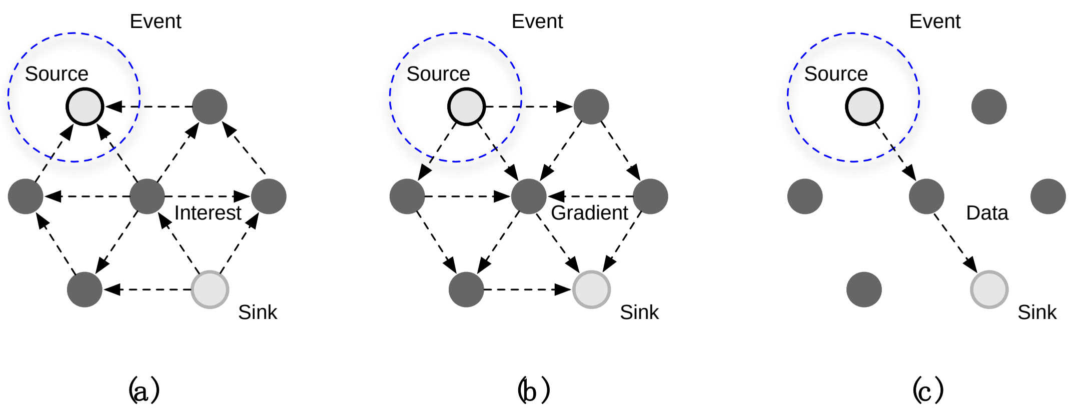


*Figure 3-2(a) Interest propagation (b) Initial gradients setup; and (c) Data delivery along reinforced path in Directed Diffusion. A simplified schematic for directed diffusion (taken from Intanagonwiwat et al., 2000)*

When a certain sink is interested in collecting data from the nodes in the network, it propagates an interest message (interest dissemination), describing the type of data in which the node is interested, and setting a suitable operational mode for its collection. Each node, on receiving the interest, rebroadcasts it to its neighbours. In addition, the node sets up interest gradients, that is, vectors containing the next hop that has to be used to propagate the result of the query back to the sink node (gradient setup). As an illustrative example (Figure 3-2), if the sink sends an interest that reaches nodes a and b, and both forward the interest to node c, node c sets up two vectors indicating that the data matching that interest should be sent back to a and/or b. The strength of such a gradient can be adapted, which may result in a different amount of information being redirected to each neighbour. To this end, various metrics such as the node's energy level, communication capability, and position within the network can be used. Each gradient is related to the attribute for which it has been set up. As the gradient setup phase for a certain interest is complete, only a single path for each source is reinforced and used to route packets toward the sink (path reinforcement and forwarding).

Data aggregation is performed when data are forwarded to the sink by means of proper methods, which can be selected according to application requirements. The data gathering tree (i.e., reinforced paths) must be periodically refreshed by the sink, and this can be expensive in dynamic topologies. A trade-off, depending on the network dynamics, is involved between the frequency of the gradient setup (i.e., energy expenditure) and the achieved performance. A valuable feature of Directed Diffusion consists of the local interaction among nodes in setting up gradients and reinforcing paths. This allows increasing efficiency, as there is no need to spread the complete network topology to all nodes in the network.

(b) COUGAR

COUGAR (Yao, 2002) is another data centric flat database system, which is similar to Directed Diffusion. It is most suitable for monitoring applications, where nodes produce relevant information periodically. COUGAR is a clustering scheme. As soon as the cluster heads receive all data from the nodes in their clusters, they send their partial aggregates to the gateway node. Cougar offers a SQL-like query interface that abstracts from the physical network layer of the sensor network. Cougar enhances the standard SQL by DURATION and EVERY clauses, which specify the lifetime of a query and the rate of query answers, respectively. Efficient data collection in COUGAR is based on the idea that local computations can reduce unnecessary traffic on the sensor network. Therefore, COUGAR uses a query proxy lay between the network and application layer that sites on each sensor node. A query optimization generates query plans that determine which nodes are involved in routing (the data flow) and which computations need to be performed at which nodes. Each query has a leader node that performs the main computation.

Thus, two computation plans are generated for a given query: one plan for the leader node and another for dissemination to all sensor nodes that have to participate in a given query. Although, the query plan can ensure an energy-efficient data collection process, the query proxy layer will introduce additional overhead to each sensor node.

Data aggregation is also used to save energy, and remove redundant data. COUGAR has a database approach in addition to Directed Diffusion. This database approach has a relational table. This relational table contains sensor node details and information collected from the node to summarize the information in the WSN (Yao, 2002).

There are three disadvantages of using COUGAR. First, the additional query layer will add extra overhead in terms of the power consumption, and memory size. Second, it needs high-level synchronization between the wireless sensor nodes. Leader nodes are required to aggregate data and send information to the base station, and these leaders should be dynamically selected to prevent nodes from being failure hops.

2) Hierarchical structure

Hierarchical network depends on dividing the job among the wireless sensor nodes into more than one level. There are a number of research issues for hierarchical in-network data processing. Node synchronization in performing partial aggregation is necessary. Data transmission reliability can be enhanced through proper modification of the routing protocol. Since finding the optimal aggregation tree to support partial aggregation can be shown to be equivalent to finding the Steiner tree, an NP-hard problem, a greedy aggregation method to support the partial aggregation is needed. A greedy incremental tree can improve path sharing and reduce transmission energy.

(a) Region based

The network can be divided into logical regions and the regions are used to facilitate in-network processing. There are three parts in the technique: region discovery, leader election and leader advertisement.

In region discovery, an initial message is sent from the sink to the network. This message contains region setup information identifying a node as being in a particular region based on its 2-D location. A node receiving the message sets its region ID attribute and forwards the setup message on. Once all nodes within the broadcast range have received the message, the leader election phase begins.

A naive selection method was used to elect region leaders. A node receiving an election message sets itself as the leader if no leader exists for the particular region. That node then sends a registration message to the sink and issues a leader advertisement message to the network. This message contains the leader's ID and the region ID for which it is the leader.

A node receiving a leader advertisement message sets its leader attribute to the leader ID in the message if they are part of the same region. Then it sends a registration message out to the leader along the reverse path of the advertisement message.

At the end of the Region Setup phase:

- All nodes within broadcast range are part of a region.
- Each region has elected a leader.
- Each leader has a path to the sink, a list of its region members and a path to each member.
- Each member knows its leader and has a path to that leader.

Since a region knows the location of its members, it can send the query to a selected subset that is in a particular proximity to the node, which detected the event. Based on the initial results the leader can decide to query further away from the event source or to stop at that point. Nodes send their data to the region leader for analysis. Each node knows its parent. Data are sent by a node to its parent until it reaches the sink. The sink upon receiving the message forwards to its parent node a packet containing the readings. Data are aggregated wherever possible as message moves up the tree and is sent in the same data packet. Once the sink has received this aggregated data, analysis is performed to determine if the data indicates event occurred. The cost of query resolution was measured by counting the number of messages generated in sending data because of query back to the sink.

3) Evaluation

Directed Diffusion, COUGAR and Region based approach are good at conserving sensor node energy by performing in-network filtering and aggregation of sensor data. But those approaches are based on different network structures sensor networks. They do not provide semantic support for different types of sensor data and as a result, they do not have ability to integrate data form heterogeneous sources in order to detect complex events within the sensor network. In the next section, we address data integration approaches for event detection where data has to be integrated from multiple sensor nodes to answer users' requirements.

## 3.3 Data Compression

In-network data compression and outlier detection are the main techniques in this category of data oriented approaches. Outlier detection is the algorithm based on the different event. Hence, we propose this algorithm combines with the data mining for event detection. In this section, we introduce existing research in data compression in WSNs.

The benefit for WSNs to employ a data compression algorithm is to minimize data size before transmitting in wireless medium is effective to reduce the total power consumption. However, most of existing data compression algorithms may not be suitable for WSNs. One reason is the limitation of memory size. For example, the size of bzip2 is 219 KB and the size of LZO is 220 KB. The instruction memory size of sensor node currently available is only 128 MB (MICA Wireless Measurement System). Another reason is the processor speed of sensor node, which is only 4MHz (MICA Wireless Measurement System). Therefore, we have to design a low-complexity and lightweight data compression algorithm for sensor networks. The following section presents some existing compression approaches for WSNs. We have divided them into two classifications: data and package funneling techniques and related data compression techniques.

1) Data and package funneling techniques

To avoid huge loss, there is a need for an efficient data gathering technique simply called as data funneling (Kumar and llango, 2015). In data Funneling, many sensor nodes have to communicate their data to a single controller node that does data-gathering and processing.

a) Coding by ordering

Petrovic et al. (2003) Introduced a "Coding by Ordering" data compression scheme, which is part of the Data Funneling Routing. Figure 3-3 shows a data pass between sensor nodes in the interested region and a collector node. In Data Funneling Routing, some of sensor nodes work as a data aggregation node (node A, B, and D). Sensing data collected by other nodes are combined at an aggregation node, and then sending to its parent node. For example, data collected by node E are combined with data collected by node D itself. Then, the aggregated data are transmitted to node B.

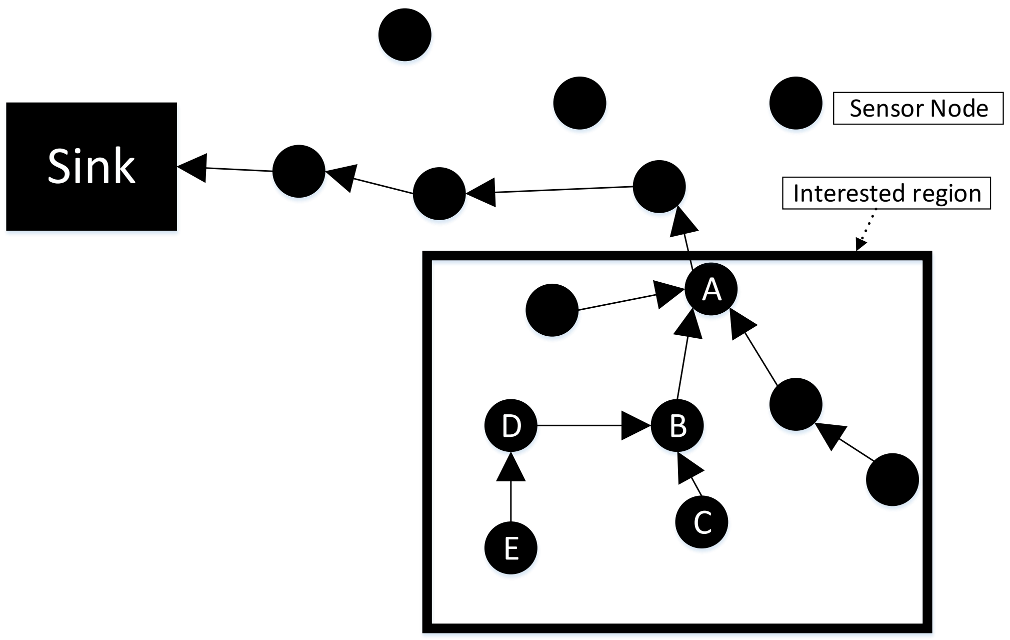


*Figure 3-3 Data Path*

In the algorithm, when data are combined at an aggregation node, some data are dropped. To include the information of dropped data in the aggregated data, the order of data packet is utilized. For example, four nodes (N1, N2, N3, and N4) send the data to an aggregation node (Na). The data value of each node can be any integer ranging from 0 to 5. The possible packets orders from other N1, N2, and N3 are 3! = 6. The data value of N4 can be included in an aggregated packed without actually including the packet of N4 by using permutation of three packets. The possible combination of permutation and data value is presented in Table 3-1.

*Table 3-1 Permutation and its Represented Integer Value*

| Packet Permutation | Integer Value |
|---|---|
| N1,N2,N3 | 0 |
| N1,N3,N2 | 1 |
| N2,N1,N3 | 2 |
| N2,N3,N1 | 3 |
| N3,N1,N2 | 4 |
| N3,N2,N1 | 5 |

In a general case, we assume that n is the total number of sensor nodes, which come with a unique node ID, m is the number of nodes sending a packet to an aggregation node, k is the possible range of data value, and l is the number of sensor node dropped at the aggregation node. Then, the number of possible combination of IDs, which dropped nodes is $\binom{n-m+l}{l}$. Since each of $l$ nodes can take any value among possible k data values, there are $k^{l}$possible data value combinations. There is total of $\binom{n-m+l}{l}k^{l}$ possible values with possible IDs and data values. This combination of values expressed by $(m-l)!$ permutations and generates the following inequality.

$$(\mathrm{m}-\mathrm{l}) \geq \binom{n-m+l}{l}k^{l} \tag{3.1}$$

Theoretically, when n = $2^7$, k = $2^4$, and m = 100, approximately 44% of packets can be dropped at the aggregation node by applying Coding by Ordering. It is possible to apply this method to WSNs for its good compression ratio with simple algorithm. The only down side of this scheme is the mapping table will significantly increase with the number of sensor nodes aggregated increasing, also.

b) Pipelined In-Network Compression (PINCO)

Arici et al. (2003) discussed the pipelined in-network compression scheme. The basic idea is to store the collected sensor data in an aggregation node's buffer for some duration of time, and combine all the arrival data packets into one packet by removing the redundancies parts during this period. For example, each data packet has the following form: <measured value, node ID, timestamp>. Then, the compressed data packet has the following form: <shared prefix, suffix list, node ID list, timestamp list>. The "shared prefix" is the most significant parts, which all measured values in combined data packets have in common. The length of shared prefix can be changed by a user based on the knowledge of data similarity. If the measured values are expected to be close to each other, the length of prefix value can be set to relatively long. The "suffix list" is the list of measured values excluding the shared prefix part. The "node ID list" is the list of node identifiers and the "timestamp list" is the list of timestamp. The compression scheme (Jolly et al., 2006) is illustrated in Figure 3-4. In this figure, three nodes send the data packets to the compression node. At the compression node, three data packets are compressed into one packet. In this example, the length of shared prefix is set to 3. In this example, the total number of bits is reduced from 33 to 27.

One advantage of this simple compression scheme is more data compression can be achieved because of the shared prefix system can be used for node IDs and timestamps. The data compression efficiency is influenced by the length of shared prefix. It increases with the longer shared prefix and commonality of measured value. However, the disadvantage is no similarity in measured sensor values. The efficiency of Pipelined In-Network Compression will decrease even with a long shared prefix. In addition, a large data buffer is required to temporary store those packets when combining a large amount of data packets. Since there is a limitation of memory space in sensor devices, enough buffer space may not be available.

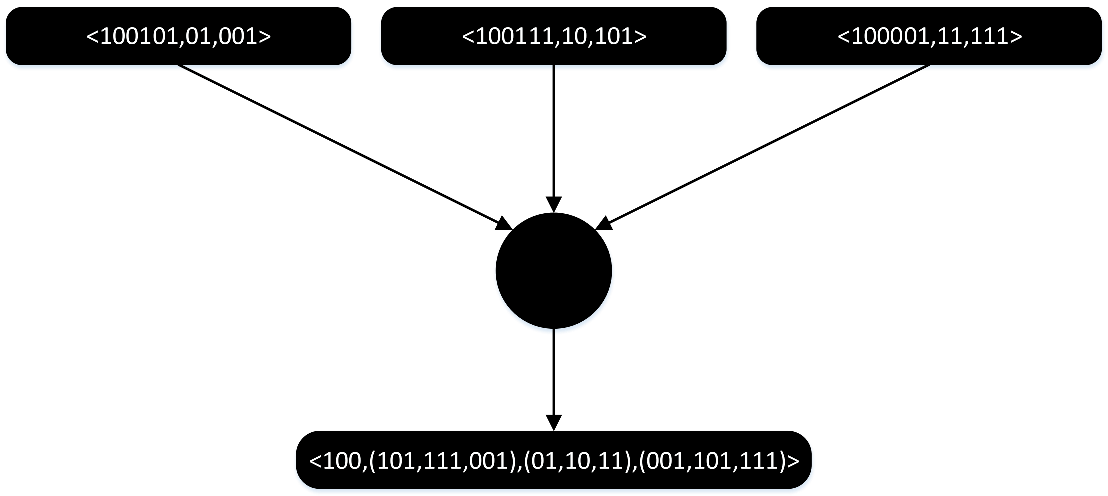


*Figure 3-4 Example of In-network Compression (Jolly et al., 2006)*

2) Related data compression techniques

a) Distributed Compression (DC)

The Distributed Compression scheme introduced in Pradhan et al. (2002） and Kusuma et al.(2001) uses a side information to encode a source information. For instance, there exist two sources (X and Y) as shown in Figure 3-5. They are correlated and discrete-alphabet independent identically distributed. Then, X can be compressed at the theoretical rate of its conditional entropy, H(X|Y), without the encoder 1 accessing Y (Slepian and Wolf, 1973). The conditional entropy can be expressed as:

$$\mathrm{H}(\mathrm{X}|\mathrm{Y}) = -\sum_{y} P_{Y}(y) \sum_{x} P_{x}(x|y) \log_2 P_{x}(x|y) \tag{3.2}$$

The general scheme of Distributed Compression is first to compose co-sets, whose code vectors are of source X. The distance of any two code vectors in the same co-set has to be large enough. An index value is assigned to each co-set. When transmitting data to a decoder, the source X only sends an index value of co-set, to which the code vector belongs. The source Y sends a code vector as a side-information. The decoder looks up the coset, which has the same index received from X. Then, it selects one code vector, which has a closest value to the code vector sent by Y.

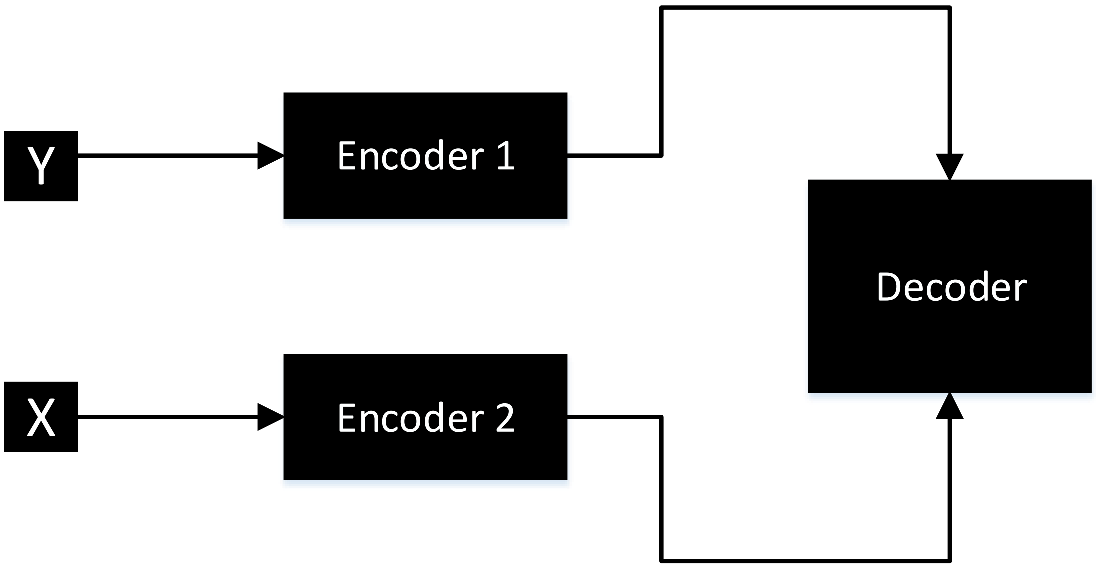


*Figure 3-5 Distributed Compression Example*

A simple example of Distributed Compression is described as follows: there are two 3-bit data sets, which are X and Y. The Hamming distance between X and Y is no more than one bit. If both Encoder 2 and Decoder know Y, X can be compressed to 2 bits. Then if Y is only known by the Decoder, according to the Distributed Compression scheme, it is not efficient to distinguish X=111 from X=000. In addition, it is the same scenario to X=001 and 110, X=010 and 101, and X=011 and 110. These sets of two X values are grouped as 4 co-sets and assigned 4 different binary index numbers as below:

co-set 1 = (000, 111) : 00

co-set 2 = (001, 110) : 01

co-set 3 = (010, 101) : 10

co-set 4 = (011, 100) : 11

If X=010 and Y=110, the Decoder received Y=110 and X=10 as side information. Then, at the Decoder, X=010 is selected from co-set 2 since 110 has a Hamming distance of 2 from 110. X is able to compress 3 bits information into 2 bits without knowing Y.

3) Evaluation:

Development of effective compression algorithm is a key to improve utilization of the limited resources of WSNs (energy, bandwidth, computation power). Hence, one of the most important requirements of data compression is lightweight for sensor computation power and memory storage. Data and package funneling techniques is a simple and lightweight data compression for in-network data processing. Related data compression techniques are suitable for large-scale network, high performance for energy efficiency and compression ratio. It also suffers in dynamic environments and network. But it requires more memory storage and computation power. Therefore, our proposal methods should be lightweight and dynamic for the environment.

## 3.4 Data Mining for Event Detection

Data mining aims to extract patterns from data. Traditional data mining technologies include Decision Trees, Rule-based Classifiers, Artificial Neural Networks, Nearest Neighbour, Support Vector Machines, Naïve Bayes etc. Most methods were initially developed for central data warehouse. In our research, it is required to provide meaning to the sensed data for it to be mined and processed by a diverse set of sensor nodes. Moreover, it is important to reason on the gathered events to deduce higher-level events from simple event. Data mining is the main approach that aims to integrate heterogeneous sensor event in a WSN to detect user-defined complex event. We review the related research, and classify two categories in the data mining: data oriented processing and event/ model driven approaches.

1) Data oriented

Outlier detection, an essential step preceding almost any data analysis routine, is used to either suppress or amplify outliers. The first usage (also known as data cleaning) improves robustness of data analysis. The second usage helps in searching for rare patterns in such domains as fraud analysis, intrusion detection, and web purchase analysis (among others).

Several factors make wireless sensor networks (WSNs) especially prone to outliers. First, they collect their data from the real world using low cost sensing devices. Second, they are battery powered and thus their performance tends to deteriorate as power dwindles. Third, since these networks may include a large number of sensors, the chance of error accumulates. Finally, in their usage for security and military purposes, sensors are especially prone to manipulation by adversaries. Hence, it is clear that outlier detection should be an inseparable part of any data processing routine that takes place in WSNs (Zhang et al., 2010).

Simply, outliers are events with extremely small probabilities of occurrence. Since the actual generating distribution of the data is usually unknown, direct computation of probabilities is difficult. Hence, outlier detection methods are, mostly, heuristics. Because the problem is fundamental, a huge variety of outlier detection methods have been developed, such as non-parametric, unsupervised methods. A simplistic implementation of these methods would require centralization of the data. Such centralization is hard and costly in WSNs as it demands high bandwidth and requires reliable message transmission over multiple hops, which is both costly and difficult to implement.

Janakiram et al. (2006) present a technique based on Bayesian belief network (BBN) to identify local outliers in streaming sensor data. This technique uses BBN to capture not only the spatio-temporal correlations that exist among the observations of sensor nodes but also conditional dependence among the observations of sensor attributes. Each node trains a BBN to detect outliers based on behaviours of its neighbours' readings as well as its own reading. An observation is considered as an outlier if it falls beyond the range of the expected class. Compared to naive Bayesian networks, this technique improves the accuracy in detecting outliers as it considers conditional dependencies among the attributes. Accuracy of a BBN depends on how the conditional dependence among the observations of sensor attributes exists. This technique may not work well in presence of the dynamic network topology change.

Spectral decomposition-based approaches aim at finding normal modes of behaviour in the data by using principal components. Principal component analysis (PCA) is a technique that is used to reduce dimensionality before outlier detection and finds a new subset of dimension, which captures the behaviour of the data. Specifically, the top few principal components capture the build of variability and any data instance that violates this structure for the smallest components is considered as an outlier (Chandola et al., 2007).

Chatzigiannakis et al.(2006) proposed a PCA-based technique to solve data integrity and accuracy problem caused by compromised or malfunctioning sensor nodes. This technique uses PCA to model the spatial-temporal data correlations efficiently in a distributed manner and identifies local outliers spanning through neighbouring nodes. Each primary node builds a model of the normal condition by selecting appropriate principal components (PCs) and then obtains sensor readings from other nodes in its group and performs local real-time analysis. The readings that significantly vary from the modelled variation value under normal condition are declared as outliers. The primary nodes eventually forward the information about outlier data to the sink. The disadvantage is that the primary node procedure of selecting appropriate PCs is computationally very expensive.

2) Event drive/ model drive

Event / model drive data mining is to extract information by sematic event or model. The authors (Yu and Taylor, 2013) propose EVENT Dashboard: an event-driven user interface through which users can describe the event constraints on sensor event. For example, reporting a high total nitrogen event when Total Nitrogen (TN) is greater than 10. The user interface also allows users to define constraints on the set of sensor data for the event detection. The event constraints are then translating by a semantic mediator to be stored in global sensor network (GSN) middleware. The GSN middleware receivers event from the sensor nodes, matches sensor data to the event constraints and notifies the mediator of it. However, in the Event Dashboard, the sensor nodes in the sensor network are decoupled from the other components in the system. Thus, all sensor nodes are sent to the middleware then processing for user specific events.

Cook et al. (2004) present MavHome smart home architecture, which focuses on the creation of an intelligent home, perceiving the state of the home through sensors and acting upon the environment through device controllers. An important characteristic of the proposed architecture is the ability to make decisions based on predicted activities. To predict the activities, an algorithm called episode discovery (ED) is proposed, which is based on the work of Srikant and Agrawal (1997) for mining sequential patterns from time-ordered transactions. Values that can be predicted include the usage pattern of devices in the home, the movement patterns of the inhabitants, and the typical activities of the inhabitants. They utilize prediction algorithms on action sequences stored in inhabitant event history to forecast user actions. Actions can then be automated based on the significance of mined patterns as well as the predictive accuracy of the next event.

A key disadvantage is the fact that the entire action history must be stored and processed off line, which is not practical for large prediction tasks over a long period. Cook et al. demonstrated the effectiveness of MavHome on synthetic smart home data and real data collected by students using X10 controllers in their homes. Experiments show a predictive accuracy as high as 53.4% on the real data and 94.4% on the synthetic data.

3) Evaluation

The above systems address the problem of heterogeneity in wireless sensor networks, however, in such systems, sensor nodes are sent all of the data to the application level.

This puts a heavy communication load on the sensor nodes of performing their operation in a continuous manner. Our proposal methods are to reduce the energy of transmission and detect the event in the network.

## 3.5 Discussion of Related Work

This section discusses the implications of the approaches analysed in this chapter. The related approaches in the surveyed research are viewed along with insights into effectiveness of the approaches where such insights are available.

*Table 3-2: Comparison of the relationship between the exiting works and WSN constraints*

| **Existing Approaches** | **Energy Saving** | **Heterogeneity** | **Real-time operation** | **Data Accuracy** | **Unpredictability** |
|---|---|---|---|---|---|
| **DD (Intanagonwiwat et al., 2000)** | √ | | | | |
| **COUGAR (Yao, 2002)** | √ | | | | |
| **EVENT Dashboard (Yu and Taylor, 2013)** | √ | √ | | | |
| **MavHome (Cook et al., 2004)** | √ | √ | | √ | √ |
| **DC (Pradhan et al., 2002）** | √ | | | | |
| **PINCO (Arici et al., 2003)** | √ | √ | | | √ |
| **Coding by Order (Petrovic et al., 2003)** | √ | | √ | | √ |

Table 3.2 provides a comparison of different approaches to the WSN constraints for the proposed work. It can be seen that the problem with existing approaches for in-network data aggregation considers different structures sensor networks and does not perform the event detection for heterogeneity sensor nodes. Both of data compression and data aggregation require the spatial or temporal correlation sensor data. This limits the potential scale of sensor networks as the sensory data cannot be communicated between the various sensor nodes. It means it can only process simple events in the sensor node level. For example, CO sensor node can only sense the

concentrations of CO to detect fire events. Moreover, the solutions provided for semantic data mining perform the operations at centralized locations in WSNs, which are normally designed, more powerful gateway nodes. For an efficient WSN, we combine these three approaches for complex in-network event processing which is to view the whole network as an in-network database.

There are no significant bounds between each of the approaches. All of the in-network aggregation, clustering, data compression and data mining approaches process sensor data in the network and can be called in-network data processing. Our proposal method is to combine all of the three approaches' advantages together to build an effective architecture to process data within the network.

## 3.6 Summary

In this chapter, we review the problems of data processing in WSNs and identify the related work for in-network data processing, data mining and data compression. In-network data processing is an intermediate node or aggregator fused or aggregates sensor data collected from a group of sensor nodes before forwarding the processed data to the base station. There are many classifications for data processing techniques especially for those designed for WSNs. There are many constraints in the WSNs, which limit memory and energy for processing and communications. These techniques can be classified into three main categories in terms of our review: the network structure based in-network data processing approaches, data compression approaches and data mining approaches. In terms of network structure based in-network data processing approaches, there are two sub-categories: flat and hierarchical. In terms of data compression approaches, in-network data compression can be divided into two subcategories: data and package funneling techniques and related data compression techniques. In terms of data mining approaches, there are two sub-categories, which are data oriented and model/event driven. Even though there are many other in-network processing techniques that can be classified under these sub-categories, the most popular ones have been reviewed in this chapter.

Not all of these in-network processing approaches are mutually exclusive, as some of them may be classified under more than one category. Furthermore, there are many methods for increasing WSN efficiently in terms of reducing the data flow, the control communication overhead, and energy consumption. For example, data aggregation and region based approaches applied a clustering technique in order to group the wireless sensor nodes and increase the lifetime. However, none of them is efficient enough and can achieve all of the performance of various methods. There are no significant bounds between each of the approaches. All of approaches process sensor data in the network can be called in-network data processing. Our proposal is to combine those three approaches' advantages together in one architecture, which can be suitable for both environment and human behaviour monitoring applications.

# Chapter 4. In-network Data Processing Architecture

The objective of this thesis, as introduced in Section 1.3 is to develop an in-network data processing architecture to mitigate the 5 WSN constraints and to be deployed in both environment and human behaviour monitoring. This Chapter proposes an in-network data processing architecture to perform data aggregation, data compression and data mining in the network for maximum energy and quality benefits.

This chapter is structured as followed. Section 4.1 gives a high-level description of the focus employed in the design of the in-network data processing for event detecting system architecture. Section 4.2 introduces the architecture and Section 4.3 provides the challenges of the new hybrid in-network data processing. Section 4.4 concludes with a summary of the chapter.

## 4.1 Design Requirements

The first step in achieving the goal of this thesis is to understand the general data processing architecture and its characteristics.

Due to the limitations of the computational and memory resources on sensors, there is no high-level data model, but rather data typically modelled as a sequence of simple data values. The most common approach is to use the spatial and temporal correlation data model. Conceptually, the sensor table is a ‘virtual’, unbounded table. The sensor readings are not physically stored in memory, but rather the data items are transient tuples in a data stream.

Figure 4-1 shows the general architecture of data processing. A sensor network consists of a large number of sensors deployed in a region for the purpose of event detection. The sensor listens to the specific event. Each node in a sensor network is responsible for observing and reporting dynamic properties of their surroundings through a path to the sink or base station in a time critical manner. Each raw data in the interested range will be sent to base station and process in centralized database for specific application.

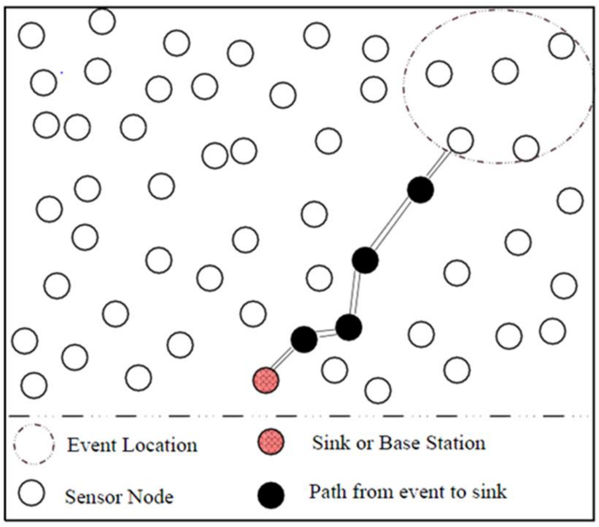


*Figure 4-1 A typical sensor network data processing (Maraiya et al., 2011)*

Keeping in mind the issues discussed above regarding sensor nodes' constraints and the initial research solutions from Section 2.4 and the goal of in-network data processing, the implementation plan is to design:

- A strategy to aggregate data by cluster to allow the sensor node to share data and energy within the sensor network. Therefore, that information can be detected and shared in an accurate and energy-efficient manner.
- A compression algorithm to compress communication data size for reducing transmission energy. It compresses data in each cluster head in the WSN for

furthering processing in gateway node or sink.

- A pattern-matching rule engine for mining complex events. It is processed in the gateway node and mined the accepted knowledge to send to specify used directly.

They are necessary to support in-network data processing in WSNs. Most importantly, the design of these components can work independent or together for different applications requirements. The next section discusses the details of the design, and these requirements for the new architecture.

## 4.2 In-Network Data Processing Architecture

This section introduces the conceptual design of in-network data processing for an energy efficient sensor network that builds on the results of the previous two chapters. Figure 4-2 shows the general architecture of in-network data processing. We divided three components of the proposed in-network data processing architecture: clustering level, data compression level and data mining level. This relates to the general architecture of the data processing, which we mentioned in Section 4.1, and is to collect the interested sensor nodes then through gateway node to users. However, the users will receive the real-time processed knowledge of the specific event in located. The base stations do not need to process data centrally. In WSNs, there are three objectives related to the data processing: data, packet, event and users. Sensor data are generated in the sensor nodes. Serval sensor data in the same sensor node can be transmitted by packet. Many sensor packets are indicated the event for the specific users. We present three levels for processing those three objectives in the WSN.

A WSN can be either heterogeneous, whereby the network is made of nodes with different data processing capability and energy capacity. In the heterogeneous network, nodes with more resources can be used for data processing or as intermediate nodes for aggregation. In cluster level component, we divide the network to several clusters and aggregate the sensor data. More resources and the correlations between sensor

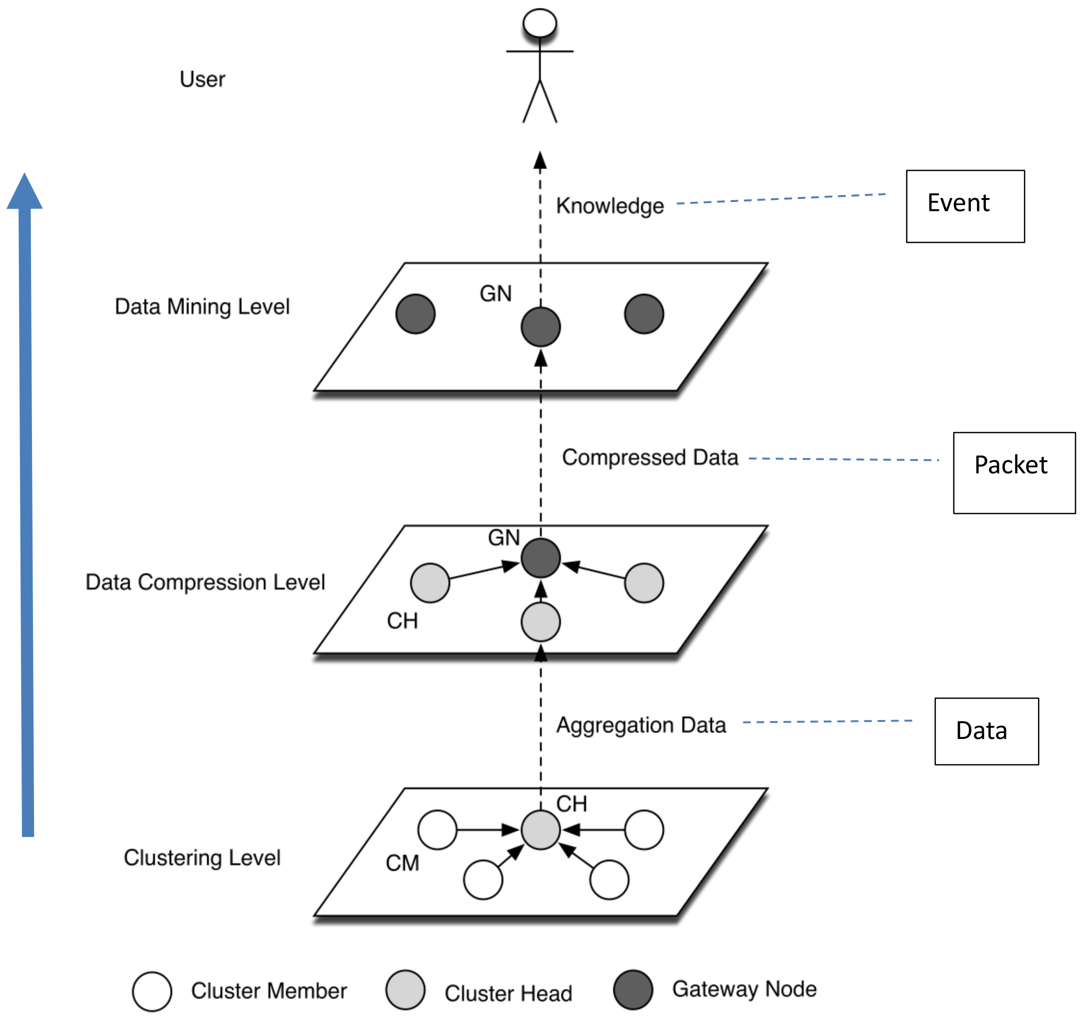


*Figure 4-2 the design of In-network data processing architecture*

nodes can be selected as cluster head (CH). We employ the correlations in sensor data to form clusters with similar node sensory values within a given threshold. Those associated sensor nodes in the cluster are cluster members (CM). Aggregation algorithms exploit the fact that there are different types of correlations in sensor readings: spatial correlations and temporal correlations. Spatial correlation is a result of the fact that the neighbour node readings are typically similar to each other. Sensor records are temporally correlated for physical feature and data sampling frequency. If the event occurred, the related CMs send to the CH for data aggregation through multi-hop routing protocol. CH is used to aggregate the homogenous sensor node data from all of CMs. Data aggregation clustering is to computer intermediate results at the node level such as the sum or average of the sensor readings by clusters. This reduces the need for communication considerably. Instead of transmitting the sensor records for each node separately, a node first aggregates the incoming reading of the nodes in the communication range and then sends the intermediate results, such as the current average of sensor readings, to the next node, which in turn aggregates this result with its own readings.

In the compression level, CHs have to compress those aggregated data in the packet to reduce the communication packet size. Data Compression is to compress several smaller records into a few larger ones. Thus, it reduces the number of packets and thereby the communication cost. It is desirable to apply compression on sensory data when highly accurate sensory data from the sensing application is required. Hence, we deploy data compression in the CH, which consists all of sensor data in the cluster and has high accuracy for further data mining. The in-network data compression has to be low-complexity due to sensor hardware limitations.

After data compression, all of the processed data are transmitted from CHs in the network to the Gateway Node (GN). In the mining level, GN is deployed data mining to extract information from the packets to get the acceptable knowledge for the specific users. It applies data mining model for detecting the specific event and given the result of current states.

Accordingly, we design in-network data processing in three levels, namely clustering level, data compression level and data mining level, to deploy in-network aggregation, data compression and data mining to fulfil the requirements in those levels, respectively. In the following, we describe some of the commonly cited proposals in the three algorithms and discuss this approach as a separate subsection.

As Figure 4-2 shows, the in-network data processing architecture is divided into three levels:

- Clustering level: To aggregate data by clusters to allow the sensor node to share data and energy consumption within the sensor network. Therefore, information can be detected and shared in an accurate and energy-efficient manner.

- Data Compression level: To compress the size of communication data to reduce transmission energy. It compresses data in each cluster head in the WSN for further processing in the gateway node.
- Data Mining Level: Data mining rule engine detects complex events. This method is processed in the gateway node, which extract the accepted knowledge from the raw for specified application

In-network clustering can be done to establish the cluster for the network and only aggregate and transmit the interested data in the cluster head, which is to limit the communication of unnecessary events for energy consumption, improve the data accuracy and deploy the flexible structure for network self-organization. Data compression can reduce energy consumption by minimizing data volume and represent data for data mining. As the sensor data must be semantically annotated, data mining is used to process data from heterogeneous sources to get the acceptable knowledge for the specified users. Be performing data processing locally, the sensor nodes make decisions quickly and remotely without the need for instructions from a centralized database. Most importantly, the design of these components can work independently or together for different applications requirements.

## 4.3 Clustering Level Approaches

The bottom level of in-network data processing architecture is clustering level, in which we focus on where, when and how data is aggregated by clusters. In this section, we address the common data aggregation clustering approaches enabling for the level and their requirements. We present the hybrid techniques, which combines the advantages of difference approaches together.

### 4.3.1 Most Commonly Used Data Aggregation Techniques

Data aggregation attempts to collect information from surroundings and then transmits only the useful information to the end (sink or base station) for the purpose of energy saving. Data aggregation is the combination of data from different sources, and can be implemented in a number of ways. The simplest data aggregation function is duplicate suppression. Duplication suppression is already practised in commercial wireless messaging networks. Other simple aggregation functions include MAX, MIN or any other function with multiple inputs.

Tree-based aggregation is to divide the nodes as the root, parent and children. It gathers data within the sensor network. The root initiates a broadcast sending its ID and its level. All nodes that receive this message and do not have a level assigned, determine their own level as the level in the message plus one. The ID with the message determines the parent for the node for the node receiving this message. The broadcasting of ID and level continues until all nodes have been assigned a level and a parent. For a network without loss and in which all nodes participate in a given query, the resulting tree is close to an optimal solution.

The main disadvantage of tree-based aggregation has been pointed out by multi-path aggregation: trees are susceptible to link loss and node loss in a sensor network. The closer a node is to the sink, the more important becomes a node for tree-based aggregation: if a link or node fails that is close to the sink, the aggregated information of an entire sub-tree could be lost. Multi-path aggregation, however, exploits the benefits of the wireless broadcast advantages in that all nodes in communication range can hear a message and propagate the aggregates toward the sink using multiple routes. The multi-path approach becomes, as a consequence, more robust for node loss or communication failures. In return, a multi-path aggregation algorithm has to deal with redundancy in data aggregation. It is often claimed that multi-path aggregation is as energy efficient as tree-based aggregation. Each node only has to transmit a message once in the same way as in any tree-based aggregation. However, a crucial factor missing in this equation: the receive time for each communicating sensor is increased. Recent studies on the energy consumption of wireless sensor nodes report that the energy requirement for the receive mode is only slightly lower compared to the energy cost for the transmission mode, i.e., a sensor node dissipates almost the same amount of energy in receiving data as in transmitting data. In multi-path aggregation, each node expects to receive data from all nodes in communication range that have a higher level than the listening node. Therefore, the duration for receiving data can be considerably longer compared to a tree-based aggregation algorithm, where each node only has to listen to its direct children, a set that can be significantly smaller. Finding for each aggregation operator a duplicate-insensitive algorithm that guarantees a desired accuracy is a key challenge for multi-path routing schemes. The obvious approach to address this challenge would be to include control information with each aggregated message. The control information contains meta information such as the node IDs, which participated in the creation of the aggregate. This meta information can be used by each forwarding node to suppress duplication. Such an approach would have the same accuracy as an aggregation algorithm on a routing tree. However, the approach cannot be used for larger sensor network due to the limited storage and processing capabilities.

Clustered in-network aggregation approach exploits the spatial correlation of sensor readings to preserve node energy. Spatial correlation in sensed data refers to the fact that sensor readings in close proximity are typically similar. Spatial correlation is a frequent phenomenon, in particular for attributes (such as the temperature for fire detector in Chapter 3). If a selective query has to retrieve an aggregate such as the average temperature in a certain area, then nearby nodes typically have similar readings and are geographically clustered in terms of the similarity of their sensor readings. In static clustering, the network is statically partitioned into grid cells. For each grid cell, one node is appointed as a cluster head node that acts as a gateway node. Each cell data is routed via a local shortest path tree, aggregated at the local gateway and then communicated directly to the sink. Clustered approaches are based on the idea that not all nodes participate in an aggregation query directly. Different clusters require variable node participation. However, the current clustered approaches are based on the assumption that participation is only determined by spatial correlation. In general, it is too restrictive to assume that participating nodes for every selective query will always be geographically clustered. Current clustered in-network aggregation approaches do not take into account that a selective query may not always translate into geographically clustered node participation. It is able to ensure energy load balancing within the network to prolong the network lifetime. However, it performs aggregation without regard to the correlation structure of its nodes.

### 4.3.2 Hybrid Aggregation Approaches

Generally, the cluster data aggregation is preferred over multi-path data aggregation in environment where events are highly mobile. Considering the advantages and disadvantages associated with both the multi-path and clustered approaches, a hybrid approach is proposed which benefits for event duration, data accuracy and event mobility. We propose three aggregations techniques: tree-based, multi-path and cluster approaches to perform data aggregation. This model would combine the strengths of all approaches.

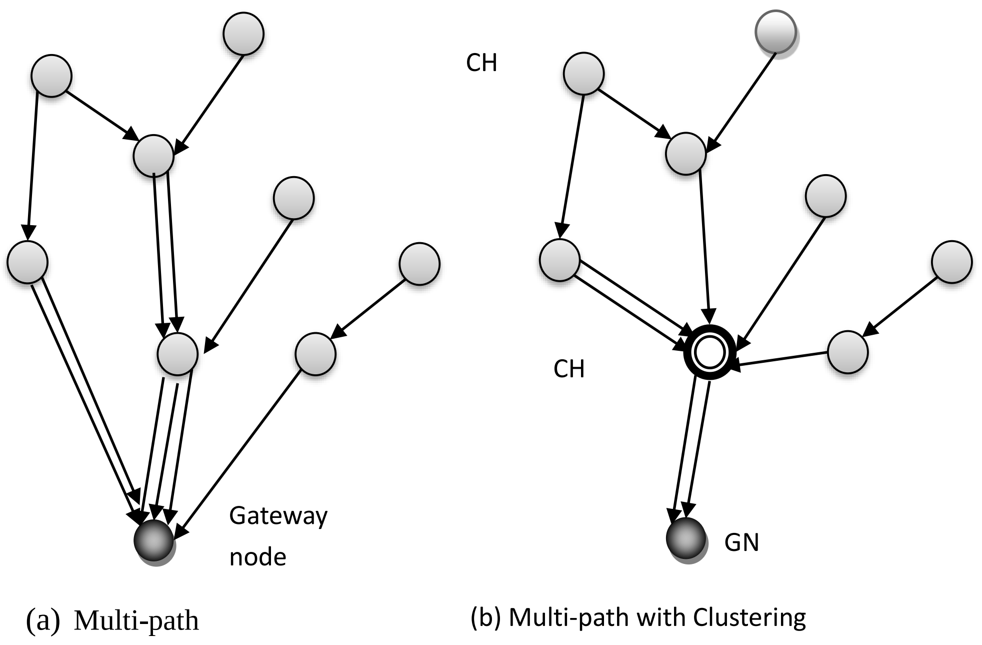


*Figure 4-3 Forwarding with and without clustering data aggregation*

Figure 4-3 shows the multi-hop approach with and without clustering data aggregation. With clustering, nodes transmit their information to their cluster heads. A cluster head aggregates the received information and forwards it over to the gateway node. Periodic re-clustering can select nodes with higher residual energy to act as cluster heads. Network lifetime is prolonged through (i) reducing the number of nodes contending for channel access, (ii) summarizing network state information and updates at the cluster heads through intra-cluster coordination, and (iii) routing through an overlay among cluster heads, which has a relatively small network diameter.

CAG (Clustered aggregation) can also be categorized as a transmission controlled-based gathering technique. CAG operates in two phases: query and response. During the query phase, CAG forms clusters using a user-defined threshold. A regular node joins a cluster if its current reading is within a range computed from the threshold. Otherwise, it becomes a CH. During the response phase, only CHs transmit aggregated data every epoch forwarded along a routing tree. The rest of the nodes do not participate in further data generation, but may be involved in forwarding if they are on the routing path. When the next query phase starts, the CHs might change except for the gateway node that always becomes the first CH every time. The final aggregated value at the gateway node is shown to be within constant error from the aggregation over all sensor readings.

The main advantage of this scheme is that most redundant nodes could sleep until the next query phase. Moreover, the cluster setup process is simple and requires very limited overhead only. However, CAG has a number of shortcomings. Firstly, it assumes only a simple correlation structure through simple edges. Secondly, a single aggregated value is obtained from the network to represent the entire region profile. Finally, for some threshold, all nodes may be forced to be CHs resulting in a flat data-gathering scheme. We will implement and improve the hybrid aggregations approaches, which would be more suitable for the fire detection of the emergency response system in Chapter 5.

## 4.4 Data Compression Level Approaches

The upper level of in-network data processing architecture is data compression level, which will compress the aggregated data in the cluster and send the compressed data to the gateway node. Most traditional compression algorithms are not directly acceptable for WSNs. First, in traditional compression approaches, the most important requirement is to save storage, not energy. However, in WSNs, energy is more important than memory. The energy efficiency is the major requirement for in-network data compression in WSN. On the other hand, it has been shown that, in terms of energy consumption, transmission of just one byte of data is equal to the execution of roughly four thousand to two million instructions.

### 4.4.1 Data Compression in Wireless Sensor Networks

In WSNs, the main objective of compression is to reduce energy consumption. Sampling compression, data compression and communication compression are three main techniques for energy efficiency in WSNs. In-network data compression is using one or more these techniques directly or indirectly for different applications.

Sampling Compression (SC) is the process of reducing the number of sensing or sampling operations while keeping network coverage and/or distortion loss within an acceptable margin. (Cardei et al., 2005) and (Subramanian and Fekri, 2006) deploy spatial correlation at to reduce the sensing tasks. These works focus on keeping the sensors in a sleep state, while a minimal number of sensors are kept active in the group. On one other hand, (Candes and Wakin, 2008) and (Haupt et al., 2008) perform the sampling-level compression by exploiting temporal data correlations at a sensor node.

Data compression is the process of converting an input data stream (raw data) into another data stream (compressed data) that has fewer bits. All non-random data has the same structure. It can be viewed as a way for discovering the structure that exists in the data and eliminating it by using more efficient encoding. This structure can be exploited to get compact data representation. After the data compression, the representation data will have no noticeable structure in it.

Communication compression is the process of reducing the number of packet transmission and receptions. Hence, it reduced the radio on time of transceivers a WSN. The longer the packet and the lower radio on-time will reduce the number of communications. The data aggregation, which we mention in in-network data processing, supports for reducing the communication cost.

There is a hierarchical relationship between the mentioned types of compressions (as shown in Figure 4-4). For instance, a reduced number of samples help in reducing the data length (data compression), which ultimately reduces the radio on-time of the transceivers (communication compression). However, very few of the existing compression techniques do so. They only operate on compressed samples or non-compressed samples. Our purpose is to exploit communication compression and data compression in WSN to reduce the energy consumption.

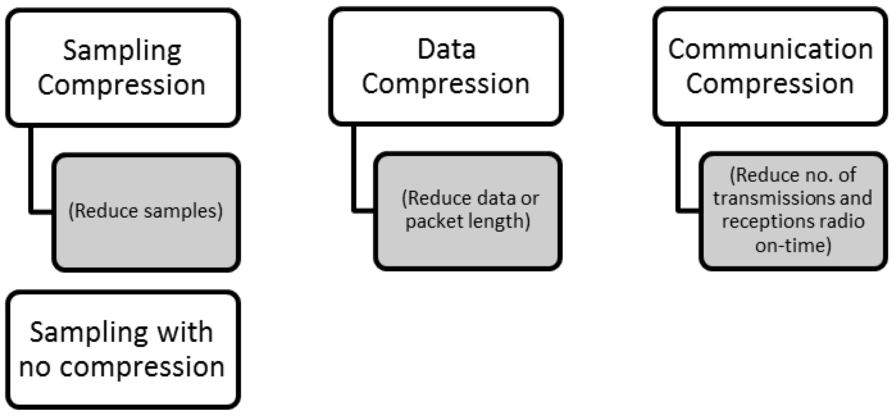

Sampling Compression
(Reduce samples)
Sampling with no compression
Data Compression
(Reduce data or packet length)
Communication Compression
(Reduce no. of transmissions and receptions radio on-time)

*Figure 4-4 the relationship between the compression algorithms*

### 4.4.2 The Proposed of Data Compression Level

In-network data processing architecture will be used in a wide range of application. This leads to a diverse range of requirements for data compression algorithms. For example, we will use it for emergency response system in simulation (see Chapter 6) and in-door location tracking system (see Chapter 8) for case study, which provide real-time user information and so can tolerate only bounded latency and data loss. We summarized the specific design requirements for data compression.

- Computation complexity and memory requirements：In Chapter 2, we mentioned WSN nodes are equipped with limited processing and memory capability. For instance, the popular WSN nodes such as Mica and Tmote Sky, are equipped with Atmel Atmega128L and Texas Instruments MSP430 micro-controllers (4-8 MHz clock speed), which have instruction and data memories of only 128 and 48 KB, respectively. Due to the limitations, we have to design a low-complexity and small code-size data compression algorithm for the in-network data processing architecture. For these requirements, data compression with asymmetric computations are often desirable, whereby most computation tasks placed in the sink or gateway node for the decoding, rather than at sensor nodes for encoding. Thus, the sensor nodes with minimal computational performance can efficiently compress data.

- On-route Compression. Traditional compression algorithms compress data at the source and decompress at the destination only. However, some WSN applications require that the data is available at intermediate nodes for on-route in-network processing or transformation. For our hybrid aggregation model, data compression schemes have to allow on-route compression and need to be sufficiently flexible to allow the inspection, modification, and aggregation at intermediate nodes. On-route compression algorithms can be particularly effective for heterogeneous networks consisting of different types of nodes. Lightweight compression at low-performance or mains-powered routing nodes.
- Real-time: WSN applications, which provide real-time user data or control solutions, such as fire detection and location tracking systems, require bounded latency. Hence, data compression needs to be performed on data at the same time. This can limit the compression ratio achieved. Therefore, spatial correlation data compression has to be exploit.

In compression level, a suitable encoding scheme is used to control the number of bits transmitted to the gateway node. However, the most significant disadvantage in such an approach is the need for all nodes to participate in all collection rounds. We present the in-network data processing architecture. It is only compressed data in the interested area. Then the compressed data is taken from the cluster head and transmitted to the gateway nodes.

For the hybrid aggregation and fire detection, there are two types of transmission packets in the network. The first category is the sensors “periodically” report what they have mentioned to the gateway node through clustering routing, which we call regular data report (RDR). Sensors usually generate RDR packets in a constant rate, and the gateway node can process them in long delay. We can compress those packets in higher compression rate. On the other hand, for fire detection systems, the second category is that sensors detect abnormal environmental data (for instance, the temperature data exceeds a predefined threshold in the fire detection system) and notify the gateway node of this specify situation, where we call them urgent event notification (UEN). Because events often appear in an unexpected manner, sensors may generate UEN packets in variable rate, but the gateway node should receive these packets as soon as possible. UEN is highest priority and accuracy due to their critical deadlines and multiple sensors in the same event detection region usually generate their UEN packets simultaneously. In hybrid aggregation, we address multi-path clustering to improve data accuracy. We have to consider both energy consumption and data accuracy in the data compression level. We decide to apply temporal-correlation and sample compression which is lightweight data compression approach for UEN packets. Hence, we address the data compression techniques for both UEN and RDR packets in Chapter 6.

## 4.5 Data Mining Level Approaches

Semantic data mining model is integrated with sensor measurements and knowledge based to convert sensor measurements into semantic measurements. For example, a temperature provided by heterogeneous sensors can be measured in Celsius. The semantic data mining model converts the sensor readings into semantically equivalent description. Data mining in sensor networks is the process of extracting application-oriented models and rules from continuous and rapid data streams from sensor networks.

Data clustering and data compression techniques have a significant benefit for energy efficiency and communication latency. However, for data mining in the gateway node, it must be designed to satisfy for not only energy efficiency and communication latency, but also the accuracy of the conveyed information. There are also constraints fault tolerance, which should effect the results in the applications.

### 4.5.1 Data Mining for In-network Event Detection

In terms of what constitutes a suitable sensor data mining algorithm, the following design requirements have to be fulfilled for in-network data processing architecture.

- Distributed

Kulakov and Davcev (2005) argued that sensor networks require "simple parallel distributed computation, distributed storage, data robustness and auto-classification of sensor readings". McConnell and Skillicorn (2005) stated that as sensors become active devices with their own processing capability, distributed algorithms in sensor networks become a new possibility. Bontempi and Borgne (2005) argued that the rationale behind distributed in-network data mining is that in-network aggregation can have two benefits: saving communication cost as well as reducing data dimensionality. Krivitski et al. (2008) also stated that in-network local algorithms have superb message pruning capabilities, thus they are "better than centralized algorithm both in terms of message efficiency and of convergence time".

- Data accuracy

The main purpose of data mining is to extract information from the sensory data and improve the data accuracy compared with the traditional event detection system. A variety of filtering methods has been proposed to improve data accuracy in sensor networks, including Bayesian Theory (Elnahrawy and Nath, 2003), Neural-fuzzy Systems (Petrosino and Staiano, 2007), Wavelet Transform (Zhuang and Chen, 2006), Kalman Filter (Tan et al., 2005) and Weighted Moving Average (Zhuang et al., 2007). Jeffery et al. (2006) proposed a pipeline framework for sensor data cleaning, which contains five sequential processing stages: point (filter out obvious outliers from individual sensor readings), smooth (aggregate sensor readings within a temporal granule), merge (aggregate sensor readings within a spatial granule), arbitrate (remove conflictions or duplicates between different spatial granules) and virtualize (combine readings from different types of devices or data sources). This framework covered the possible steps involved in sensor data cleaning in general. It can be implemented fully or partially according to specific scenarios.

- Data-driven/event-driven/service-driven rather than synchronization

Kulakov and Davcev (2005) noted “all previous work on distributed clustering assumes tight cooperation and synchronization between the processors containing the data and a central processor that collects the sufficient statistics needed in each step of the hill-climbing heuristic”. However, they argued that such synchronization controlled by a central point might not be suitable in sensor networks. Instead, they believed that data-driven is one of the most important features which qualify an algorithm for sensor networks. Bontempi and Borgne (2005) used the Lazy Learning algorithm in the upper level prediction of their two-level modular adaptive architecture. They claimed that such an algorithm assumes no a prior knowledge on the process underlying the data, only driven by “available information represented by a finite set of input/output observations”, which makes it appealing in the sensor network context.

Krivitski et al. (2008) also stated that the local majority voting algorithm they used has good performance in terms of message load and convergence time because it is event driven and requires no form of synchronization. Silberstein et al. (2007) proposed data-driven processing in sensor network. Rezgui and Eltoweissy (2007) introduced service-driven query routing.

In summary, the concept of data-driven/event-driven/service-driven rather than synchronization has been widely supported by research in sensor data mining.

### 4.5.2 Semantic Data Mining Framework

We deploy four components in the semantic data mining framework for fulfilling the re for emergency response system (see Chapter 7).

A. Problem formulation

In this stage, information and the users' requirements have to be formulated and identified. In general, these are questions, which are related to pattern and relations between data. Examples of data mining questions in emergency response include "How can we characterize a fire event?" and "Which groups of multi-sensor data at the current time, appear to suggest the occurrence of a fire event?" Data mining comes into its own when there are many possible relations between large amounts of questions that have to be evaluated (Feelders et al., 2001). In this stage, an intelligible description of interesting knowledge must be presented clearly. Subsequently, any discovered relations are used to make predictions about the situations occurring in current event.

According to the processing of emergency event, the set of system problems we need to consider can be divided into three stages. Firstly, the problems are "Does the system run correctly?" and "how can we build and update the event model?" These two questions are used to make sure the emergency response system will run smoothly. Secondly, we need to find the characters of the fire event, the location and danger level of the event, and the predicted spread. Finally, we have to evaluate the results and report this knowledge for decision making.

B. Pre-processing the data

Even if all related data are collected by WSN, it will often be necessary to pre-process the data before analysis (Crone, 2005). Data pre-processing generally includes the following two tasks:

Outlier detection: Outliers are data values that are not consistent with most other observations. The reasons for outliers' generation are usually measurement errors, coding and recording errors and other environmental elements. In emergency response, outliers can seriously affect the results of fire detection and could be one of the main causes of false alarms. There are two strategies for dealing with outliers. The first is to detect and eventually remove outliers as a part of the data pre-processing stage. The second is to develop robust modelling methods that are insensitive to outliers.

Data reduction and data projection: Data pre-processing tasks usually reduce significantly the amount of data, which decreases the size of the dataset for feature selection, while data projection, which alerts the representation of data. Crone et al. (2005) analyse various methods applied regarding whether parameter tuning was carried out, and which methods of data reduction and projection could be observed and present results. In the in-network architecture, it can be realized in the data compression level.

C. Real-time data mining methods

Real-time data mining (Blum et al. 2014) is the process of extracting interested and previously unknown or unexpected knowledge from large amount of real time data. Its correctness depends on not only logical correctness but also timing constraints. Various methods in the area of data mining have been developed in order to discover knowledge effectively and correctly. Data mining aims to extract patterns from data. Traditional data mining technologies include Decision Trees, Rule-based Classifiers, Artificial Neural Networks, Nearest Neighbour, Naive Bayes, Support Vector Machines, Logistic regression, etc. Based on a different dataset, the recent research on mining sensor data mainly focused on distributed in-network data mining.

D. Event detection

A knowledge base is a database that stores data and presents the features of specific events. It contains and represents discovered knowledge during real-time data mining.

One of any emergency response requirements is that the emergency response systems must be suitable for different environment and users' requirements. Accordingly, we built various models of emergency response systems. Model selection is conducted to select the relative models from a knowledge base for upcoming data analysis. In this process, as we mentioned above, it is mainly real-time classification methods that are used for choosing and classifying the various models. Many methods of model selection employ a simple approach, which prefers the use of the simplest model that fit the data (Zellner, 2001). For emergency response, model selection is mainly based on environment models such as global or local models. Next, the model evaluation step is used to evaluate model performance. There are two purposes of model evaluation: the prediction of how well the final model will work in the future, and as an integral part of methods, which help to find the best performance for data mining. Souza et al. indicate five patterns, which deal with the first case for the evaluation of data mining models. Finally, the last step in this stage is to update the existing model (model updating) which is to update models in the knowledge base and this may identify dynamic models for real-time emergency response systems.

## 4.6 Summary

The problem of efficient data gathering of correlated data in WSNs is a challenging one but promises a large saving in data transmission costs. This is particularly advantageous to sensor nodes where energy constraints are rather severe. In a periodic monitoring scenario, any form of compression or aggregation may result in significant energy savings. Based on the proposals, we design the in-network processing architecture into three levels: clustering level, data compression level, and data mining level. The clustering level aims to form an efficient aggregation by clusters that reduce the number of bits drastically. Such a structure should ideally be mapped onto the routing structure to realize opportunistic in-network aggregation clustering. Data compression level aims to reduce the number of bits transmitted after obtaining some information about correlations. Data mining is to design a rule for event detection to extract information from the raw data. In the following three chapters, we introduce the approaches for each level and the evaluations in details.

# Chapter 5. In-network Clustering Aggregation Level

This chapter introduces the clustering level in the in-network data processing architecture. In-network aggregation clustering includes an intermediate node or aggregates sensor collected from a group of sensor nodes before sending the processed data to the base station. For example, in a wireless sensor network comprised of temperature sensors, the aggregator can collect temperature readings from different sensor nodes in a given time interval, locally process these readings, and then forward and average these readings. It is the global process of gathering and routing information through a multichip network, processing data at intermediate nodes with the objective of reducing resource consumption (in particular energy), and thereby increasing network lifetime.

The chapter is structured as follows. Section 5.1 addresses the introduction of in-network aggregation. Section 5.2 proposes Neighbour-aware Multi-path Cluster aggregation (NMCA) Strategy for clustering level. Section 5.3 simulates and evaluates the performance of NMCA. Section 5.4 gives the conclusion of the chapter.

## 5.1 In-network Data Aggregation

In-network aggregation techniques require three basic ingredients: suitable network protocols, effective aggregation function, and efficient ways of representing the data. Several theoretical studies provide limitations and bounds on the performance of in-network data aggregation techniques and thus focus on the design of suitable algorithms. They can be classified into three classes: tree-based, cluster-based and multi-path.

Classic tree-based strategies (Erramilli et al., 2004 and Ding et al., 2003) are usually based on a hierarchical organization of the nodes in the network. A node may be marked as special depending on many factors such as its position within the data gathering tree (Yu, 2004), the type of data stored in its queue (Lindsey et al., 2002 and Yao et al., 2003), or the processing cost due to aggregation procedures (Zhou et al., 2004). Power-Efficient Gathering in Sensor Information System (PEGASIS) (Fasolo et al., 2007) is a tree-based aggregation algorithm, which is to organize the sensor nodes in a chain. Moreover, nodes take turns acting as the chain leader, where at every instant the chain leader is the only node allowed to transmit data directly to the gateway node or sink. But link failures and packet losses may affect the performance of this algorithm. In fact, the failure of any intermediate node compromises the delivery of all data aggregated sent by the previous nodes in the chain. Hence, some improvements to the scheme may be needed in order to increase its robustness.

Similar to the tree-based algorithms, the cluster-based schemes (Govindan et al., 2002, Akkaya and Younis, 2005, Mahimkar and Rappacit, 2004, and Nath et al., 2004) also consist of hierarchical organization of the network. Low-Energy Adaptive Clustering Hierarchy (LEACH) (Govindan et al, 2002) is a self-organizing and adaptive clustering algorithm using randomization to distribute the energy expenditure among the sensors evenly. But, several problems may arise in highly dynamic environments. In this case, continuous updates are needed in order to keep the clusters consistent with the underlying topology. This requires sending many control messages, which, in turn, may substantially affect performance. In addition, mobility additional problems may arise. A node close to a cluster head at a given instant in time may move away from the cluster head. Therefore, the node needs to increase its power, thereby spending much more energy to transmit to the cluster head than expected.

In order to overcome the robustness problems of aggregation trees, multi-path approaches were recently proposed (Al-Karaki and Kamal, 2004, Solis and Obraceka, 2005 and Manjhi, 2004). Synopsis Diffusion (Al-Karaki and Kamal, 2004) is a multipath algorithm where data aggregation is performed through a multipath approach. The underlying topology for data dissemination is organized in concentric rings around the gateway node. But, as the main feature of Synopsis Diffusion is that data can flow over multiple paths, a node may receive information for more than one time. This may affect the aggregation result, especially when aggregation functions are duplicate sensitive.

## 5.2 Neighbour-aware Multi-path Cluster Aggregation Strategy Description

In order to benefit from the advantage of tree-based, cluster-based and multi-path schemes, it is possible to define a hybrid approach we mentioned in previous chapters. It adaptively tunes the data aggregation structure for optimal performance. We try to overcome the problems of tree-based, cluster-based and multipath schemes and improve the energy efficiency and data accuracy of the network. The solution is a hybrid algorithm where the three data aggregation structures may simultaneously run in the network.

After establishing the network, all group members (sensor nodes) in each cluster are assigned to a cluster head to determine the priority and each cluster head in the network knows now its parents toward the gateway node through multiple paths. We propose neighbour-aware multi-path clustering aggregation (NMCA) algorithm for data processing and transmission in the network. NMCA works in three phases. The idea is that under low packet loss rates, a clustered-based aggregation is the most suitable structure and has good efficiency in representing and compressing the data. On the other hand, in case of high loss rates or transmitting partial results accumulated from many sensor readings, a multipath approach may be the option due to its increased robustness. Hence, we combine cluster-based and multipath approach, which can process different packet loss rates. Moreover, because of the characteristics of the spatial and temporal related, sensor data in the 'neighbour' range should be get the similar performance: increased, decreased or static. Hence, the neighbours' sensor data can be as the benchmark to decide the data accuracy of source node.

In the Neighbour-aware step, the source node ($S_i$) tracks the event and sends request to its one-hop neighbours ($N_i$). All one-hop neighbour nodes table containing a lists of neighbours, including their respective hop distance from cluster head, ID, level weight which is used to calculate data value for path determine, and residual energy. In cluster formation phase, the network is partitioned into clusters, each with a cluster head ($CH_i$). In the data transmission phase, the cluster head ($CH_i$) collects data and forwards them to gateway node along adaptive multiple or double-path paths (Figure 5-1). Dash arrow is the broadcast from the source nodes to its one hop neighbours. Solid line is the data transmission between the sensor nodes.

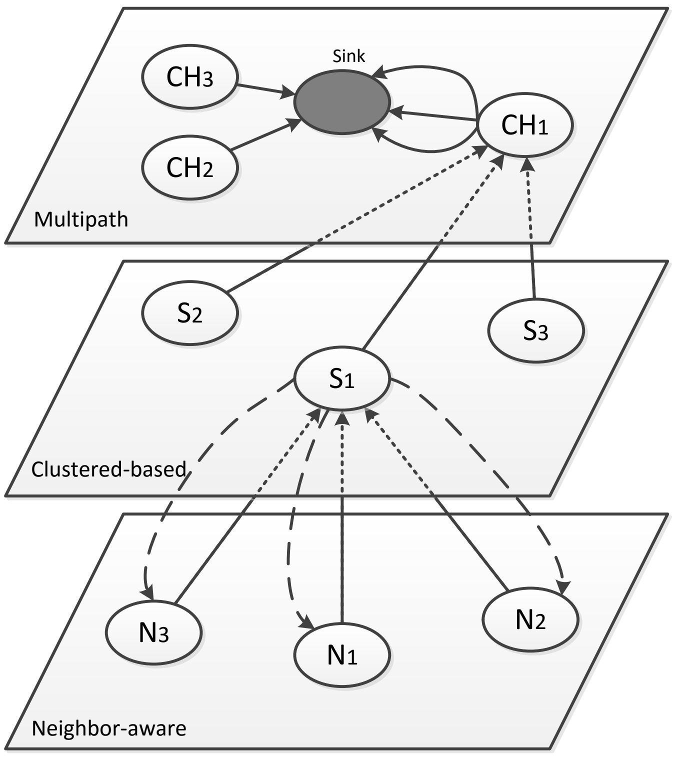


*Figure 5-1 The structure of NMCA*

### 5.2.1 Neighbour-aware Data Processing

The neighbour-aware data processing algorithm is the method used to decide the priority of each unusual data according to its neighbour data. If a set of nodes detects the event, they begin to send the data to the cluster-head. After a cluster-head receives the data from all nodes in the same cluster, the cluster-head performs data aggregation and sends data to the sink along multiple paths. The level of data will be the parameter of the number of paths. The more important data will decide more multiple paths to transmit the data to the sink.

The neighbour-aware processing is implemented in each of the sensor and forwarding nodes. It detects the unusual event and divides levels from the source and its neighbourhood. Table 6.1 shows the level weight for different status of source node and its neighbour. The neighbour weight is calculated according to the nearest neighbour table as in Table 5-1.

*Table 5-1 Level Weight*

| | Source threshold exceed | Source data increased | Neighbour threshold exceed | Neighbour data increased |
|---|---|---|---|---|
| Level weight | 1 | 1 | 1 | 1 |

The level weight is calculated by Equation 5-1:

$$L_i = ST_i + SI_i + NT_i + NI_i \quad (5\text{-}1)$$

If the source data exceeds threshold, $ST_i$ is 1. Then, $SI_i$ is based on whether the data in the packet increased significantly or not from the content of the previous from the same source. If the data increased over the given threshold, $SI_i$ will be set to 1. $NT_i$ and $NI_i$ are the status of the nearest neighbour of the source. $L_i$ is the total level weight for the source data. For example, there is an event in node A. Node A exceeds the threshold and increased indeed significantly. But there is not any change about its neighbour. So the level weight $L_i$=1+1+0+0=2. Because of its neighbour status, the data may be an outlier. Hence, we do not need to transmit through more paths to sink.

### 5.2.2 Cluster Construction for Data Aggregation

In wireless sensor networks, energy consumption of cluster-head nodes is much greater than other nodes because cluster-head nodes need to receive and aggregate the sensor data of other nodes. To equalize the energy load of all nodes in the network, cluster-head should randomly rotate among nodes. LEACH is one of the most popular clustering algorithms for WSN (Govindan et al, 2002). It forms clusters based on the received signal strength and uses the Cluster Head (CH) nodes as gateways to the BS. All the data processing such as data fusion and aggregation are locally performed within the cluster. LEACH forms clusters by using a distributed algorithm, where nodes make autonomous decisions without any centralized control. Initially a node decides to be a CH with a probability p and broadcasts its decision. Each non-CH node determines its cluster by choosing the CH that can be reached using the least communication energy. The role of being a CH is rotated periodically among the nodes of the cluster in order to balance the load. Guo et al. (2010) point out LEACH randomly selects the cluster heads without considering the residual energy of these nodes. As a result, the elected CH nodes can have the least energy level and consequently die very soon. Hence, in the cluster construction, we will consider about the residual energy for cluster head selection to improve the network lifetime.

First, in the setting phase, all nodes are organized to form cluster first and then cluster-head allocates time division multiple access time slot to cluster members and at the same time data gathering trees are formed among cluster-heads. In steady operation phase, cluster member nodes send data to cluster-heads transmitting aggregated data to sink node via the gathering tree. When residual energy of cluster head is less than a certain threshold value, new cluster heads are selected and the data gathering tree is reconstructed.

NMCA algorithm separates sensor network into a two-layer structure. Cluster-head nodes form the upper layer of backbone network nodes and cluster member nodes form the lower layer of other nodes.

In the cluster-head election algorithm, the ratio of nodes' residual energy and the mean residual energy of all the neighbour nodes is considered as a major parameter for cluster-head competition and the mean transmission distance of the node and its entire neighbour as the minor parameter. The advantage is that nodes with higher residual energy and shorter mean transmission distance more easily become a cluster-head node, which efficiently reduces the energy consumption of data transmission and prolongs the sensor network lifetime.

The procedure of clustering algorithm is shown as follows.

At the beginning of each round of rotation, when the node value in a random position exceeds the threshold, the node broadcast message of the event E_Msg (ID, s(i)) with radius r which includes sensor node ID, residual energy and node coordinate. Any other node within communication radius r is considered as neighbour from where the nodes receive the message and updates the neighbour information table.

Every node with residual energy higher than threshold has the chance to participate in cluster-head competition and therefor becomes a cluster-head candidate, which obtains the mean residual energy $E_{ai}$ of all neighbour nodes according to the updated neighbour information table, as Equation 5-2.

$$E_{ai} = \sum_{j=1}^{m_i} \frac{E_{rj}}{m_i} \tag{5-2}$$

We consider $v_j(1 \leq j \leq mi)$ as neighbour of node vi. $E_{rj}$ represents the residual energy of nodes of $v_j$. $m_i$ represents the total amount of neighbour nodes of $v_i$ and it is easy to determine the mean communication distance and mean data value amount node $v_i$ and its neighbour nodes.

$$d_i = \sum_{i=1,j \neq i}^{m_i} \frac{d_{ij}}{m_i} \tag{5-3}$$

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{5-4}$$

$$data_i = \sum_{i=1,j \neq i}^{m_i} \frac{data_i + data_j}{m_i} \tag{5-5}$$

$d_{ij}$ is the distance between node $v_i$ and $v_j$. $data_i$ is the mean data value of node $v_i$ and its neighbour.

Each cluster-head candidate calculates the competition bits of being elected using Equitation 5-6 in which the value α and β is determined by the distribution of nodes within cluster and their residual energy situation. Then, the cluster-head candidate broadcasts cluster-head competition message cluster header ($ID_i$, $CB_i$) with radius r.

$$CB_i = \alpha \frac{E_{ri}}{E_{ai}} + \beta \frac{1}{d_i} \tag{5-6}$$

All the cluster head candidates are set in receive state and wait a time T. the length of

T is determined to make sure that the nodes can receive the competition message from all nodes within communication radios r. The computation formula is as Equation 5-7.

$$T = \frac{k}{B} m_{max} \tag{5-7}$$

k represents the bit number of single packet and B is the channel bandwidth. $m_{max}$ is the maximum number of member nodes in one cluster.

After a time length, if no packet is received by cluster-head candidate, it demonstrates that there is no other competing for cluster-head within the coverage region of the node so that the node broadcasts competition success message with radius r. Otherwise, comparing the residual energy of itself and all others to choose node with the highest energy as the cluster-head.

Every cluster-head node calculates the data level weight $W_i$ using level weight table (Table 5-1).

The aggregated data CH_AD (ID, Wi, s(i)) are sent to the sink through multiple paths which will introduce in the following section.

### 5.2.3 Multi-path Data Transmission

When the scope of WSN gets larger, the diversity of energy consumption among cluster-heads as well gets larger.

Our loss model is simple: drops are independent and the loss probability *p* is fixed for all links and is kept constant throughput the simulation. A packet is successfully transmitted with the probability (1-*p*). The strategy of course creates a trade-off between consuming energy to send packets and increasing the accuracy probability of delivery. Hence, the single path schemes might not be suitable for all scenarios. In fact, when data are generated sporadically rather than periodically, the error recovery will likely to more cost-effective.

We use three different transmission schemes for multi-path aggregation, namely double-path, max-path, and adaptive-path. While double-path, as its name implies, sends every packet twice through two different paths, max-path sends each packet to as many as reading are aggregated in the packet. We assume aggregation counter in the data packet. The reason of max-path is that packets carrying or processing more readings are more important and valuable and we therefore want to increase their accuracy and chances of getting through.

Adaptive-path is a more complex algorithm compared with double-path and max-path. The goal of adaptive-path is to achieve certain delivery guarantees (expressed by number of acceptable losses) for a given loss rate. Equation 5-8 describe the relationship between the number of acceptable losses $l$ and the drop probability $p$, where $a$ and $t$ are the level weight of aggregation reading and the number of transmissions, respectively. The number of transmissions to achieve $l$ under loss probability is given by Equation 5-8.

$$l = L_i p^t \tag{5-8}$$

$$\mathrm{t} = log_p\left(\frac{l}{L_i}\right) \tag{5-9}$$

According to the Equation 5-9, we calculate the number of delivery paths for different level aggregation data. Note that, these equations apply to a single link. In other words, if l=0.05, adaptive-path will perform the necessary number of retransmission to guarantee a 95% delivery data guarantee over a particular link given that link’s loss rate. The overall network’s delivery guarantee may depend on the interdependencies between the aggregates in the cluster head.

## 5.3 Simulation and Results of the Network Model

First, assume that the network contains m sensor nodes that sense and acquire information. A unique ID is assigned to each node after nodes are deployed randomly in the sensing field. The network composed of a sink and many nodes in an interesting area is considered. Nodes keep static or less movement. The coverage area is a square region of n×n and other parameters or properties are as follows.

- Sink nodes are deployed at a fixed and exclusive location inside the network region.

- It is a high density and static network that nodes are motionless after deployed and with the same properties and initial energy.

- Nodes are all with the function of NMCA and the transmit power is controllable.

- The transmission packet includes the location information of the special event, the aggregation results and transmission route.

Second, the node originally consists of several parameters. Figure 5-2 shows the format of sensor nodes. Third, we propose the assumption of energy model. There is a great deal of research in the area of low energy radios. Different assumptions about the radio characteristics, including energy consumption in transmit and receive modes will affect the result of energy model. In our work, we assume a simple energy model where the radio dissipates $E_{elec}$=50nJ/bit to run the transmitter or receive circuitry and $\varepsilon_{amp}$=100 pJ/(bit.m-2) for the transmit amplifier. Thus, to transmit a k-bit message a distance d using the energy model, which expends:

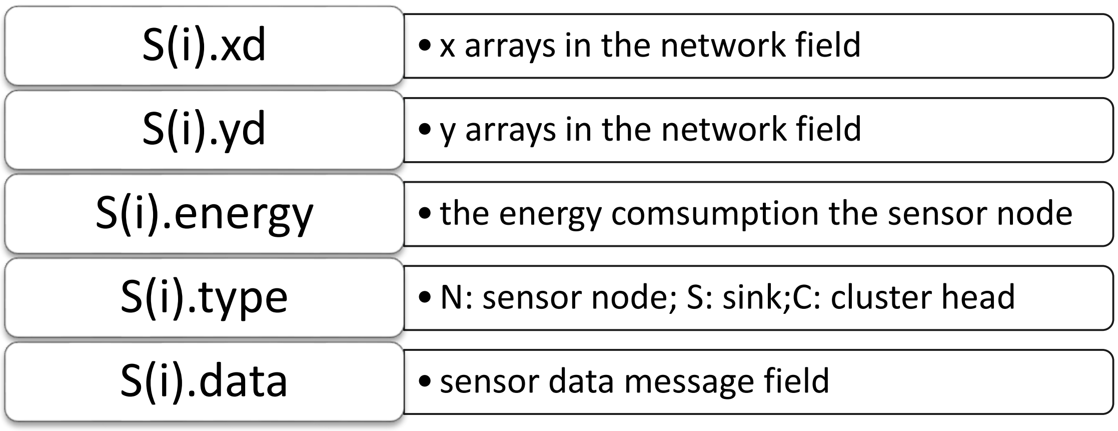

S(i).xd
• x arrays in the network field
S(i).yd
• y arrays in the network field
S(i).energy
• the energy comsumption the sensor node
S(i).type
• N: sensor node; S: sink;C: cluster head
S(i).data
• sensor data message field

*Figure 5-2 The format of sensor data structure*

$$E_{\mathrm{TX}}(k,d) = E_{TX-elec}(k) + E_{TX-amp}(k,d)$$

$$= E_{elec}k + \varepsilon_{amp}kd^2 \quad (5\text{-}10)$$

and to receive this message, the energy expends:

$$E_{RX}(k) = E_{RX-elec}(k) = E_{elec}k \quad (5\text{-}11)$$

In Equation. 5-10. $E_{TX}$(k,d) denotes the total energy dissipated in the transmitter of the source node, while $E_{RX}$(k) in the Equation. 5-11 represents the energy cost incurred in the receiver of the destination node for transmission and reception, respectively, and $\varepsilon_{amp}$ is the transmitted amplifier parameter to denote the energy required by the transmission amplifier to maintain an acceptable radio for transferring data reliably. The set of parameters given and assumed for all experiments in the work: $E_{TX}$=$E_{RX}$=50 nI.bit, $\mathcal{E}_{amp}$=1-pJ/b/m2.

For these parameter values, receiving a message is not a low cost operation. Thus, we should try to minimize not only the transmit distances but also the number of transmit and receive operation for each message.

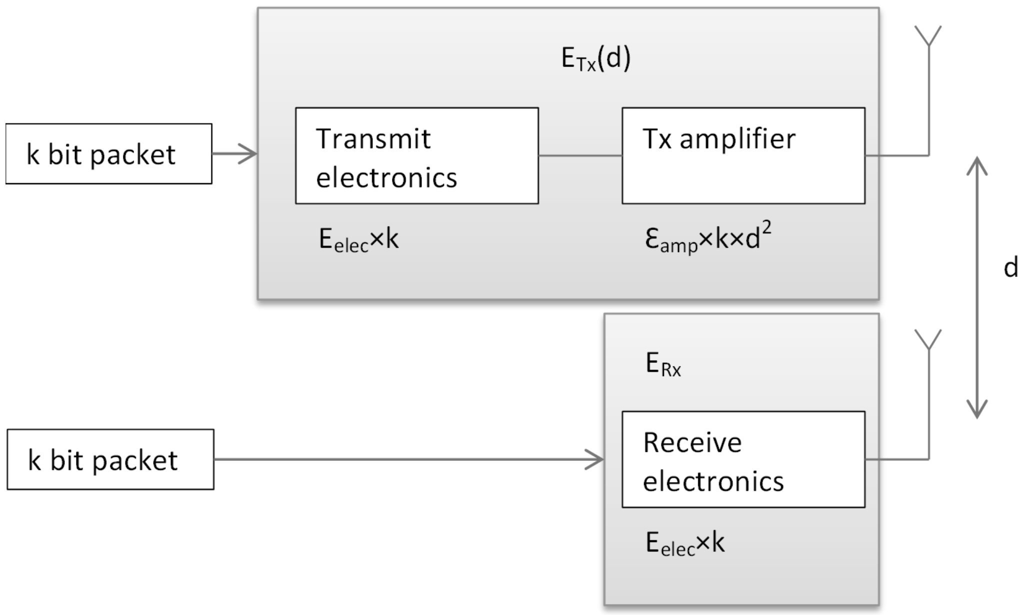


*Figure 5-3 Energy model*

In addition, clustering aggregation can reduce the message transmission operation. Data aggregation costs some energy and the energy consumption for aggregating a certain data signal is represented as $E_{DA}$. Thereby, the energy consumption of each aggregation node for aggregating the data itself and m neighbour nodes is:

$$E_c = (m+1)E_{DA}k \quad (5\text{-}12)$$

k is the amount of raw data generated by each node. The energy cost for the data aggregation was set to $E_{DA}$=5nJ/b/message. The energy model set up as followed.

```
%Energy Model (all values in Joules)
%Initial Energy
Eo=0.5;
%Eelec=Etx=Erx
ETX=50*0.000000001;
ERX=50*0.000000001;
%Transmit Amplifier types
```

```
Efs=10*0.000000000001;
Emp=0.0013*0.000000000001;
%Data Aggregation Energy
EDA=5*0.000000001;
```

In this network model, the energy consumption patterns mainly include the data processing and transmission. Data transmission cost is generally more expensive than data processing.

In order to evaluate the performance of NMCA, several simulation experiments with various random topologies were run. We implement out proposed algorithm using Matlab. We simulate the network of size 100 m × 100 m deployed with 100 sensor nodes in randomized grid. The sink is located in the centre of the network. Table 5-2 shows our simulation environment and other parameters used in our simulation.

*Table 5-2 Simulation Parameters*

| **Parameters** | **Symbol** | **Parameters Values** | **Unit** |
|---|---|---|---|
| **Network scale** | S | 100×100 | $m^2$ |
| **Sink node coordinate** | sink.x, sink.y | (50,50) | m |
| **Total amount of nodes** | n | 100 | |
| **Numbers of Cluster-head** | | 5 | |
| **round** | nr | 5000 | |
| **Communication radius** | r | 20 | m |
| **Initial energy** | | 2 | J |
| **Radio dissipates** | $E_{elec}$ | 50 | nJ/bit |
| **Transmit distance** | $\varepsilon_{amp}$ | 0.0013 | $pJ/(bit \cdot m^{-2})$ |

| | | | |
|---|---|---|---|
| **Reference distance** | $d_0$ | 87 | M |
| **Packet size of single sampling** | k | 4000 | bit |

The network simulation scene under randomly dissemination is as Figure 5-4. After interactivity between each node and its neighbours, nodes determine their own status (cluster-head nodes or regular nodes) according to their competition bids and form the network topologies. Then, each cluster-head node constructs the data aggregation and multi-path data transmission according their level, which we mentioned before. The ‘x’ in the centre of region is the sink. The ‘+’ is the cluster head which should be changed in each round. The black ‘*’ is the node which transmit or collect data. If the node’s energy is lower than threshold, it will set as a red point, which means the node is dead.

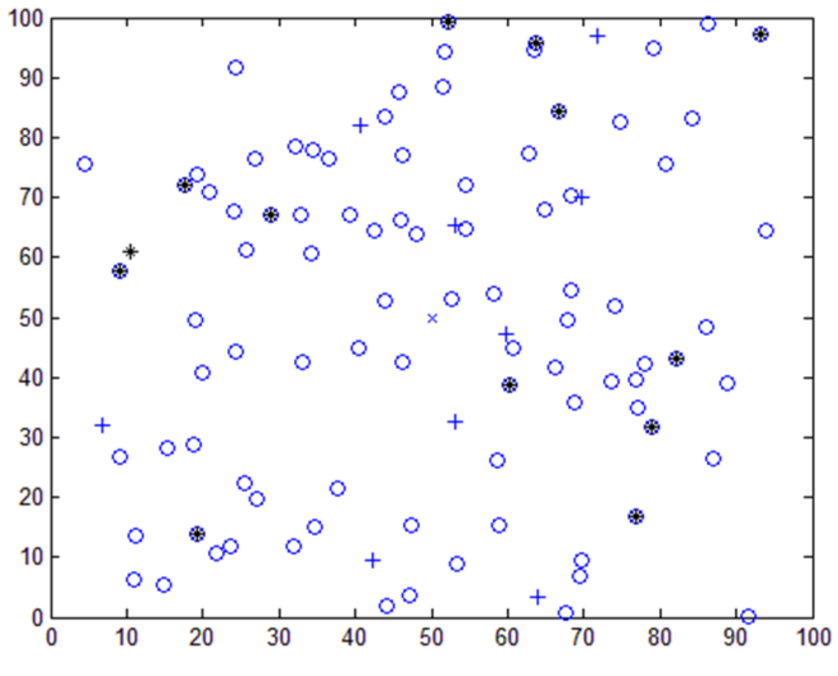


*Figure 5-4 Scene of randomly generated 100 nodes, 10 cluster-head nodes and a sink*

To evaluate the performance of NMCA, such as network lifetime, data accuracy and data throughput, we have compared NMCA, no aggregation algorithm, Clustering

Aggregation, double-paths, max-paths and single path algorithm.

### 5.3.1 Results and Analysis

In this subsection, we choose to use the work of LEACH (Govindan et al, 2002) as the benchmark for comparison. LEACH is the most popular clustering aggregation for WSN (Md and Koo, 2012). The performance is evaluated mainly, according to network lifetime, data accuracy and data throughput.

#### a) Network lifetime

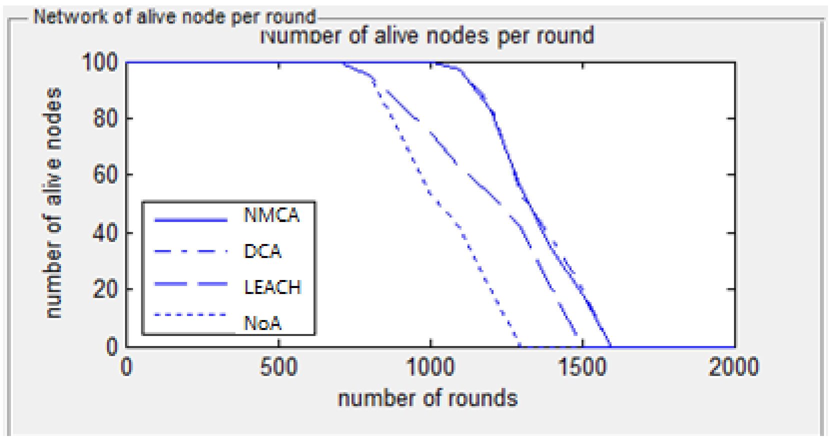


*Figure 5-5 Network lifetime*

Network lifetime is one of the most important performance indicators of energy consumption in WSNs. Figure 5-5 shows the comparison of network lifetime using NMCA, Douth-path Cluster aggregation(DCA), LEACH, and No-aggregation Algorithm (NoA).

LEACH forms clusters by using a distributed algorithm, where nodes make autonomous decisions without any centralized control. Initially a node decides to be a CH with a probability p and broadcasts its decision. Each non-CH node determines its cluster by choosing the CH that can be reached using the least communication energy. The role of being a CH is rotated periodically among the nodes of the cluster in order to balance the load. The rotation is performed by getting each node i to choose a random number T(i) between 0 and1. A node i becomes a CH for the current rotation round if the number T (i) is less than the following threshold:

$$T(i)= \frac{p}{1-(r.mod\frac{1}{p})}1, i\in\ G\,(1) \qquad 5\text{-}13$$

Where p is the desired percentage of CH nodes in the sensor population, r is the current round number, and G is the set of nodes that have not been CHs in the last 1/p rounds. LEACH forms one-hop cluster topology where each node can transmit directly to the CH and thereafter to the gateway nodes or base station.

In Double-path cluster aggregation, we set up two shortest path to transmit the aggregated data. The network lifetime of NMCA and double-path are almost similar. Due to NMCA using an extended max-path algorithm, it may cost more energy than double-path algorithm. In the 1100 rounds, the proposed NMCA increases the nodes' lifetime by 20% for LEACH and 55% for no-aggregation algorithm.

### b) Data accuracy

We present the result for different path schemes: max-path (MAX-P), double-path (DP) and single-path (SP) in Figure 5-7. However, the current work did not employ error recovery, which should be in the future work.

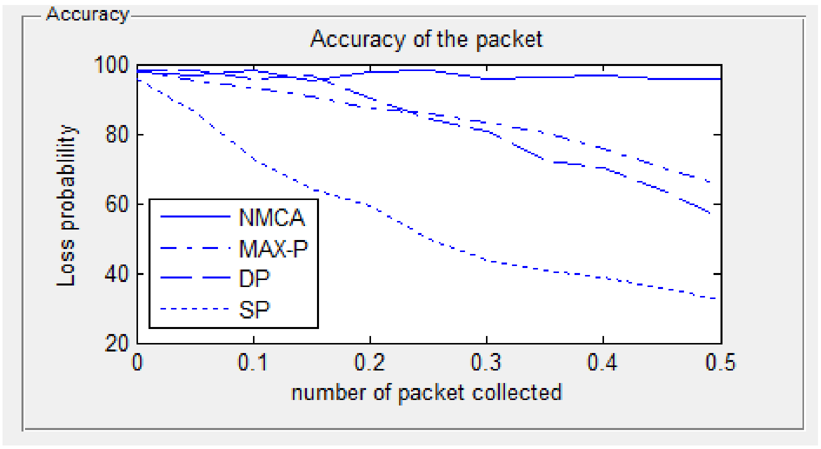


*Figure 5-6 Data accuracy*

The results obtained for the different algorithms bring out some interesting points. Figure 5-6 shows the data accuracy (percentage) in the sink. At the beginning, we notice that double-path may be slightly better than max-path and NMCA. Because the double-path is to transit each data twice to the sink node, even the packets coming with single reading or less level weight. But in a long time, there are more and more aggregated data, which means more loss probability in the transmission. NMCA can guarantee data accuracy by 95%.

### c) Data throughput

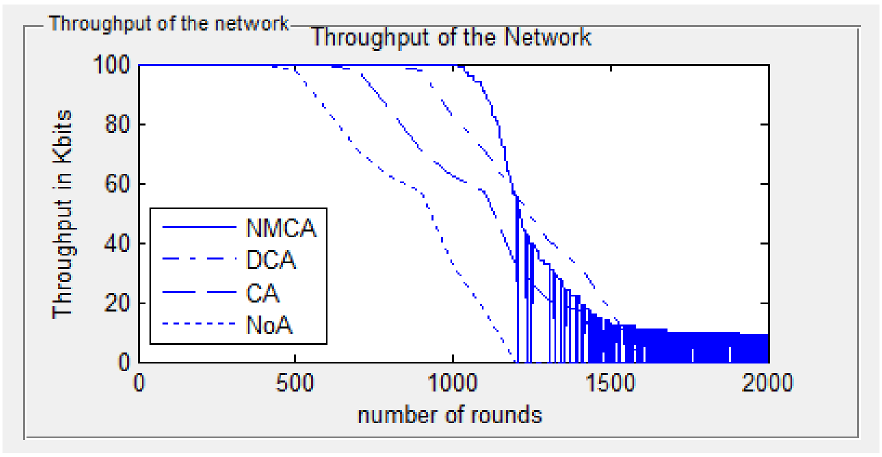


*Figure 5-7 Data throughput*

The data throughput can be represented by standard deviation, which gives an average variance between energy levels on all nodes. All nodes in the network are sources. In Figure 5-7, the mean throughput of our algorithm is shown compared with other algorithms: Double-path Cluster Aggregation (DCA), Cluster Aggregation (CA) and No-Aggregation algorithm (NoA). The throughput is measured as the total number of packets received at the sink over the simulation period. NMCA can select multiple paths of better link quality and minimum delay leads to load balancing and efficient utilization of the wireless spectrum. As the result, it achieves higher throughput and much better performance than the other algorithms.

## 5.4 Summary

This chapter explored in-network cluster aggregation as a power-efficient and accurate mechanism for processing and collecting data in wireless sensor networks. Our focus was on applications where a large number of nodes produce data periodically, which is consumed by fewer sink nodes. Using a simple energy model, we present NMCA algorithm. Through simulations, we evaluate the performance of different in-network aggregation algorithms, including network lifetime, data accuracy and throughput of the network. Our algorithm divides the senor data in five levels by their neighbour data before forwarding data onto the next hop; aggregates data in cluster head and transmit through multi-path toward the gateway node. The NMCA is a simple and lightweight algorithm that is specially designed for WSN. It improves the energy consumption and the accuracy of event detection significantly. It is more suitable for emergency system, which require the fast and accurate data processing in the network. However, for streaming data sources, data compression is used at the nodes to reduce the amount of data transmitted over network hence reducing energy used for transmission. In-network Data compression requests the intermediate node has amount of dataset. Data compression in the cluster head can be achieved the requirements. In the following chapter, we will propose the in-network data compression algorithms.

# Chapter 6. Data Compression Level

This chapter introduces the data compression level in the in-network data processing architecture. The design of data compression level is based on the clustering level structure. We proposed the NMCA technique in Chapter 5, which indicates the architecture of in-network data processing. It is good for detect the event and process the event data in a short period. However, for human behavior monitoring or the alter environment information, the sensor data have to be processed by lossless processing approaches. In this chapter, we will present in-network lossless data compression approaches, which gives more effective data. We propose the communication compression scheme for both regular and alert packets and Optimal Dynamic Huffman algorithm and compare their performance in the simulation. The data compression algorithm can make the energy efficient for the continuous data stream and a long period. But compressing the data at the node itself requires more storage and compute resources, by using the cluster head compression strategy best of both worlds can be achieved.

The chapter is structured as follows. Section 6.1 gives a high-level description of the focus and motivations employed in the design of data compression level. Section 6.2 presents the data compression scheme for both general data and event data packets. Section 6.3 presents Optimal Dynamic Huffman algorithm for compression data in the cluster head. The evaluation and performance of data compression level discussed in the Section 6.4 and the chapter concludes with a summary in Section 6.5.

## 6.1 Data Compression Level

In the data compression level, we point out the effective ways to represent and compress the data for energy consumption. A wireless sensor network includes a large number of small sensor nodes. Due to limited resource constraint, it needs to decide whether to store, compress, discard or transmit data. This entire requirement wants a suitable way to represent the information any type of structure are common to all sensor node in the network. WSNs can offer mobility and versatility for a variety of applications, such as object detection/tracking, environment monitoring and traffic control. The main challenge is that they often rely on batteries for power supply and limiting energy consumption becomes essential to ensure network survivability. As we mentioned in Chapter 2, the size of each sensor node is expected to be small. To achieve its requirements, the sizes of each component in sensor node have to be small, such as the power source, processing ability and data storing memory. In addition, a large number of sensor nodes will be often deployed to the location, where it is hard to access. It is not practical to perform maintenance operations, such as changing batteries, on deployed sensor nodes. Because of the above reasons, WSNs, or more specifically each sensor node, are resource constrained. They have limited power supply, bandwidth for communication, processing speed, and memory space. Therefore, many studies conducted so far have focused on how to achieve the maximum utilization of limited sensor resource.

One field of resource utilization studies for sensor networks is data compression. Researchers seek the optimal way to compress the sensing data. By doing so, it will reduce the power consumption due to processing and transmitting data in each node, and thus extends the lifetime of the sensor network. In addition, by reducing data size less bandwidth is required for sending and receiving data. The data compression is one effective method to utilize limited resources of WSNs. When data is acquired at multiple correlated sources, aggregation involving in-network data compression can offer a more efficient representation of measurements.

## 6.2 Communication Compression Scheme of UEN and RDR

According to most WSN applications and their limitations, we classified two categories for the data generated from sensors in Section 4.4.2. The general continued sensor report is regular data report (RDR). The special or interested event data is urgent event notification (UEN). NMCA proposed to process the UEN by the neighbour-aware clustering method and aggregate the result for further transmission. It is noteworthy that RDR and UEN packets can coexist in a WSN. For example, fire detection for emergency response also can periodically send RDR packets to the gateway node in order to construct the temperature map of the building. Nevertheless, once the sensor detected unusually high temperature, the system has to send UEN packets immediately to the fire station for emergency response as quickly as possible.

For RDR traffics, sensors will send a large amount of packets to the gateway node, which not only occupies most bandwidth but also reduces the lifetime of the WSN. This means UEN packets may be processed after completing the amount of RDR, and thus they cannot arrive in time to the gateway node. As we mentioned before, the non-chargeable batteries is another critical concern to them. One of the feasible solutions to deal with the aforementioned problems is for NMCA to cluster and aggregate data for in-network data reduction. However, NMCA only gets an average of related data in the cluster. If the users need the whole situation in the area, we have to use a data compression algorithm to compress data for in-network data reduction.

On the other hand, for UEN packet, when an event occurs, will be detected by many nearby sensors, which instantly and simultaneously generate a large number of UEN packets to describe the same event. These packets not only contend for sending to the gateway node, but also need the wireless medium transmission, which requires the data compression technique to be lightweight and on-route. We call such a phenomenon the neighbour-aware congestion because the network will be congested by these UEN packets during a small period of event occurrence. We will present NCS for neighbour-aware UEN when an event occurs.

We introduce the identification of communication compression in Section 4.4.1. In the following section, we propose the communication compression scheme for UEN, which is used for compressing the number of packets and reducing redundancy before data compression in the cluster head. Data compression scheme for RDR is to set the waiting time for collecting data in the cluster head, which can compress by Optimal Huffman Algorithm.

### 6.2.1 Communication Compression for UEN Packets

When an event appears, there could be multiple sensors detecting its appearance. Furthermore, some types of events such as temperature increment usually occur in a small region containing a number of sensors at the beginning of a fire. In this case, we set the sensing field with an event detection region, which all sensors are aware of the same event. Those sensors may be the neighbours of each other. Because the sensors in an event detection region actually observe the same event, these UEN packets should be processed by spatial correlation. We proposed neighbour-aware algorithm in Chapter 5. In the neighbour-aware area, the source node only needs to consider the nearest neighbour, which should be a small local range of the network. If the query is required to collect a range of source nodes data, not only one source node data, we can extend one hop neighbour to a small range of neighbours for compression scheme. In the small range, it can be feasible to apply distributed source coding locally within the neighbour's range of target node. In this case, the source node collects the knowledge of local correlation structure to perform coding by neighbour-aware algorithm.

Therefore, we propose Neighbour-aware Compression Scheme (NCS) a simple but practical communication compression method to observe the event region without the position information of sensors to reduce redundant packages. Because sensors will transmit the UEN packets to the same destination node (gateway node) and they are located inside the same event region. There may be same neighbours in the same event region. It can process neighbour-aware algorithm in the NMCA to determine whether UEN packets describes the same event or not. If so, it can send out only one compressed UEN packet rather than all of them. The details of the approaches are followed. We define a five-tuple format for UEN packets generate by a sensor $S_i$:

$$(ID_i, t_i, type_i, value_i, RE_i) \quad (6\text{-}1)$$

When $ID_i$ is the identification of sensor $s_i$, $t_i$ is the current timestamp of event being detected, $type_i$ is the type of the sensor, $value_i$ is the value of reading, and REi means the region of event. If $s_i$ detects the event and generate the corresponding UEN packet, its $RE_i$ set as an initial value $\gamma \in \mathrm{N}$. Then, when UEN packet through neighbour-aware algorithm to relayed in one hop, its $RE_i$ will be minus 1 (until $RE_i$=1). The region $RE_i$ is depending on the expected diameter of an event detection region by hop count. For example, in location tracking system, we expect that a target tracking region in three hops of sensor, we can set $\gamma = 3$. But for fire detection system, the events may spread out over a large range, $\gamma$ can be assigned as a larger value.

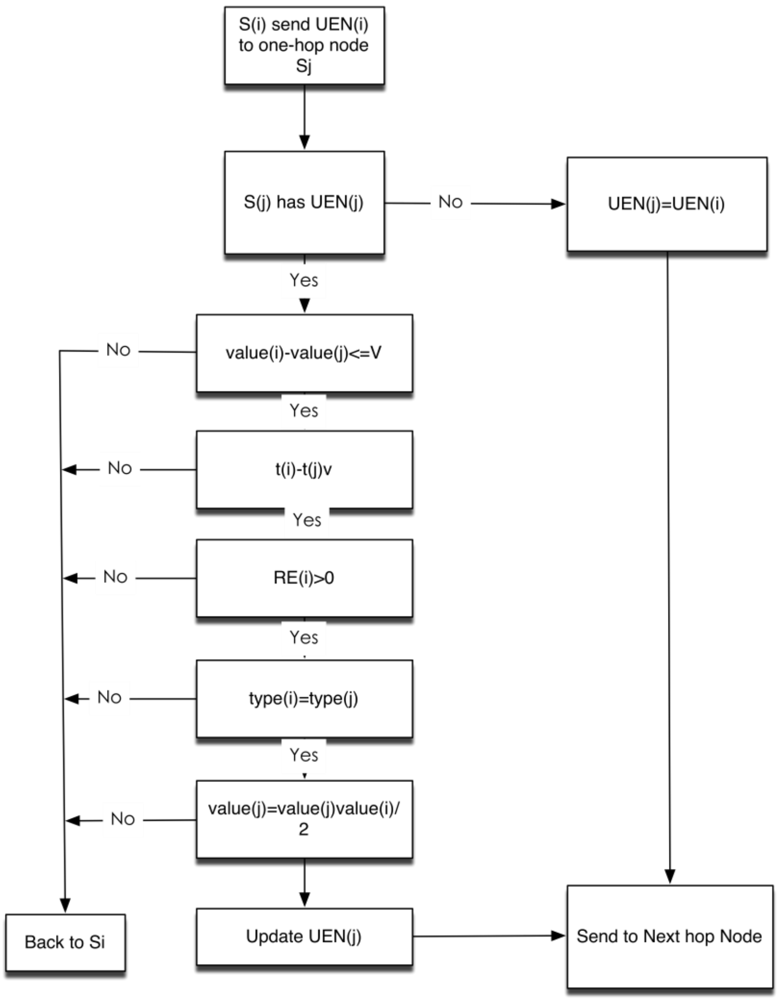


*Figure 6-1 the flow chart of Neighbour-aware compression scheme*

We define two threshold value I and V for sensor to determine whether the received UEN is in the same event to reduce redundant in WSN, where I and V depends on the application requirements. The received UEN packet, as the neighbour-aware compressed result, is sent to the cluster head for furthering processing. Figure 6.1 shows the flow chart of UEN packet generation. For example, the sensor $s_j$ receives a UEN package from its one-hop neighbour $s_i$. The UEN package is:

$$p_i^u=(s_i, t_i, type_i, value_i, RE_i) \qquad (6\text{-}2)$$

If the sensor $s_j$ generates its own UEN packet and then aggregates new UEN package of the following four conditions are all satisfied (shown in Figure 6-1):

1. $\left|t_i - t_j\right| \leq I$: this condition presents that both sensor $s_i$ and $s_j$ observe the event at the similar time stamp. It means they are temporal correlation.
2. $type_i = type_j$: this condition indicates these two sensor observe the same type of event.
3. $\left|value_i - value_j\right| \leq V$: the condition means the readings of those two sensors are similar, which indicates their observations have spatial correlation.
4. $RE_i > 0$: RE is the region of event. It means $s_i$ and $s_j$ in the same event region.

If $s_j$ does not have its UEN, we set $UEN_j$=$UEN_j$ and transmit this package to next hop neighbour sensor. Those four conditions together indicate there are high correlation between $s_i$ and $s_j$; we can discard them for communication compression to reduce the packet of transmission. It is safe to discard the redundant packet.

Another problem is simultaneous multi-event in the WSN. If there be two or more events occur in the same time in one cluster, we cannot determine whether those conditions are all satisfied to discard those UEN packets. Because both of them are very import packets for different events. Hence, RE has another function to discuss whether the UEN package is the end package of one event. It RE=0, it means the UEN package is on behalf of the event. The package cannot discard any other packages. Figure 6-2 shows an example of NCS for UEN. First, in region 1, when we assume that $\gamma$=3. Sensor s3 will send only UEN package $p_3^u$ on behalf of s1, s2, s3. The UEN packages of s2 and s1 will be discarded. On the other hand, for s4, s5, s6, only send the package $p_6^u$ to the sensor s8. In the sensor s8, it will discard $p_6^u$ and take $p_7^u$ for the end UEN package. Therefore, only two UEN packages are sent to the cluster head: $p_3^u$ and $p_7^u$ for the event in the region 1. But for different event at the same time, $p_9^u$ is the end UEN package in the region 2. In sensor s10, it will receive $p_7^u$ and $p_9^u$. Because both of them have a zero REi. s10 will reply both of them to the cluster head. We use REi to help distinguish different events in the spatial domain

.

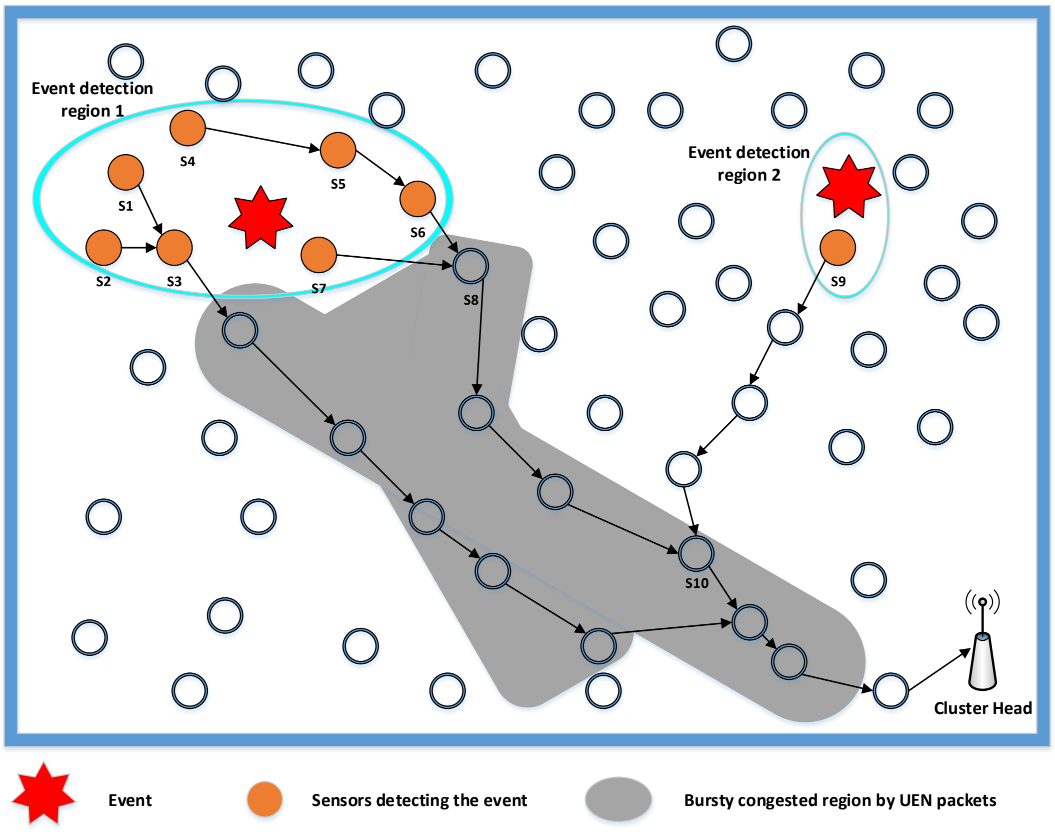


*Figure 6-2 Communication compression scheme for UEN packets*

One of the most important constraints for UEN is that it should be real-time and lightweight. For emergency system, data compression in cluster head should be as quick as possible and needs to consider the storage of the cluster head.

In this section, we propose the neighbour-aware compression by Huffman coding for processing UEN packets, when the cluster head receives the low redundant UEN packet from the event region. Because the spatio-temporal correlation among sensor observations are significant and unique characteristics of WSN. Therefore, it is unnecessary for every sensor node to send redundant information to the gateway node. The proposal techniques are to combine the data and package funneling techniques for data compression and related data compression techniques for communication compression.

This compression scheme is suitable for the small size area and UEN package. If we apply the compression scheme in cluster for collecting RDR package, it is impossible to collect data from the neighbour cluster, which would cost more power in the transmission scheme. In other words, the larger sensor network and the more clusters are required to apply other communication compression schemes for RDR in the following section.

### 6.2.2 Communication Compression for RDR

For RDR or larger amount of sensor networks, those data should be not important and urgent like UEN. However, there is a large amount of data, which can indicate the whole situation of the network. If the user requests the details of the whole network, NMCA can offer the general results of the network. Alternatively, if the user prefers to get every data in each sensor in the network, we have to use the communication compression to reduce the data transmission for the network lifetime. In this section, we present the communication compression for the RDR packets.

First, each sensor $s_i$ has its own data rate $r_i$ to generate RDR packets in order to report its monitoring to the gateway node. $\delta$ is the threshold number of RDR packets required to be compressed into one compressed data packet. For example, where $\delta \geq 2$ is a predefined parameter, sensor data in the cluster head will be compressed and decompressed in the gateway node. But if $\delta$ is less than the parameter, that means only a few packets in the cluster, RDR does not need to compress and send to the gateway directly. Each cluster head has two choices to determine whether to send out CDP or RDR.

1. Accumulate $\delta$ RDR packets in the cluster head and then only send one CDP packet to the gateway node by ODH. In this case, the neighbour cluster in the routing just relays the CDP without any other further processing.
2. Send a number of RDR to the neighbour cluster head. In this case, the neighbour may compress these packets together with its own RDR packets in the cluster by ODH.

We should guarantee that $b_i < \delta$, where $b_i$ is the number of RDR packet residing in the buffer of sensor $s_i$. In case of $b_i \geq \delta$, sensor $s_i$ has to immediately compress δ RDR packets to one CDP. The queuing time of a CDP, which means the sensor data packet stay in the cluster head's buffer:

$$T_i^A = r_i \times (\delta - b_i) + f_c(\delta) \tag{6-3}$$

where $f_c(\delta)$ is the time spent to compress δ RDR packets and while $r_i \times (\delta - b_i)$ indicates the expected waiting time to calculate CDP, $r_i$ is the RDR packet generating rate in the cluster region i. Notice that because the cluster head $s_i$ already has $b_i$ RDR packets in its buffer, it needs to wait to generate the remaining $(\delta - b_i)$ RDR for compression, where $s_i$ spends $r_i$ time to generate each RDR packet. After preparing to set the compression waiting time for each cluster head, we have to use data compression algorithm to process the RDR packets in one CDP packet in the following section.

## 6.3 Optimal Dynamic Huffman (ODH) Algorithm for WSNs

The spatio-temporal correlation among sensor observations are a significant and unique characteristic of the WSN and for RDR packets, which can be exploited to increase drastically the overall network performance. Recently, it addressed distributed source compression and other approaches in the previous literature review (Chapter 3). It utilizes the spatial correlation in a sensor network for compression. In the following section, we propose optimal dynamic Huffman compression (ODH), which explores the optimal Huffman coding in the cluster head of NMCA.

### 6.3.1 Huffman Coding for WSN

The key of Huffman coding is the dictionary and Huffman tree. Its codes express more frequent data values with shorter bit sequences and less frequent values with longer value. Huffman-style coding converts each possible value into a variable length string based on the frequency of the data. Higher frequency values are assigned shorted strings. So the more concentrated the data is over a small set of values, the more the data can be compressed. UEN may be any values compared with the regular RDR packets. Due to the spatiotemporal correlation of the sensory data, the continuous time stamp UEN packet has a limited difference between the current and the next time stamp. We can design the Huffman table for the changes of sensory value, which is a lightweight and real-time data compression approach.

Each sensor node measurement $m_i$ is converted to binary representation $a_t^i$using A bits. The compression algorithm computes the difference between the continuous time stamps in the same sensor node, which is spatial correlation sensory data:

$$d_t^i = a_t^i - a_{t-1}^i \tag{6-4}$$

The difference between the one-hop neighbours in the same stamp, which is temporal-correlation sensory data:

$$d_t^i = a_t^i - a_t^{i-1} \tag{6-5}$$

If the query requires the sensory data in the same sensor node, we use equation 6-4 for compressing data by Huffman. On the other hand, equation 6-5 means the whole network sensory data at the same time.

For Huffman coding, we have to propose a dictionary for different sensory datasets. The fixed dictionary requires prior knowledge of the probabilities of the source sequence. Static Coding requires prior knowledge of the probabilities of the source sequence. From (Khalid Sayood, 2004), if this knowledge is not available, Huffman coding becomes a two-pass procedure: the statistics are collected in the first pass and the source is encoded in the second pass. In order to convert this algorithm into a one-pass procedure, (Faller, 1973) and (Gallagher, 1978) have independently developed Adaptive algorithms to construct the Huffman code based on the statistics of the symbols already encountered. These were later improved by (Knuth, 1985) and

(Vitter, 1987). Using adaptive Huffman algorithm, we derived probabilities, which dynamically changed with the incoming data. Therefore, the adaptive Huffman provides effective compression. However, unlike static Huffman algorithm the statistics of sensor data need not be known for encoding the data. The dictionary is dynamic, which is not suitable for our compression level in the cluster head. This is because the base station does not know the latest dictionary in the cluster head, which makes the encoding difficult in the base station.

### 6.3.2 Optimal Dynamic Huffman (ODH) algorithm

In static and dynamic algorithms, the ultimate objective of compression was achieved with fixed and dynamic probabilities, respectively. The main disadvantage of Static Huffman algorithm is that requires the prior knowledge of incoming source sequence. However, for most applications, the sensor nodes are changeable and the data is hard to expectable, which could make the fixed dictionary not effective enough in the data compression. In Adaptive Huffman algorithm (Tharni and Ranjan, 2009), the probabilities are assigned dynamically. Due to the increased number of data available in the source sequence, the number of levels and hence the number of bits transmitted increases and is found to be effective only for very frequent data and data occurring at initial level of dictionary table. The dynamic dictionary would be changed dynamically based on the arrival of incoming data. It makes data compression most effective. However, because of the compression level is in the cluster head, which is the intermediate level in the in-network data processing architecture. In the gateway node, we have to decode the compressed packet for data mining level. However, the dynamic dictionary table is unknown for the gateway node or the base station. It has to send the dynamic dictionary table for decoding which would waste more energy in the transmission. Hence, we design an Optimal Dynamic Huffman (ODH) to solve the drawbacks of both static and dynamic Huffman algorithms.

The ODH is a method to improve the dynamic Huffman algorithm to solve the decoding in the gateway node without transmitting the dynamic dictionary table. We build two same initial dictionary table in both cluster head and the gateway node: one for encoding and another one is the decoding. Both of the dictionary tables are dynamic which is updated by the incoming data. But in the cluster head, the dynamic dictionary table is different to the Adaptive Huffman algorithm.

Figure 6-3 shows the data compression processing between the cluster head and the gateway node. The ODH algorithm is to compress sensor data in the cluster head and transmit the compressed data to the gateway node. In the gate node, the compressed data is decoded and calculates the original sensor data by the previous data and difference. The encoder performs compression lossless by encoding differences by statistical characteristics. Both encoder and decoder has to use the Huffman dictionary table. We put the initial table D in compression components in cluster head and the gateway node.

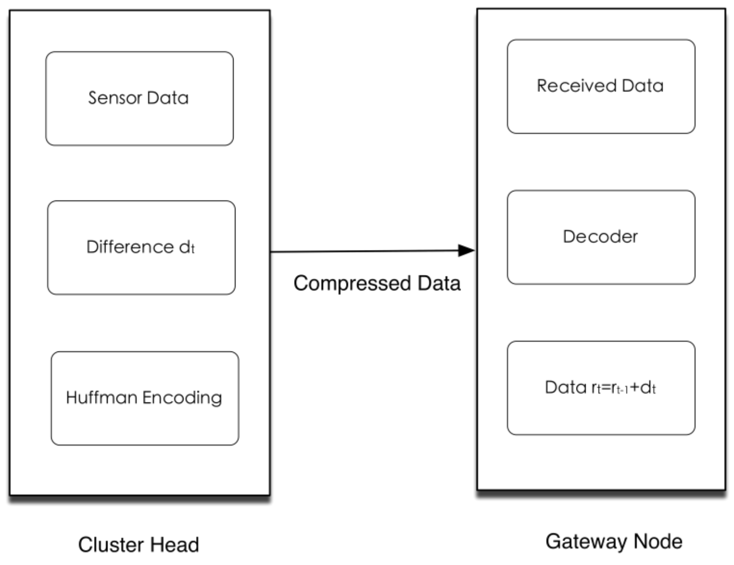


*Figure 6-3 the Process of Data compression*

Adaptive Huffman is based on the dynamic table for compressing data effectively. Figure 6-4 shows that the incoming data is to update the dynamic table first, and then compressed by the new table to get the effective compressed data. Even though adaptive Huffman coding is suitable for saving the memory storage in the sensor node, it is difficult to decode in any other device with the dynamic Huffman table.

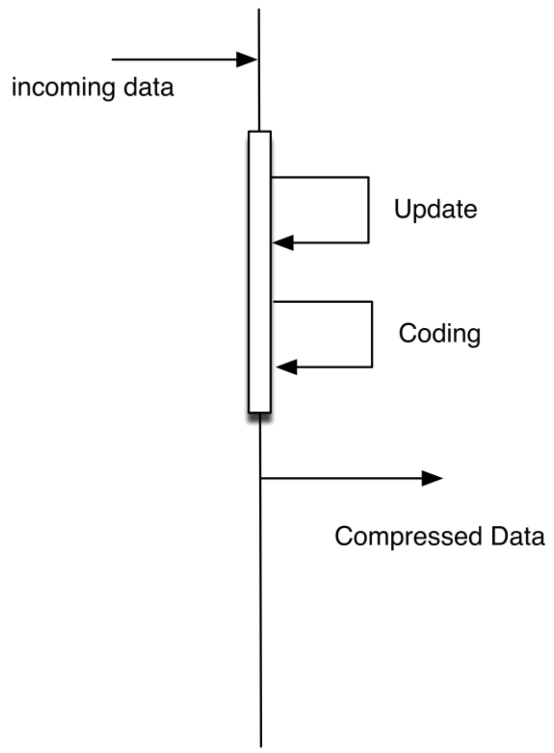


*Figure 6-4 The Flow chart of Adaptive Huffman Coding*

Hence, we improve the Adaptive Huffman to design ODH in Figure 6-5. Consider about the synchronization of dynamic Huffman Table in cluster head and gateway node, it compresses the current data by the previous Huffman table and then updates the Huffman Table for the next coming data. In the gateway node, the decoder is decoding the compressed data by the current Huffman Table. Then the decoded data is used to update the Huffman table, which is prepared for the next compressed data. The Huffman Tables in both cluster head and gateway node are dynamic and synchronised. It can compress and decode the data from any unknown network without transmitting the Huffman Table. The ODH has the advantages both Static and Adaptive Huffman algorithm.

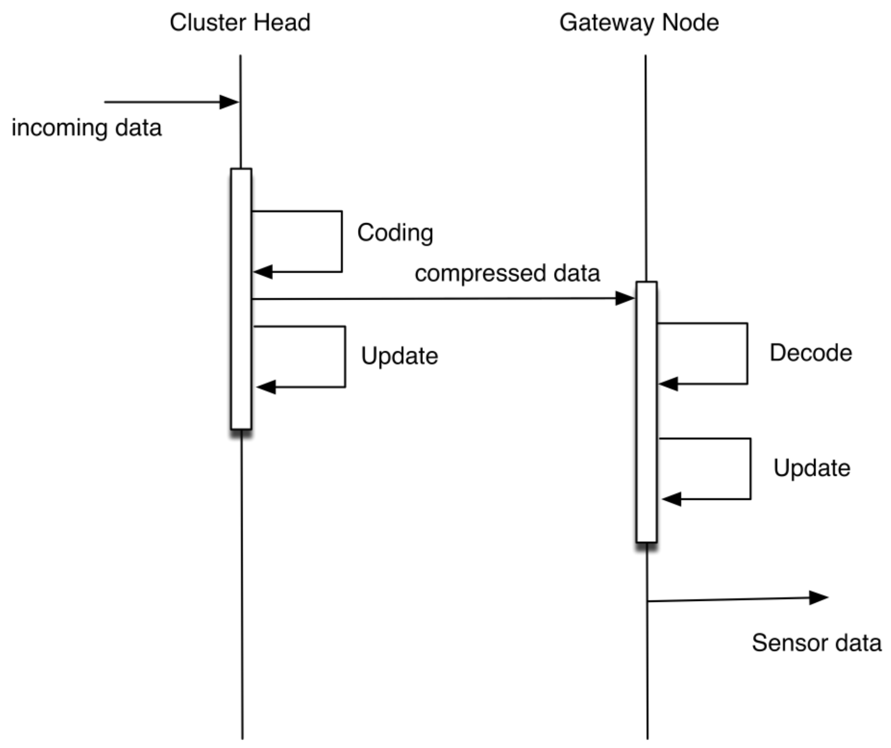


*Figure 6-5The Flow Chart of Optimal Dynamic Huffman Coding and Decoding*

In our data compression level, the data is a temporal correlation dataset. Hence, Equation 6-5 can give the value of $d_t=r_t-r_{t-1}$ which means the difference value for the same sensor node between the current and the previous time stamp. Figure 6-3 shows the data compression of encoding and decoding between the cluster head and gateway node. There are two initial dictionaries D in both of Huffman encoding and decoder components.

The encode function encodes each $d_t$ as a bit sequence, $h_t$ composes two parts $a_t$ and $n_t$, where $n_t$ gives the number of bits required to represent $d_t$ and $a_t$ is the representation of $d_t$. Code $n_t$ is a variable length code generated by using Huffman coding.

$$n_t = \lceil log_2 |d_t| \rceil \qquad (6\text{-}6)$$

when $d_t$=0, $n_t$=0. The basic idea of Huffman coding is that symbols that occur frequently have a smaller representation that those that occur rarely. The maximum bits length of coding is R which is the length of uncompressed data. The part of $a_t$ is a variable length integer code generated as follows:

$$a_t = \begin{cases} d_i & d_t \geq 0 \\ (2^{n_t} - 1) + d_t & d_t < 0 \end{cases} \tag{6-7}$$

$n_t$ is the category (group number) of $d_t$. It is the lower order bits needed to encode the value of $d_t$. Note that if $d_t$=0, $a_t$ is not represented. The Huffman coding is to make sure each bits sequence is the unique one. Although we will connect each sequence together without symbol, we still can divide each data from a compressed dataset.

According to the Huffman coding, we build the dictionary table. Table 6-1 is the initial dictionary used to encode and decode the incoming data. $n_t$ is the number of bits used for packets transmission, and it also means the frequency of occurrence level from source. The higher the frequency means the higher the level in the dictionary table and the lesser the value of $n_t$. $a_t$ is the code, which represent the value of $d_t$ by Huffman Coding.

*Table 6-1 Huffman coding Table D*

| $n_t$ | $a_t$ | $d_t$ |
|---|---|---|
| 0 | 00 | 0 |
| 1 | 01 | -1,1 |
| 2 | 11 | -3,-2,2,3 |
| 3 | 101 | -7…,-4,4,…7 |
| 4 | 1001 | -15…,-8,8…15 |
| 5 | 10001 | -31…,-16,16…31 |
| 6 | 100001 | -63…,-32,32…63 |

| | | |
|---|---|---|
| 7 | 1000001 | -127…,-64,64…127 |
| 8 | 10000001 | -255…,-128,128…255 |
| 9 | 1000000000 | -511…,-256,256…511 |
| 10 | 10000000010 | -1023…,-512,512…1023 |

After the initial Huffman Table establish, q[j] is the sets where puts all of difference $d_t$. p[j] is the set of the number of $d_t$ frequency.

In the initialization, q[j] is:

$$q_{init}[j] = \left\{0,1,-1,2,-2,3,-3,\ldots\frac{(j-1)+1}{2},-\frac{j-1}{2},\frac{j+1}{2},-\frac{j}{2}\right\} \quad (6\text{-}8)$$

Then q[j] can get from j:

$$q[j] = \begin{cases} 0 & j = 0 \\ -\frac{j}{2} & j \ is \ even \\ \frac{j+1}{2} & j \ is \ odd \end{cases}$$

(6-9)

For example, the 6th data in the set of q[j]=q[5]=(5+1)/2=3, the 7th data is q[6]=-6/2=-3. P[j] is the set to count the frequency of $d_t$. When the same value of $d_t$ occur, p[j] plus 1. The initial p[j]=0.

Figure 6-6 is the flow chart of OGH coding. The first step is the initialization of ODH, which builds the initial dictionary table D to get the initial $n_t$ and $a_t$, and set $q_{init}[j]$ and p[j]=0. Second, is the data receiver, which get the previous data $r_{t-1}$ and then calculate the difference $d_t=r_t-r_{t-1}$. Third, according to j, it can get the n and a to represent $d_t$ for compression. $(n_t,a_t)$ is the compressed data to send to the gateway node. The last step is use the p[j] and q[j] to update the order of Huffman dictionary table by the value of p[j]. If p[j] < p[j+1], the value of q[j] and q[j+1] should be exchange.

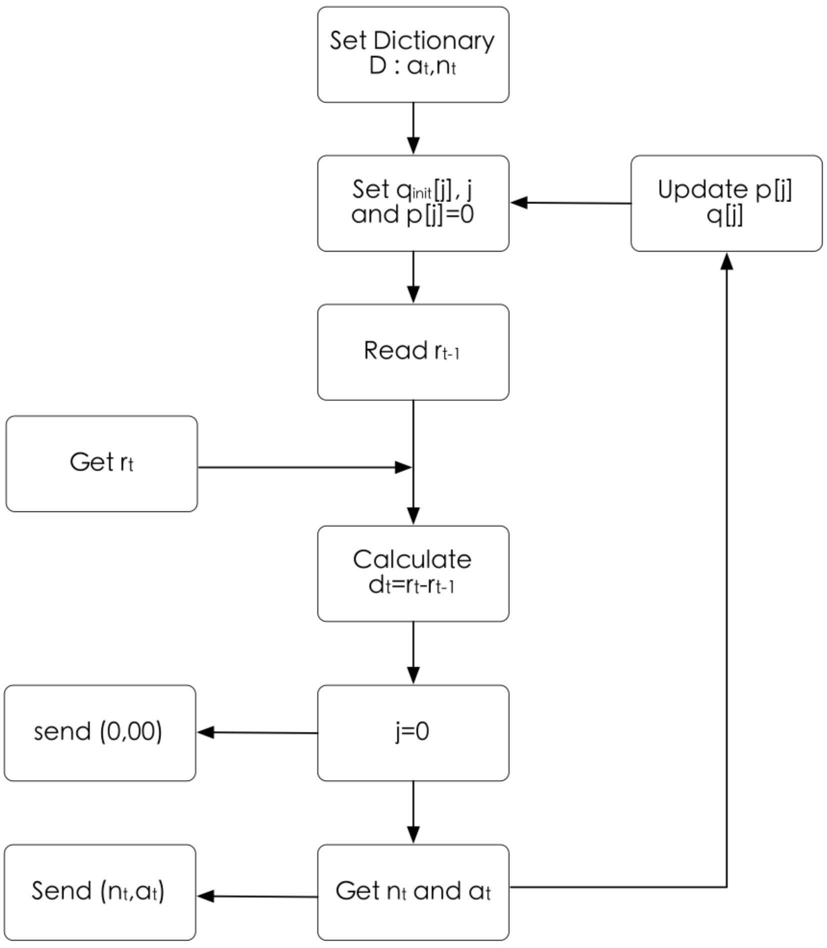


*Figure 6-6 the flow chart of Optimal Dynamic Huffman*

Here is an example of the Huffman updating. For example, $d_{23}=r_{23}-r_{22}=3_{(10)}$. We can find $q_t[j]=d=3$ in table 6-2, get j=2. To use j can get the n and a. Then send (n,a) to the gateway node for decoding by j. Because we get value in q[2], the new p[2]=p[2]+1=5+1=6. But p[1]<p[2], Hence, we exchange all values in j=2 and j=1, and then get the new table.

*Table 6-2 The Example of Update p[j] and q[j]*

| j | 0 | 1 | 2 | 3 | 4 | 5 | … |
|---|---|---|---|---|---|---|---|
| $q_{init}[j]$ | 0 | 1 | -1 | 2 | -2 | 3 | … |
| $P_{init}[j]$ | 0 | 0 | 0 | 0 | 0 | 0 | … |
| $q_t[j]$ | 2 | 1 | 3 | -1 | 0 | -2 | … |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $p_t[j]$ | 7 | 5 | 5 | 3 | 2 | 1 | … |
| $q_t[j]$ after update | 2 | 3 | 1 | -1 | 0 | -2 | … |
| $p_t[j]$ after update | 7 | 6 | 5 | 3 | 2 | 1 | … |

The decoder in the gateway node is similar to the encoding. It use (n,a) to get the original value $d_t$ and then updates the table for the next compressed data.

## 6.4 Performance Analysis and Evaluation

The objective of the proposed algorithms is to employ the in-network processing architecture to improve energy efficiency and the compression ratio in the implementation of WSN in the data compression level. In this section, we evaluate the effects of neighbour-aware compression scheme for UEN and ODH algorithm to compressed both UEN and RDR packets in the cluster head.

The metrics chosen to analyse the performance of our compression techniques are:

1. For energy consumption, we compare the NCS, ODH based on NMCA algorithms and no data compression by NMCA. Node Energy consumption is defined as the communication (transmitting and receiving) energy the network consumes. The idle energy is not counted. The energy model and parameter is presented in Chapter 5.
2. In assessing the goodness of proposed algorithms in terms of compression ratio, ODH is simulated and compared with the static Huffman algorithm and Adaptive Huffman Algorithm. Compression Ratio is the ratio of number of bits saved by compression to the uncompressing file size.

### 6.4.1 Compression Ratio

In this section, we simulate and compare the compression ratio for Static, Adaptive and Optical Dynamic Huffman algorithms.

The static Huffman algorithm involves grouping of the sample space of the source set of symbols and assignment of fixed probabilities table for each of the group. The transmitted code is determined and stored up as a lookup fixed table and is referenced for each data group. This algorithm hence requires prior knowledge of probabilities of the incoming source sequence.

The Adaptive Huffman involves dynamic assignment of probabilities of the incoming data through dynamic dictionary table constriction. Every time new data arrives, it is updated the dynamic dictionary table and encoded by Huffman coding. However, the dynamic dictionary table in the node is unknown for the base station and gateway node. Hence, the ODH algorithm overcomes the drawback of the above two algorithms. The efficiency of the algorithm is computed based on the compression ratio percentage, which is given by Equation:

Compression Ratio= (1-compressed no. of bits / original bits) *100 (6-10)

Compressed size is the number of bits obtained after compression and original size will be the total number of bits required without using compression algorithm.

*Table 6-3* shows the details of two dada sets for the simulation from Weather (Underground, 2012, http://www.wunderground.com).

| | Location | Range(°C) | Samples | Data |
|---|---|---|---|---|
| Set 1 | Hagerstown ,MD,USA | -16 to+37 | 3000 | 01/01/09 to 07/08/11 |
| Set 2 | Manaus, AM, Brazil | +21-+36 | 3000 | 07/01/11 to 11/30/11 |

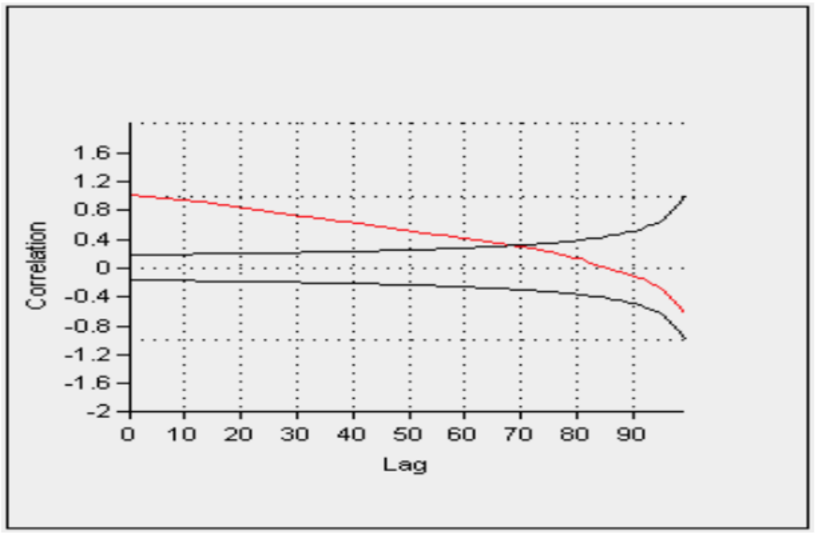


*Figure 6-7 the data correlation of data set 1 in Table 6-3*

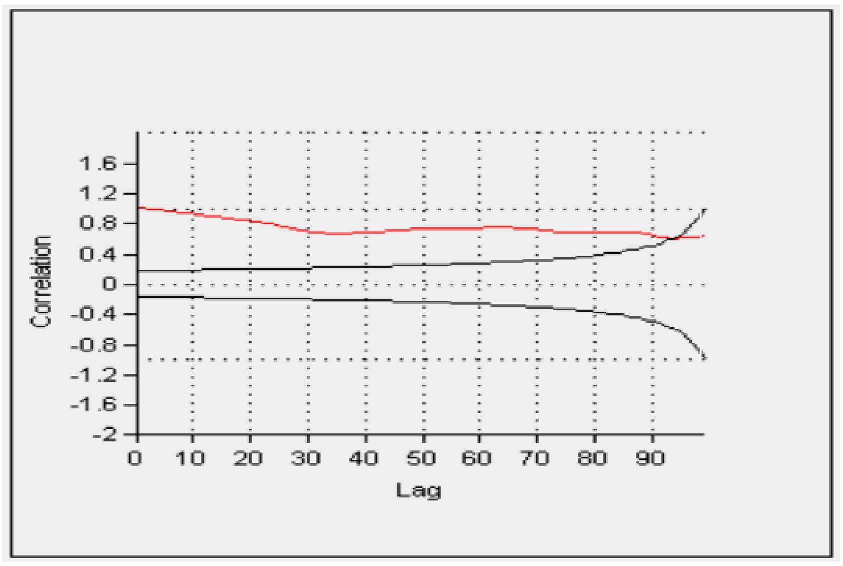


*Figure 6-8 the data correlation of data set 2 in Table 6-3*

We employ the two environment datasets in Table 6-3 to evaluate the performance of the propose algorithm. Figure 6-7 and 6-8 show the data correlation to the two sets of data. It can be seen set 2 has higher data correlation of data sets compared with set 1.

The performance of ODH, static Huffman and Adaptive Huffman algorithm were analysed in terms of compression in number of bits required for transmission and is shown in Figure 6-9 and the compression ratio by equation 6-10 is shown in Figure 6-10. For Adaptive Huffman and ODH, the number of bits' transmission is almost similar, and both of them are good at the higher correlated data.

ODH algorithm provides a compression ratio 58% of higher correlated data set 2 and nearly 23% for lower correlation data set 1.

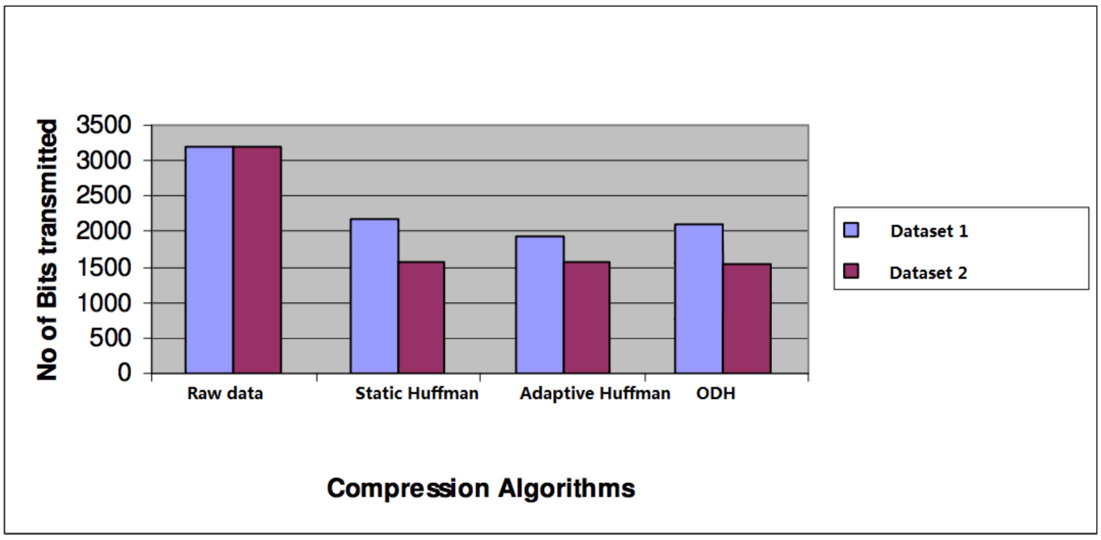


*Figure 6-9 No. of bits transmitted for compression algorithms*

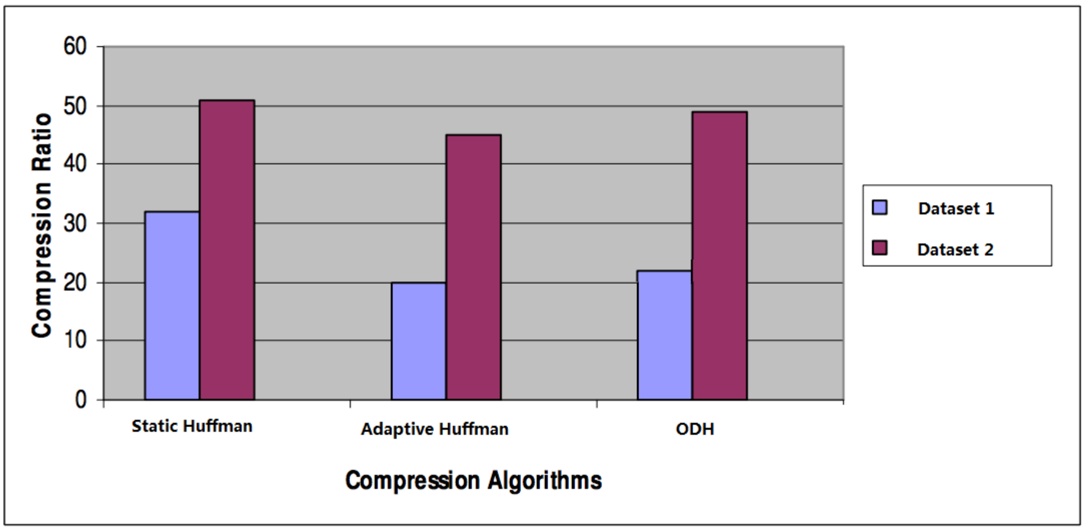


*Figure 6-10 Compression ratio for compression algorithms*

### 6.4.2 Node Energy Consumption

The energy model for the communication is similar to that mentioned in Chapter 5. We improve the energy model for data compression algorithms.

Let s be the size of the data, the energy of receiving the data is

$$\mathrm{E} = \mathrm{m} \times \mathrm{s} + c_s + t_i \times p_i$$

where m is the energy to receive a unit of data, $c_s$ is the network communication start-up cost, $p_i$ is the power during system idle where $t_i$ is the total idle time between packet arriving. For example, in communication compressed scheme for RDR, we present the waiting time for compressed.

For UEN:

Figure 6-11 is the case of 2 events in the network. Neighbour-aware compression scheme for UEN packets consume more energy than NMCA. Because NCS reduces the data redundant in the same event at the similar prior event. However, NMCA is to make sure the packets are accurate by neighbour-aware weight and sends all of them to the cluster head. If we set more events in the network, NCS should have more advantage in the energy consumption.

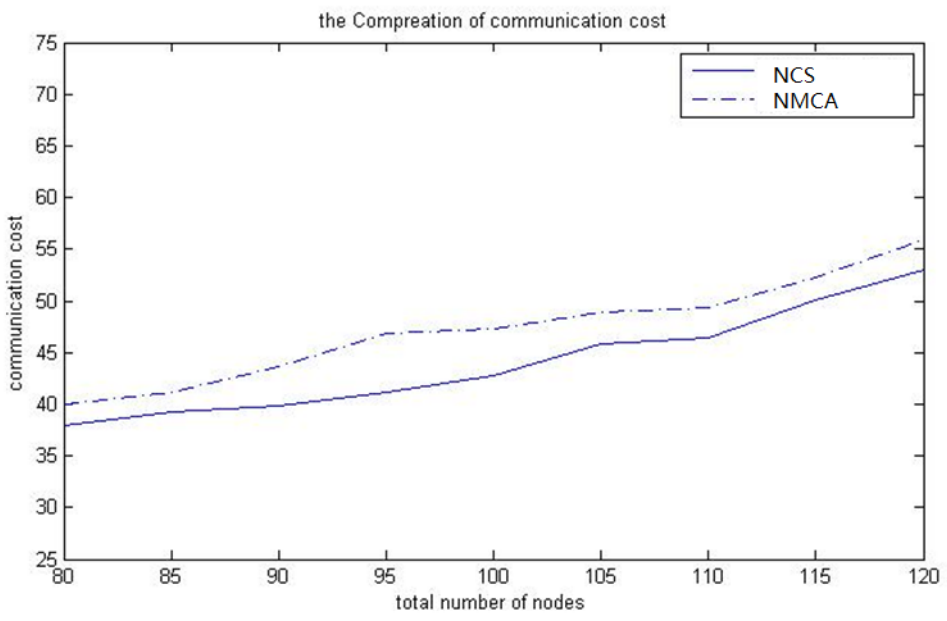


*Figure 6-11 the communication cost of Neighbour-aware compression scheme (NCS) and NMCA*

For RDR:

δ is the threshold number of RDR packets required to be compressed into one compression data packet (CDP). We set δ =10 by Equaition6-3, which is the threshold number of RDR packets required to be compressed. Figure 6-12, ODH can reduce more than 40% communication energy compared with NMCA without data compression. Obviously, without data compression, the NMCA always lets sensors consume the most amount of energy because they have to send out a great number of RDR packets continually. If we increase the value of δ, the sensor nodes can save more energy because they compress more RDR packets into one compression packet for transmission by ODH. But ODH requires more storage and compute resources compare with NMCA. NMCA is lightweight and accurate approach for fast response system.

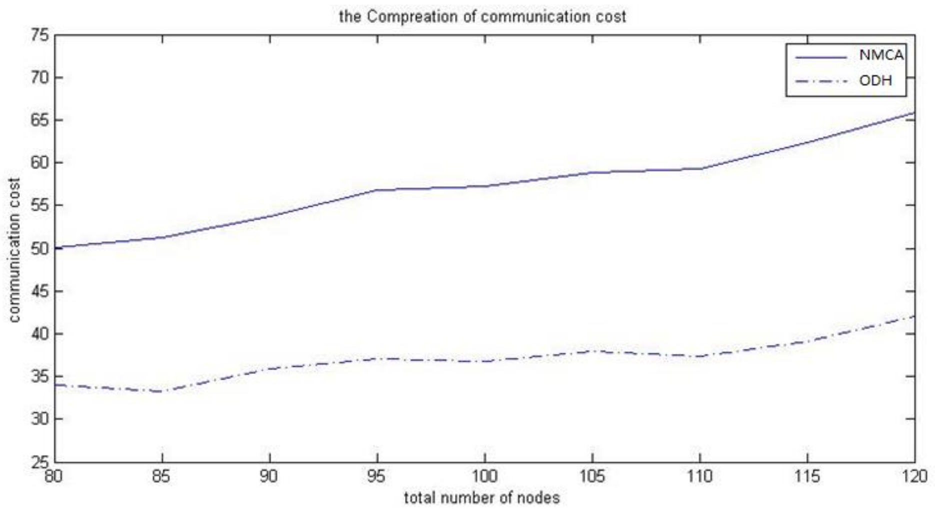


*Figure6-12 the communication cost of ODH and NMCA*

## 6.5 Summary

With the features of WSNs, we proposed neighbour-aware compression scheme for UEN packet and the Optimal Dynamic Huffman compression for both UEN and RDR packets in the cluster head. It combined both data and package funneling techniques and related data compression techniques in the architecture. The most significant advantages of designing WSN are how to improve the energy efficiency and make the compression algorithm lightweight for the intermediate sensor nodes within network.

The in-network data compression techniques can significantly decrease the whole network energy consumption. Neighbour-aware scheme is high performance for several events at the same time. It reduces the energy consumption and response quickly in the event detection. Optimal Dynamic Huffman algorithm can be easily employed in practice and any wireless sensor network for in-network data compression. In order to evaluate the algorithm, we computed the compression ratio obtained in several real datasets containing temperature collected at different locations and during distinct periods. The dataset also have the different correlated rate. Both the Adaptive Huffman algorithm and ODH are high performance for different correlated and repeated data. However, ODH is more suitable for in-network data processing. After, the data compression in the cluster head, the gateway node will decode the data and mining the acceptable knowledge for the users. In the following chapter, we will propose the data mining level in the in-network data processing architecture.

# Chapter 7. Data Mining Level

This chapter will employ semantic real-time data mining techniques for emergency response of fire detection to improve the reduction of false alarms, which could be beneficial to the effectiveness of any determined evacuation. In this chapter, we present data mining level in the architecture and investigate WSNs real-time data mining dealing with multi-sensor datasets for emergency response systems in WSNs. The motivation is to reduce various existing problems in traditional emergency response such as the "delay warning", "a false alarm", and "simple management" etc. In the data mining level, we not only integrate the event from a large amount of sensor data, but also design to satisfy constraints on metrics assessing energy efficiency, communication latency and accuracy of the conveyed information and there is a fundamental trade-off over these metrics. Furthermore, we will perform the simulation to illustrate how the real-time data mining technique fulfils the requirements of emergency response. This chapter describes the proposed data mining approach, namely real-time data mining. The work presented in this chapter was based on the publisher paper (W. He, et al., 2010). Section 7.1 presents the focus of data mining methodology for fire detection. Section 7.2 and 7.3 propose the model of semantic data mining. Section 7.3 exploits the fire detection simulation to evaluate the performance of the semantic data mining. Section 7.5 is the conclusion of the chapter.

## 7.1 Focus of the Data Mining Methodology for Fire Detection System

This section explains what data mining is and how real-time data mining fulfils the requirements of the fire detection system. (Fayyad, 1997) viewed data mining to be a particular step in large process of knowledge discovery in database (KDD). KDD is a more encompassing process that includes understanding application domain, data integration and selection, data mining, pattern evaluation and finally consolidation and use of extracted “knowledge” (Fayyad, 1996). The Gartner Group stated that “data mining is the process of discovering meaningful new correlations, patterns and trends by sifting through large amounts of data stored in repositories, using pattern recognition technology as well as statistical and mathematical techniques” (Kantardzic, 2003). Applying this definition, to the context of emergency response systems, suggests a need to extract useful and previously unknown or unexpected emergency knowledge from a large amount of real-time environmental data in order to detect a fire incident and support emergency services to enable them to provide a more effective response. Real-time data mining combines modern data mining techniques with time series analysis. The traditional data mining method focus on a limited perspective, which is inappropriate for analysing data from an assumed model. Therefore, it cannot process time series data and newly emerged types in real time. As shown in (Box and Jenkins, 1994) and a vast volume of time series literature, traditional time series analysis and modelling tend to be based on non-automatic and trial-and error approaches. Time series can deal with temporal sequences of datasets and forecasting data. However, development of time series modelling using a non-automatic approach for large amount of data would become impractical. Therefore, automatic model building is necessary.

Real-time data mining is focused on the development of a novel approach, which aims to provide fast and effective emergency response. In emergency response, firstly, we have to collect the UEN packages from the clusters to the gateway node as soon as possible. Secondly, UEN packages are processed with data mining model in the gateway node. Thirdly, the systems have to provide feedback to the emergency services or response system according to the results detection. Finally, the output should provide intelligent support for the users' decision-making and development of rescue plans. Real-time data mining is the methodology that links and realizes the functions mentioned above. Most importantly, the design of real-time data mining should get the adaptable knowledge in the limited time and node storage.

## 7.2 Semantic Data Mining Model Design

This section introduction the conceptual design of data mining level in the in-network data processing architecture in chapter 4. Figure 7-1 shows the propose data mining model in the gateway node consists primarily of three parts: Semantic Annotator, Data Mining Rule Engine and Knowledge Base. Moreover, there is the general architecture of the sensor node in the gateway node, which the communication unit is apart from the receiver and transmitter components. The processing unit is split into semantic annotator and data mining rule engines components, where it performs the annotation, mining and integration of the sensor data. The available storage in the processing unit is used for storing the facts and rules in the knowledge base of the sensor node. The following section presents the individual components of the model in detail.

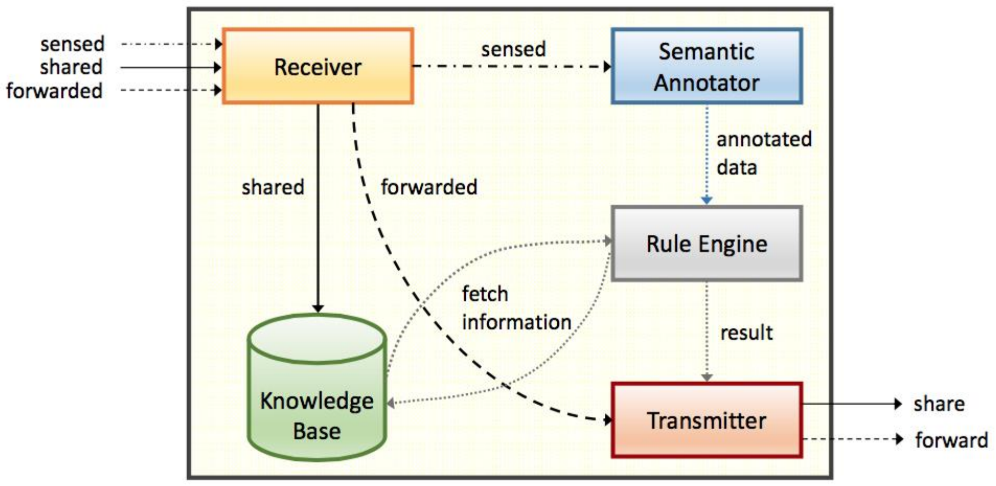


*Figure 7-1 Architecture of Semantic Data Mining Model*

### 7.2.1 Receiver and Transmitter

The receiver component in the gateway node purpose is to distinguish between shared, sensed and forwarded data. Shared data is used to build the knowledge base in the node at the beginning of application execution and it the triggers the rule engine for evaluation rules against the shared knowledge base. Sensed data is collected from the cluster heads in the heterogeneous WSNs, which includes UEN packets and RDR packets. Forwarded data should be directly sent out of the gateway node to other nodes or base station by the requirements.

In fire detection for emergency response system, the monitored phenomenon is transited as the event data by UEN packets and other data by RDR packets at regular intervals. The sensed event is then annotated and pre-proceeded in the semantic annotator and compared against the rules in the data mining rule engine.

The transmitter component is responsible for sending the formatted and acceptable information to different specific users. For example, in the emergency response system, there are firefighters, controller station and fire station. Shared and forwarded data will be sent to other nodes without processing by the transmitter in the gateway node.

### 7.2.2 Knowledge Base

The knowledge base contains a facts base and rules base. The facts base is related to problems we have considered in the real world for the application requirements. In the facts base, information and the users' requirements have to be formulated and identified. In general, these are questions, which are related to pattern and relationships between data. Examples of data mining questions in emergency response include "How can we characterize a fire event?" and "Which groups of multi-sensor data at the current time, appear to suggest the occurrence of a fire event?" Data mining comes into its own when there are many possible relations between large amounts of questions that have to be evaluated. In this base, an intelligent description of interesting knowledge must be presented clearly. Subsequently, any discovered relations will be used to make predictions about the situations occurring in the current event.

According to the processing of emergency events, the set of system problems we need to consider can be divided into three stages. Firstly, the problems are "Does the system run correctly?" and "how can we build and update the event model?" These two questions are used to make sure the emergency response system run smoothly. Secondly, we need to find the characters of the fire event, the location and danger level of the event, and the predicted spread. Finally, we have to evaluate the results and report this knowledge for decision making.

The knowledge base is also a collection of rules, which is defined by the applications. Rules are updated as the applications change its requirements or new requirements arrive. The rules include both of the sets of conditions and the actions to be performed when those conditions are met. In the fire detection system, we use the form of IF-ELSE expressions to make sure all of processing data under the threshold of event. The rules of fires are all its conditions are true. The IF statement does not specify the conditions to which the rule should apply in order to make the ELSE statement valid. The knowledge base builds the event condition rule (ECR) to the data mining rule engines to extract the application event.

In many applications, such as the environmental monitoring, the data typically changes slowly, and most of them are similar in regular intervals. The sensors will generate new readings several times per second. Although data clustering and data compression level can reduce most duplicate data in the same event, it still forwards the sensed event to the gateway node. In this case, data mining rule engines also will combine and discard the duplicate data for the same event. Hence, the knowledge base has to send the rules to distinguish the duplicate facts. Due to the changing in the application and limited memory, the knowledge base has to update and discarded the previous similar facts.

### 7.2.3 Semantic Annotation

A semantic annotator is used to pre-process the data before analysis. There is a need to add an explicit description to the measurement data and convert it to the formatted data. For example, a temperature provided by heterogeneous sensors can be measured in Celsius. The semantic annotator converts the sensor readings into semantically equivalent descriptions. There are two tasks included in the semantic annotation:

- Outlier detection: Outliers are data values that are not consistent with most other observations. The reasons for outliers' generation are caused by measurement errors, coding and recording errors and other environmental elements. In emergency response, outliers can seriously affect the results of fire detection and could be one of the main causes of false alarms. There are two strategies for dealing with outliers. The first is to detect and eventually remove outliers as a part of the data pre-processing stage. The second is to develop robust modelling methods that are insensitive to outliers.

- Data format and representation: Data format is based on triples, which consists of subject, types and object. For example, in “sensor_1 measures 25 degrees Celsius”, sensor_1 is the subjects, temperature sensor is the type, 25 degrees is the object data. The formatted data is put into different classes for further processing by rule engines.

### 7.2.4 Data Mining Rule Engine

Data mining rules engines are employed for filter and integrating sensor data in heterogeneous sensor networks. Real-time data mining is the process of extracting interested and previously unknown or unexpected knowledge from large amounts of real time data. Its correctness depends on not only logical correctness but also timing constraints. Various methods in the area of data mining have been developed in order to discover knowledge effectively and correctly.

The knowledge base contains and represents discovered knowledge during real-time data mining. One of any emergency response requirements is that the emergency response systems must be suitable for different environment and users’ requirements. Accordingly, we built various rules of emergency response systems. Rule selection is conducted to select the relative models from a knowledge base for upcoming data analysis. Many methods of rule selection employ a simple approach, which prefers the use of the simplest rule that fit the data. For emergency response, rule selection is mainly based on environment models such as global or local models. Next, the rule evaluation step is used to evaluate model performance. There are two purposes of rule evaluation: the prediction of how well the final rule will work in the future, and as an integral part of methods, which help to find the best performance for data mining. Finally, the last step in this stage is to update the existing rule (rule updating) which is to update rules in the knowledge base and may identify dynamic rules for real-time emergency response systems.

## 7.3 Data Mining for Emergency Response Fire Detection

The section explains how the data mining fulfils the requirements to detect fire events, improve the data accuracy and reduce false alarms. We use a simplified example of a real-time fire detection system to illustrate how the real-time data mining process works in practice. The aim of the fire detection system is to reduce false alarms, which is one of the most important and challenging application areas in emergency response systems. The aims for this example are to improve the correctness of fire alarm detection and compare the results with the traditional fire detection system. Because the features of data are similar in most fire states, we design a fixed feature data model to detect fire.

The design of real-time data mining consists of five components: receiver, transmitter, semantic annotator, knowledge base and data mining rule engine, which we mentioned in the previous section. Figure 7-2 shows those components according to in-network data processing in which they participate: annotation process, event detection process and communication process.

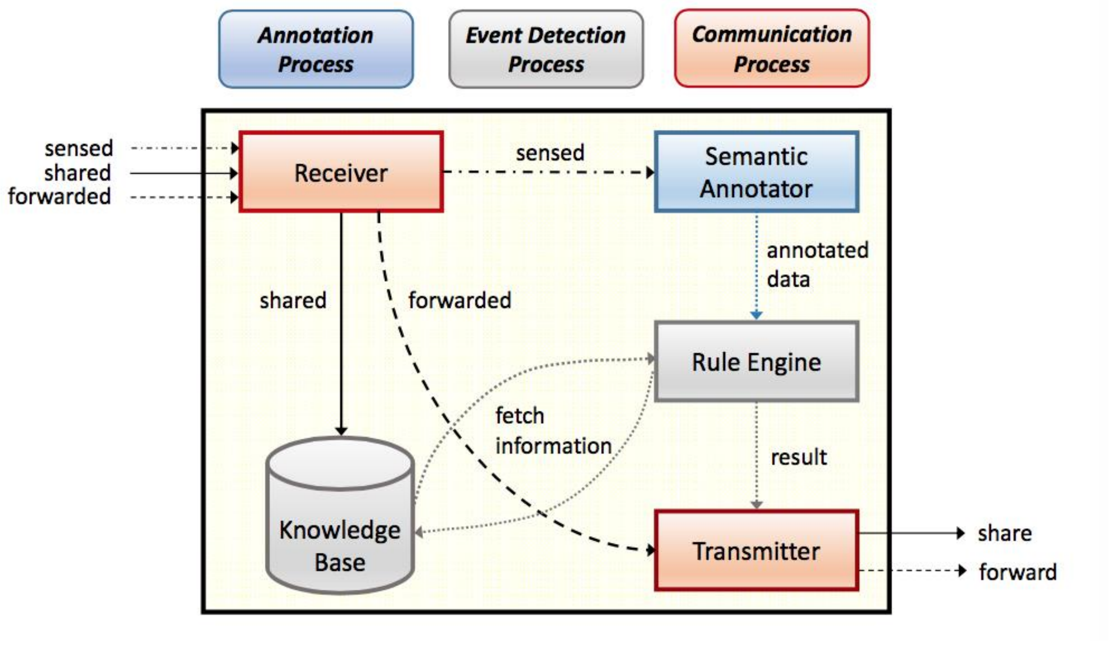


*Figure 7-2 visualization of the processes involved data mining architecture*

First, the annotation process is to pre-process data before analysis. Under the pre-processing, both of the other components can effectively and correct understand the real environment and operate the mining process. Second, the data mining rule engine operates the data mining in real-time context. The real-time data mining process engine imports the feature data rules from the knowledge base. Then the formatted data is sent to fire event detector, which gives the details of the fire (e.g. fire location, current time). Finally, the communication process is to get data from the cluster heads by receiver, and use a GUI to provide figures and charts to demonstrate the results and evaluate the correctness of the alarm in the transmitter. The result will be shown in the simulation.

### 7.3.1 Annotation Process

The annotation process is implemented in the semantic annotator component of the data mining model. The semantic annotation has two proposals in section 7.2: data formatted and outlier detection.

Figure 7-3 shows the corresponding graph of nodes (subjects) and edges (predicates), and the meaning of statement “RoomTemperature is observed by a TemperatureSensor”. All of the nodes are labelled. Edges are shown directed and labelled.

Temperature Sensor
Room Temperature

*Figure 7-3 The corresponding graph of the statement*

Semantic annotation provides types for subjects and objects for the statements. It begins by formatting data from sensor. It is formatted by the fixed data structure and put in to the classes, which are linked with the use of an edge. The edges represent the features of ObjectProperty and DataProperty for defining classes and their properties.

Table 7.1 and 7.2 shows the fields of the nodes and edges. Here is an example of a description for temperature sensor in our fire detection system. Table 7.3 shows the links between the subjects to the predicates through property instances defined in the edges.

*Table 7-1 the field of nodes*

| **Variable** | **Description** | **Definition** |
|---|---|---|
| Name | Node's name | Class name |
| Edges | Null/link to edges | Links to other class |

*Table 7-2 the field of edges*

| **Variable** | **Description** | **Definition** |
|---|---|---|
| Name | Edge's name | Property name |
| Parent | Link to parent class | Domain of the property |
| Child | Link to child class | Range of the property |
| Property type | Property description | Data property |

*Table 7-3 The link between subjects to the predicates through property in the edges*

| **Variable** | **Description** |
|---|---|
| Id | Uniqule id |
| Subjects | Link to parent of edges |
| predicate | Link to edges name |
| object | Link to child of the edges |

*Table 7-4 The data format for temperature sensor in sensor node*

| **No.** | **Subject** | **Predicate** | **Object** |
|---|---|---|---|
| 1 | Temp_Sensor(1) | .type | temp |
| 2 | Temp_Sensor(1) | .id | 1 |
| 3 | Temp_Sensor(1) | .unit | C |
| 4 | Temp_Sensor(1) | .value | 20 |
| … | … | … | … |
| n | Temp_Sensor(1) | .node | (0,1) |

Once we have collected the data and formatted them, another purpose of the data semantic annotator is outlier detection. In the fire detection system, we apply a Kalman filter for outlier detection. Before we discuss the system, we have to introduce what a Kalman filter is and how it works. A Kalman filter is a mathematical method named after Rudolf E. Kalman. The purposed of a Kalman filter is to use measurements that are observed over time that contain noise and other inaccuracies, and produce values that tend to be close to the true values and their associated calculated values. It can also be used in linear and dynamic systems.

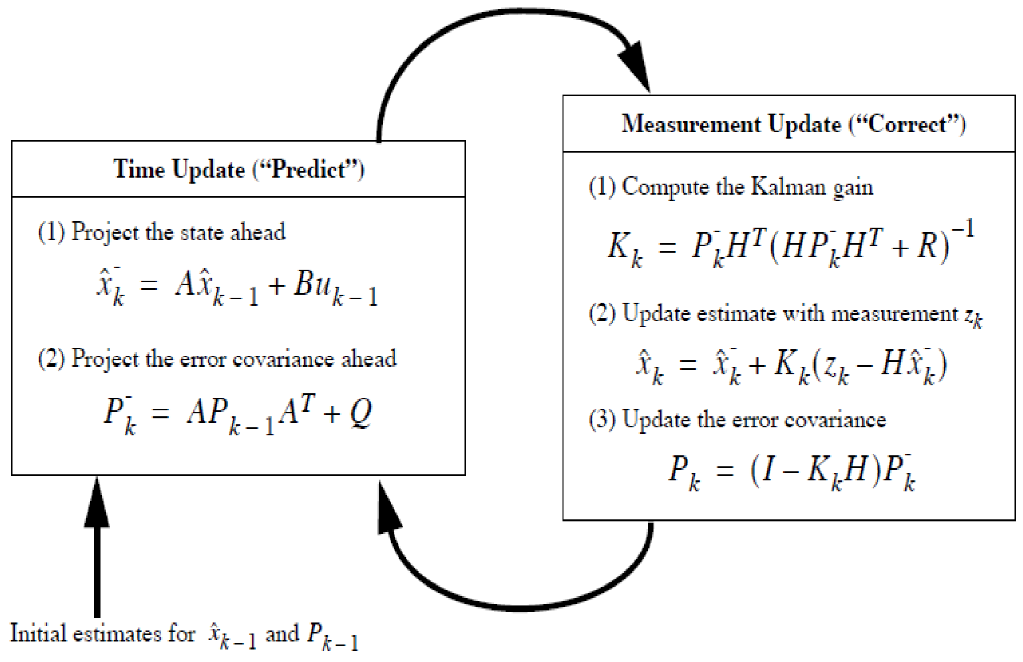

Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
(2) Project the error covariance ahead
$P_k^- = AP_{k-1}A^T + Q$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement $z_k$
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and $P_{k-1}$

*Figure 7-4 Kalman filter algorithm*

Figure 7-4 is combing the Kalman filter algorithm and stages. It shows that the initial estimate is processed and reprocessed by "predict" and "correct" function. we will introduce the details of using a Kalman filter in fire detection system in the following section. In the system, the Kalman filter method is not only used to remove the outliers, but also to predict data in the next time stamp that could be used for knowledge updating and environment data adjusting. Figure 7-4 shows that there are five steps in using the Kalman filter method in the system: project the estimation ahead, project error covariance ahead, calculate the Kalman gain, update the estimate with measurement which helps to detect whether it is a real event or an outlier, and update the error covariance. The processing will be repeated during the operation. The purpose of a Kalman filter is to update the estimate of the state vector of a system based upon the information in a new observation. In this project, we assumed fire occurs at fixed discrete time intervals. This system is a linear system, meaning that the time evolution of the state vector can be calculated by means of a state transition matrix.

In this system, we assumed the state estimate "optiA" and its covariance "optiB" which are updated by this Kalman filter function, when the other fields describe the mechanics of the system without changing. A calling routine may change these other fields as needed if state dynamics are time-dependent; otherwise, they should be left alone after initial values are set.

The exceptions are the observation vector “pb” and the input control. However, in this system, the input control is not calculated. Because there is no input function in this system, input control should be set to zero value by the calling routine.

First, we have assumed A =1 and H=1 and try to find out the optimal Q and R for Kalman filter operation. According to simulation, we can find out when Q=1 and R=1, the Kalman filter signal is not smooth enough so that it is too closed to raw signal. When Q=2 and R=2, this Kalman filter signal is distinct. Hence, the values of Q and R are decided to be between 1 and 2. Through assuming many groups of numbers, I found out Q=1.9 and R=1 result is optimal. In conclusion, I selected Q=1.9 and R=1 as system default value.

Initialize set Q=1.9, R=1, A=1, H=1

Define initial state estimate: optiA(1)=raw data(1) the initial state estimate equals to the first raw data from each sensor

```
    For t=1….300,

    Sample optiB(:,:,ts)=A*optiA_t(:,:,ts-1)+w(:,:,ts-1); see Eq.(7.1)

    pb(:,:,ts)=A.^2*pa(:,:,ts-1)+Q; see Eq.(7.2)

    Kg(:,:,ts)=H*pb(:,:,ts)./(H.^2*pb(:,:,ts)+R); see Eq.(7.3)

    optiA_t(:,:,ts)=A*optiB(:,:,ts-1)+Kg(:,:,ts).*(temp(:,:,ts)-A*H*optiB(:,:
,ts-1)); see Eq.(7.4)

    pa(:,:,ts)=pb(:,:,ts)-H*Kg(:,:,ts).*pb(:,:,ts); see Eq.(7.5)
```

### 7.3.2 Event Detection process

The event detection process is performed by the data mining rule engine component based on the facts and rules gathered from the knowledge base component. Before event detection, the facts have been obtained through the annotation process with the user-specified rules are part of the knowledge base. In the fire detection system, we employ the historical data from other research to build the data mining rules and then to detect the event by the data mining rule engines.

The goals of event detection are to identify whether an event of interest has occurred and to characterize the event (e.g., the time, the affected area, the type and the severity of the event). The most important one is to detect the event accurately and as early as possible. Hence, the main requirements of event detection is timeliness, a high true detection rate, and a low false alarm rate. In the case of fire detection from multi-sensory data, the requirements on detection timeliness and accuracy still apply, in addition, special challenges (e.g., temporal spatial information incorporation, information integration from multiple sensor streams, computational complexity) presented by sensor data should be taken into consideration.

Threshold-base event detection is based on the underlying intuition that an event occurring will result in changes in the sensor reading. For example, a fire will result in an increased temperature reading. Therefore, normal behaviour can be defined as a threshold (e.g., maximum values, rates of increase and combination thereof from multiple sensors) based on statistics of historical data (or domain knowledge), and alarms can be raised if the predefined threshold is exceeded. The advantage of threshold-based event detection is its simplicity of implementation and low computation complexity. However, tuning the threshold is highly dependable on the specific detection problem and the environment that sensors are monitoring, and some events cannot be fully captured by discrete threshold values. The accuracy of detection is limited.

Event detection process includes neighbour-aware, temporal-aware and spatial-aware algorithms to improve the accuracy of detection. In the NMCA and data compression level, we employed neighbour-aware to reduce the false alarm of event detection. In the fire detection, there are multi-sensory data in the sensor node: temperature, smoke, CO and flame sensor. Hence, we use the temporal spatial-aware to set the rules of data mining. When a fire occurs, there is not only one type of sensor reading increasing. It may affect every type of sensor in the same sensor node at the same time.

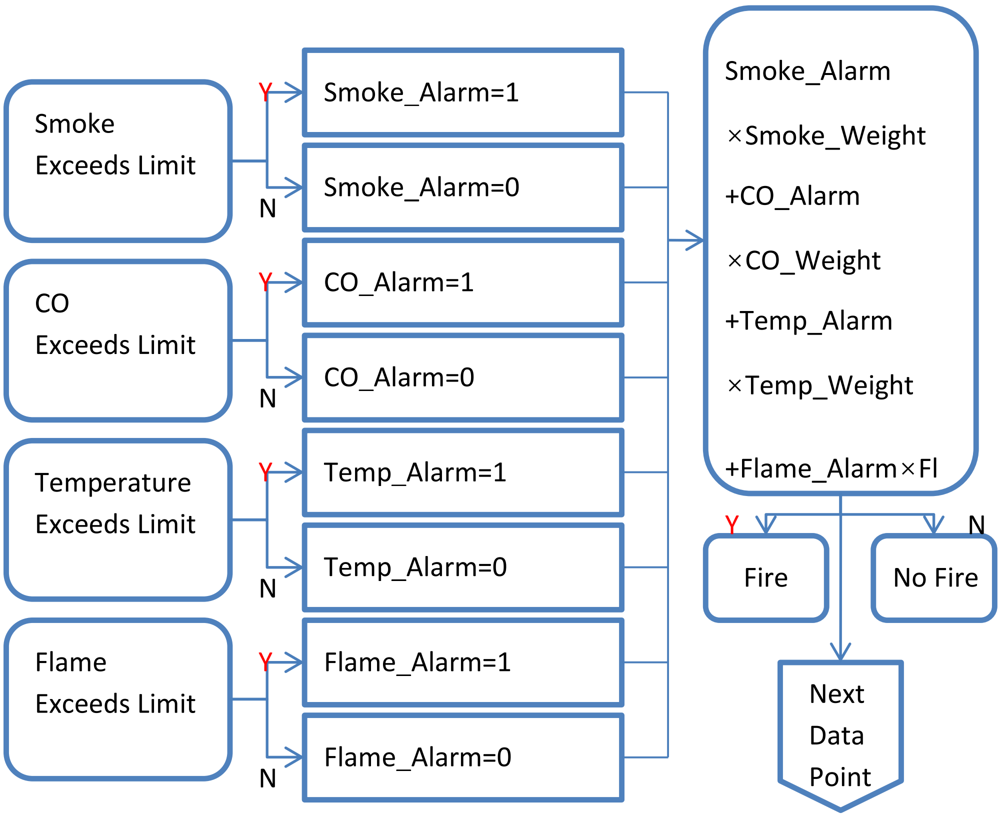


Figure 7-5 Threshold-base rules

In this simplified case, the environment would be simply compared with the real world. In order to analyse false alarm problem only, we integrated the model updating and selection steps and then built threshold-based model for fire detection to integrate different types of sensory data in the same node. The threshold-based model is to apply the threshold of normal patterns of behaviour to detect emergency event. The process of threshold-based model can be shown in Figure 7-5. According to real experience, we fix the threshold values for each sensor (maximum value, the increase rate) and the weight of each sensor that means the essentiality for abnormal event detection. Then repeating the process by attaching a time label. We apply the historical shared data into the threshold-based model to get the weight rate for different types of sensor. In our simulation, we set the initial value of weight rate:

Temperature weight rate: 30%

Smoke weight rate: 10%

CO weight rate: 35%

Flame UV weight rate: 25%

The threshold-base rule model can be applied in an FOR and IF-ELSE as:

```
    1. Initialize:
    Smoke_alarm =0; CO_alarm = 0; Temp_alarm =0; Falme_alarm = 0;
    Smoke_Weight=0.3; CO_Weight = 0.35; Temp_Weight = 0.3; Falme_Weight = 0.25.
    Total_Alarm = 0.
    2. For (1~t, 1~n, 1~n) : check each data
    If Smoke≥1.5    Smoke_alarm = 1;
            Else Smoke_alarm = 0;
    If Temp≥1.5    Temp_alarm = 1;
            Else Temp_alarm = 0;
    If CO≥1.5    CO_alarm = 1;
            Else CO_alarm = 0;
    If Falme≥1.5    Falem_alarm = 1;
            Else Flame_alarm = 0;
    3. Total_alarm = Smoke_alarm × Smoke_weight + CO_alarm × CO_weight + Temp_alarm × Temp_weight + Flame_alarm × Flame_weight
    4. If Total_alarm≥0.5 Fire
Else no Fire
```

## 7.4 Performance Evaluation

Data simulation is the first step for the fire detection. The sensor data include four parts: the normal data, fire data, system noise and outliers. We assume the range of normal environmental multi-sensors data. The fire event generates random fire data for each sensor. According to the different types of sensors, we assume different noise and outlier rates for each sensor. The raw data are from four sensors: temperature sensor, CO sensor, flame UV sensor and smoke sensor.

Network size (N), which is the length of the square N × N grid, was varied from 5 to 20, to evaluate the computation efficiency of the selected event detection methods.

Outlier rate (o), simulates temporary errors caused by signal conflictions in the network or sensor malfunctioning over the quality of the sensor data. The outlier rate was varied from 1% to 20%, to evaluate how robust the event detection methods can be when processing data with low to high uncertainty.

Fire spread model (F), simulates the characteristics of an occurrence of fire. Two fire spread models were used, to evaluate the detection time required under different fire scenarios. Fire spread model A represents fire spreading that reflects in all types of sensors at the same time, whereas fire spread model B represents fire spreading that reflects in different types of sensors at different time.

We assumed the size of this WSN is 5×5 and duration is 300. Hence, the normal data have to be changed to 5×5×300 form for further calculation.

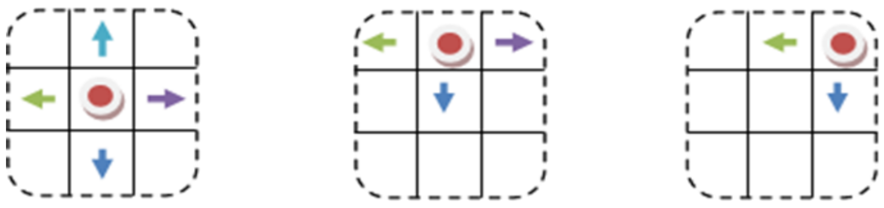


*Figure 7-6 Fire spread model*

The noise is white Gaussian noise. Gaussian noise is a probability distribution describing random fluctuation in a continuous physical process and named after Karl Friedrich Gauss, an 18th century German physicist. It can be used to simulate the effect of thermal activation, and be represented by a stochastic process. According to the character of this system, the noise is assumed as 1.5 times normally distributed random numbers.

Fire data were from the process of fire simulation. Before fire simulation, we should design the fire spread model first. There are three kinds of fire position presented by sensor nodes in WSN: centre, boundary and cornet. We divide nodes in those three ways. Then the fire is spread based on the fire spread model in Figure 7-6. In the fire simulation, we have to assume various data for each process. As we can see in Figure 7-7, to define a random time ‘rt’ and duration ‘rd’ for fire is used for making sure when and how long fire burst in the network. Random ‘x’ and ‘y’ is to define the range of fire spread. Random position (xrp,yrp) in the network is used for defining fire point location. When the initial values are prepared for this system, it can start the process of fire simulation.

The fire bursts when the time equals to random starting time ‘t’. The normal data in the random position (xrp, yrp) was increased until its value exceeds the limitation value of fire alarm. In next time slice, the neighbour data of this ‘random position’ would be changed similarly. This process would be continued until each sensor node data changed in the defined fire range, which was mentioned in the previous section. For example, as we can see in Figure 7-7, we assumed (5,3) sensor node as fire point. When t=rt, (5, 3) data would be changed. Then (5,2), (5,4) and (4,3) would detect fire and appear in the sensor data. Repeat this process until the fire was extinguished.

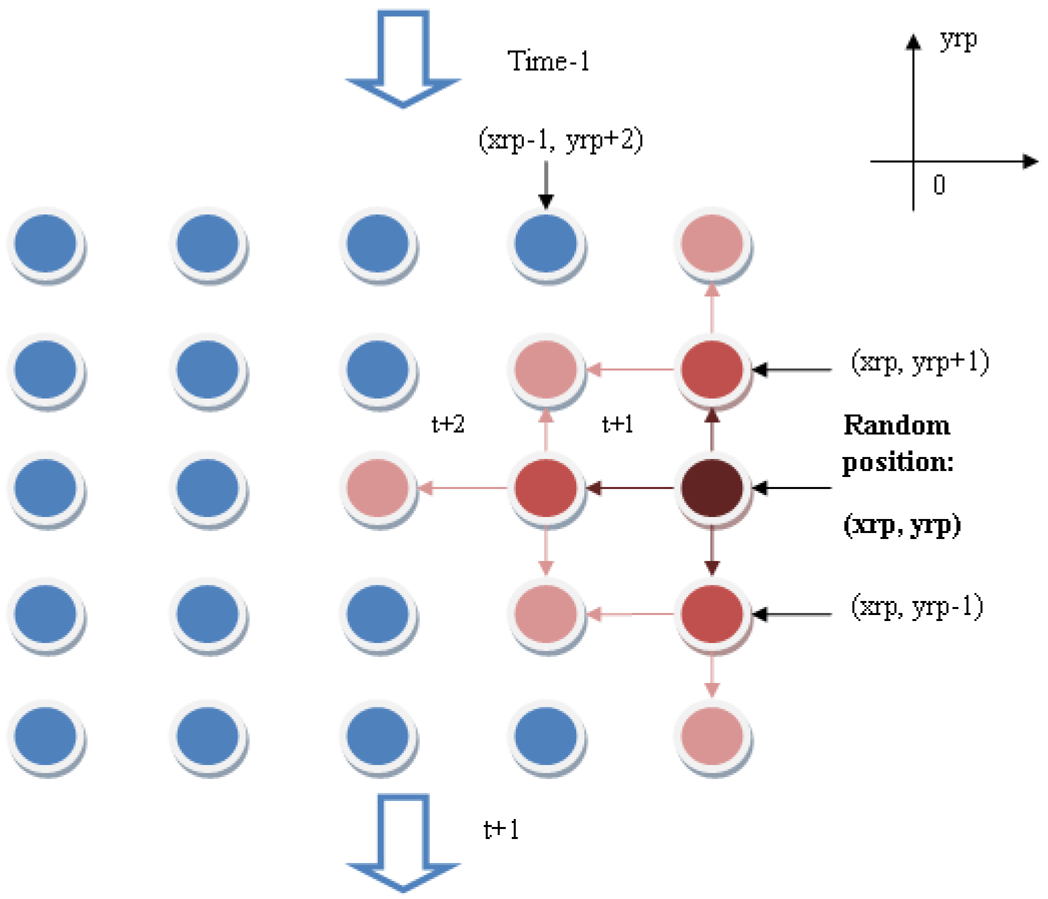
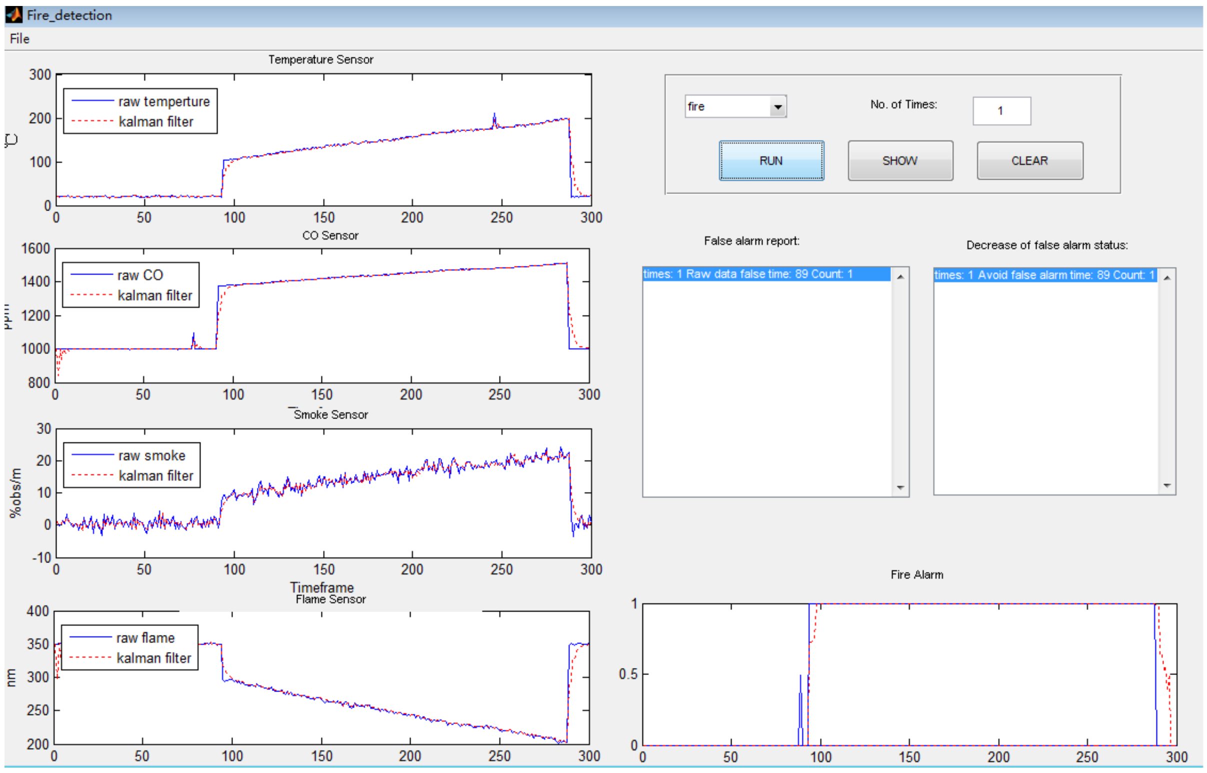


*Figure 7-7 Fire simulation architecture*

In Figure 7-8, we assumed the system operated once in fire case. In the output data parts, it shows four sets of data of length time = 300 obtained by the output of four sensors. It can be seen that the fire occurred in the time = 92 and ended in time = 285. The blue line in the figure represents raw data and the red line represents the Kalman filter data. The Kalman filter reduces the noise and outlier in the original signal. Moreover, we can find out the results are identical with the fire point in the fire alarm chart. According to the false alarm report, the original data generated a false alarm in time = 89. However, the Kalman filter methods remove this false alarm, which can be seen in the “Decrease of false alarm status”.



fire

No. of Times: 1

times: 1 Raw data false time: 89 Count: 1

times: 1 Avoid false alarm time: 89 Count: 1

*Figure 7-8 Test system: operation times =1 (fire case)*

Figure 7-9 shows the last 10 operations. It shows four set of data of time=300 obtained from temperature, CO, smoke and flame sensors. The output of the fire detection as defined above is also graphically represented (“fire alarm”). The fire point is time =160. The detection results are correct. According to the fire detection system, we can calculate the false alarm rate by the equation 7-1 below.

$$False\ \ alarm\ \ rate = \frac{Totle\ \ false\ \ alarm\ \ times}{time\ \ \times operation\ \ times} \times 100\%$$

(7-1)

False alarm rate by original data:$\frac{8}{300\ \times 10} \times 100\% = 0.26\%$

False alarm rate by Kalman filter: $\frac{2}{300\ \times 10} \times 100\% = 0.067\%$

Decreased false alarm rate:$\left(1 - \frac{0.067}{0.26}\right) \times 100\% = 74.2\%$

We employ the test system by fire case and non-fire. The Table 7-5 is shown the decreased false alarm rate for different operation times.

*Table 7-5 The result for fire case and non-fire case by different operation times*

| | Fire case | | | Non-fire case | | |
|---|---|---|---|---|---|---|
| Operation times | 10 | 20 | 100 | 10 | 20 | 100 |
| False alarm rate by original data | 0.26% | 0.5% | 0.34% | 0.26% | 0.23% | 0.28% |
| False alarm rate by Kalman filter | 0.0067% | 0.2% | 0.15% | 0.033% | 0.1% | 0.17% |
| Decreased false alarm rate | 74.2% | 60% | 55.9% | 87.3% | 56.5% | 39.3% |

## 7.5 Summary

This chapter indicates the semantic data mining level of in-network data processing. Real-time data mining is a developing field of study, which has raised many challenges to be addressed (Dong, 2003; Gaber, Kargupt, 2002). Emergency response is a highly complex process that needs the support of WSNs. Real-time data mining technologies for WSNs could greatly improve the ability of emergency response system and reduce loss of life and false alarm, as well as reducing the risks of emergency. For fire case, data mining approaches can reduce 58% false alarm on the average. For non-fire case, it can reduce 50% false alarm.

The existing techniques of real time data mining cannot meet the requirements of the real world. Some of them try to develop complex computational algorithms, which introduce a significant margin of error in real experience. These purposed algorithms may not be discovered in the simulation study. For example, in our emergency response system, the environment is simpler than the real world. The fire spreads in the simulation system at a fixed speed. However, in the real world, it may be affected by a changing and complex environment. Hence, in future work, we have to improve the models and algorithms to fulfil the more complex requirements from the real world.

The chapter contributes further introduce the data mining level by introducing and implementing the conceptual design described in Chapter 4. This study focuses on the event detection of emergency response system. Thus in the simulation and rule engine of data mining, it was developed that related to fire detection. Real-time data mining is still in its infancy state. Further developments would be realized over the next few years in various applications, such as physical and astronomical applications, as well as business and financial ones etc. In the following chapter, we will embed the in-network data processing architecture in the case study-indoor location tracking system.

# Chapter 8. A Case Study: WSN Based Location Tracking System Embed In-Network Data Processing

The previous three chapters described the level design and evaluated simulations for in-network data processing architecture. The goal of the architecture, following the objective of the thesis, is to perform complex event detection and process data within network to migrate WSN constraints. To determine whether the architecture reaches this goal, it needs to apply it in the case study.

The third research question in Chapter 1 asks about “How much benefit could be gained by performing in-network data processing?”. In this chapter, a case study of a WSN based indoor tracking system will be introduced. With the presented in-network data processing algorithm that was introduced in the previous chapter, the target trail system obtains significant improvement on performance and energy saving. The case study proves the reliability and value of our research in real world applications.

This chapter is structured as follows. Section 8.1 introduces the indoor location and tracking system’s goals, principles and objectives and achievements. Section 8.2 describes the design of our system, which comes with hardware introductions, algorithm design and the code for specified sections. Section 8.3 presents the test result. Finally, this chapter is concluded in section 8.4.

## 8.1 Introduction

Indoor location and tracking is currently a hot topic amongst the consumer industry and other fields. It provides the user the current position inside the building where Global Positioning System (GPS) cannot reach. This chapter describes the details of such low cost WSN based localization and tracking. This system is designed for applications such as trailer tracking, roll cage tracking, and forklift tracking in a warehouse/retailer’s distribution centre, beds and equipment tracking in a hospital, or any other similar scenarios of indoor location and tracking purposes. It aims to develop a low cost, WSN based localization and tracking system with acceptable tracking accuracy. Further, the core of this objective was to build an efficient, low cost, and affordable location and tracking WSN. After that, the back-end gateway collects the distance measurements and calculate the target’s position. The energy consumption is always the biggest problem in a wireless system, to improve system life circle without affecting the performance significantly. The demo system is constructed based on a WSN designed for in-door location and tracking, and with our in-network data processing strategy enabled. Figure 8-1 shows the conceptual model of the demo system.

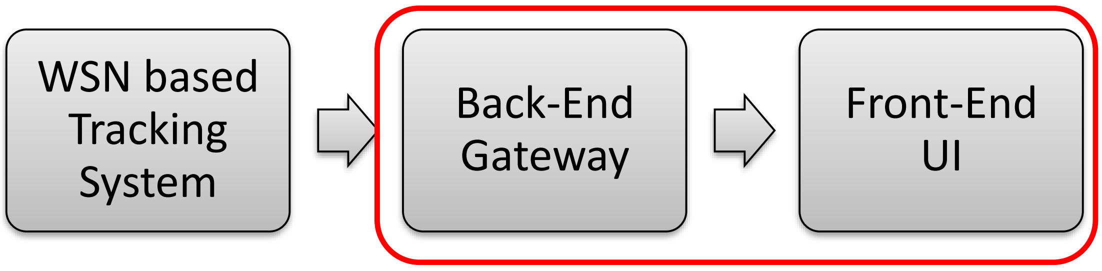


*Figure 8-1 the conceptual model of the demo system*

By communicating through the middle part of “Back-End Gateway”, the demo system achieves the distance measurement data from the WSN, calculates the target position and pushes to the user interface. In this demo system, we bind the Back-End Gateway and the Front-End UI as one component.

The energy saving objective of the demo system is developed through an optimized data archiving approach. It is balanced between the energy consumption and performance. We introduced an improved Huffman table encoding to reduce the traffic between the cluster head point, which is also the heaviest loaded device in the network. The enhanced cryptographic algorithm is created from learning a large amount of example data from the localisation system, and is specifically design for the demo system. With the benefits from in-network data processing mechanism, the wireless network system has an optimised data transmission strategy to minimize the energy consumption on 2.4GHz wireless communication. Even more, a wireless power control synchronizing architecture has also been developed in the lab trial phase. This feature can be added to the current system in the future.

## 8.2 An Object Tracking Prototype System

The demo system is depicted in Figure 8-2. The specific devices and layout pictures for the system are depicted later in this chapter, which consisted of Anchor nodes, Target device, Gateway and UI. In the demo system, we are using EVB1000 evaluation kit (Figure 8-3) as the demo node to act as Anchor, Target and Coordinator. It is a specific (Real Time Location System) RTLS designed hardware based on the IEEE802.15.4a UWB RF protocol.

The Tag is the target node, which the system does not know at the beginning but will find out its real world physical position in a localization system. In the system, the Target is the trigger of the system. Once the Target node joined the network, it starts to send ranging request to the entire Anchor nodes within the network. We introduced the In-network aggregation with data compression mechanism between the data transmission of the Target node and the coordinator to reduce the size of data packets to optimise the in-network data transmission performance for the energy saving proposes.

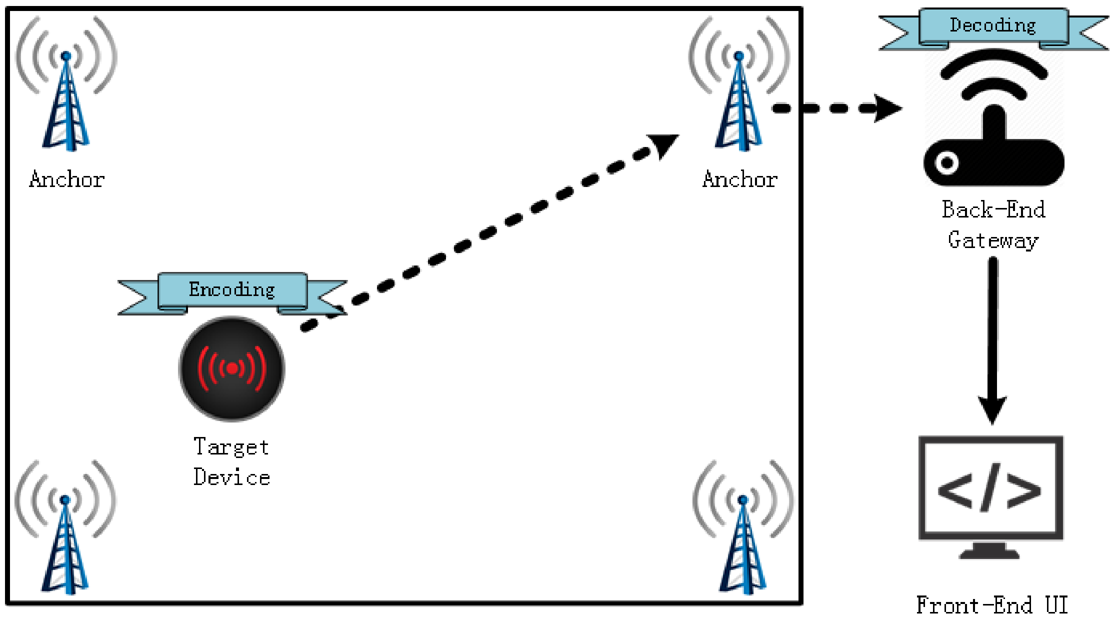


*Figure 8-2 Prototype system architecture*

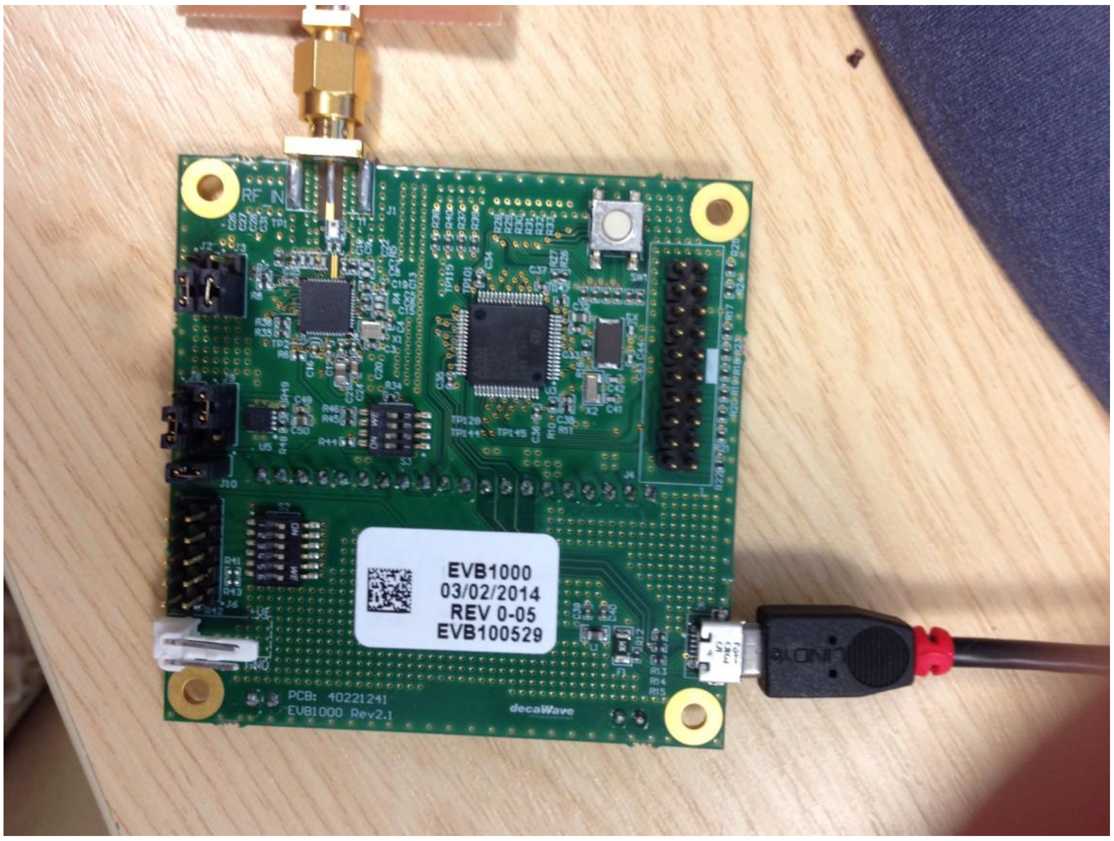


*Figure 8-3 EVB1000 evaluation board with antenna*

The Anchor nodes is the node device with their real world physical position known by the system and help to calculate the target's position. In the system, the Anchor are passive nodes, and they always waits request either for ranging or data transmission. When the Tag is out of range of the pan coordinator, the anchor works as a coordinator (second level coordinator) and helps Tag to relay the ranging data package to the Gateway by multi-hop.

The Gateway is the device to link with the WSN localization system; it is connected with the pan coordinator of the WSN, which all the other devices join the established IEEE 802.15.4a network after identifying the existence of the coordinator and successfully obtaining permission to join the network from the coordinator. The demo system is using UWB RF (Radio Frequency) component as the transmission device. The ranging process is designed to send a specified value from the Target device to the Anchor nodes. Then send the distance value to the coordinator after fetching. The Gateway is the destination of all the ranging data, and the data processing component in the system. It receives the incoming ranging data of each Anchor from Tag node and calculates the target‘s real world position by using trilateration algorithm. The Front-End UI is for user monitoring purposes. In the current stage, we combine the localization Server and the User Interface into one part as it may be easier for testing.

The algorithm for the demo system includes in-network data compression for reducing energy of transmission and data mining for distance calculation between the nodes, trilateration for determining the tracking location These processes can be considered as part of data mining under in-network data processing strategy.

### 8.2.1 In-network Data Processing

The system is featured with In-network data processing technique to enhance the performance and running efficacy. The data in a localisation and tracking system is more like linear data. When the target object is moving, the distance data from each anchor node are changing within a certain range. In the current case study, the amount of the devices is limited to seven, so we cannot choose the cluster head dynamically. As we discussed in the previous chapters, there are three levels of data processing in total:

1. Clustering level: Because of the limitation of devices, we choose the target device as the cluster head. The Anchor nodes as cluster members, the cluster is static.

2. Data Compression level: In this level, the ODH optimal dynamic Huffman algorithm is used to optimise the data transmission between target device and gateway. It aims to reduce the size of packets, to minimal the transmitter's active time, so that it can improve the energy consumption.

In this study, we mainly focused on the use of the optimal dynamic Huffman table that was introduced in chapter six to record the different values and to translate the different values to the value that costs less space than the real value. It means the transceivers may work in a shorter time to save more energy. In this dynamic Huffman table algorithm, the sender and receiver only update their own Huffman table in a certain rule, the rule will keep the Huffman tables matching each other, so the receiver can fetch the data as exactly the same as the one before encoding in sender device. An example of the data processing descripted below:

At the very beginning, because the data being sent in the system is based on the different value, the very first group of data will be sent in original value. In the meantime, the Huffman table has been initialized by a default table. We assume the sending value is (20，30，30，40), with the actual sending value (00010100，00011110，00011110， 00101000). The two Huffman tables in each sides shows below:

*Table 8-1 The initialization of Huffman table*

| Sender | | | Receiver | |
|---|---|---|---|---|
| Real Value | Replacement | | Real Value | Replacement |
| 0 (00000000) | 0 (00000000) | | 0 (00000000) | 0 (00000000) |
| 1 (00000001) | 1 (00000001) | | 1 (00000001) | 1 (00000001) |
| 2 (00000010) | 2 (00000010) | | 2 (00000010) | 2 (00000010) |
| 3 (00000011) | 3 (00000011) | | 3 (00000011) | 3 (00000011) |
| 4 (00000100) | 4 (00000100) | | 4 (00000100) | 4 (00000100) |
| 5 (00000101) | 5 (00000101) | | 5 (00000101) | 5 (00000101) |

| | | | | |
|---|---|---|---|---|
| 6 (00000110) | 6 (00000110) | | 6 (00000110) | 6 (00000110) |
| 7 (00000111) | 7 (00000111) | | 7 (00000111) | 7 (00000111) |
| 8 (00001000) | 8 (00001000) | | 8 (00001000) | 8 (00001000) |
| 9 (00001001) | 9 (00001001) | | 9 (00001001) | 9 (00001001) |
| …… | …… | | …… | …… |
| 50(00110010) | 50(00110010) | | 50(00110010) | 50(00110010) |
| …… | …… | | …… | …… |
| 99(01100011) | 99(01100011) | | 99(01100011) | 99(01100011) |

This phase happened in the sender device. As the initial raw data package has been transmitted, from the second data package, the transmitted data will start to use Huffman table to replace the different values compared with the previous package. For example, the second wave real data is (22，33，33，44), so the different value is: (2，3，3，4), the replacement is (00000010，00000011，00000011，00000100). After the despatching of the second package, the sender will upgrade its 99(01100011)99(01100011)99(01100011) Huffman table with the despatched data package. Any value appears in the second package will shift forward by one space, shown in bold character in the following table. The value “2”, “3” and “4” move forward and the “1” fall to the fifth place.

*Table 8-2 Updated Huffman Table for Encoding in Target Device*

| Sender | | | Receiver | |
|---|---|---|---|---|
| Real Value | Replacement | | Real Value | Replacement |
| 0 (00000000) | 0 (00000000) | | 0 (00000000) | 0 (00000000) |
| **2 (00000010)** | 1 (00000001) | | 1 (00000001) | 1 (00000001) |
| **3 (00000011)** | 2 (00000010) | | 2 (00000010) | 2 (00000010) |
| **4 (00000100)** | 3 (00000011) | | 3 (00000011) | 3 (00000011) |
| **1 (00000001)** | 4 (00000100) | | 4 (00000100) | 4 (00000100) |
| 5 (00000101) | 5 (00000101) | | 5 (00000101) | 5 (00000101) |
| 6 (00000110) | 6 (00000110) | | 6 (00000110) | 6 (00000110) |
| 7 (00000111) | 7 (00000111) | | 7 (00000111) | 7 (00000111) |
| 8 (00001000) | 8 (00001000) | | 8 (00001000) | 8 (00001000) |
| 9 (00001001) | 9 (00001001) | | 9 (00001001) | 9 (00001001) |
| …… | …… | | …… | …… |
| 50(00110010) | 50(00110010) | | 50(00110010) | 50(00110010) |
| …… | …… | | …… | …… |
| 99(01100011) | 99(01100011) | | 99(01100011) | 99(01100011) |

In this phase, the receiver decodes the second package using the default Huffman table and converts back to the real value. After that, it will update its Huffman table with the decoded value to make the two Huffman tables from sender and receiver match each other for the next wave data.

*Table 8-3 Updated Huffman Table for Decoding in Gateway node*

| Sender | | | Receiver | |
|---|---|---|---|---|
| Real Value | Replacement | | Real Value | Replacement |
| 0 (00000000) | 0 (00000000) | | 0 (00000000) | 0 (00000000) |
| 2 (00000010) | 1 (00000001) | | **2 (00000010)** | 1 (00000001) |
| 3 (00000011) | 2 (00000010) | | **3 (00000011)** | 2 (00000010) |
| 4 (00000100) | 3 (00000011) | | **4 (00000100)** | 3 (00000011) |
| 1 (00000001) | 4 (00000100) | | **1 (00000001)** | 4 (00000100) |
| 5 (00000101) | 5 (00000101) | | 5 (00000101) | 5 (00000101) |
| 6 (00000110) | 6 (00000110) | | 6 (00000110) | 6 (00000110) |
| 7 (00000111) | 7 (00000111) | | 7 (00000111) | 7 (00000111) |
| 8 (00001000) | 8 (00001000) | | 8 (00001000) | 8 (00001000) |
| 9 (00001001) | 9 (00001001) | | 9 (00001001) | 9 (00001001) |
| …… | …… | | …… | …… |
| 50(00110010) | 50(00110010) | | 50(00110010) | 50(00110010) |
| …… | …… | | …… | …… |
| 99(01100011) | 99(01100011) | | 99(01100011) | 99(01100011) |

The Huffman table only contains one hundred initial value at beginning. If a new value appears, they will update the table by inserting the new value into 50th place, for example as the following table, it inserts "-4" and "-2" into the 50th place, that makes all the following values shift two spaces behind.

*Table 8-4 Updated Huffman Table with New Data Incoming*

| Sender | | | Receiver | |
|---|---|---|---|---|
| Real Value | Replacement | | Real Value | Replacement |
| 0 (00000000) | 0 (00000000) | | 0 (00000000) | 0 (00000000) |
| 2 (00000010) | 1 (00000001) | | 2 (00000010) | 1 (00000001) |
| 3 (00000011) | 2 (00000010) | | 3 (00000011) | 2 (00000010) |
| 4 (00000100) | 3 (00000011) | | 4 (00000100) | 3 (00000011) |
| 1 (00000001) | 4 (00000100) | | 1 (00000001) | 4 (00000100) |
| 5 (00000101) | 5 (00000101) | | 5 (00000101) | 5 (00000101) |
| 6 (00000110) | 6 (00000110) | | 6 (00000110) | 6 (00000110) |

| | | | | |
|---|---|---|---|---|
| 7 (00000111) | 7 (00000111) | | 7 (00000111) | 7 (00000111) |
| 8 (00001000) | 8 (00001000) | | 8 (00001000) | 8 (00001000) |
| 9 (00001001) | 9 (00001001) | | 9 (00001001) | 9 (00001001) |
| …… | …… | | …… | …… |
| 49(00110001) | 49(00110001) | | 49(00110001) | 49(00110001) |
| **-4(10000100)** | 50(00110010) | | 50(00110010) | 50(00110010) |
| **-2(10000010)** | 51(00110011) | | 51(00110011) | 51(00110011) |
| **50(00110010)** | 50(00110010) | | 50(00110010) | 50(00110010) |
| **51(00110011)** | 51(00110011) | | 51(00110011) | 51(00110011) |
| **……** | …… | | …… | …… |
| **97(01100001)** | 99(01100011) | | 99(01100011) | 99(01100011) |

Once the receiver fetches the value, it will decode the data with the existing Huffman table first and update the table with the decoded value. The processing above repeats as the system runs. The dynamic Huffman table strategy is like a self-learning process, it gently pushes the value with more frequency to the top of the table to reduce the data package length, so to reduce the time of data transmission.

3. Data Mining level: In this demo system, the data mining process is focused on calculating the real position of the target node from the raw distance data to make the value more meaningful for human users. First, annotation process is to pre-process data before analysis. Under the pre-processing, both of the other components can effectively and correctly understand the relationship between the information and sensors data. Second, the mining process operates data mining in real-time context. The real-time data mining process engine imports the feature data rules from the knowledge base and formatted data into location tracking.

- Annotation Process

The recent research of RTLS has investigated several technologies that can be used. Such as using dedicated RFID tags and readers while others use existing WLAN networks and add RLTS ability to those networks. But the most common methods used by RLTS to locate an object in 2D space one way ranging are Time of Arrival

(ToA), Time Difference of Arrival (TDoA) and Received Signal Strength (RSSI). However, to achieve precision up to the nanosecond scale, which results in a more precise distance measurement, an elaborate clock synchronization system must be developed which has high costs in terms of development time and effort.

Compared with one way ranging, two way ranging has much softer conditions. The Time of Flight method uses measured elapsed time for a transmission between a tag and a reader based on the estimated propagation speed of a typical signal through a medium. As this method is based on a time value, clock accuracy becomes significantly more important than in previous methods. In annotation process, the system set the link between Time of Fight (TOF), the distance between the tag and the reader, the position in the test room.

- Location Tracking Process

Trilateration is based on the theory that the position of a point projected on a 2-D plane can be determined by its distances from three non-collinear reference points. In trilateration, the positions of three reference points are known, as well as the distances from each of them to the unknown target point. Therefore, regarding to the location tracking, trilateration algorithm uses the geometrical property of circles, or triangles (O.S. Oguejiofor .etc. 2013) to calculate the positions of target points based on the reference distances from the anchor nodes. Figure 8-4 shows the principle of trilateration.

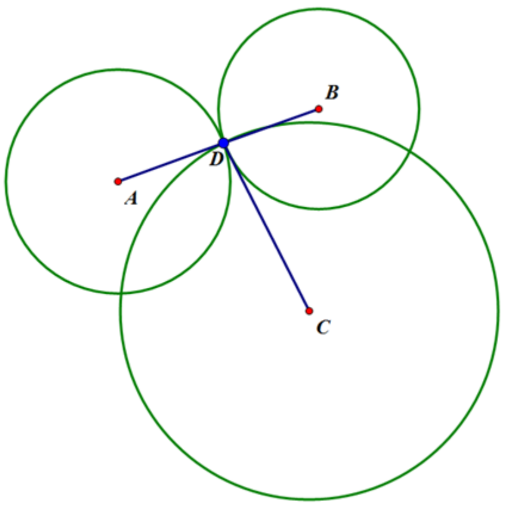


*Figure 8-4 Principle of trilateration*

The idealized model assumes that three reference circles A, B and C intersect at the point of a Target node position (). Then the intersection point is the position of the unknown target node. The location coordinates of the point D can be calculated by solving any two circles' mathematic equations. Generally, in any study involving indoor radio signals, one must be tool into consideration is the effect of multipath propagation (i.e. reflection and absorption of signals). Trilateration problem occurs since the physical measurement errors cannot be avoided. The circles centres and radii are dynamic and the circles might overlap in a region rather than intersecting at a single point and the unknown node location N is somewhere within this region. The measurement errors between the tag node and the anchors have great impact on the size of this overlapping region. Worse still, the three reference circles even do not intersect at all based on useless error correction. Thus, the localization accuracy of Trilateration will be significantly reduced. In this demo system, we design the localisation algorithm meet all the possible chances between these three circles to improve the error acceptance of the system.

## 8.3 Embedded Software Design

The Embedded software design includes two parts. The first part introduced the main processing between Target device and Anchor node, the code is built based on the distance ranging demo project of the EVB1000 development kit. The second part presents the algorithm of dynamic Huffman data processing.

### 8.3.1 Two Way Ranging Process between Anchor and Tag

Once power is on, the main controller (STM32F105 ARM Cortex M3) takes the responsibility of initializing hardware and software. The hardware includes system clock, SPI interface, and interrupt source. The software includes ranging algorithm and ad hoc routing protocol. Once initialization completes, the operating system periodically calls the scheduled functions. If any desired event is captured, the corresponding function will handle it at the closest time. The Anchor node is designed to place at the edge of the tracking field and its position is fixed through the whole localization process. In addition, the Target device measures the distance between itself and each Anchor node. The system using UWB radio to establish a WPAN, it handles the distance measurement request and data transmission request by different packet header. (DecaWave Ltd, 2013)

```
void instanceconfigframeheader(instance_data_t *inst, int ackrequest)
{
	inst->msg.panID[0] = (inst->panid) & 0xff;
	inst->msg.panID[1] = inst->panid >> 8;

	//set frame type (0-2), SEC (3), Pending (4), ACK (5), PanIDcomp(6)
	inst->msg.frameCtrl[0] = 0x1 /*frame type 0x1 == data*/ | 0x40 /*PID comp*/;
	inst->msg.frameCtrl[0] |= (ackrequest ? 0x20 : 0x00);
#if (USING_64BIT_ADDR==1)
	//source/dest addressing modes and frame version
	inst->msg.frameCtrl[1] = 0xC /*dest extended address (64bits)*/ | 0xC0 /*src extended address
(64bits)*/;
#else
	inst->msg.frameCtrl[1] = 0x8 /*dest short address (16bits)*/ | 0x80 /*src short address
(16bits)*/;
#endif
	inst->msg.seqNum = inst->frame_sn++;

}
```

A UWB network consists of a number of DW1000 (UWB chip on EVB1000) devices that are capable of detecting distance between any pair of two devices (for these two devices, one acts as a Tag, and another one acts as an Anchor). The code presents the process of finding the first Anchor (Coding see Appendix I).

A ranging algorithm is implemented by two kinds of ranging devices (Tag and Anchor). A Tag is responsible for issuing ranging request by sending a poll message to an Anchor with specified anchor address. Once the message arrives on the Anchor, the Anchor will respond to the Tag. If the local device is pre-configured as a TAG, a timer will be started. After that, the TAG will periodically perform the defined functions of measuring distance to Anchor 1, 2 … 6, and report ranging results to a gateway. If the local device is pre-configured as an Anchor, it will perform ranging algorithm when certain conditions are satisfied (Coding see Appendix II).

### 8.3.2 In-Network Data Processing

In the proposed system, the target device acts as a sender (cluster head), which collects all the distance information and sends it to the gateway device (coordinator). There are several flag bit in the target device’s software, the “u8Flag” is used to present it the data is original data (value = 1) or the difference data (value = 2). The “u8Src” is a pointer, which points to data passed by application.

If data is offset data, the application should implement proper processing in advance. For example, if offset is 100, passed value is 01100100, if offset is -100, passed value is 11100100, where the highest bit indicates positive or negative (see Appendix III).

After the first wave of data sending, the following data will be compressed by the Huffman table. When the parameter “u8Src” is the pointer points to data passed by application, “u8Dest” pointer points to data to be sent out, “u8Len” is the length of data. Dest[0] is for indicating "existing value" and "new value", where "0" means existing value and "1" means it is a new value. When a value can be searched in the default Huffman table, it will compress the data into replacement value. If the value cannot be found in the table, it will be sent as it is, and will update the Huffman table by inserting the new value into the fiftieth place (Coding see Appendix IV).

The destination of data is the gateway device in the network. It connects the server and the WSN as a bridge. In the receiver, the first step after receiving a new data is to validate its eligibility. If the length of the packet is less than 3, it means the package is broken and will be dropped. In the packet, the first element is the packet type, “DATAREADING” means is compressed value, “ORIGINALDATA” means it is the very first value (raw data), the second element is the length of valuable content excluding the first three elements, and the third element is data flag to indicate if the corresponding data is an old value or new value. The details can have been presented in the code comments.(Coding see Appendix V)

## 8.4 Lab Test

The lab test is concluded which focuses on the energy saving for the demo system. The test is to exam the designed system’s functionality with the in-network data processing strategy in the lab environment. Moreover, to show the performance of the designed strategy. To compare the differences between the strategies OFF and ON, we did the test two times and recorded the voltage of battery and the matching time stamp. The third floor open area was chosen as the lab-testing site (Figure 8-5). To test the system functionality, we took over an accuracy tests are done by collecting 10 groups of coordinates. Each group contains the real-world coordinates and the calculated coordinates of the target at one specific position.

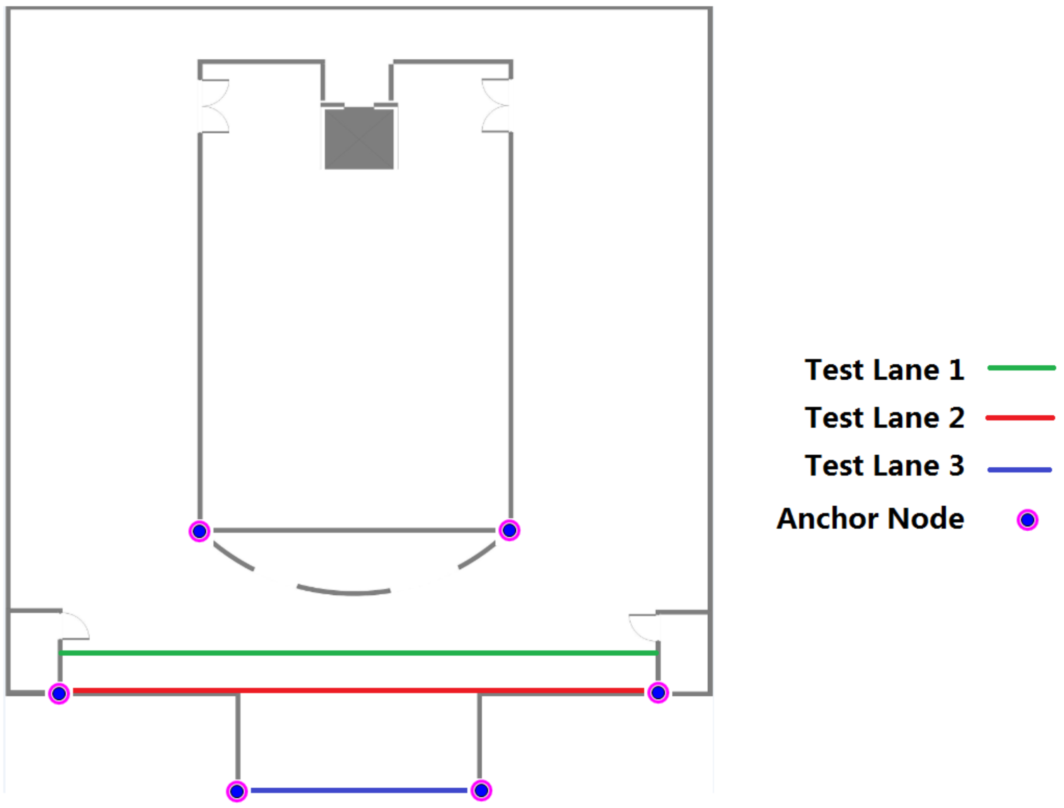


*Figure 8-5 Lab Testing Area*

The measurements of the entire third floor are 26.20m long and 23.45m wide. We randomly picked 20 groups of sample and measurement the real world position by using a Bosch laser measure device. We pick 20 groups of sample each on Lane 1 and Land 2, and pick 10 groups of sample on Lane 3, with the interval of 1 meter. In the result graphs, the red points are the real-world positions of the target. The blue curve shows that the estimated positions fluctuated around the red points.

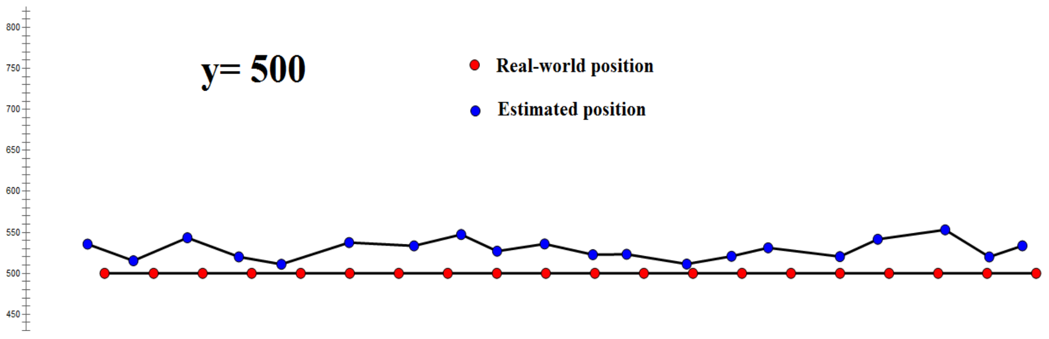


*Figure 8-6 Test result of Lane 1*

Figure 8-6 shows that test Lane 1 is roughly the middle line of the area which is surrounded by the 6 anchors. Thus the localization on this lane is expected to be more accurate than other positions since tag can receive all the six anchor’s signal directly without passing through the walls with cables inside or other equipment which might interfere the signals. We add all the D-value between estimated and real-world X coordinates together and then divide the sum by the numbers of samples. The result reveals that the average error of X equals to 0.226m and similarly the average error of Y is around 0.322m.

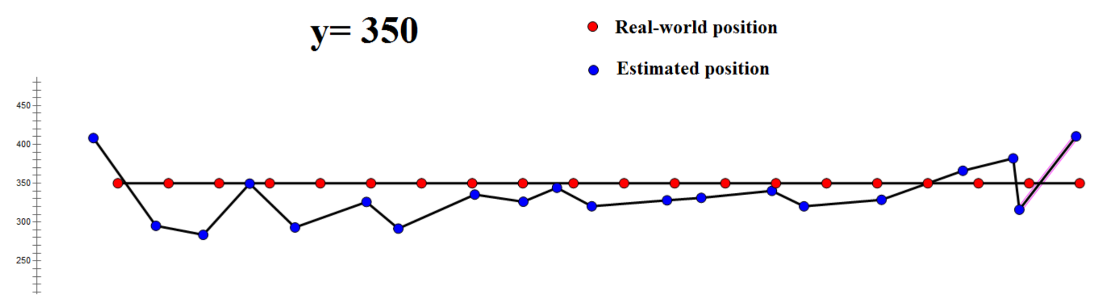


*Figure 8-7 Test result of Lane 2*

Test Line 2 is a line that has two segments adjoins the wall. As expected, the estimated error will be larger that Test Line 1 especially when the targets are sitting at the positions within these two segments. In Test 2, the Average error of X = 0.367m and the Average error of Y = 0.395m.

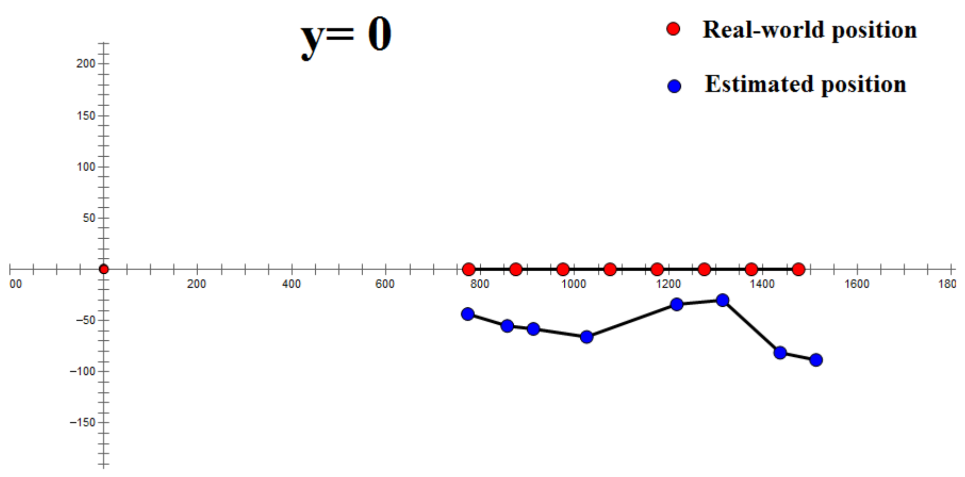


*Figure 8-8 Test result of Lane 3*

Test line 3 is a windowsill line which is the bottom edge of the test field. When the target is sitting on this line, the radio signals from the far left and far right anchors need to permeate through the wall. Hence the errors might get further enlarged. In Test 3, the average error of X = 0.417m and the average error of Y = 0.62m.

To draw a conclusion of the tests, the estimated position distributes around the real world positions without any rule. When the target is sitting in the middle area far away from the walls or windows, the localization errors is around 0.3m with minimal 0.05m, which can be considered as an acceptable accuracy for an indoor localization system. Due to the effects from the walls, cables inside the walls, radiators and other large metal objects, the accuracy gets significantly decreased when these barriers are sitting on the path between the tag and anchor. The calculation error is roughly between 0.4m and 0.6m. Overall, the accuracy of the whole system is around 0.4m on average. That result is much better than the RSSI Fingerprint based tracking method which can only achieve around 2 meters according to the test we had done in lab.

To indicate how the in-network data processing improve the energy consumption for the individual node device, we use the measurement of batteries' voltage to identify the power consumption level. To do this, we pick two groups of brand new type D alkaline batteries with very similar voltage (±0.01V) at 1.58V. Uploading the software with in-network data processing strategy disabled and installing the batteries for Group A anchor node devices. There is an on-board ADC on each node device to monitor the voltage level, we enable that component and send out the voltage value with localisation data package. At the end of the Group A test, the final voltage level is recorded. For the second group, uploading the software with in-network data processing strategy enabled, pick a set of batteries with the same energy capability, repeating the steps above and recording the final voltage level for Group B. The testing period of each group lasts four hours with sampling rate set to 4 second once. The target tag is held by a person who moves from the left border to the right border and returns every 15 minutes. This act is to make sure the whole network is involved in the tracking process, and the result is shown in the following figure.

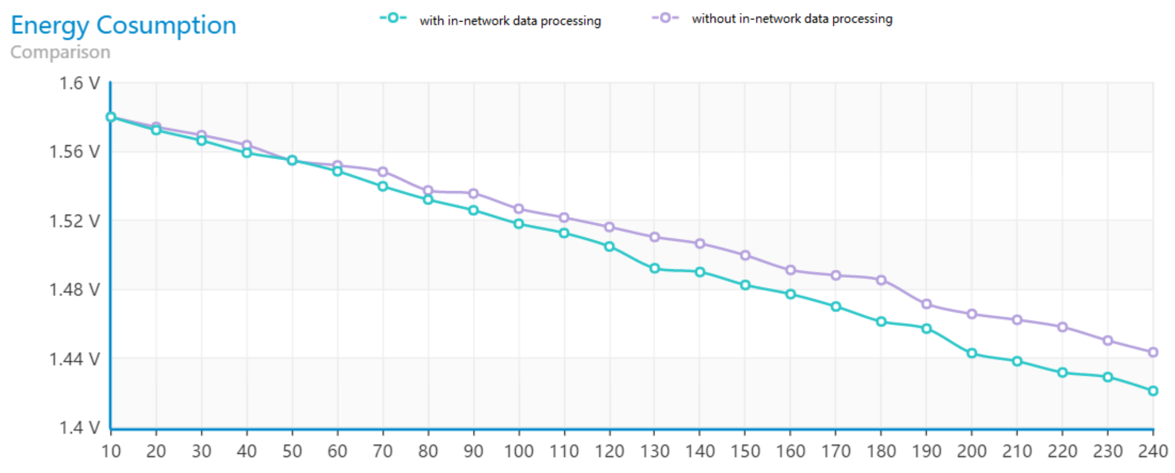


*Figure 8-9 Energy consumption comparison*

In Figure 8-6, the abscissa presents the time with minute as unit, and the ordinate presents the mean value of each group's battery on-board voltage level. The figure clearly shows the energy consumption saving on the group with in-network data processing enabled system. The dynamic Huffman strategy is a self-learning process, so the power consumption for these two groups is nearly the same. However, the strategy starts to show its benefit with the time passing, the Huffman table was updated to the value set, which suits the testing site environment, like the speed and the moving behaviour of the target. At the point of the first hour (60 minutes in the figure), the mean voltage value of Group A is 1.5485V, and 1.5493V for Group B. It is 3% energy saving compared with none in-network data processing strategy. At the point of the second hour, the mean voltage value of Group A is 1.5049V and 1.5142V for Group B. At that time point, the energy saving is about 5%. At the end of the test, the mean voltage value of group A is 1.4212V, and the mean voltage value of group B is 1.4436V. The energy saving is about 14%. The result shows the longer the system is running; the more energy is saved by the In-network data processing strategy.

## 8.5 Summary

This case study focuses on the test of the energy saving performance with in-network data processing strategy. With the highlighted energy saving feature by deploying the dynamic Huffman data compressing strategy, as a main objective of the test, the system reaches the expected performance by providing localisation accuracy between 0.3m and 0.5m. Furthermore, the in-network data processing helps to reduce the energy consumption for the entire system with 13% maximum per hour compared with a plain system without the strategy. The strategy has proved its value to save the times of data transmission inside the sensor network, and so to save the energy for the entire network. The current system only contains six anchor devices, which means the packets amount is small compared with a real world WSN in use. Overall, the accuracy of the whole system is around 0.4m on average. That result is much better than the RSSI Fingerprint based tracking method which can only achieve around 2 meters according to the test we had done in lab. We believe the performance of the in-network data processing will be greater in a larger size WSN with bigger amount of packets being processed inside the network.

# Chapter 9. Conclusions and Future Work

## 9.1 Conclusions

In-network processing is one of the most important techniques in sensor network DB and a key emerging area in the design and development of WSNs. In one form of in-network processing, an intermediate node or aggregator will fuse or aggregate sensor data, which are collected, from a group of sensors before transferring to base station. Gathering information from a WSN in an energy effective manner is of paramount importance in order to prolong its life span. This thesis proposed a new architecture of processing heterogeneous sensor data within WSNs. Its objective was to minimise the energy consumption and migrate other constraints in WSNs (Chapter 2). This thesis addressed the challenges of WSN for data processing. The in-network data processing architecture for migrating the constraints of WSNs was proposed. Three main levels (clustering level, data compression level and data mining level) were further investigated in detail. We present the development approaches for the three levels, separately (Chapter5, 6 and &7). The research reported in this thesis shows that the combination of the three levels in the indoor location tracking case study reduces the energy consumption in the WSNs (Chapter 8).

This chapter is structured as follows. Section 9.1 gives the conclusion of this thesis. Section 9.2 outlines research objectives and questions Revisited in chapter 1. Section 9.3 presents the contributions of this thesis. Section 9.4 discusses the limitations of the work and gives the direction for future work.

## 9.2 Research Objectives Revisited

**RQ1**: What are the factors that need to be considered for WSN constraints?

**Objective associated with RQ1:** Investigate the existing literatures to understand the restrictions of wireless communication, i.e., energy consumption, heterogeneity, data accuracy.

**Objectives Revisited:** Through the applications' requirements, the WSN constraints include energy constraints, heterogeneity, real-time operation, data accuracy and unpredictability. The constraints analysis managed to identify the existing challenges for data processing. As a result, we propose the research method in section 2.4 list the concept development, and three technology opportunities were identified. Consequently, those three opportunities were taken forward to technical development.

**RQ2:** How can the technology capabilities of WSNs be implement in different applications (emergency response system and in-door location system) and mitigate the WSN constraints?

**Objective associated with RQ2:**

- Understand the goals, tasks and information requirements from the chosen user group raised form the WSNs constraints.

- Research the existing literature available on data processing for sensor data within the sensor network.

- Design a suitable in-network data processing architecture for both environment and human monitoring applications

**Objectives Revisited:** Through the literature review on data processing from data aggregation, data compression and data mining were analysed. The challenges of make sense of the relationship between WSN constraints and the existing data processing approaches was revealed. Through the proposed in-network data processing architecture, the essential approaches to fulfil the WSN constraints were identified to be 1) data aggregation to merger the sensor data in the cluster. 2)data compression is to compressed packets in cluster head 3) data mining is to extract event from a large amount of data in the gateway node.

**RQ3:** What constitutes a suitable data aggregation to mitigate WSN constraints within the network?

**Objective associated with RQ3:**

- Propose a suitable data aggregation technique for sensor data to limit the communication of unnecessary events for energy consumption, improve the data accuracy and deploy the flexible structure for network self-organization
- Evaluate the efficiency of the proposed data aggregation and its cost-effectiveness

**Objectives Revisited:** Through the existing data aggregation, what constitutes a suitable network structure was evaluate. The proposed data aggregation approach (NMCA) is a hybrid algorithm which combine multi-path and cluster aggregation. It overcome the disadvantages of them. The performance evaluation demonstrated to improve network lifetime by 20% compared with LEACH and grantee packet transmission under different loss rate.

**RQ4:** What constitutes a suitable data compression to mitigate WSN constraints within the network?

**Objective associated with RQ4**

- Analyse and propose a suitable data compression for data mining for both alert and regular sensor data to reduce energy consumption by minimizing data volume and represent sensor data

- Evaluate the effectiveness of the proposed data compression approach in comparison to the existing approaches

**Objectives Revisited:** Through the existing data compression approaches analysis and the WSN constraints, the two communication compression schemes for both regular packets and alert packets in the network. The neighbour-aware commination compression for regular packets is used to reduce the data redundancy for further compression in the cluster head. The proposed ODH is a dynamic and lightweight algorithm and high performance for different correlated and repeated data comparison with the adaptive Huffman algorithm.

**RQ5:** How to extract information from sensor data to mitigate WSN constraints within the network?

**Objective associated with RQ5**

- Analyse and specify what "meaning" can be in the context of WSN-based onsite emergency response system, and propose a data mining approach.

- Evaluate the performance of the proposed meaning extraction approach and discuss its application

**Objectives Revisited:** Through the existing meaning extracting analysis and the constraints of event detection, sematic data mining model is proposed to process the data by semantic annotator and then extract event by data mining rule engines. The sematic process is to establish the link between data and information by semantic annotator and reduce the outliers by Kalman filter. Then, a generic state model of emergency event detection in Wireless Sensor Networks was proposed, which addresses the key requirements of identifying the occurrence and characteristics of an incident based on sensor networks

## 9.3 Contribution of Knowledge

**Contribution 1:** The proposal of in-network data processing architecture that addresses three levels of in-network data processing: data clustering level, data compression level and data mining level.

The existing data WSN constraints were investigated and analysed by the application requirements. The major application requirements are network lifetime (Heinzelman et al., 2002), event detection (Alsheikh et al., 2015), information quality (Martinez et al., 2007), data timeless (Paradis and Han,2008) and self-organization (Walpola, 2009). They make the WSN able to consider about energy consumption, heterogeneity, data accuracy, real-time and unpredictability. They are the factors that are necessary for sensor data processing within a heterogeneous WSN. The existing data aggregation approaches can limit the communication of unnecessary events for energy consumption, improve the data accuracy and deploy the flexible structure for network self-organization (Sasirekha and Swamynathan, 2015). But for the important data, the sensor data have to be processed by lossless processing approaches. Data compressing is a lossless energy efficiency approaches (Haupt et al. 2008). In heterogeneity WSN, there are several different types of sensors in the same nodes. Data mining is to process sensor data for mining the acceptable information for users from heterogeneous sources (Mao et al., 2012). The investigate of existing approaches, most of them cannot fulfil all the constraints.

> Therefore, there is a need for design an architecture to combine data aggregation, data compression and data mining for mitigating the constrains of energy, heterogeneity, unpredictability, real-time and data accuracy.

The proposed architecture can mitigate the WSN constraints. The design of the in-network data processing architecture includes clustering layer, data compression layer and data mining layer. The clustering level is the method to build the network structure for data processing. The data compression level is the data oriented algorithm to reduce the energy consumption and process data within the network. The data-mining model is for the specific event detection to convert the sensory data to the acceptable knowledge for the users. The in-network data processing architecture processes data locally and the sensor nodes make decisions quickly and remotely without instructions from a centralized database. Most importantly, the designed components can work either independently or together for different application requirements.

**Contribution 2:** A Proposal Neighbour-aware Multi-path Cluster Aggregation (NMCA)

In-network data aggregation can be done in the sensor node is to organize the WSN into a structural architecture such as tree, chain or cluster (Sasirekha and Swamynathan, 2015). The main disadvantage of tree-based aggregation has been pointed out by multi-path aggregation: trees are susceptible to link loss and node loss in a sensor network. The closer a node is to the sink, the more important becomes a node for tree-based aggregation: if a link or node fails that is close to the sink, the aggregated information of an entire sub-tree could be lost. the cluster-based schemes (Govindan et al., 2002, Akkaya and Younis, 2005, Mahimkar and Rappacit, 2004, and Nath et al., 2004) also consist of hierarchical organization of the network, which is a self-organizing and adaptive clustering algorithm using randomization to distribute the energy expenditure among the sensors evenly. But, several problems may arise in highly dynamic environments. In addition, mobility additional problems may arise. A node close to a cluster head at a given instant in time may move away from the cluster head. Therefore, the node needs to increase its power, thereby spending much more energy to transmit to the cluster head than expected. Multi-path aggregation can improve the data accuracy under the high loss rate network. But it required more power for the sensor node.

Therefore, we propose Neighbour-aware Multi-path Cluster Aggregation (NMCA), which make the maximum efficiently for both energy saving and data accuracy.

NMCA explored in-network aggregation clustering as a power-efficient and accurate mechanism for processing and collecting data in wireless sensor network. Our focus was on applications where a large number of nodes produce data periodically, which is consumed by fewer sink nodes. Multi-path data transmission can overcome the robustness problems. But it requires more energy compare with the single path. Hence, we propose the neighbour-aware phase, which is used to discard the redundant packages, only send the satisfied important data to aggregated in cluster head for further transmission. NMCA presents the network structure to employ in-network data processing. By classifying the network into clusters by the region and exploited neighbour-aware and multi-path phase to improve data accuracy of the event. After the clustering level, the aggregated data is prepared in the cluster head for further processing by compression level. It also can be used independently for a small-area data processing.

**Contribution 3:** The proposal of communication compression scheme for UEN packets and Optimal Dynamic Huffman algorithm for data compression level.

Development of effective compression algorithm is a key to improve utilization of the limited resources of WSNs (energy, bandwidth, computation power). Hence, one of the most important requirements of data compression is lightweight for sensor computation power and memory storage. Data and package funneling (Petrovic et al. 2003) techniques is a simple and lightweight data compression for in-network data processing. Related data compression techniques (Pradhan et al., 2002 and Kusuma et al., 2001) are suitable for large-scale network, high performance for energy efficiency and compression ratio. It also suffers in dynamic environments and network. But it requires more memory storage and computation power. Therefore, our proposal methods should be lightweight and dynamic for the environment. The majority of the existing research techniques on in-network data compression include sample compression, data compression and communication compression. A reduced number of samples help in reducing the data length (data compression), which ultimately reduces the radio on-time of the transceivers (communication compression). However, very few of the existing compression techniques do so. They only operate on compressed samples or non-compressed samples. Our purpose is to exploit communication compression and data compression which is satisfied lightweight and dynamic in WSN.

> Therefore, we propose the communication compression scheme for UEN packets to reduce the redundant packages and a lightweight Optimal Dynamic Huffman (ODH) algorithm for data compression level.

Neighbour-aware compression scheme for UEN is high performance for several events at the same time. It reduces the energy consumption and data redundancy in the event detection for communication compression. Optimal Dynamic Huffman algorithm can be easily employed in practice and any wireless sensor network for in-network data compression. In order to evaluate the algorithm, we computed the compression ratio obtained in several real datasets containing temperature collected at different locations and during distinct periods. The datasets also have the different correlation rates. Both of Adaptive Huffman algorithm and ODH are high performance for different correlated and repeated data. However, ODH is more suitable for in-network data processing.

**Contribution 4:** A semantic real-time data-mining model for emergency event detection.

Existing research has revealed the data mining that is an emerging field that identifies elements of information contained in datasets that imply meaning in the context of application and can be interpreted by the users to facilitate their tasks. The existing meaning extraction research is highly domain-specific. We review the related research, and classify two categories in the data mining: data oriented processing and event/ model driven approaches. Data oriented processing (Chatzigiannakis et al., 2006) is to model the spatial-temporal data correlations efficiently in a distributed manner and identifies local outliers spanning through neighbouring nodes. The disadvantage is that the primary node procedure of selecting appropriate PCs is computationally very expensive. Event / model drive data mining is to extract information by sematic event or model (Yu and Taylor, 2013). A key disadvantage is the fact that the entire action history must be stored and processed off line, which is not practical for large prediction tasks over a long period.

Therefore, there is a need for the propose data mining, which is used to detect the event in the network and improve the data accuracy by a lightweight computation.

Data clustering and data compression techniques have a significant benefit for energy efficiency and communication latency. However, for data mining in the gateway node, it must be designed to satisfy for not only energy efficiency and communication latency, but also the accuracy of the conveyed information. There are also constraints fault tolerance, which should effect the results in the applications. Semantic data mining includes the semantic annotator, knowledge base and data mining rule engines to detect fire and improve the detection accuracy. The semantic process is applied to the data format structure and outlier detection to pre-process data for data mining rules. Event detection process uses the threshold-base rules from the knowledge base to extract the event by data mining rules engines. The simulation demonstrated that applying this mode could improve efficiency and accuracy of the existing event detection algorithms.

**Contribution 5:** A case study for in-door location tracking system embed in-network data processing.

Indoor location and tracking is currently a hot topic amongst the consumer industry and other fields. It provides the user the current position inside the building where Global Positioning System (GPS) cannot reach. This system is designed for applications such as trailer tracking, roll cage tracking, and forklift tracking in a warehouse/retailer's distribution centre, beds and equipment tracking in a hospital, or any other similar scenarios of indoor location and tracking purposes. It aims to develop a low cost, WSN based localization and tracking system with acceptable tracking accuracy. Further, the core of this objective was to build an efficient, low cost, and affordable location and tracking WSN. After that, the back-end gateway collects the distance measurements and calculate the target's position. The energy consumption is always the biggest problem in a wireless system, to improve system life circle without affecting the performance significantly.

> Therefore, there is a need for in-door location tracking system embed the in-network data processing to improve the energy efficiency.

We employ ODH optimal dynamic Huffman algorithm to optimise the data transmission between target device and gateway. It aims to reduce the size of packets, to minimal the transmitter's active time, so that it can improve the energy consumption. Data mining process is focused on calculating the real position of the target node from the raw distance data to make the value more meaningful for human users. First, annotation process is to pre-process data before analysis. Under TOF, both of the other components can effectively and correctly understand the relationship between the information and sensors data. Second, the mining process operates data mining in real-time context. Trilateration algorithm is as the mining rules to calculate target's' location. The designed in-network data processing architecture has been tested and successfully implemented in the human behaviour monitoring and helps to reduce the energy consumption for the entire system with 13% maximum per hour compared with a plain system without the strategy.

## 9.4 Summary and Future work

The key contribution of this research in the area is to detect sensor events within the network. The in-network data processing architecture was introduced to provide and support for aggregation clustering, compression and mining data within the WSNs. The sensory data can be processed and filtered to detect events of interest at the sensor nodes. This work is an initial step towards the realization of processing event in network for energy benefits and data accuracy.

The concept of a sensor network is something that provides the end users with the capability to obtain environmental information. The rapid development of in-network data processing techniques in recent years enables the rapid response and energy efficiency and be an easily deployed sensor without any restriction imposed by the need for cables. There are a number of promising points that could be pursued to address the limitations and to extend this work in further. This section describes the directions in which this research can be extended.

**Improvement data mining model and other application domains**

As stated by (Römer ,2008), "it remains a major challenge to make sense of the collected data, i.e., to extract the relevant knowledge from the raw data." This thesis investigated what constitutes meaning in the context of on-site ER. Due to time constraints, it only investigated the occurrence and characteristics of an incident as an example meaning. Further investigation on extracting the real-time development of an incident is needed.

In Chapter 7, the data-mining model extracted knowledge for fire detection by the simple rule model. For future work, we have to improve the data-mining model for different application and mining the information of the event for further investigation. It can improve the efficiency and accuracy of decision making for the users. On the other hand, the current data-mining model is designed for annotating sensor data.

Future work could focus on the automatic infusion by the classifying events.

# References

Alemdar, H. and Erosy, C., “Wireless sensor networks for healthcare: A survey”, *In Journal of computer network*, 2010.

Alsheikh, M., Hoang, D., Niyato, D., Tan, H. and Lin, S., “Markov Decision Processes with Applications in Wireless Sensor Networks: A Survey”, *Communications Surveys & Tutorials, IEEE,* 2015.

Al-Karaki, J. N. and Kamal, A. E., “Routing Techniques in Wireless Sensor Networks: A Survey”, *IEEE Wireless Commun*ication, Vol. 11, No. 6, pp. 6–28, Dec. 2004.

Akkaya, K. and Younis, M., “A Survey of Routing Protocols in Wireless Sensor Networks”, *Elsevier Ad Hoc Network J.*, Vol. 3, No. 3, pp. 325–49, May 2005.

Akyildiz, I.F., Su, W., “A Power Aware Enhanced Routing (PAER) Protocol for Sensor Networks”, *Georgia Tech Technical Report*, Jan. 2002.

Antoine-Santoni, T., Santucci, J. F., Gentili, E. D. and Costa, B., “Using Wireless Sensor Network for Wildfire Detection. A Discrete Event Approach of Environmental Monitoring Tool”, *Environment Identities and Mediterranean Area, ISEIMA '06 First international Symposium,* pp.115-120, 2006.

Arici, T. B. et al., “PINCO: a Pipelined In-Network Compression Scheme for Data

Collection in Wireless Sensor Networks", *In Proceedings of 12th International Conference on Computer Communications and Networks*, Oct. 2003.

Avison, D. and Fitzgerald, G., "Where Now for Development Methodologies?", *Communications of the ACM 46 (1),* 2003.

Barontib, P., Pillaia, P., Chooka, V., Chessab, S., Gottab, A. and Hua Y., "Wireless Sensor Networks: A Survey on The State of the Art and The 802.15.4 and Zigbee Standards", *Computer Communications,* Vol. 30, Issue 7, pp. 1655–1695,2007.

Benini, L., Farella, E., and Guiducci, C., "Wireless sensor networks: Enabling technology for ambient intelligence"*, Microelectronics Journal,* Vol. 37, No. 12, pp.1639– 1649, 2006.

Bharathidasan, A. and Ponduru, V., "Sensor Networks: An Overview", *Technical Report, University of California,* 2003.

Blasch, E. and Plano, S., "DFIG Level 5 (User Refinement) Issues Supporting Situational Assessment Reasoning", *Information Fusion, 8th International Conference,* Vol. 1, pp. xxxv-xliii, 2005.

Blum, J. et al., "Real-time emergency response: improved management of real-time information during crisis situations", *Journal on Multimodal User Interfaces*, Volume 8, Issue 2, pp 161-173, 2014.,

Bontempi, G. and Borgne, Y. L., "An Adaptive Modular Approach to The Mining of Sensor Network Data", *In Proceedings of 1st International Workshop on Data Mining in Sensor Networks as part of the SIAM International Conference on Data Mining*, SIAM Press, pp. 3-9, 2005.

Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. "Time Series Analysis, Forecasting and Control", *3rd ed. Prentice Hall, Englewood Clifs*, 1994.

Brooke, T. and Burrell, J., “From Ethnography to Design in A Vineyard”, *Design User Experiences (DUX) Conference,* 2003.

Candès, E. J. and Wakin, M., “An Introduction to Compressive Sampling”, *IEEE Signal Processing Society,* pp.21-30, Mar. 2008.

Cardei, M. and Wu, J., “Energy-efficient coverage problems in wireless ad-hoc sensor networks”, *Computer Communications,* Vol. 29 (4), pp. 413-420, 2006.

Cardei, M., Thai, M. T., Li, Y. and Wu, W., “Energy-efficient Target Coverage in Wireless Sensor Networks”, *In Proceedings of the IEEE INFOCOM,* pp. 1976–1984, 2005.

Cardell-Oliver, R., “ROPE: A Reactive, Opportunistic Protocol for Environment Monitoring Sensor Networks”, *Journal of Proc. The Second IEEE Workshop on Embedded Networked Sensors*, 2005.

Cerpa, A., Elson, J., Estrin, D., Girod, L., Hamilton, M. and Zhao, J., “Habitat Monitoring: Application Driver for Wireless Communications Technology”, *In ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean,* 2001.

Chandola, V., Banerjee, A. and Kumar, V., “Anomaly Detection: A Survey”, *Technique Report 07-017, Computer Science Department, University of Minnesota*, 2007.

Chanin, J. I. and Halloran, A. R., “Wireless Sensor Network for Monitoring Applications”, *A Major Qualifying Project Report,* 2008.

Chatzigiannakis, V., Papavassiliou, S., Grammatikou, M. and Maglaris, B., “Hierarchical anomaly detection in distributed large-scale sensor networks”, *Computers and Communications, 11th IEEE Symposium,* pp761-767, 2006.

Chen, Y. et al., "In-Network Data Processing for Wireless Sensor Networks", *Mobile Data Management, 7th International Conference*, pp. 26, May 2006. (Missing ref. at Page 32, the literature is classified in-network data processing into two categories: hierarchical and non-hierarchical)

Cook, D. J. and Youngblood, M., "Smart Homes", *Journal of Berkshire Encyclopedia of Human-Computer Interaction, Berkshire Publishing Group*, 2004.

Crone, S., Lessmann, S. and Stahlbock, R., "Support Vector Machines Versus Artificial Neural Networks - New Potential in Data Mining for Customer Relationship Management", *Neural Network Applications in Information Technology and Web Engineering, Borneo Publishing,* pp. 80-93, 2005.

DecaWave Ltd, Distance Measurement Application for EVB1000 development kits Demo Code, 2013.

Ding, M., Cheng, X. and Xue, G., "Aggregation Tree Construction in Sensor Networks", *IEEE VTC '03*, Orlando, FL, Oct. 2003.

Durisic, M. er al., "A survey of military applications of wireless sensor networks", *In proceeding of 2012 Mediterranean Conference on Embedded Computing (MECO),* 2002.

Elnahrawy, E. and Nath, B., "Cleaning and Querying Noisy Sensors", *In Proceedings of 2nd ACM international Conference on Wireless Sensor Networks and Application*, pp. 78-87, 2003.

Elnahrawy, E. and Nath, B., "Online Data Cleaning in Wireless Sensor Networks", *Research Poster*, 2003.

Erramilli, V., Matta, I. and Bestavros, A., "On the Interaction between Data Aggregation and Topology Control in Wireless Sensor Networks", *IEEE*

*SECON ’04*, Santa Clara, CA, Oct. 2004.

Faller, N., “An adaptive system for data compression”, *In Record of the 7th Asilomar Conference on Circuits, Systems and Computers (Pacific Grove, Calif., Nov.), Naval Postgraduate School, Monterey, Calif.,* pp. 593-597, 1973.

Fasolo, E., Rossi, M., Widmer, J. and Zorzi, M., “In-network aggregation techniques for wireless sensor networks: A survey”. *IEEE Communication Magazine*, April 2007.

Fayyad, U. M., Piatetshy-Shapiro, G., Smyth, P. and Uthurusamy, R., “Advances in Knowledge Discovevy and Data Mining”, *AAAI/MIT Press,* 1996.

Feelders, A. J. and Daniëls, H. A. M., “Integrating Economic Knowledge in Data Mining Algorithms”, *CentER Discussion Paper,* Vol. 2001-63, 2001.

Gallager, R.G., “Variations on a theme by Huffman”, *IEEE Trans. Inf. Theory 24, 6 (Nov.)*, pp. 668-674, 1978.

Govindan, R., Hellerstein, J., Hong, W., Madden, S., Franklin, M. and Shenker. S., “The Sensor Network as a Database”. *Technical Report 02-771*, Computer Science Department, University of Southern California, Sep. 2002.

Guo, L. Q., Xie, Y., Yang, C. and Jiang, Z., “Improvement on LEACH by Combining Adaptive Cluster Head Election and Two-Hop Transmission”, *In Proceedings of the Ninth International Conference on Machine Learning and Cybernetics,* Qingdao, p. 1678-1683, 2010.

Haupt, J., Bajwa, W. U., Rabbat, M. and Nowak, R., “Compressed Sensing & Network Monitoring”, *Reprinted with permission of IEEE, originally published in IEEE Signal Processing Magazine,* pp 92-10, 2008.

He, W., Yang S. and Yang, L., “Real-time data mining methodology and emergency knowledge discovery in wireless sensor networks", *PGNet,* 2010.

He, W., Yang, S. and Yang, L., "NMCA: Neighbour-aware Multi-path Clustering Aggregation in Wireless Sensor Networks", *In Networking, Sensing and Control (ICNSC)' 10th IEEE International Conference,* 2013.

Heinzelman, W., Chandrakasan, A. and Balakrishnan, H., "An Application-specific Protocol Architecture for Wireless Micro Sensor Networks", *IEEE Transactions On Wireless Communications*, Vol. 1, No. 4, pp. 660-670, October 2002.

Hill, J., Szewczyk, R., Woo, A., Ciller, D., Hollar, S. and Pister, K., "System Architecture Directions for Networked Sensors*", ACM SIGPLAN Notices*, Vol. 35(11), pp.93-104, No. 2000.

Intanagonwiwat, C., Govindan, R. and Estrin, D., "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks". *In Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, Boston, Massachusetts, ACM Press,* pp. 56-67, August, 2000.

Janakiram, D., Adi, V. and Reddy, M., "Outlier Detection in Wireless Sensor Networks Using Bayesian Belief Networks", *Communication System Software and Middleware, First International Conference*, 2006.

Jeffery, S. R. et al., "A Pipelined Framework for Online Cleaning of Sensor Data Streams", *In Proceedings of the 22nd International Conference on Data Engineering,* pp. 140-143, 2006.

Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L. and Rubenstein, D., "Energy Efficient Computing for Wildlife Tracking: Design Trade-Offs and Early Experiences with Zebranet", *In Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*

*(ASPLOS 2002),* pp. 96–107, 2002.

Kantardzic, M., “Data Mining: Concepts, Models, Methods, And Algorithms", *Wiley-Blackwell,* pp.1-343, 2003.

Knuth 1985 Knuth, D.E., “Dynamic Huffman Coding”, *Journal of Algorithms 6, 2(June),* 1985.

Krivitski, D. et al., “A Local Facility Location Algorithm for Large-Scale Distributed Systems”, *Journal of Grid Computing,* Vol. 5, No. 4, pp. 361–378, 2007.

Kumar, S. and Ilango P., “Data Funneling in Wireless Sensor Networks: A Comparative Study”, *Indian Journal of Science and Technology,* 2015.

Kulakov, A. and Davcev, D., “Distributed Data Processing in Wireless Sensor Networks Based on Artificial Neural-Networks Algorithms”, *In Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC '05), IEEE Computer Society,* pp. 353-358, 2005.

Kulakov, A. and Davcev, D., “Distributed Data Processing in Wireless Sensor Networks Based on Artificial Neural-Networks Algorithms”, *In Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC '05), IEEE Computer Society,* pp. 353-358, 2005.

Kusuma, J., Doherty, L. and Ramchandran, K., “Distributed Compression for Sensor Networks,” *In Proceedings of 2001 International Conference on Image Processing,* Oct. 2001.

Leopold, G., "U.S. Wind Power Is Rising, But Will It Fly?" *EE Times*, 2007.

Lewis, F.L. “Wireless Sensor Networks”, *D.J. Cook, S.K. Das, Editors, Smart Environments: Technology, Protocols, and Applications, Wiley*, 2004.

Lindsey, S., Raghavendra, C. and Sivalingam, K. M., "Data Gathering Algorithms in Sensor Networks using Energy Metrics," *IEEE Trans. Parallel Distributed. System,* Vol. 13, No. 9, pp. 924–35, Sept. 2002.

Lu, J., Valois, F., Dohler, M. and Wu, M.Y., "Optimized Data Aggregation in WSNs Using Adaptivearma", *In Proceedings of the International Conference on Sensor Technologies and Applications. IEEE Computer Society,* pp. 115–120, 2010.

Madden, S., Franklin, M. J., Hellerstein, J. M. and Hong, W., "TAG: A Tiny Aggregation Service for Ad Hoc Sensor Networks." *In Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI),* pp 131-146, 2002.

Madden, S., Franklin, M. J., Hellerstein, J. M. and Hong, W., "The Design of an Acquisitioned Query processor for Sensor Networks", *In Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data,* pp. 491-502, 2003.

Madden, S., Franklin, M. J., Hellerstein, J. M. and Hong, W., "TinyDB: An Acquisition Query Processing System For Sensor Networks", *ACM Transactions on Database Systems (TODS),* Vol. 30(1), pp. 122-173, 2005.

Mahimkar, A. and Rappaport, T. S. "SecureDAV: A Secure Data Aggregation and Verification Protocol for Sensor Networks," *IEEE Globe com 2004,* Nov. 2004.

Mainwaring, A., Polastre, J., Szewczyk, R., Culler, D. and Anderson, J., "Wireless Sensor Network for Habitat Monitoring", WSNA'02, Atlanta, Georgia, USA, Sep.28, 2002.

Mao, Y. et al., "An Integrated Data Mining Approach to Real-time Clinical Monitoring and Deterioration Warning", *Washington University in St. Louis,* 2012.

Maraiya, K., Kant, K. and Gupta, N., "Architecture Based Data Aggregation Techniques in Wireless Sensor Network: A Comparative Study", *International Journal on Coumpyer Science and Engineering,* Vol. 3, No. 3, pp. 1131-1138, 2011.

Martinez, K., Elsaify, A., Zou, G., Padhy, P. and Hart, J., "A Low Power Reliable Sensor Network for Glacier Deployments", *At Proc. 4th European Conference on Wireless Sensor Networks (EWSN 2007),* 2007.

Manjhi, A., Nath, S. and Gibbons, P. B., "Tributaries and Deltas: Efficient and Robust Aggregation in Sensor Network Stream," *ACM SIGMOD 2004*, June 2004.

McConnell, S. M. and Skillicorn, D.B., "A Distributed Approach for Prediction in Sensor Networks", *1st International Workshop on Data Mining in Sensor Networks as part of the SIAM International Conference on Data Mining, SIAM Press*, pp. 28–37, 2005.

Md, S. and Koo, I., "Performance Analysis of LEACH and LEACH Protocols for Wireless Sensor Networks", *In Journal of information and communication convergence engineering*, p. 384-389, 2012.

Mhatre, V. and Rosenberg, C., "Homogeneous vs heterogeneous clustered sensor networks: a comparative study", *Communications 2004 IEEE International Conference*, Vol. 6, pp. 3646–3651, 2004.

Nath, S. et al., "Synopsis Diffusion for Robust Aggregation in Sensor Networks," *ACM SenSys 2004*, Nov. 2004.

Oguejiofor, O.S., Aniedu, A.N., Ejiofor, H.C. and Okolibe, A.U., "Trilateration Based localization Algorithm for Wireless Sensor Network", *International Journal of Science and Modern Engineering (IJISME),* ISSN: 2319-6386, Vol. 1, Issue 10, Sep. 2013.

Paradis, L. and Han, Q., "TIGRA: Timely Sensor Data Collection Using Distributed Graph Coloring", *Sixth Annual IEEE International Conference on Pervasive Computing and Communications*, 2008.

Petrovic, D., Shah, R. C., Ramchandran, K. and Rabaey, J., "Data Funneling: Routing with Aggregation and Compression for Wireless Sensor Networks," *In Proceedings of First IEEE International Workshop on Sensor Network Protocols and Applications,* May 2003.

Pradhan, S.; Kusuma, J. and Ramchandran, K. "Distributed Compression in a Dense Microsensor Network", IEEE *Signal Processing Magazine,* Vol. 19, Issue 2, pp. 51-60, March 2002.

Qi, H., Kurugani, P. and Xu, Y., "The Development of Localized Algorithms in Wireless Sensor Networks", *Sensors 2002(2),* pp.286-293, 2002.

Ratnasamy, S. B., Karp, S. and Estrin, D., "Data-Centric Storage in Sensornets with GHT, A Geographic Hash Table" *Mobile Networks and Applications (MONET),* Special Issue on Wireless Sensor Networks, Kluwer, 2003.

Rezgui, A. and Eltoweissy, M., "Service-oriented Sensor-actuator Networks: Promises, Challenges, and the Road Ahead", *Journal of Computer Communications,* Vol. 30, Issue 13, pp. 2627-2648, 2007.

Romer, K., "Discovery of Frequent Distributed Event Patterns in Sensor Networks", *Wireless Sensor Networks, 5th European Conference, EWSN 2008,* Vol. 4913, pp. 106-124, 2008.

Sadagopan, N., Krishnamachari, B. and Helmy, A., "Active query forwarding in sensor networks", Elsevier Journal of Ad Hoc Networks, 2003.

Sasirekha, S. and Swamynathan, S., "A Comparative Study and Analysis of Data Aggregation Techniques in WSN", *Indian Journal of Science and Technology*

*8(26)*, 2015.

Saywood, K., “Introduction to Data Compression”, *3rd Edition, Elsevier,* 2004.

Sheth, A. P. “Changing Focus on Interoperability in Information Systems: From System, Syntax”, *In Interoperating Geographic Information Systems Conference,* pp. 5–30, 1999.

Seo, Y., Song, J., et al., “Hospital-Based Influenza Morbidity and Mortality Surveillance System for Influenza-Like Illnesses: A Comparison with National Influenza Surveillance Systems”, *Influenza and Other Respiratory Viruses,* 2013.

Shnayder, V., Hempstead, M., Chen, B., Allen, G. W. and Welsh, M., “Simulating the power consumption of large- scale sensor network”, *SenSys ’04,* pp. 191, 2004.

Silberstein, A. et al., “Data-Driven Processing in Sensor Networks”, *In Proceedings of the 3rd Biennial Conference on Innovative Data Systems Research (CIDR),* pp. 7-10 Jan. 2007.

Slepian, D. and Wolf, J. K., “Noiseless Coding of Correlated Information Sources,” *IEEE Trans. on Information Theory,* Volume: IT-19, pp. 471-480, July 1973.

Solis, I. and Obraczka, K., “Isolines: Energy-Efficient Mapping in Sensor Networks”, *IEEE ISCC ’05,* Cartagena, Spain, June 2005.

Subramanian, R. and Fekri, F., “Sleep Scheduling and Lifetime Maximization in Sensor Networks: Fundamental Limits and Optimal Solutions”, In Proceedings of the 5th International Conference on Information Processing in Sensor Networks. ACM, pp. 218–225, 2006.

Tan, Y. L. et al., “SensoClean: Handling Noisy and Incomplete Data in Sensor Networks using Modelling”, Technical Report, University of Maryland, http://www.cs.umd.edu/~viveks/finalreport.pdf, 2010.

Tanenbaum, A.S., “Computer Networks” *Fourth Edition, Prentiee Hall,* 2003.

Tharini, C. and Ranjan, P., “Design of Modified Adaptive Huffman Data Compression Algorithm for Wireless Sensor Network”, *In Journal of Computer Science 5 (6),* pp. 466-470, 2009.

Tulone, D. and Andmadden, S., “Time Series Forecasting For Approximate Query Answering In Sensor Networks”. *In Proceedings of the 3rd European Workshop on Wireless Sensor Networks (EWSN),* pp.21–37, 2006.

Underground 2012, http://www.wunderground.com, 2012.

Vitter, J.S., “Design and analysis of dynamic Huffman codes”, *J. ACM 34, 4 (Oct.),* pp. 825-845, 1987.

Wagenknecht, G., Anwander, M., Braun, T., Staub, T., Matheka, J. and Morgenthaler, S., “MARWIS: A Management Architecture for Heterogeneous Wireless Sensor Networks”, *Wired/Wireless Internet Communications,* Vol. 5031, pp. 177-188, 2008.

Walpola, M., “Energy-Efficient Self-Organization of Wireless Acoustic Sensor Networks for Ground Target Tracking”, *Florida International University*, 2009.

Yao, Y. and Gehrke, J., “Query Processing for Sensor Networks”, *ACM CIDR 2003 Conference,* Jan. 2003.

Yao, Y. and Gehrke, J., “The Cougar Approach to In-network Query Processing in Sensor Network”, *SIGMOD record,* Vol. 31, No.3, September 2002.

Yu, J. and Taylor, K., “Event dashboard: Capturing user-defined semantics events for event detection over real-time sensor data”, *SSN ISWC,* pp. 19–34, 2013.

Yu, Y., Krishnamachari, B. and Prasanna, V., “Energy-Latency Trade-offs for Data Gathering in Wireless Sensor Networks,” *IEEE INFOCOM ’04,* Mar. 2004.

Zellner, A., Keuzenkamp, H. A., and McAleer, M. (Eds.), “Simplicity, Inference and Modelling: Keeping it Sophisticatedly Simple”, *Cambridge University Press*, 2001.

Zhang, Y., Meratnia, N. and Havinga, P. J. M., "Outlier Detection Techniques for Wireless Sensor Network: A survey", *IEEE Communication Survey & Tutorials,* Vol. 2 12, pp. 159-170. 2010.

Zhao, Y. J.; Govindan, R. and Estrin, D. “Computing Aggregates for Monitoring Wireless Sensor Networks”, *The First IEEE International Workshop on Sensor Network Protocols and Applications (SNPA’ 03),* May 11, 2003.

Zhou, B. et al., “A Hierarchical Scheme for Data Aggregation in Sensor Network,” *IEEE ICON ’04 Conference,* Nov. 2004.

Zhuang, Y. and Chen, L., “In-Network Outlier Cleaning for Data Collection in Sensor Networks”, *CleanDB Workshop,* pp. 41-48, 2006.

Zhuang, Y. et al., “A Weighted Moving Average-based Approach for Cleaning Sensor Data”, *In Proceedings of the 27th International Conference on Distributed Computing Systems,* pp. 38-45, 2007.

# Appendix

## I. The process of finding the first Anchor

```
int testapprun(instance_data_t *inst, int message)
{
        switch (inst->testAppState)
        {
                case TA_INIT :
                        // printf("TA_INIT") ;
                        switch (inst->mode)
                        {
                                case TAG:
                                {
                                          dwt_enableframefilter(DWT_FF_DATA_EN | DWT_FF_ACK_EN); //allow
data, ack frames;
                                        dwt_setpanid(inst->panid);
                                          dwt_seteui(inst->eui64);
#if (USING_64BIT_ADDR==0)
                                        {
                                                uint16 addr = inst->payload.tagAddress & 0xFFFF;
                                                dwt_setaddress16(addr);
                                        }
#endif
                                        //set source address into the message structure
                                          memcpy(&inst->msg.sourceAddr, inst->eui64, ADDR_BYTE_SIZE);
                                        //change to next state - send a Poll message to 1st anchor in the list
#if (DR_DISCOVERY == 1)
                                          inst->mode = TAG_TOF ;
                                        inst->testAppState = TA_TXBLINK_WAIT_SEND;
                                          memcpy(inst->blinkmsg.tagID, inst->eui64, ADDR_BYTE_SIZE);
#else
                                        inst->testAppState = TA_TXPOLL_WAIT_SEND;
#endif
                                        dwt_setautorxreenable(inst->rxautoreenable); //not necessary to auto RX
re-enable as the receiver is on for a short time (Tag knows when the response is coming)
#if (DOUBLE_RX_BUFFER == 1)
                                        dwt_setdblrxbuffmode(0); //disable double RX buffer
#endif
#if (ENABLE_AUTO_ACK == 1)
```

```
				dwt_enableautoack(ACK_RESPONSE_TIME); //wait for 100 symbols before
replying with the ACK
#endif
```

## II.	A ranging algorithm

```
			uint64 tagCalculatedFinalTxTime ;
			// Embbed into Final message: 40-bit pollTXTime,    40-bit respRxTime,    40-bit
finalTxTime
			// Write Poll TX time field of Final message
			memcpy(&(inst->macdata_msdu[PTXT]), (uint8 *)&inst->txu.tagPollTxTime, 5);
			// Write Response RX time field of Final message
			memcpy(&(inst->macdata_msdu[RRXT]), (uint8 *)&inst->rxu.anchorRespRxTime,
5);
			// Calculate Time Final message will be sent and write this field of Final message
			// Sending time will be delayedReplyTime, snapped to ~125MHz or ~250MHz
boundary by
			// zeroing its low 9 bits, and then having the TX antenna delay added
			tagCalculatedFinalTxTime = inst->delayedReplyTime & MASK_TXDTS; // 9 lower
bits mask
			// getting antenna delay from the device and add it to the Calculated TX Time
			tagCalculatedFinalTxTime = tagCalculatedFinalTxTime + inst->txantennaDelay;
			tagCalculatedFinalTxTime &= MASK_40BIT;
			// Write Calculated TX time field of Final message
			memcpy(&(inst->macdata_msdu[FTXT]), (uint8 *)&tagCalculatedFinalTxTime, 5);
			//set destination address
			memcpy(&inst->msg.destAddr, &inst->relpyAddress, ADDR_BYTE_SIZE);
			setupmacframedata(inst, TAG_FINAL_MSG_LEN,
RTLS_DEMO_MSG_TAG_FINAL, !ACK_REQUESTED);
#if (DEEP_SLEEP == 1)
			if(inst->tag2rxReport==0) //if not going to wait for report, go to sleep after TX is
complete
			{
				dwt_entersleepaftertx(1);
				dwt_setinterrupt(DWT_INT_TFRS, 0); //disable all the interrupts (wont be
able to enter sleep if interrupts are pending)
			}
#endif
			if(instancesendpacket(inst, DWT_START_TX_DELAYED))
			{
				//error - TX FAILED
				inst->txu.txTimeStamp = 0;
```

```
                    // initiate the re-transmission
                    inst->testAppState = TA_TXE_WAIT ;
                    inst->nextState = TA_TXPOLL_WAIT_SEND ;
#if (DEEP_SLEEP == 1)
                    dwt_entersleepaftertx(0);
#endif
                    break; //exit this switch case...
                }
                else
                {
                    stateCount = 0;
                    inst->testAppState = TA_TX_WAIT_CONF;
// wait confirmation
                    inst->previousState = TA_TXFINAL_WAIT_SEND;
                    inst->done = INST_DONE_WAIT_FOR_NEXT_EVENT; //will use RX FWTO to
time out    (set below)
                }
                if(inst->tag2rxReport) //if waiting for report - set timeout to be same as a Sleep
timer... if no report coming time out and send another poll
                {
                    dwt_setrxtimeout(0);
                    inst->done = INST_DONE_WAIT_FOR_NEXT_EVENT_TO;
                }
                else //if Tag is not waiting for report - it will go to sleep automatically after the
final is sent
                {
#if (DEEP_SLEEP == 1)
                    inst->done = INST_DONE_WAIT_FOR_NEXT_EVENT_TO; //kick off the
TagTimeoutTimer (instancetimer) to initiate wakeup
                      inst->nextState = TA_TXPOLL_WAIT_SEND;
                    inst->testAppState = TA_SLEEP_DONE; //we are going automatically to sleep
so no TX confirm interrupt (next state will be TA_SLEEP_DONE)
                    inst->txmsgcount ++;
#endif
```

## III. Compressed data transmission

```
void vSendData(uint8 u8Flag, uint8 *u8Src, u8Len)
{
    uint8 i;
    // Clear output buffer
    vClearBuffer(u8OutputBuffer,MAX_OUTPUT_BUFF_LEN);
```

```
        u8OutputBuffer[0] = u8Flag;    //ORIGINAL_DATA or COMPRESSED_DATA
        u8OutputBuffer[1] = u8Len;      // Length is specified by application, can be
ORIGINAL_DATA_LEN, or COMPRESSED_DATA_LEN
        // If it is for original data, simply copy data into buffer
        if(u8OutputBuffer[0] == ORIGINAL_DATA)
        {
                // If the input data is original data
                for(i=0; i<DATA_LEN; i++)
                {
                        u8OutputBuffer[2+i*2] = u8Src[i*2];
                        u8OutputBuffer[2+i*2+1] = u8Src[i*2+1];
                }
        }else if (u8OutputBuffer[0] == COMPRESSED_DATA)
        {
                // If the output data requires compression
                vFillCompressData(u8Src, &u8OutputBuffer[2],u8Len-1);
                // Update huffman table
                vUpdateHuffmanTable(u8Src);
        }
}
```

## IV. ODH (data compression algorithm)

```
void vFillCompressData(uint8 *u8Src,    uint8 *u8Dest, uint u8Len)
{
        uint8 i;
        uint8 u8CompressedData
        u8Dest[0] = 0x00;
        // Search HuffmanTable
        for(i=0; i<u8Len; i++)
        {
                if(locateHuffmanValue(u8Src[i], (u8Dest+i))!=-1)
                {
                        // It is an existing value
                        u8Dest[0] = u8Dest[0] | 0x00<i;
                }else{
                        // Desired data is not located, it is a real offset data (i.e. non-huffman data)
                        u8Dest[0] = u8Dest[0] | 0x01<i;
                }
        }
}
int locateHuffmanValue(uint8 u8OffsetData, uint8 *u8CompressedData)
```

```
{
        uint8 i;
        for(i=0; i<HUFFMAN_LEN; i++)
        {
                if(u8PracticalValue[i] == u8OffsetData)
                {
                        *u8CompressedData = u8HuffmanTable[i];
                        break;
                }
        }
        if(i<HUFFMAN_LEN) return 0;
        if(i>=HUFFMAN_LEN)
        {
                // huffman data is not located, directly copy offset data
                // to output buffer
                *u8CompressedData = u8OffsetData;
        }
        return -1;
}
void vUpdateHuffmanTable(uint8* u8Src)
{
        uint8    i,m;
        for(i=0; i<4; i++)
        {
                if(isExistingValue(u8Src[i]))
                {
                        // Data already exists, just swap it with the data ahead.
                        swapPracticalData(u8Src[i]);
                }else{
                        // New value
                        insertPracticalData(u8Src[i]);
                }
        }
}
void insertPracticalData(uint8 u8Data)
{
        uint8 i;
        for(i=(HUFFMAN_LEN-2);i>(HUFFMAN_LEN/2);i--)
        {
                u8PracticalValue[i+1] = u8PracticalValue[i];
        }
u8PracticalValue[HUFFMAN_LEN/2]=u8Data;
}
void swapPracticalData(uint8 u8Data)
```

```
{
        uint8 i;
        uint8 u8TempData
        for(i=0; i<HUFFMAN_LEN; i++)
        {
                if(u8PracticalValue[i] == u8Data)
                {
                        break;
                }
        }
        // i<100    it is a valid data
        // i>0 it is not the very first data
        if(i<HUFFMAN_LEN && i>0)
        {
                u8TempData = u8PracticalValue[i-1];
                u8PracticalValue[i-1] = u8Data;
                u8PracticalValue[i] = u8TempData;
        }
}
uint8 isExistingValue(uint8 u8Data)
{
        uint8 ret,i;
        for(i=0; i<HUFFMAN_LEN; i++)
        {
                if(u8PracticalValue[i] == u8Data)
                {
                        return 1;
                }
        }
        return 0;
}
void vClearBuffer(uint8 *dest, uint8 u8Len)
{
        uint8 i;
        for(i=0; i<u8Len; i++)
                dest[i]=0x00;
}
```

## V. Decode in the gateway node

```
uint16 u16SensorReading[SENSOR_READING_LEN];
uint8 u8ProcessDataPacket(uint8 *ptrPacket, uint8 u8Len)
```

```
{
        uint8 *workPtr, u8PacketType,u8PacketLen,u8DataFlag,i;
        u8PacketLen = 0;
        // Record start position of packet
        workPtr = ptrPacket;
        if(u8Len < 3) return 1;
        u8PacketType = workPtr[0];
        u8PacketLen = workPtr[1];
        u8DataFlag = workPtr[2];
        if((u8PacketLen+3)!=u8Len) return 1;
        switch(u8PacketType)
        {
                case DATAREADING:
                         // "Data reading" is for delivering compressed data
                         u8RecoverData(workPtr+2,u8PacketLen,DATAREADING);
                         // Update huffman table (re-use update method employed at sender side
                        vUpdateHuffmanTable(workPtr+2);
                break;

                case ORIGINALDATA:
                         // It is the very first data (i.e. long data)
                         vRecoverData(workPtr+2,u8PacketLen,ORIGINALDATA);
                         // After processing, report sensor reading newly generated
                         print(...u16Data);
                break;
                default:
                break;
        }
        return 0;
}
void u8RecoverData(uint8* u8Ptr, uint8 u8Len, uint8 u8DataType)
{
    uint8 i;
    uint8 u8HuffmanData, u8DataFlag, u8PracticalData,index, operator;
        if(u8DataType == DATAREADING)
        {
                // Get data flag from the received data
                u8DataFlag = u8Ptr[0];
                // Use "datareading" as offest to update sensor reading
                for(i=0; i<SENSOR_READING_LEN;i++)
                {
                        u8HuffmanData = u8Ptr[1+i];
                        if(((u8DataFlag>i)&0x01)==0)
                        {
```

```
                    // It is an existing huffman data
                    index = u8LocatePracticalDataIndex(u8HuffmanData);
                    u8PracticalData = u8PracticalValue[i];
                    operator = (u8PracticalData>>7)&0x01;
                }else{
                    // It is a new data. The data received is not a haffman data, but a real offset
                    operator = (u8HuffmanData>>7)&0x01;
                    u8PracticalData = u8HuffmanData;
                }
                if(!operator)
                        u16SensorReading[i] = u16SensorReading[i]- (u8PracticalData&0x7f);
                else
                        u16SensorReading[i] = u16SensorReading[i]+ (u8PracticalData&0x7f);

            }
    }else if (u8DataType == ORIGINALDATA)
    {
            for(i=0; i<SENSOR_READING_LEN; i++)
            {
                u16SensorReading[i] = u8Ptr[1+i*2];
                u16SensorReading[i] = u16SensorReading[i] << 8;
                u16SensorReading[i] = u16SensorReading[i] | u8Ptr[1+i*2+1];
            }
    }
}
uint8 u8LocatePracticalDataIndex(uint8 u8HuffmanData)
{
    uint8 i;

            for(i=0; i<;i++)
            {
                if(u16HuffmanTable[i] == u8HuffmanData)
                    return i;
            }
    return 0;
}
```