A Cross-Layer Design for Sensor-based Ambient Intelligence Systems

Yang Liu

A thesis submitted to

Auckland University of Technology

In fulfilment of the requirements for the degree of

Doctor of Philosophy (PhD)

2014

School of Engineering

Auckland University of Technology

New Zealand

Declaration

I, Yang Liu, hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university of institution of higher learning.

Signature   Date

Yang Liu

Abstract

The wireless sensor network (WSN) is an enabling technology of ambient intelligence (AmI) where an intelligent system can sense the presence of and respond to the context or situations of people in the environment. AmI relies on the massive deployment of interconnected and distributed sensor devices to provide personalised services via intuitive interfaces and natural interactions in a manner consistent with the user contexts.

Cross-layer approaches have been widely used for WSN management and play an important role in designing solutions for protocol optimisation. The cross-layer approaches allow the sharing of information in a protocol stack across different layers for significant improvements on network performance and efficiency.

After an extensive literature review, it emerges that there exists research opportunities on cross-layer designs for WSNs in context-aware systems. Therefore, the research presented in this thesis is to develop a cross-layer optimisation approach for WSNs by utilising the user and environment context information from an AmI system. This approach can provide the resource-constrained sensor devices with the capability to understand the situations of their surroundings for the purpose of optimising WSN communications.

The thesis is structured into eight chapters. The first three chapters provide the introduction, background, and literature review. The fourth chapter proposes a network partitioning and data storage formation mechanism for WSNs based on the Anchor-Free Localization (AFL) algorithm. Optimal parameters of this mechanism are derived based on the network size. With the derived parameters, this mechanism can organise a WSN into balanced partitions and construct in-network data storage units. The data storage units provide the in-network data exchange ability to the sensor devices and AmI applications.

In Chapter Five, a publish/subscribe (pub/sub) based data-centric communication mechanism is proposed. This bi-directional communication model does not only allow the sensor devices to publish and enable the AmI applications to subscribe to the sensor data, but also let the AmI applications to publish their inferred context information that can be subscribed to by the sensor devices. In addition, the mechanism for the storage units to handle the publishing and subscription procedures of both the sensor data and AmI context information is proposed. In

Chapter Six, a generic ontology-based context modelling and reasoning approach is proposed for context-aware WSN management. This approach can model both the network- and application-related context information. In addition, the context information can be categorised according to where and how it has been inferred. Using the contextualised information can allow WSN management to become context-aware, whereby the management tasks can be performed and optimised according to the context information of both the WSNs and AmI systems. The seventh chapter proposes a context-aware cross-layer protocol optimisation mechanism according to AmI context information. This mechanism includes a generic framework for managing and handling context information for WSNs. This framework is then applied to two existing protocols on the medium access control and network layers for joint protocol optimisations by utilising the context information. Through the evaluations, the proposed optimisation approach improves network performance and efficiency in terms of throughput, delivery ratio, delay and energy cost. Chapter Eight concludes the thesis.

Acknowledgements

It would not have been possible to complete this thesis without the help and support of number of people.

I would like to sincerely express my gratitude to my supervisors Dr Boon-Chong Seet and Professor Adnan Al-Anbuky for their guidance and inspiration over the past couple of years. Without their constant supervision, encouragement and great support throughout the years, I would not have been where I am now.

I am grateful to all members of the Centre for Sensor Network & Smart Environment (SeNSe) for their generosity in sharing knowledge and experience in work and life.

I would like to thank Ms Sarah Grimes for proofreading of this thesis.

I would like to acknowledge the financial support I have received during the work. This work has been supported by AUT Tuition Fee Scholarship and AUT University Graduate Assistantship Award.

Finally, I must thank my family, especially my wife Ting, and friends for their understanding and for believing in me.

Table of Contents

List of Figures

List of Tables

Glossary

ACK	—	Acknowledgement
ADC	—	Analog-to-Digital Converter
AFL	—	Anchor-Free Localization
AH	—	Aggregator Head
AmI	—	Ambient Intelligence
AODV	—	Ad hoc On-Demand Distance Vector
APP	—	Application
CACH	—	Context-Aware Clustering Hierarchy
CA-MAC	—	Context-Aware MAC
CAMS	—	Context-Aware Multi-path Selection
CCA	—	Clear Channel Access
CH	—	Cluster Head
CL	—	Cross-Layer
CS	—	Context Storage
CSMA/CA	—	Carrier Sense Multiple Access / Collision Avoidance
CUA	—	Context User Agent
DAAM	—	Delay-Aware AODV-Multi-path
DBP	—	Derivative-Based Prediction
DD	—	Directed Diffusion
DEEC	—	Distributed Energy-Efficient Clustering Algorithm
DHCS	—	Dual Head Clustering Scheme
DHT	—	Distributed Hash Table
DL	—	Description Logic
DR-MAC	—	Dynamic Reconfiguration MAC
DCXP	—	Distributed Context eXchange Protocol

EDL	—	Event Definition Language
EEHC	—	Energy Efficient Heterogeneous Clustered Scheme
EMA	—	Environmental Monitoring Aware
E-LEACH	—	Energy-LEACH
GHT	—	Geographic Hash Table
GPS	—	Global Positioning System
HMN	—	Heterogeneous Multi-tier Network
HSFC	—	Hilbert Space Filing Curve
IEEE	—	Institute of Electrical and Electronics Engineers
INS	—	In-Network Storage
ISM	—	Industrial Scientific Medical
IST	—	Information Society Technologies
ISTAG	—	IST Advisory Group (European Commission)
LEACH	—	Low Energy Adaptive Clustering Hierarchy
LUCA	—	Location-based Unequal Clustering Algorithm
K-NNG	—	K-Nearest Neighbour Graph
KOCA	—	K-hop Overlapping Clustering Algorithm
LR-WPANs	—	Low-Rate Wireless Personal Area Networks
MAC	—	Media Access Control
MB	—	Mobile Base-station
MCCD	—	Multi Criteria Context-based Decision
MIMO	—	Multiple-Input Multiple-Output
MPMP	—	Multi-Path multi-Priority
NET	—	Network (layer)
NMS	—	Network Management System
OSI	—	Open System Interconnection
OWL	—	Web Ontology Language
PHY	—	Physical (layer)
PRESTO	—	Predictive Storage

PSO-DH	—	Double cluster-Head clustering using Particle Swarm Optimization
Pub/Sub	—	Publish/Subscribe
QoS	—	Quality of Service
ROI	—	Region of Interest
RERR	—	Route Error
RREP	—	Route Reply
RREQ	—	Route Request
SARIMA	—	Seasonal Autoregressive Integrated Moving Average model
SM-AODV	—	Service-oriented Multipath AODV
TDMA	—	Time Division Multiple Access
UCI	—	Universal Context Identifier
VB	—	Virtual Broker
VBN	—	Virtual Broker Node
VCH	—	Vice Cluster Head
VLSI	—	Very Large-Scale Integration
VSU	—	Virtual Storage Unit
Warp-5	—	Wireless Adaptive Routing Protocol, Version 5
WSN	—	Wireless Sensor Network
XML	—	Extensible Markup Language

Chapter 1 Introduction

The vision of Ambient Intelligence (AmI) is a new and interdisciplinary paradigm where technologies become invisible and hidden in our daily environments. Through intuitive and intelligent interfaces, an AmI system can sense and respond according to the context of people and their behaviour [1].

In recent years, wireless sensor networks (WSNs) have emerged as an enabling technology for AmI. WSNs can address the information needs of AmI applications by providing a flexible underlying infrastructure for information sensing and gathering. The rapid development of Very Large-Scale Integration (VLSI) chip design and micro-fabrication techniques over the last decades have made WSN devices capable of accomplishing more complex tasks. Given their small physical size, they can be easily embedded into objects in our daily life. By incorporating a group of those devices, a large amount and variety of important information about the monitored environment can be wirelessly collected and processed by applications that are running on top of the WSNs.

Data-centric communication in WSNs is a novel paradigm whereby WSN devices can be identified by their sensor data content instead of their addresses [2]. This is a key feature to distinguish WSNs from other types of wireless communication technologies [3]. Recent advances in data storage technologies have enabled in-network storage for WSN applications, where sensor devices can cooperatively store and maintain sensor data within, instead of externally to, the WSN [4]. By adapting this in-network storage approach, an AmI system can become more distributed, less dependent on the centralised controls, and be able to provide more personalised services to meet the unique requirements of the users.

The data-centric characteristic of WSNs raises new challenges to the design and optimisation of communication protocols. Common restrictions – for example resource-constrained sensor devices, short transmission range and maintenance-free – must be considered when designing protocols in WSNs for a wide variety of applications and services.

Protocol architectures in WSNs mostly follow strict layered principles. Under a layered principle, a protocol is normally designed for a particular layer, and implemented by either software or firmware. A protocol in a layer can only communicate with its immediate adjacent layers, and each protocol can be implemented independently of other layers. Such modularity is a key feature in layered principles, which promotes interoperability and allows simple and fast protocol implementation and deployment.

Cross-layer interaction between protocols in non-adjacent layers is observed as a violation of the layered principles. However, it can potentially address some of the performance limits and inefficiencies of communication under layered principles [5, 6].

This thesis argues that WSNs can be improved through protocol optimisation via cross-layer interaction, and proposes a cross-layer design that adapts WSN protocol parameters to AmI context information. To realise this objective, the methods for modelling context, exchanging context and context adaptation must be conceived.

1.1 Motivation and Scope

The cross-layer design approach offers a greater opportunity and potential over standalone or alternative methods for optimising protocol performance and efficiency in WSNs.

An important requirement of an AmI system is the awareness of the situation or context of its surrounding area. Context awareness is a prerequisite for systems to have the ability to self adapt to changing user behaviour or environmental situations [7]. As the common underlying infrastructure, WSN communications are an important aspect for context awareness in AmI by allowing sensor data to be collated for processing into higher-level context information.

Until lately, context adaptation has not been considered in cross layer designs. The basic question that this thesis aims to answer is how to effectively optimise WSN protocols by exploiting AmI context via cross-layer interactions. Of interest is in understanding what AmI context information is useful for WSNs, and how they can be disseminated and interpreted by WSN nodes for protocol optimisation via cross-layer interactions.

Figure 1.1 conceptually illustrates the mutually-supportive roles of WSN and AmI and the synergistic information flow between them: raw sensor data is collected from WSN nodes by AmI systems for contextualising into information useful to human-centric AmI applications. In turn, this context information is disseminated by AmI systems to WSN nodes to support their protocol optimisation through cross-layer interaction.

For instance, context awareness is achieved in smart home environments for facilitating intelligent home automation services by gathering and contextualising raw data from distributed sensors. On the other hand, by providing feedback loops from the upper level context-aware systems, it is possible to provide underlying networked devices with abilities to understand system-wide situations and high-level events in the monitored environments. With such opportunities, network control and management tasks, such as network structuring and sensor duty scheduling, could be realised more intelligently according to the inferred and exchanged AmI context information.

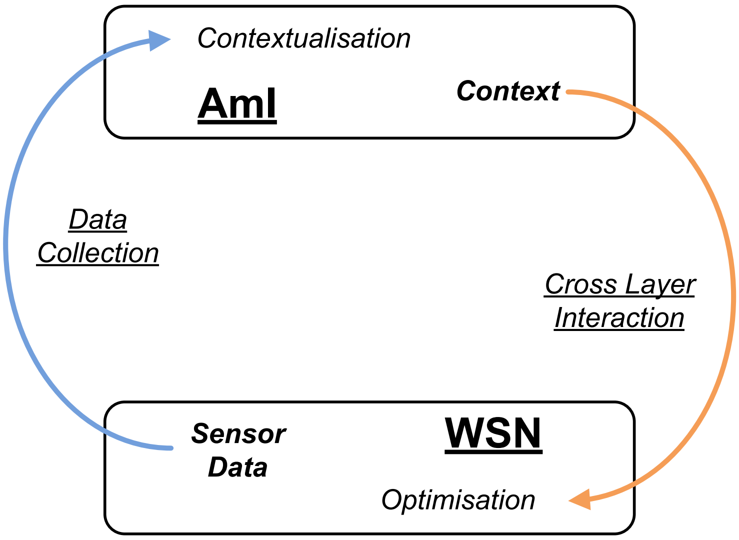


Figure 1.1 Information flow between WSNs & AmI

1.2 Contributions

This thesis has three major contributions, which are outlined as follows:

- **Network structuring for data-centric communication in WSNs** – a Publish/Subscribe (Pub/Sub) based communication architecture is proposed in this thesis. A novel concept of Virtual Broker (VB), which is inspired by the Virtual Multiple-Input Multiple-Output (MIMO) technology [8], is presented. A VB is a virtual grouping of co-located ordinary sensor nodes that jointly provide the brokerage functionality for Pub/Sub in WSN. Under this design, both sensor nodes and AmI systems can publish and subscribe to their information on the VBs, as shown in Figure 1.2.

Firstly, a set of optimal formulae is derived from an analytical study to determine the optimal network structure in terms of the number of partitions and VB size that minimise the communication overhead of the WSN. Then, based on the relative location of the sensor nodes in the WSN, a mechanism is designed for splitting the WSN into balanced partitions and forming VBs in each partition. Another important contribution is the development of a Pub/Sub mechanism for VB-based in-network storage (INS) of the data published and subscriptions.

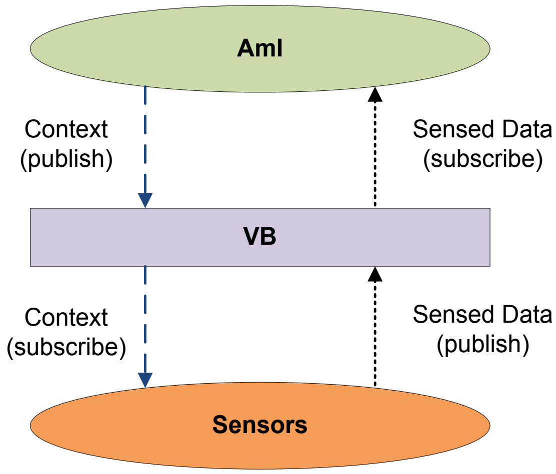


Figure 1.2 Publish and Subscribe over VB

- **Ontology based context modelling and reasoning** – this thesis also proposes a generic ontology-based model for modelling a range of contexts relevant to WSNs and AmI. Under this model, context information can be classified as low or high level context. Low level context is one that can be inferred directly from multiple raw sensor data, while high level context can only be deduced from multiple low level context. The context information can also be further classified as local or global context, depending on whether it can be deduced from information locally within a node, or only from that received externally. A reasoning mechanism is also developed for this context model. As a use case study, the Context-Aware Multi-path Selection (CAMS) algorithm [9] is modified and adapted to utilise the ontology based context information. This model shows that only minimum modifications to an algorithm are required, i.e. the original CAMS algorithm, for context adaptation.

- **Context-aware cross-layer protocol optimisation** – finally, this thesis presents a novel context-aware cross-layer protocol optimisation framework, which enables a sensor node to optimise the parameters of its WSN protocols based on AmI context. Using the modelled context information from the previous ontology-based model, an innovative approach to joint optimisation via cross layer interaction between the Medium Access Control (MAC) and network layers is presented. Two existing WSN protocols: Dynamic Reconfiguration MAC (DR-MAC) [10], and Delay-Aware AODV-Multi-path (DAAM) routing [11], are chosen for optimization using the proposed framework. Results showed that both network performance and efficiency are improved with this context-aware cross-layer optimisation design.

The following is a list of publications generated during the period of this research:

Yang Liu, Boon-Chong Seet, and Adnan Al-Anbuky. "Ambient Intelligence Context-Based Cross-Layer Design in Wireless Sensor Networks." Sensors, vol. 14, no. 10, pp. 19057-19085, 2014.

Yang Liu, Boon-Chong Seet, and Adnan Al-Anbuky. "AmI Context-based Cross-Layer Optimization of MAC Performance in WSNs." Proc. 1st International Electronic Conference on Sensors and Applications, June, 2014.

Yang Liu, Boon-Chong Seet, and Adnan Al-Anbuky. "An Ontology-Based Context Model for Wireless Sensor Network (WSN) Management in the Internet of Things." Journal of Sensor and Actuator Networks, vol. 2, no. 4, pp. 653-674, 2013.

Yang Liu, Boon-Chong Seet, and Adnan Al-Anbuky. "In-Network Storage for Virtual Broker-based Publish/Subscribe in WSNs." Proc. IEEE International Symposium on Personal Indoor and Mobile Radio Communications, Sydney, Australia, September 2012.

Yang Liu, Boon-Chong Seet, and Adnan Al-Anbuky. "Virtual brokers for large-scale publish/subscribe in wireless sensor networks." Proc. IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, Hong Kong SAR, China, December 2010.

Yang Liu, Boon-Chong Seet, and Adnan Al-Anbuky. "Sustainable Living through Ambient Intelligence: A Bi-directional Publish-Subscribe System Using a Cross-Layer Design." (Poster), Proc. 2nd International Workshop on Sensor Networks and Ambient Intelligence, Hiroshima, Japan, December 2009.

Under Submission:

"Analytical optimization of WSN structure with virtual brokers for publish/subscribe systems." submitted to a journal, 2014.

1.3 Thesis Structure

The rest of the thesis is organised as follows:

Chapter 2 describes in detail the background of the research for this thesis. An overview on various aspects of WSNs, AmI, and cross-layer optimisation is presented.

Chapter 3 presents a literature review of related works. This chapter explores the literature relevant to the research, including topics on partitioning/clustering algorithms, ontology-based context modelling schemes, and context-aware cross-layer optimisation mechanisms.

Chapter 4 propose a network partitioning and virtual storage formation design for WSNs, with the aim of minimising the overall communication overhead. This design takes advantage of relative node localisation information based on the Anchor Free Localisation (AFL) algorithm [12]. The resulting partitioned network with virtual storages is used to support information exchange in the following chapters.

Chapter 5 presents a Pub/Sub based mechanism for VB-based INS. It describes the functions of data publishing and subscription on VBs, e.g. structured VSUs, in WSNs. The mechanism has a generic design, which can work with any data type. In this thesis, the mechanism is adapted to work with AmI context information, which is described in the next chapter.

Chapter 6 presents the design of the Ontology-based context modelling and reasoning scheme. This scheme can contextualise relevant information of AmI systems and their underlying WSNs to establish context representation for a cross-layer optimisation of the WSN in AmI.

Chapter 7 presents a context-aware design for joint MAC and network cross-layer optimisation in WSNs. In this design, protocol parameters of DR-MAC and DAAM routing algorithms are adapted to AmI context information. Evaluations and analysis are presented and discussed.

Chapter 8 concludes the thesis and discusses the future directions of the research.

Chapter 2 Background

2.1 Introduction

In this chapter, the background concepts of three technologies related to this thesis, namely Wireless Sensor Network (WSN), Cross Layer Optimisation, and Ambient Intelligence (AmI), are introduced. The literature review of the state-of-the-art research relevant to this thesis will be covered separately in Chapter 3.

2.2 Wireless Sensor Network

A WSN can consist of a large number of inexpensive and resource-constrained sensor devices or nodes, embedded in an indoor or outdoor environment to cooperatively detect certain phenomena or events occurring in the environment. Data will then be generated by the sensors and forwarded to some base station or sink at the edge of the network for storage and processing. Given their small physical size and wireless connection, sensor nodes are typically deployed densely for broad geographic coverage [13]. In recent years, WSNs have been considered for deployment in industry sectors such as manufacturing, agriculture, logistics/transportation, and home automation, to provide monitoring, surveillance, and object tracking services [14].

2.2.1 Wireless Sensor Hardware Platform

The hardware platform of a sensor node incorporates the components of sensing unit, radio transceiver, memory storage, microcontroller and energy source, as shown in Figure 2.1. Through an analogue-to-digital converter (ADC), a sensing unit can quantify the detected attributes of its surrounding environment for further processing by WSN applications [15]. Sensor data can be stored temporarily on the node's local memory, which can range from a few kilobytes to a dozen megabytes [16]. Radio transceiver of a sensor node operates in the unlicensed Industrial, Scientific, Medical (ISM) frequency bands at 868/915 MHz, and at 2.45 GHz, with data rates of 20–40 kbps, and 250 kbps, respectively [14]. These components are interconnected by data buses with a width of 8/16/32 bits, and controlled by a microcontroller with a clock speed of between 4–16 MHz (to as high as 180 MHz) [17]. A battery is a common energy source in WSNs [18, 19]. During sensor operation, the transceiver can consume the majority of the energy in a sensor node [18, 20], especially during active operations [21]. Other factors can also result in high power consumption, such as clocking microcontrollers at high speed [22] and long range transmission [23].

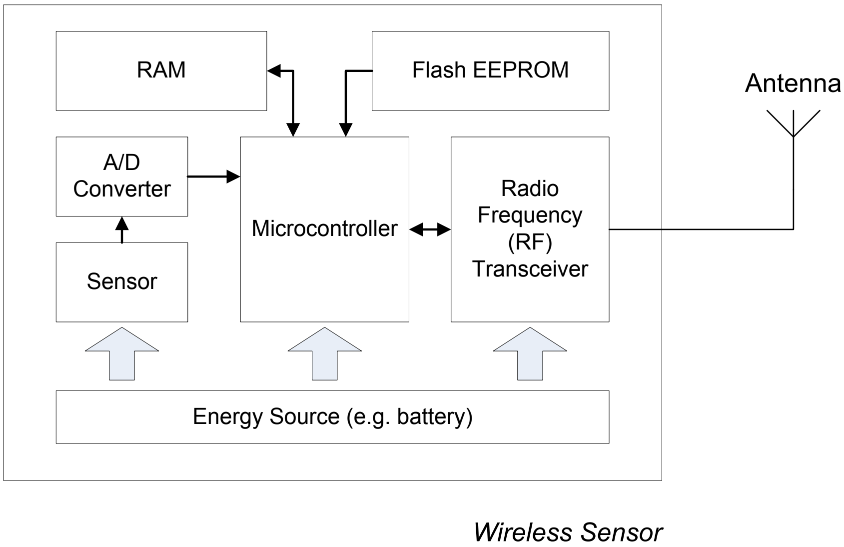


Figure 2.1 Wireless sensor hardware platform

2.2.2 Protocol Architecture in WSNs

The majority of communication designs in WSNs are still based on conventional layered protocol architecture, which follows the Open System Interconnection (OSI) model (ISO/IEC 7948-1) [24], as shown in Figure 2.2. Many protocols have been proposed for the lower three layers, i.e. the physical, medium access control (or data link), and network layers [25, 26].

This thesis focuses on the medium access control (MAC) and network layers. The following few paragraphs describe the unslotted carrier-sense multiple access with collision avoidance (CSMA/CA) MAC protocol of the IEEE 802.15.4 specification [27], and the ad-hoc on-demand distance vector (AODV) routing protocol on the network layer of the Zigbee specification [28].

Zigbee [28] is a communication technology for creating low-energy wireless mesh networks for low data rate applications. Zigbee is built on the physical layer and MAC layer specifications of the IEEE 802.15.4 standard. It employs AODV at the network layer for mesh routing, and defines the specifications for the application layer.

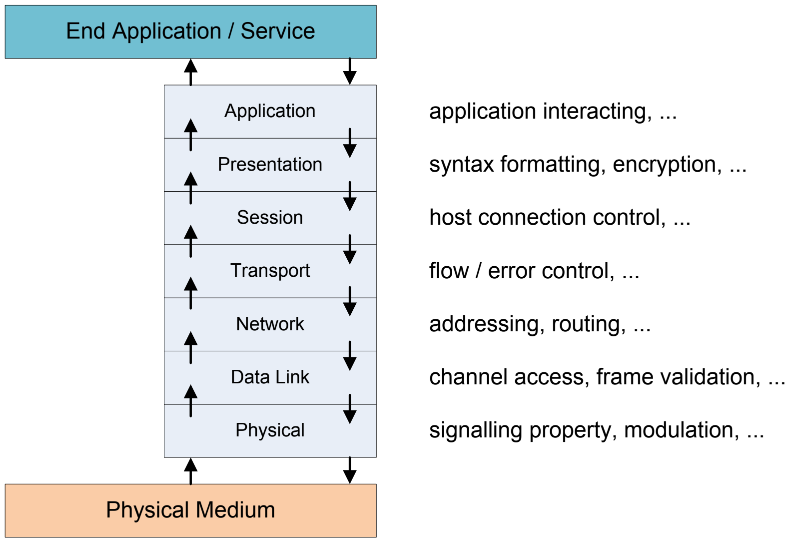


Figure 2.2 ISO/OSI 7-layer reference model (ISO/IEC 7498-1)

2.2.2.1 Unslotted CSMA/CA

The IEEE 802.15.4 standard [27] uses a contention based mechanism called CSMA/CA to access a shared medium, and can operate either as a slotted or an unslotted model. This thesis only focuses on the unslotted CSMA/CA channel access model, as slotted CSMA/CA will present an additional requirement for time synchronisation in WSNs. Figure 2.3 shows the unslotted CSMA/CA channel access model.

Similar to the IEEE 802.3 and IEEE 802.11 standards, the CSMA/CA algorithm tries to break the symmetry of simultaneous frame transmissions by performing random exponential backoffs to avoid frame collisions.

Table 2-1 summarises the parameters used by unslotted CSMA/CA. The corresponding frame structures for the MAC and acknowledgement (ACK) frames are shown in Figure 2.4[1].

It has been shown in [10] that CSMA/CA performances can be improved by simply varying the recommended values as in Table 2-1. Under low/normal traffic loads, increasing the macMinBE value can reduce the probability of packet collections and CCA failure, while

[1] 1 symbol period = 4 bits as specified in IEEE 802.15.4 standard

increasing the macMaxCSMABackoffs value can reduce the packet loss rate. Therefore, this approach is later adapted by this research in its cross-layer mechanism for the MAC layer protocol optimisations.

Table 2-1 MAC layer parameters of the IEEE 802.15.4 standard

Variable	Description	Value
aMaxBE	Maximum allowed backoff exponent value	5
aMaxFrameRetries	Maximum number of frame retries before declaring a transmission failure	3
aTurnaroundTime	Radio switch time between Rx and Tx states	12 symbol periods
aUnitBackoffPeriod	Duration of one backoff procedure	20 symbol periods
macAckWaitDuration	Delay before sending back an ACK frame after receiving a successfully delivered frame	120 symbol periods (channels 0 - 10) 54 symbol periods (channel 11 - 26)
macMaxCSMABackoffs	Maximum number of backoff attempts before declaring a channel access failure	0-5 (default 4)
macMinBE	Minimum backoff exponent value	0-3 (default 3)
CCA duration	Duration of one clear channel assessment procedure	8 symbol periods

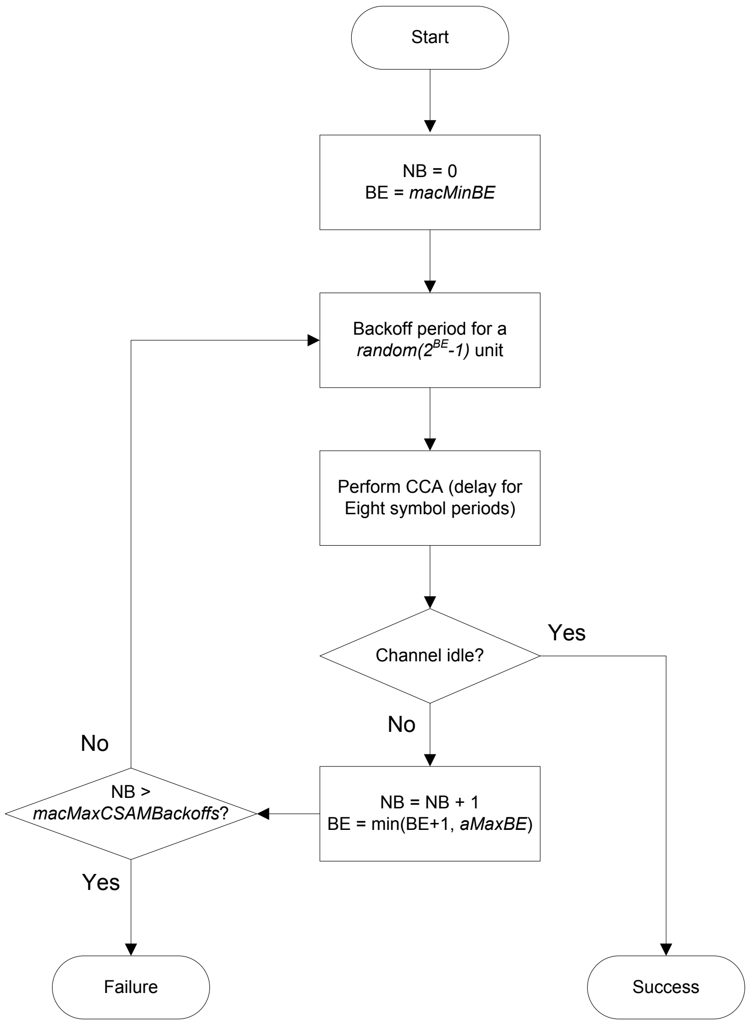


Figure 2.3 IEEE 802.15.4 Unslotted CSMA/CA channel access mechanism

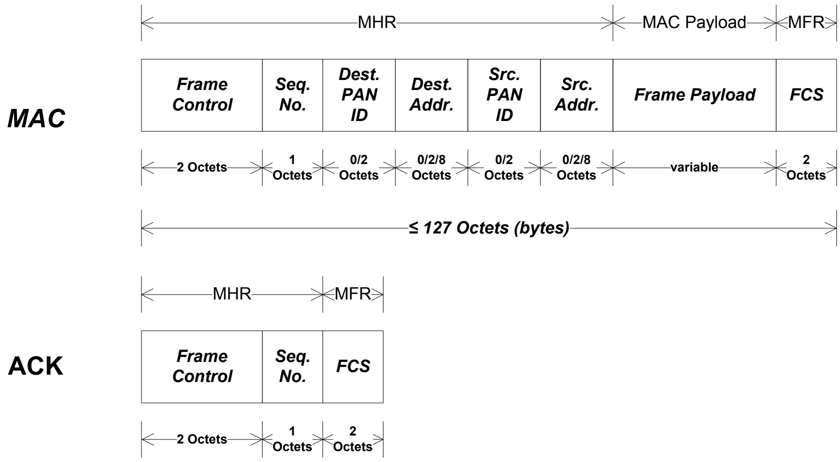


Figure 2.4 MAC and ACK Frames of unslotted CSMA/CA

2.2.2.2 The AODV Algorithm

Due to the resource-constraint nature of sensor nodes, it is typically more efficient to perform multiple short-range transmissions than one single long-range transmission. Such multi-hop communications require the use of intermediate nodes to relay packets, and a routing algorithm is therefore necessary to discover relaying nodes and multi-hop paths for routing in the network.

Ad-hoc On Demand Distance Vector (AODV) [28, 29] is an on-demand routing algorithm, which does not require a node to maintain or exchange any path information unless data is ready to be sent. When a source node has data to transmit to a destination node to which it has no valid path, a route request (RREQ) packet is broadcast in the network to discover a path.

An intermediate node receiving the RREQ may reply with a route reply (RREP) packet if it knows a valid path to the required destination. Otherwise, the RREQ packet is rebroadcast to the network by the intermediate node. The destination node will reply with a RREP packet when it receives the RREQ packet. All nodes that receive the RREP packet can add or update their routing table entries to both the source and destination nodes. A source node can start to transmit the actual data to the destination node once it receives the first RREP packet. Upon expiration or breakage of a path, a new path will be discovered following the same procedure.

The frame structures of AODV in ZigBee [28] are shown in Figure 2.5[2].

[2] 1 octet = 1 byte = 8 bits

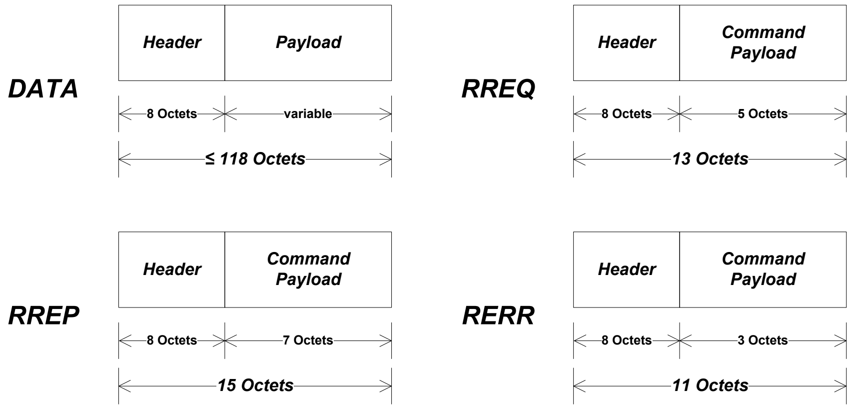


Figure 2.5 AODV frame structures in ZigBee

2.2.3 Network Topology and Structure in WSNs

Network topology describes the inter-node connectivity of a network. Unlike many wired networks with fixed network topologies, WSN topologies can be easily reconfigured, e.g. by a change of the transmission power or node location [30]. In WSNs, the primary goal of topology management is to maintain node connectivity in a way that optimises network efficiency and performance [31, 32]. The common WSN topologies are: star, tree, and mesh [33–35].

The topology of more complex WSNs with hundreds or thousands of nodes will need to be organised into hierarchical structures in order to enable tasks such as routing [36] and data aggregation [37] to be simplified and made more efficient.

A network can be organised in a tier hierarchy in which nodes are grouped into partitions or clusters. Each partition or cluster can contain one coordinator node (also known as clusterhead) and multiple member nodes. Member nodes can only communicate with their coordinator node on the lower hierarchy. On the higher hierarchy, the coordinator nodes from different partitions or clusters can communicate with each other in response to the needs of their member nodes. The highest hierarchy are the WSN applications (normally located on some servers) which collect the sensor data from the coordinator nodes.

2.2.4 Data Centricity in WSNs

In WSNs, **data centricity** is a major feature that distinguishes WSNs from other types of wireless networks. What most applications want from a WSN is the **data** gathered by the sensor nodes, not communication with specific sensor nodes. Thus, having the means to identify individual nodes is less important in a WSN than in traditional networks where every node has to be uniquely identified by an address [3]. Although it is not necessary for nodes in a WSN to be assigned with addresses, WSN applications do at least need to know the approximate locations of the sensor nodes from which data is collected [38].

Traditionally, data queries may be sent periodically from WSN applications to sensor nodes. This approach is called polling [39]. Such constant transmission of queries across the network can waste resources such as wireless bandwidth and nodal energy, which in turn reduces the network's performance and lifetime.

Publish/subscribe (pub/sub) is an emerging communication paradigm that has been proposed for **data-centric applications** on the Internet, and recently for **data-centric sensor networks** [40–42]. A description of the pub-sub mechanism is given in the next section.

2.2.4.1 The Pub-Sub Mechanism

As mentioned above, WSN applications are only interested in the **data** from the sensor nodes. They are not concerned with the identity of the nodes from which they retrieved the sensed data. Most sensor nodes simply sense and notify the WSN application of events when they occur, e.g. when temperature is above or below some predefined threshold. The sensor nodes can also schedule their operations, e.g. alternating between 'sleep' and 'active' states periodically to conserve energy. Most sensor nodes cannot anticipate who will be interested in their data, and when and where their data must be delivered to. Under such circumstances, which are typical of a large number of WSN applications, the **pub/sub** mechanism provides a suitable means of retrieval and dissemination of sensor data.

As shown in Figure 2.6, the pub-sub mechanism has three entities: i) **publishers**; ii) **subscribers**; and iii) one or more **brokers**. The publisher refers to the data source, which makes its data available for others to access by publishing it to the broker entity. The subscriber is the entity that subscribes to and consumes the supplied information from publisher. The broker is an intermediate entity that stores information from the publishers, receives subscriptions from the subscribers, and forwards the information matching the subscription to them. Under this paradigm, sensor nodes can publish their data under different sensory types, such as temperature, light, humidity, and pressure etc., to the broker without having to know when and who will subscribe them; and subscriber devices that operate the WSN applications can subscribe to the broker to receive particular data without having to know when and who will publish them. Such unbundling of publishers and subscribers in time and space can allow the system to scale in size more efficiently and making it easier to add new data sources and consumers, or to replace existing ones [40].

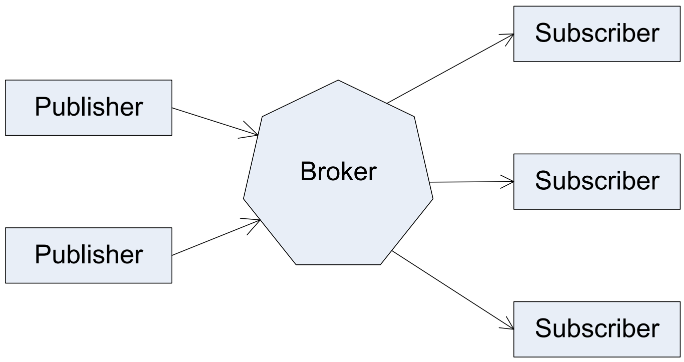


Figure 2.6 Publish and Subscribe mechanism

2.3 Cross Layer Optimisation in WSNs

As described in Section 2.2.2, current protocols designs in WSNs commonly adopt a layered architecture. For more efficient communications with limited network and nodal resources, there is a need for better ways to utilise the resources optimally [43]. By sharing information in a protocol stack more widely across different layers, network performance and efficiency can be improved significantly [5, 44]. This is the cross-layer approach adopted in this thesis.

2.3.1 Cross Layer Approach

Figure 2.7 shows the traditional layered protocol architecture in which the protocol of one layer can only exchange data with protocols in its adjacent layers. A protocol encapsulates data arrived from its upper adjacent layer and forwards it down to its lower adjacent layer. Similarly, it can decapsulate data received from its lower adjacent layer and forward it up to its upper adjacent layer. There is no interaction between any non-adjacent layers as in such a layered architecture.

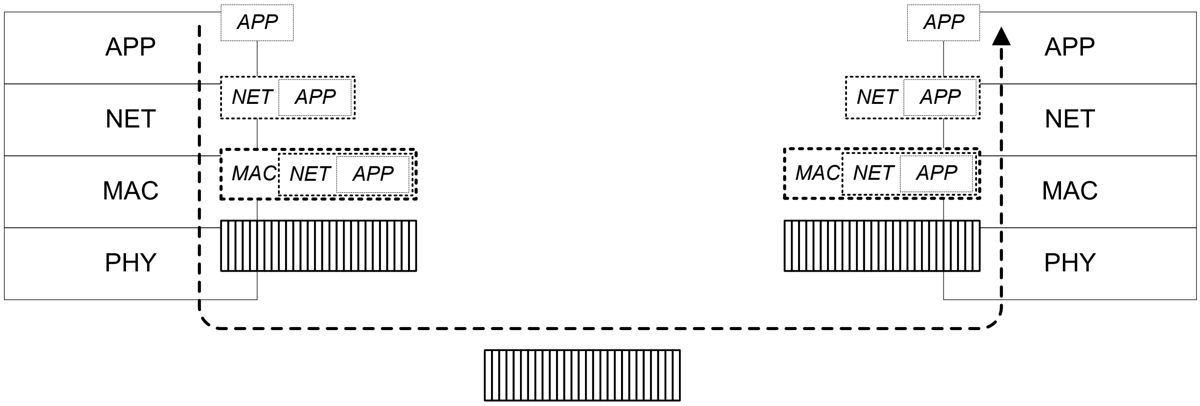


Figure 2.7 Data encapsulation and decapsulation in traditional layered architecture

Although layered architectures have served well for wired networks, they present restrictions to achieving the best possible performance in wireless networks [45, 46]. The layered architecture based on the OSI reference model forbids the direct interactions between non-adjacent layers in a protocol stack, while the interactions between adjacent layers are limited to encapsulation and decapsulation of data [47]. In order to overcome the limitations of protocol designs in wireless networks, researchers have proposed the cross-layer concept. A cross-layer design may refer to a *"design by the violation of reference layered communication architecture"* but *"with respect to the particular layered architecture"* [45]. In WSNs, due to a more severe constraint in resources that impedes the network performance, it becomes even more necessary for protocols to have access to all relevant information in different layers, such as information of the link quantity, residual node energy, status of applications, etc, through non-adjacent and adjacent layer interactions, as shown in Figure 2.8.

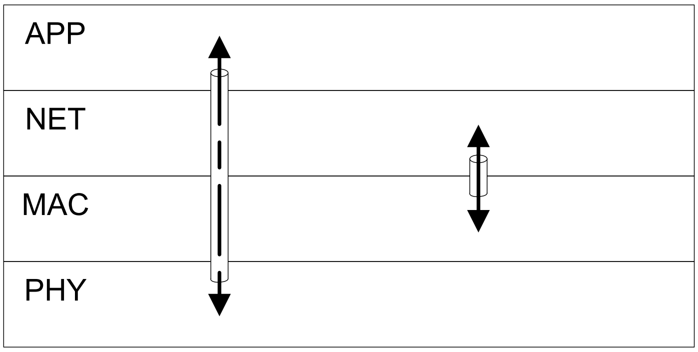


Figure 2.8 Interactions between non-adjacent (left) and adjacent (right) layers

By breaking protocol boundaries in a layered architecture, it allows not only protocol interactions across the layers, but also the possibilities of merging and removal of layers, and the creation of new interfaces and entities for interactions between the layers [45].

2.3.2 Types of Cross Layer Optimisation

Existing cross-layer design approaches may be categorised by the way information is exchanged or shared between non-adjacent layers, which can be either: i) **direct layer communication** [48–51]; or ii) **indirect layer communication via shared entity** [52–54], as shown in Figure 2.9.

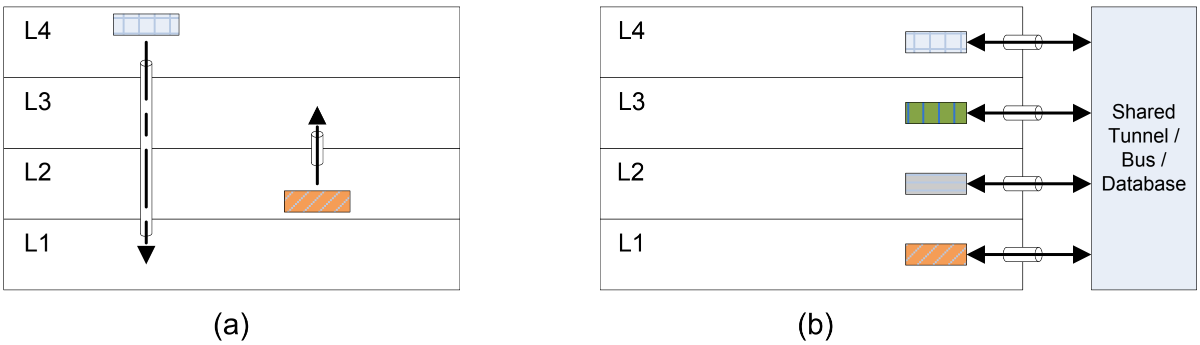


Figure 2.9 Cross-layer interaction. (a) Direct layer interaction; (b) Indirect layer interaction via shared entity

For direct layer communication, information is exchanged between layers directly using cross-layer signalling, which can be either in-band or out-of-band [38]. On the other hand, in indirect layer communication, all communications go through a shared component, which functions like a shared database/storage where layers store and retrieve information to be used by others.

Given that energy constraint is one of the most critical challenges in WSNs, a number of cross-layer designs for WSNs have focused on optimising energy efficiency [18, 55]. Other optimisation objectives can be improvements to the network quality-of-service (QoS), network scalability, and robustness to node failures [56]. Selected relevant cross-layer designs will be reviewed in the next chapter.

2.4 Ambient Intelligence

WSN is an enabling technology of Ambient Intelligence (AmI). AmI relies on the massive deployment of networked, distributed and embedded devices in the physical environment to sense and respond to the context or situations of people in the environment, as shown in Figure 2.10. In addition, AmI should provide personalised responses through intuitive interfaces and natural interactions in a manner consistent with the user contexts [57].

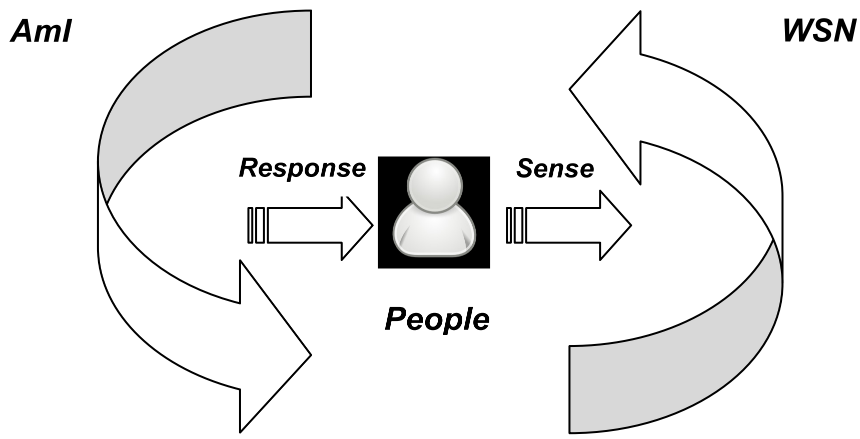


Figure 2.10 Ambient intelligence

2.4.1 Formal Definition of Ambient Intelligence

AmI is a novel vision of a user-centric ubiquitous computing future where people are surrounded by distributed networks consisting of tiny sensors, computational devices, and smart tags seamlessly integrated into our everyday objects. Within an AmI environment, the "smart" objects can anticipate the needs of human beings in their daily lives [58]. AmI illustrates technology developments that could make consumer electronics interact and adapt to human needs while remaining invisible to people. A formal, and widely accepted, definition of the concept of AmI and its vision is set forth by the ISTAG [59]:

"The concept of Ambient Intelligence (AmI) provides a vision of the Information Society where the emphasis is on greater user-friendliness, more efficient services support, user-empowerment, and support for human interactions. People are surrounded by intelligent intuitive interfaces that are embedded in all kinds of objects and an environment that is capable of recognising and responding to the presence of different individuals in a seamless, unobtrusive and often invisible way."

The above definition shows that AmI is focused on technologies for humanity. In an AmI world, technologies should be moved into the background, embedded in the objects of environment, and without being noticed by people [60, 61]. People can interact with the AmI environments while not having to deal with the complex technologies behind it. Essentially, AmI systems are able to sense the presence of people, detect their characteristics, and understand their needs. In addition, AmI systems can adapt their autonomous and intelligent services to human activities and behaviours in a natural way through intuitively connected devices [62, 63].

In 1998, the notion of "Ambient Intelligence" was first introduced by Philips and Palo Alto Ventures [61]. The European Commission quickly adapted the AmI's vision to their Sixth Framework Programme (FP6) in Information, Society and Technology (IST), and supported by the Information Society Technologies Advisory Group (ISTAG) [59, 64]. Over the years, there are many related research projects in both industry and academia, such as HomeLab of Philips Research [65] and MIT's Media Lab [61].

2.4.2 Context-Awareness in Ambient Intelligence

An AmI context information describes a particular situation of an AmI environment. The context information is generated or inferred by AmI applications and services [66] typically from multiple sources of raw sensor data [67].

The term context has been defined by many authors according to the types of information that may affect the user. It is common to refer context as *location*, *identities*, *time*, *temperature*, etc [68]. This is further defined and categorised into four groups: **where**, **who**, **what**, and **when** [69]. The '**where'**, '**who'**, and '**what'** reflect the '**location'**, '**identity'** and '**activity'** of the involved entity, respectively; while '**when'** indicates the **time** or **date** at which this context was generated. With further application developments on a wider range of user activities, it became more difficult to apply the above classification. A more general and widely accepted definition by Day [70] embraced the previous ones and generalised that:

"Context is any information that can be used to characterize the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interactions between a user and an application, including the user and the application themselves."

This definition describes that context holds and represents appropriate knowledge about the user and environment. This can affect the behaviour of an AmI system. In addition, context has to be structured for sharing knowledge.

Context-awareness is a key feature of AmI [58] that enables the system to intelligently adapt its behaviours to the users' circumstances or characteristics, and deliver personalised intelligent services according to user context, e.g. current activity of the user.

2.4.3 Context Modelling and Reasoning

Humans can easily share and understand context information or knowledge. However, this can be a difficult task for non-human entities such as machines in AmI [71]. To be context-aware, an AmI system must be able to capture and collect the circumstances of users and/or environments. The AmI system may not be able to incorporate or understand the context if it is not structured [72]. Context has to be modelled, i.e. organised in a structure, to simplify the ability of capturing, sharing and interoperability in AmI applications. Contextualising information into context can be done through context modelling mechanisms, and decision-making based on context can be referred to as context reasoning.

In AmI, context modelling refers to the usage of a context model that integrates low-level sensed information from a multitude of sensors, inputs from external systems, as well as any user-related information, such as profiled information or social interests, to derive high-level context information. A context model can capture facts from information, represent them in expressions, and organise them structurally [73]. With a proper context model, the markup of information can be standardised and interpreted by different AmI applications and services, facilitating the sharing of context knowledge in uniform format.

Usually, a context model is designed for a particular application scenario, due to the diversity of AmI objectives, which in turn affects how context information will be processed and organised. Hence, a specific context model designed for one application may not be suitable for other AmI applications. To overcome this limitation, there are recent proposals on generic modelling approaches for general-purpose context models [74, 75].

While context modelling is an approach for standardising information representation for easy knowledge exchange [76], context reasoning is a process for matching situations with context, performing decision makings, and inferring new contexts from existing ones [72, 77]. Context-awareness in AmI requires reasoning capabilities to evaluate context for the detection of changes relevant to the users and their environments.

Often, a context reasoning design is tightly associated with its relevant context model. This is due to the high volatility of context represented by an AmI application [73]. Hence, to achieve better interoperability, the developments of both context modelling and reasoning mechanism should be standardised for easy context sharing and reuse, as in the Amigo project [78]. To achieve better context sharing, the context model design should be generic so that one context model can be used in different application scenarios. It is also important to standardise context reasoning to ensure that different context representations of a single situation can always produce the same result using different context reasoning mechanisms. This can lead to better interoperability across different AmI implementations [73].

2.4.3.1 Ontology-based Context Modelling

There are different approaches to modelling and representing context, including key-value pair, graphic-based, and logic-based [75]. However, most of them are designed only for specific circumstances, and lack interoperability outside their design scopes. This can make it difficult to interoperate different context model designs for different scenarios.

To overcome the issues of interoperability, usability and extensibility of existing context modelling methods, the ontology-based context modelling approach has been studied lately [79]. Ontology, which can be defined as "*explicit formal specifications of the terms in a domain and the relations among them*" [80], can formalise taxonomies to represent types and values of properties to be modelled. Ontology can represent semantics, concepts and their interrelationships onto machine-understandable data constructs [81, 82]. An ontology based model can represent both domain-independent and specific contexts, while a piece of context information represents an instance of a context model. For ease of understanding, an ontology model can be imagined as a data structure, and a piece of context information is an instance of this structure with all the fields filled with some values.

Under an ontology model design, context should be interpreted through context reasoning, which may use *rules* to represent derivation axioms that can infer context and perform decisions. This can provide a design with the capacity for contextual knowledge comparison, new or complex contextual knowledge deduction, and reduction of incomplete or ambiguous contextual knowledge.

There are several advantages of modelling context based on ontology: share knowledge, reuse knowledge, and knowledge inference [83]. Ontology-based modelling can allow contextual knowledge interactions using a common set of context representations. Therefore, it is possible to incorporate and reuse existing ontology models from different designs to create a new ontology model rather than having to create one from scratch.

2.5 Chapter Summary

This chapter introduces the background concepts of wireless sensor network (WSN), cross-layer optimisation, and ambient intelligence (AmI). Cross-layer protocol optimisation has potential to achieve significant improvements in wireless sensor network performance and efficiency. There are possibilities to realise cross-layer protocol optimisations by using high-level contexts from an ambient intelligence system. The next chapter will present a review of the recent research literature on the topics described in this chapter.

Chapter 3 Literature Review

3.1 Introduction

In Chapter 2, the background concepts of three technologies (WSN, AmI, and cross-layer design) are introduced. This chapter presents a review of the state-of-the-art on four relevant research issues in the domain of WSNs, namely Network Structuring, Data-Centric Communication, Context Modelling, and Context-Aware Cross-Layer Design in WSNs.

3.2 Network Structuring in WSNs

In this thesis, network structuring refers to the process of structuring the network through clustering or partitioning mechanisms. These mechanisms involve organising WSN nodes into groups (clusters or partitions) for different design objectives such as enhancing the network scalability, reducing communication overheads, enabling better topology control, and improving energy efficiency to prolong the network lifetime [84, 85].

Low Energy Adaptive Clustering Hierarchy (LEACH) [86] is one of the well-known energy efficient clustering algorithms for WSNs. LEACH is a probabilistic and distributed clustering algorithm in which no centralised control is present. An individual node can declare itself as a cluster head (CH) based on a probability value *P*, and broadcast a notification. Other nodes can join this CH if they are one-hop away and have a lower communication cost to this CH than to other CHs. All CHs are required to communicate with the data sink directly. For a node to decide if it should become a CH, the node compares its randomly generated number (T_{CH}) with a threshold value ($T(n)$) given by:

$$T(n) = \begin{cases} \dfrac{P}{1 - P * (r \bmod \frac{1}{P})} & \text{if } n \in G \\ 0 & \text{otherwise} \end{cases} \qquad [86]$$

where *P* is the desired percentage of CHs in a network of *N* nodes, *r* is the current iteration count, and *G* is a set of nodes that have not become CHs in the previous 1/*P* rounds. A node can become a CH only when $T_{CH} < T(n)$. While rotating the role of CH among the nodes may distribute the node energy consumption in the network, it is still possible for some low energy nodes to become a CH, which can result in premature death of these nodes. In addition, LEACH is limited to operating in a single-hop cluster. It is not suitable for use in multi-hop clusters when short-range transmission is used.

Energy-LEACH and Multihop-LEACH [87] were proposed to improve LEACH in two aspects. In Energy-LEACH, the first iteration round follows exactly the same procedure of LEACH, where n nodes ($n = P \times N$) become CHs randomly. However, from the second iteration round onwards, only the top n nodes with the highest residual energy will become CHs. This prevents low energy nodes from becoming CHs, but could result in CHs and therefore traffic being unevenly distributed in the network. In Multihop-LEACH, a CH can forward data to the sink through multiple relay nodes instead of a direct communication with the sink as in LEACH. However, Multihop-LEACH requires the transmission power of a CH to be an order higher than non-CH nodes. This can draw a significant amount of energy from a node that functions as a CH during an iteration round.

KOCA [88] is another probabilistic based distributed clustering algorithm for load balancing. In KOCA, a probability p is pre-defined to determine the number of CHs in a network. The average number of CHs is $p \times N$, for a network of N nodes. The cluster radius is limited to k hops for a CH. Similar to LEACH's CH selection phase, any node can perform autonomous decision according to the predefined probability value p. Once a node becomes a CH, it notifies all sensor nodes within k hops. A sensor node sends a join request (JREQ) to the CH from which it receives the notification. In KOCA, a sensor node can belong to more than one cluster as long as the node is within k hops of the CH. Although KOCA can create partitions of uniform geographic size, i.e. up to k hops, the CHs can still be distributed unevenly. Moreover, restricting the cluster size to a fixed number of hops may result in the isolation of some nodes which do not belong to any clusters, i.e. due to there being no CH available within their k hops.

The Distributed Energy-Efficient Clustering (DEEC) algorithm [89] is proposed for forming hierarchical clusters in heterogeneous WSNs. Here, CHs are chosen according to a probability value based on the ratio between the residual energy of a node and the average energy level of a network. There are two types of nodes in DEEC: normal nodes, and advanced nodes that have α times more initial energy than the normal nodes. For a network with N nodes, there are $m*N$ advanced nodes and $(1-m)*N$ normal nodes, where m is the fraction of advanced nodes. During each rotating epoch r, i.e. clustering iteration round, all sensor nodes have to know the average network energy level $\bar{E}(r)$ from the base station (BS). Every node i calculates its probability of becoming a CH as:

$$P_i = \begin{cases} \frac{P \cdot E_i(r)}{(1+a \cdot m)\overline{E}(r)} & \text{if } i \text{ is a normal node} \\ \frac{P \cdot (1+a) \cdot E_i(r)}{(1+a \cdot m)\overline{E}(r)} & \text{if } i \text{ is an advanced node} \end{cases} \quad [89]$$

where r is the current rotating epoch, P is the desired fraction of nodes that are to be CHs, $E_i(r)$ is the current residual energy of the node i. DEEC uses the same approach as LEACH to determine the threshold value T of node i to become a CH in the network at the current rotating epoch. DEEC requires the $\bar{E}(r)$ to be known by every node, which can result in significant amount of communications with the BS to obtain this information.

The Energy Efficient Heterogeneous Clustered scheme (EEHC) [90] is an algorithm to enhance the WSN lifetime and stability by cooperating with heterogeneous nodes. Similar to DEEC, EEHC assumes three types of nodes with different initial energy. Out of a total of n nodes, there are $m_o * m*n$ super nodes, $(1-m_o)*m*n$ advanced nodes, and $(1-m)*n$ normal nodes, where m is a fraction of n nodes that are equipped with β times more energy than normal nodes, and m_o is a fraction of m nodes that are equipped with α times more energy than normal nodes. EEHC provides three weighted probability values for each of the normal, advanced and super nodes according to their energy:

$$P_n = \frac{P}{1+m*(\alpha + m_o * \beta)}$$
$$P_a = \frac{P}{1+m*(\alpha + m_o * \beta)} * (1+\alpha) \quad [90]$$
$$P_s = \frac{P}{1+m*(\alpha + m_o * \beta)} * (1+\beta)$$

where P is the desired fraction of nodes that are to be CHs, and P_n, P_a and P_s are the weighted probabilities for the normal, advanced, and super nodes, respectively. The corresponding threshold value T for a node φ to become a CH is computed as:

$$T(\varphi) = \begin{cases} \dfrac{P_\varepsilon}{1 - P_\varepsilon * (r \bmod \dfrac{1}{P_\varepsilon})} & if\ \varphi \in G \\ 0 & otherwise \end{cases} \qquad [90]$$

$$where \quad \varepsilon \in \{n, a, s\}$$

Both DEEC and EEHC are designed for heterogeneous WSN nodes, and they will function very similarly to LEACH if only homogeneous nodes are deployed in a WSN.

The Base Station Initiated Clustering scheme [91] is a centralised algorithm proposed for improving network lifetime. In this scheme, heterogeneous sensor nodes, some of which are 'power' nodes, i.e. nodes equipped with more residual energy, more computation power, and with location awareness, are deployed. CHs are organised in levels according to their distance to the BS. A low level CH is one that is close to the BS, and vice versa. The BS initiates the cluster formation, and clusters are formed according to the energy level and location of the power nodes, which are selected as CHs. A hierarchical tree is established among all CHs and BS. A non-power node joins the closest CH and sends its data to it. Communications between a CH and its member nodes are single-hop while communications between the CHs are multi-hop as shown in Figure 3.1. Data is forwarded from the high level CH to the low level CH until it reaches the BS. This scheme can evenly distribute CHs if the advanced nodes are uniformly distributed during the deployment phase. However, it requires the BS to control the cluster formation. Moreover, nodes that are more than one hop away from any CH will be isolated.

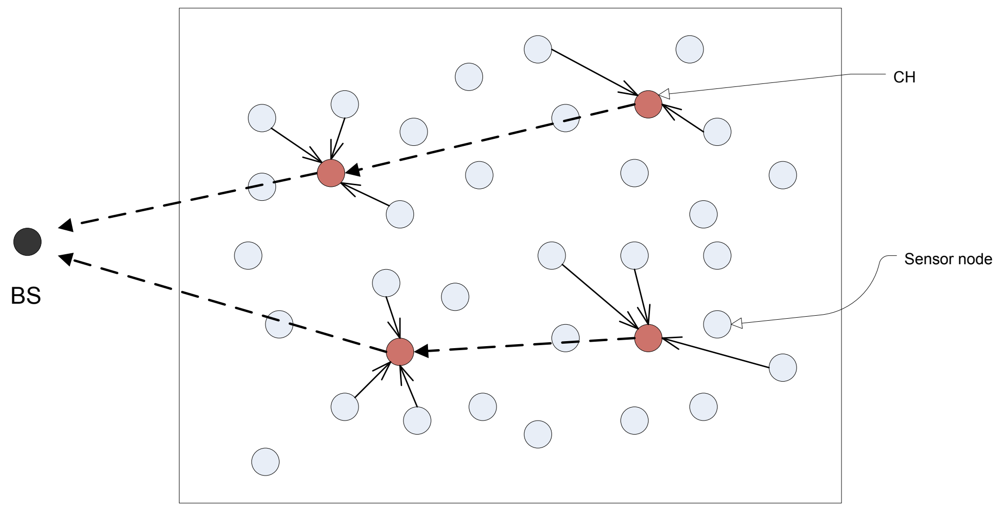


Figure 3.1 Base Station Initiated Clustering [91]

Location-based Unequal Clustering Algorithm (LUCA) [92] forms non-uniformly distributed clusters according to the distance from a CH to the sink. The objective is to minimise energy dissipation for all sensors in a network. LUCA does require each sensor node to be equipped with a GPS-like device for measuring its distance to the sink located at the centre of a network. LUCA uses a two-tier multi-hop communication model to organise a network. In each cluster, a sensor node transmits its data to the CH, which in turn forwards the data to the sink using geographic routing that can minimise path length in terms of hop counts and avoid path loops. A CH can arrange the sleep/wake-up schedule for its member nodes to be more energy efficient. Therefore, a member node can sleep over a long period, wakes up to transmit data, and sleep again.

In LUCA, the optimal cluster size r in terms of hop counts, which is the coverage radius of a CH that optimises the energy consumption, is given as:

$$r = \left\lceil \frac{1}{r_0} \sqrt[3]{\frac{3D}{\lambda \pi}} \right\rceil \text{ [92]}$$

where D is the distance between a CH and the sink in a Poisson distributed network with intensity λ.

After sensor nodes are deployed, each node selects a random backoff time between 0 and 1. At end of the backoff time, if a sensor node receives no CH advertisement, it will declare itself as a CH by broadcasting a CH advertisement to other sensor nodes within r hops away. Otherwise, the sensor node joins the nearest CH from which it receives a CH advertisement. LUCA requires absolute node position information during the clustering phase, which may not be always available in a WSN.

Most of the clustering schemes consider only one CH for each cluster. However, due to the tasks performed by a CH, which include data gathering and forwarding, significant amount of energy can be drawn from the CH. Therefore, a number of network structuring schemes that incorporate multiple CHs in one cluster have been developed, as described below.

In the Dual Head Clustering Scheme (DHCS) [93], two CHs exist in each cluster. One CH provides the function of data gathering and storing while another performs data forwarding. In DHCS, a sensor node can determine its distance to the static sink based on received signal strength. This distance is used to calculate the cluster size if the sensor node becomes a CH. The formula for a node j to decide its cluster size R_j is given as:

$$R_j = \frac{x_j \cdot (R_{\max} - R_{\min})}{d_{\max}} + R_{\min} \text{ [93]}$$

where R_{max} and R_{min} are the maximum, and minimum radius of a cluster, respectively, x_j is the estimated distance to the sink based on received signal strength, and d_{max} is the estimated distance between the farthest node and the sink in a network. Every node has to calculate its cluster radius *R*, and broadcast the result. One node with the highest *R* radius is selected as a CH while all other nodes within that radius join the CH as its member nodes. After the CH has been determined, the next phase is to elect an Aggregator Head (AH) for the cluster by selecting the node with the highest energy and minimum distance to the CH. Inter-cluster communications are handled by the AH based on shortest routing path to the sink. For each cluster, the CH collects data from its member nodes and sends it to the AH for forwarding to the sink. DHCS presents the concept of collaborating multiple nodes for the tasks of a single CH. However, if the selected AH in a cluster is far from its corresponding CH, significant amount of intra-cluster communications between CH and AH can occur, which will increase the energy consumption of the nodes.

A Double cluster-Heads clustering algorithm using Particle Swarm Optimization (PSO-DH) is proposed in [94] to extend network lifetime. Two CHs are generated by the algorithm: Master Cluster Head (MCH) and Vice Cluster Head (VCH). The MCH gathers the data from the sensor nodes within a cluster and transmit it to the VCH, which in turn sends the aggregated data to the sink directly. PSO-DH determines the fitness of a node *i* as follows:

$$\begin{aligned} f(i) &= \varepsilon \cdot f_1(i) + (1-\varepsilon) \cdot f_2(i) \\ f_1(i) &= E(i) / \sum_{k=1,k\neq i}^{m} E(k) \qquad \text{[94]} \\ f_2(i) &= (m-1) / \sum_{k=1,k\neq i}^{m} d(i,k) \end{aligned}$$

where $f(i)$ is the fitness function of node *i*, $f_1(i)$ is the ratio of node *i*'s energy to the total cluster energy, $f_2(i)$ is the ratio of the number of remaining nodes in the cluster over the total Euclidean distance of the cluster nodes to node *i*, ε is a user defined weight constant (set to 0.6 in [94]), $E(i)$ and $E(k)$ are the energy of node *i* and *k* respectively, *m* is the total number of nodes in the cluster, and $d(i,k)$ is the distance between a pair of nodes *i* and *k*.

PSO-DH starts the clustering process by using the LEACH algorithm. After the initial clusters are constructed by LEACH, the original CH from LEACH selects the MCH and VCH based on the fitness value of each individual node in the cluster. A node with the optimal PSO solution is selected as the MCH while another with the sub-optimal PSO solution is selected as the VCH. Once this selection phase is completed, the original CH announces its MCH and VCH selection to its member nodes. Next, a TDMA schedule is set up by the MCH to all cluster member nodes, which can only transmit during their assigned time slot. Once the MCH completes data collection from the member nodes, the aggregated data is sent to the VCH which in turn forwards to the sink. It is noticed that the PSO-DH requires global information about the average energy and total distance in each cluster to be known by all cluster member nodes. In real-world implementations, such information can only be known through node communications in the network. The clustering process is repeated periodically.

A multi-sink partitioning algorithm is proposed in [95] to maximise the network lifetime by reducing the communication distance between sensor nodes and sink during data communication. This algorithm is based on K-Nearest Neighbour Graph (K-NNG) and uses a flooding-based routing algorithm. The algorithm has two phases – initial phase and incremental phase. Initially, a network with p number of sinks can create p number of partitions. In the initial phase, each sink can generate its K-NNG, which contains the k-nearest neighbour nodes to itself. In the incremental phase, the farthest neighbour node from the sink is set as the *NextNode*. Then, the farthest node in the K-NNG of the sink to this *NextNode* is marked as the new *NextNode*. This phase repeats until the number of nodes processed by the sink matches the total number of nodes deployed in the network. It is observed that the flooding-based routing used by this algorithm can potentially increase communication overheads during the partition formation phase.

Discussion and Analysis

This section discussed the aforementioned algorithms in terms of their structuring approach, requirement for non-local (global) information and centralised control, and analyses their implications on the quality and efficiency of network structuring.

- *Structuring approach*: Existing algorithms for network structuring use different approaches, such as clustering or partitioning based on probability models, weight values, signal strength, residual energy, and location/distance. Algorithms based on probability and energy can generate unevenly distributed partitions/clusters, i.e. the partition/cluster size can vary widely between different partitions/clusters in a network. This can result in unbalanced traffic loads and as a consequence unbalanced energy consumption among sensor nodes. On the other hand, although the distance based algorithms can evenly distribute the partitions/clusters across the network, they do require the sensor nodes to be equipped with hardware functionalities such as GPS or software resources for location/distance computation. Such requirements of a sensor node may reduce the applicability of the algorithm to be implemented in the real world.

- *Requirement for non-local information*: This refers to the need of an algorithm to use non-local information (or have a global view of the network) for its clustering/partitioning process. For an algorithm that requires non-local (global) information, each sensor node will have to acquire some information from other nodes, and incur a communication cost in the process. For instance, LEACH requires no additional information other than a local probability p value to determine if it can become a CH. On the other hand, DEEC requires the current average network energy level $\bar{E}$ to be known by every sensor node. There is no way for a node to obtain this information without communicating with the BS.

- *Requirement for centralised control*: A centralised algorithm refers to one which is initiated and controlled by some central nodes, such as a sink. The advantage of a centralised algorithm over a distributed one is that the central node may already have a "global" view of the network and, hence can perform the clustering/partitioning process more efficiently, such as the direct selection of a CH without any information exchange. However, nodes in a centralised algorithm can behave more passively and act only on the directives of the central node. This can make the algorithm less responsive to local conditions and more vulnerable to single point (central node) of failure.

Table 3-1 summarises the comparison of the discussed algorithms in terms of the aspects mentioned above.

Table 3-1 Comparison of the partitioning/clustering algorithms

Algorithm	*Approach based on*	*Non-local information*	*Centralised Control*
LEACH	Probability	No	No
Energy-/Multihop-LEACH	Probability/Energy	Yes	No
KOCA	Probability/Hop Distance	Yes	No
DEEC	Energy	Yes	No
EEHC	Weight/Energy	No	No
[91]	Received Signal Strength/Distance	Yes	Yes
LUCA	GPS Distance	No	No
DHCS	Received Signal Strength/Distance	Yes	No
PSO-DH	Energy/Distance	Yes	No
[95]	Distance	Yes	Yes

3.3 Data Centric Communication in WSNs

Data centricity is a key characteristic of WSNs where applications are more interested in the data gathered by the sensor nodes than the identity/location of the sensor nodes themselves [2]. Commonly, the term "data" refers to sensed data. However, it can also refer to any information from any entities of a WSN, including software/hardware related information of the network, e.g. sensor software/hardware performance, and application information, e.g. data/status of a WSN application. Extending beyond the existing model-based [96], query-based [97], and data driven [98] classifications for data centric communication, this section categorises algorithms for data centric communication in WSNs according to their data acquisition approach, namely pull-based, push-based, and pull/push based data acquisition.

3.3.1 Pull-based Data Acquisition

As a common approach in WSNs, sensor data is periodically transmitted from the sensors to a centre node such as a sink. This data acquisition approach exhibits a "request-reply" communication pattern. The term "pull" can refer to a centre node requesting for data from selective or all sensor nodes. The centre node decides and controls how and where to collect the sensor data from. The sensor nodes only passively respond to commands and/or requests from the centre node. Flooding and directed diffusion are common approaches under this category.

The Directed Diffusion (DD) algorithm [99] is a data-centric scheme designed for environment monitoring. A tree structure is established by a sink which collects data from the sensor nodes in a network. The sink periodically broadcasts *interest* packets to the network, which are query packets that contain the attributes of the required sensor data, such as sensor data type and transmission interval. Each sensor node senses and stores the data in its local cache, while waiting for an *interest* packet. Upon receiving an interest packet, a node sets up a *gradient* node that indicates the neighbour node from which this *interest* packet is received. A sensor node with data that matches the attributes of the interest packet replies with the data to the sink. The data is forwarded by all *gradient* nodes until it is received by the sink. In this algorithm, network-wide broadcasting is used, which is very inefficient as all nodes will receive every *interest* packet from the sink even if they have no data matching with the content of the packets. Moreover, network-wide broadcasting can create communication issues, such as packet collision.

The Data Spider scheme [100] is a solution that can allow a mobile base-station (MB) to gather data opportunistically from static WSN nodes. A MB uses a Dynamical Tree Reconfiguration (DTR) protocol, which is a variant of the dynamic spanning tree algorithm, to construct a routing tree in the network. The sensor node closest to the MB (called the *anchor* node) is selected as the root and announces itself to other nodes by flooding the network. Upon receiving, the nodes update their parent nodes, i.e. the nodes one-hop closer to the anchor node. The nodes have to update their parent nodes in response to any new anchor node announcement when the MB changes its location. In Data Spider, only the nodes within the region of interest (ROI) of the MB will forward data though their parent nodes to the anchor node, which in turn forwards the collected data to the MB. Since Data Spider selects only a single node as the anchor node, and the selection is made solely based on proximity to the MB, the anchor node can become a single point of failure if it has been low on energy.

3.3.2 Push-based Data Acquisition

As opposed to pull-based mechanisms, the data gathering process is initiated by sensor nodes in push-based mechanisms, i.e. the sensor nodes can initiate and decide what and when data is to be transmitted or "pushed" to the centre node (sink). The centre node has no knowledge on where and when the data may come from, and only provides the storage and/or processing capability for the data received. In this paradigm, the sensor nodes have more control over the data gathering phase. Each sensor node can autonomously transmit its data to the centre node according to a data exchange or dissemination model. The model-driven approaches are the most common approach in this category, and both the centre node and sensor nodes will follow a particular model for their data operations.

The Sensor-TDMA protocol [101] allows push-based data communications in a clustered network. A sensor node is only allowed to send its data when the data value reaches a threshold level. Otherwise, the data will be kept on hold. A priority for each sensor is calculated based on its data value, and is used for allocating time slots. The higher priority a node has, the faster the node can access to a time slot and push its data if the threshold is reached.

The predictive storage (PRESTO) scheme [102] is a two-tier architecture for data acquisition. There are two major components in this architecture: sensor proxies at the higher tier and remote sensors at the lower tier. A sensor proxy contains more computation and storage capability than the remote sensors, whereas the remote sensors are mainly used to sense and transmit data to the sensor proxies. PRESTO provides the remote sensors with the ability of pushing data asynchronously to the sensor proxies instead of being dependent on data pulling from the sensor proxies. The sensor proxies construct a time series of the Seasonal Autoregressive Integrated Moving Average (SARIMA) model for each remote sensor according to past observations of the received sensor data and send the model parameters to the remote sensor. Each remote sensor then uses the model to predict its future data value, and only transmits its data to the proxy when the difference between the predicted and actual data values is over a certain threshold. PRESTO cannot be applied to homogeneous WSNs where no powerful/resourceful nodes are available as sensor proxies. In addition, PRESTO is less useful in WSNs where the phenomena sensed are not always predictable, such as industrial fires or security break-ins.

The Derivative-Based Prediction (DBP) [103] is a model-driven data acquisition scheme that uses a linear modelling technique to allow sensor nodes to predict their future data locally. Periodically, a sensor node constructs a line approximation model based on a small sequence of sensed data. This model together with the sequence of sensed data is then sent to the sink and stored there. Any subsequent sensed data from a sensor node is compared with the predicted data from the constructed model. If the predicted data is within a certain tolerance from the true sensed data, then no data is sent to the sink, as it can *predict* this data for that sensor node according to the earlier received model. Otherwise, a new model is constructed and sent to the sink along with the sensed data used to construct the new model. It was shown DBP can reduce data transmissions by over 90% under certain scenarios. Unlike PRESTO, DBP allows an individual sensor node to construct its own prediction model for the centre node to predict its sensed value. Similar to PRESTO, DBP is only useful for WSNs where sensor events are predictable.

3.3.3 Pull/Push-based Data Acquisition

Generally, pull-based approaches can be implemented more easily than push-based approaches, but they are less communication efficient. Under the push-based approach, both the centre node and the sensor nodes presume that the sensed data is predictable and its value shall follow some trends or patterns defined by a model. Therefore, the sensor nodes generate data, and the centre node acquires the generated data, simply according to the model. However, due to the dynamicity of WSNs, a predefined model may not always reflect the necessities of WSN applications, e.g. a predicted value by a model may have a large variation from its true value. Therefore, a hybrid paradigm that takes advantage of both pull and push based approaches was proposed. Here, both the centre node and sensor nodes can have some control on how data is going to be transmitted and received. Similar to the pull approach, the centre node can still send queries, which set the criteria for searching the sensed data, to the sensor nodes. The sensor nodes also have control of the data acquisition by deciding autonomously when and what sensed data will be sent to the centre node. A common approach in this category is the Publish/Subscribe (Pub/Sub) model where sensor nodes only *push* necessary data that will be used by WSN applications to a centre node. Therefore, both the application and the network efficiency and performance can be improved, as the application has less data to process while its underlying network has less data to transmit.

PSWare [104] is a middleware pub/sub solution for primitive and composite event detections. Using the event definition language (EDL), events can be expressed by SQL-like syntax. EDL defines events as a group of attributes. Each attribute may correspond to a type of sensed data. With EDL, the events can be newly created or composited with the existing events. PSWare uses an event detection protocol for composing events. The basic idea is that it is unnecessary to let all sensor nodes to monitor continuously, but only to monitor when necessary. There are two steps in the protocol: parent node selection and event detection for each sensor node. After receiving an event subscription query from the application, each sensor node decides if it should participate in detecting this event. The decision is broadcast by the sensor node to its one-hop neighbours. If so, the sensor node selects its parent node according to the shared participation information from its neighbours, and sends a notification to the selected parent node to join as a child node. Once the parent node is selected, the sensor node can forward its events to its parent nodes, until the events are received by the application.

Geoserv [105] is a two-tier sensor networking platform that allows location tagged sensed data to be shared with mobile users across the Internet-based P2P overlay nodes. Geoserv uses the Hilbert space filing curve (HSFC) function to linearise a 2-D space into fixed size grids, and associates each grid ID to a Distributed Hash Table (DHT) key space. A network is partitioned according to the grid IDs. To support location-aware publish/subscribe services with a group of mobile users, GeoPS is proposed for Geoserv. The aim of GeoPS is to *publish* data updates for all users *subscribed* to a region. GeoPS divides a grid into a hierarchy of smaller grids to build a multicast tree for mapping data to geographic locality. Data can be published on the Internet servers according to their geographical locations. Data lookup can be performed via DHT with geographic location as key space. Geoserve requires the GPS location of sensor nodes to function, and thus may be suitable for only outdoor WSN applications.

RUNES [106] is a general middleware framework to provide dynamic reconfiguration and publish/subscribe functionality to support a range of services, such as advertising, discovery, and data gathering for *heterogeneous* embedded systems comprising of gateway nodes, sensor routing nodes, and basic sensor nodes. A gateway node is capable of powerful processing, long-range communication, and support IP functionality for interconnection to external networks. The sensor routing nodes are more resource-constrained but can still perform advanced processing tasks such as data aggregation, fusion and multi-radio communication for connecting the basic sensor nodes that perform only basic sensing and computation to the gateway. RUNES may not be appropriate for WSNs where no powerful/resourceful nodes are available. In addition, it mostly focused on reconfiguration and adaptation at device level only, which implies the prospects for network reconfiguration and adaptability is quite limited.

There are schemes that adopted similar approaches to those described above for communicating context information in WSNs [107]. For example, the Distributed Context eXchange Protocol (DCXP) [108, 109] is an Extensible Markup Language (XML) based application level protocol for IP-based mobile and sensor nodes. The DCXP forms a Context Storage (CS) in the network for context information indexing and adopts a DHT approach based on Chord [110] to map Universal Context Identifiers (UCIs) (each of which identifies a particular piece of context information) with node addresses. A logical DHT ring is constructed over all nodes, and a node joins the CS by using a Context User Agent (CUA) that registers the node address in the DHT ring to hold particular context information. To retrieve context information via UCIs from the CS, a node firstly sends a query to the CS. The CS then returns an address of a node with the desired context information to the requesting node, which in turn retrieves the context information, as shown in Figure 3.2.

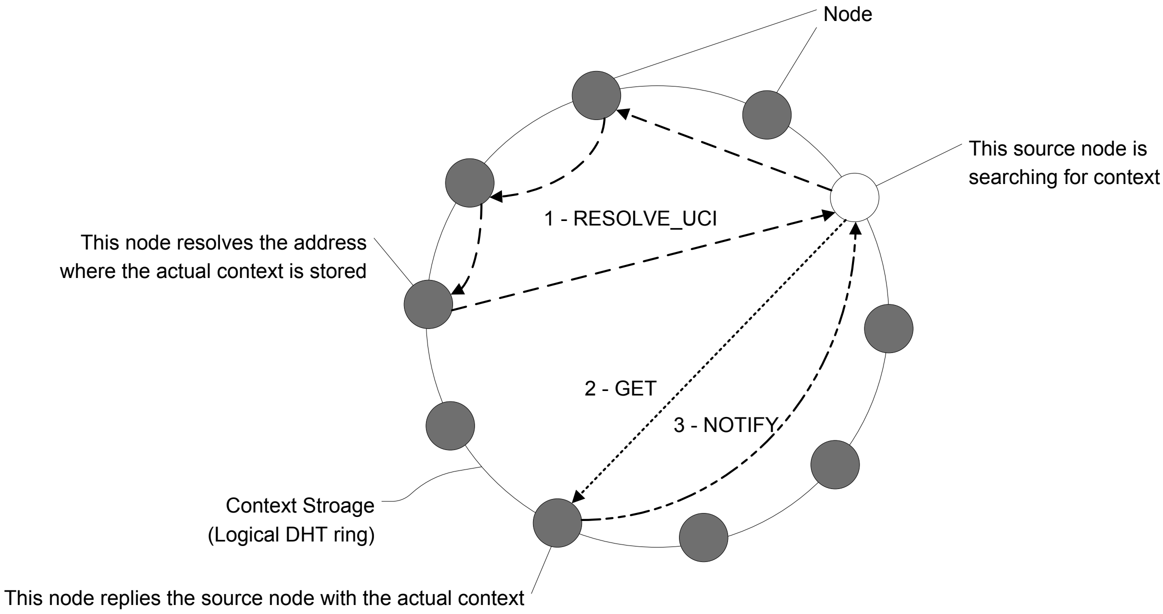


Figure 3.2 Distributed Context eXchange Protocol [108]

3.3.4 Discussion and Analysis

Data centricity is a key feature that distinguishes WSNs from other types of wireless networks [3]. In data centric communication, the communication functions can be performed according to the content or meaning of the data, rather than the geographic or identity information of the nodes that generate the data.

In *pull*-based approach, an instruction received by the sensor node from the WSN application only controls how the sensor node responds, e.g. to start or stop sensing, but not the information it holds. Information processing in pull-based mechanisms is mainly performed at the WSN applications' side. The pull based approach often results in relatively high data communication overhead due to its use of neighbourhood- or network-wide broadcasting [99, 100].

On the other hand, in a *push* based approach, the WSN application mainly functions as a data consumer, and makes predictions about sensor data that it does not receive using a model. The model can be constructed by the application or the sensor node. It can be expressed as a value comparison [101] and linear/non-linear model [102, 103]. The push based approach is often more communication efficient as the applications do not "request" for any sensor data but only "wait" to receive the sensor data. However, there is an aspect of weakness in this approach, which is the requirement for the sensor data to be predictable, and resiliency of the prediction under noisy and dynamic conditions of the WSN. For applications such as in health and safety environments, data accuracy is of critical importance and only sensed (not predicted) data may be acceptable in these applications.

The *pull/push* based approach aims to take advantage of both *pull* and *push* based approaches, i.e. acquiring sensed data (not using predicted data) as in a pull based approach, but in a more communication efficient manner as in a push-based approach. The pull/push based approach is commonly using the pub/sub [104, 106] or based on data hashing [105, 109], in which data is distributed on multiple nodes instead of on a single centre location. While hash based mechanisms can make data search in large-scale WSNs more scalable, their requirement to discover the right node for storing the right data could still incur considerable costs in terms of communication and hence energy [109].

3.4 Context Modelling for WSN Management

Context modelling is the process of representing context information in a data structure according to a set of expressions or rules in a system [111]. The function of a context aware system can be adapted to the modelled context information. For instance, the operation of a network can be optimised according to the context information [112].

It can be a difficult task to define the scope of the context as it is usually scenario dependent. Different context aware systems may require different contexts to be modelled to represent different knowledge domains. A context model can be established only when the objectives of a context aware system are determined.

Context is usually inferred from some lower-level information, such as raw or pre-processed sensor data, through modelling mechanisms. Existing works on context modelling range from as simple as a regular expressions compression between variables [113], to some policy regulations [114], to a full scale XML framework design [115]. In general, a particular modelling technique is designed for a particular scenario. In literature, context models have been classified into key-value, markup scheme, logic based, and ontology based model [74, 75, 116].

- *Key-value model* is a simple context model in which a context instance, i.e. a piece of context, is represented as a paired data structure, which includes an attribute key and its associated value. This model can be implemented easily, but lacks capabilities of structuring context systematically. A key-value model may only be suitable for a small quantity of context information. It cannot handle more complex context structure with large amount of information. Moreover, it also lacks the mechanisms for data validation [117].

- *Markup scheme model* represents context in a hierarchical data structure consisting of tags with the corresponding attributes and contents. It can be considered as an improved key-value model [75]. Markup languages, such as XML, are commonly used to represent structured context information. It can allow context to be stored and exchanged across various components of a context aware system. It can also provide the function of data validation, but still lacks interoperability and re-usability [111].

- *Logic based model* exploits the simplicity of logic formalism to construct context with facts, expressions and rules, and can deduce new context through inference. Expressions under a logic based model can be incorporated by other types of context modelling, such as the ontology based model as described below. The logic based model can achieve high degrees of reasoning capability but with low reusability and applicability due to lack of standardisations [111].

- *Ontology based modelling model* represents context information through semantic based techniques. Through ontology, context and its relationships can be mapped to the modelled knowledge, allow reuse of previous works, and sharing of common vocabularies across domains. This model allows a high degree of expression and possibility for context modelling and reasoning as they are simple, flexible and extensible [74].

This thesis has adopted the method of ontology base modelling due to its clear advantages over other modelling methods. The following reviews some of the most relevant literature on ontology based context modelling.

A centralised ontology based model for context aware management in WSNs is presented in [118]. This model provides a WSN with the capability of autonomous management according to network conditions. It contains the following core components located on a base station: Context Manager, Context Reasoner, Ontology Manager, one or more Sensor/Actuator Managers, Context Repository and Rule Repository, as shown in Figure 3.3. Context querying and reasoning process are also performed on the base station. The Context Manager is responsible for gathering sensor- and actuator-associated data. The Context Reasoner uses the Logical Rules to infer the network conditions. The Context Manager controls the Context Repository, which stores the ontologies on a MySQL relational database. In this model, all contexts are organised in an ontology structure, as shown in Figure 3.4, and expressed by Web Ontology Language Description Logic (OWL DL). This ontology can model context of computer device, location, and activity. This is a centralised design that requires all information processing to be performed at a centre location (base station).

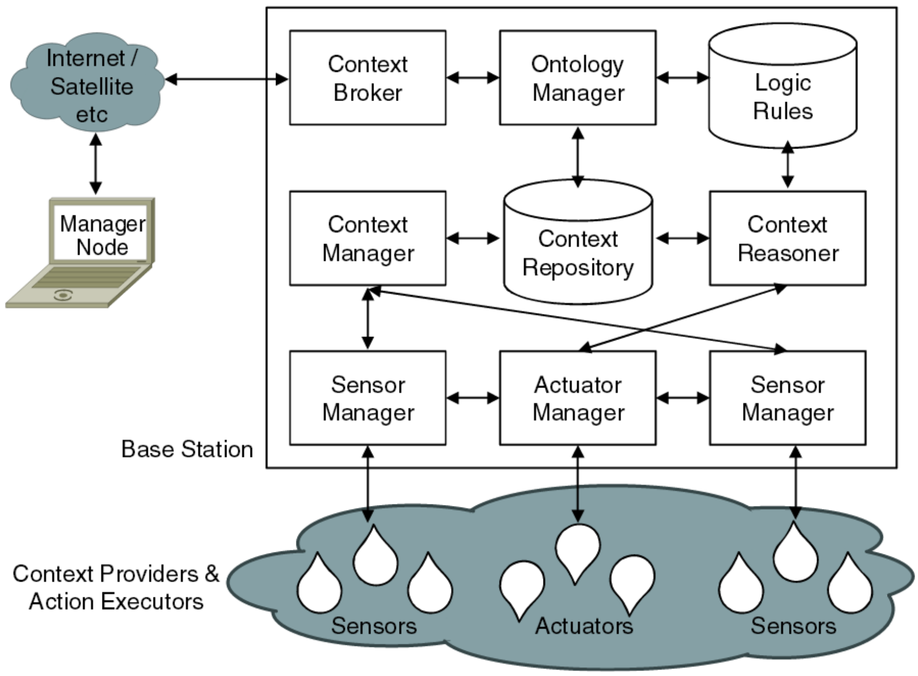


Figure 3.3 The management model overview. Source: [118]

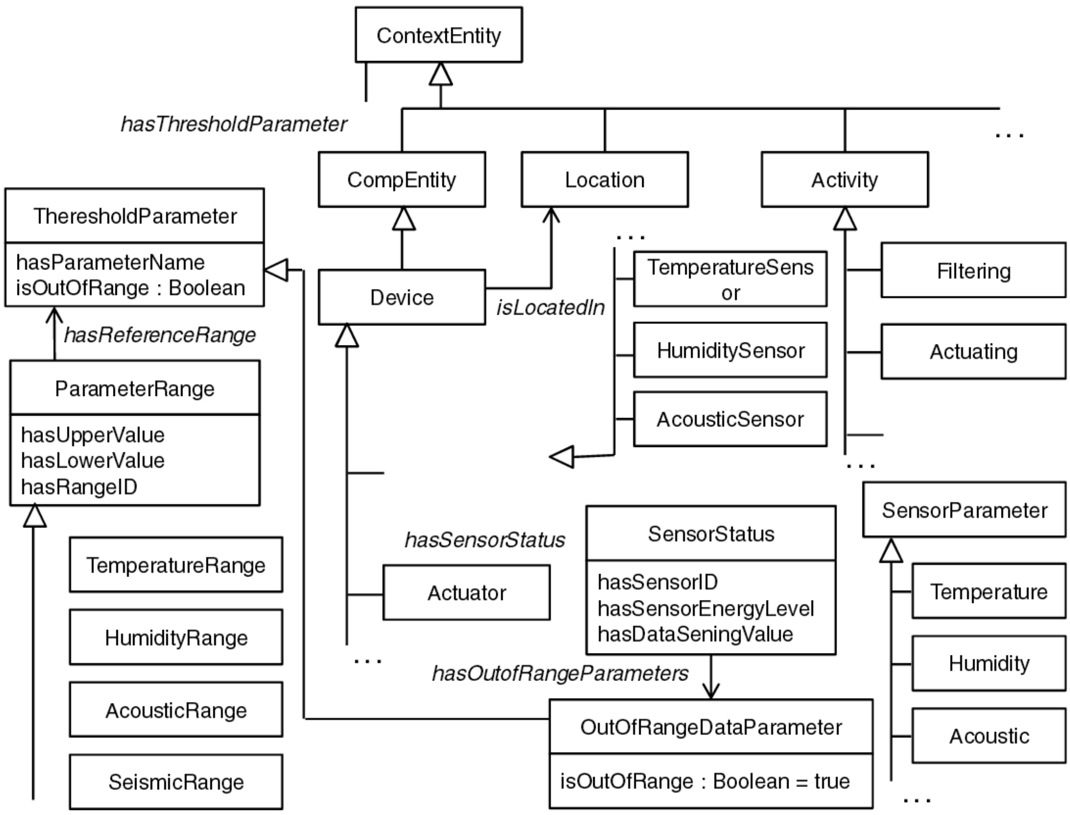


Figure 3.4 Context-aware management ontology. Source: [118]

In [119], a scheduling approach for distributed Description Logic (DL) reasoning is proposed. Unlike [118], where context reasoning is performed centrally at the base station, here the reasoning tasks can be offloaded to a number of sensor nodes in WSN based on context parameters such as resources of sensor nodes and network characteristics.

An ontology based abstraction to define API for managing resource-constrained networks such as WSNs is proposed in [120]. The network management tasks can be separated from the user level applications, i.e. an application does not require managing, but only utilising the underlying network. This can allow development of applications to be independent from the characteristics of the underlying networks. However, this service ontology is designed for networks with IP functionality, as shown in Figure 3.5 as well as only presentation context that can be deduced and exchanged directly with any of the upper-level network management and application tasks, which may limit applicability of this ontology in WSNs.

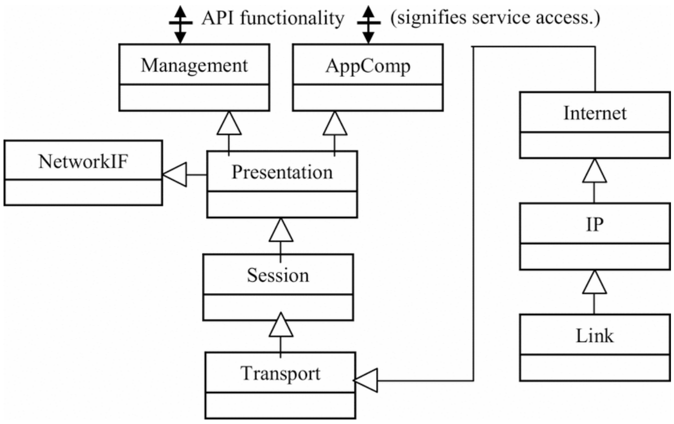


Figure 3.5 Service ontology of M2M networks. Source: [120]

In [121], an ontology based Network Management System (NMS) is proposed for heterogeneous multi-tier networks (HMN). NMS is developed in particular for topology control of a HMN, whose management complexity increases with network size, number of tiers, and device types. With an ontology based management model, the management complexity can be reduced. As shown in Figure 3.6, the core component of NMS is the Ontology Sub-system, which consists of Ontologies, Knowledge Base and Reasoner. A unique ontology design is required for each device type. Some simple ontologies can be derived from concepts of other individual ontologies through ontology mapping. The Knowledge Base stores ontologies and their instances. An instance is created by collecting all necessary device attributes with raw data from heterogeneous multi-tier networks through the Ontology Instances Inference and Management Protocols. Multiple instances of a single ontology type can be created while each instance is a representation of context information for a particular device. The Reasoner can allow the NMS to interact with the Knowledge Base using DL. However, the NMS does require user involvements for managing a network.

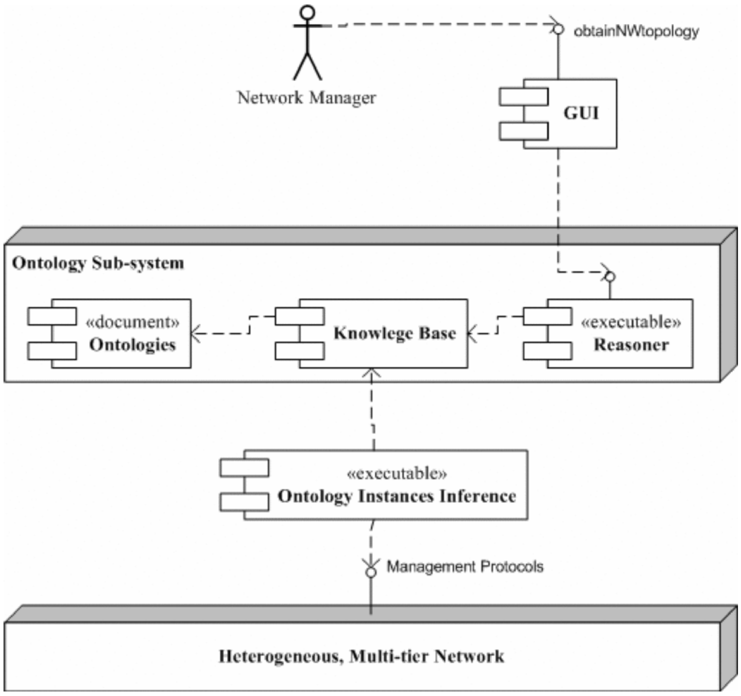


Figure 3.6 NSM system diagram. Source: [121]

Discussion and Analysis

Context modelling is a necessary part of many context aware systems. There are ontology based context modelling mechanisms proposed for context aware applications, but few exist for context aware networking, particularly for WSNs.

In these mechanisms, context can be inferred from various information sources or characteristics of a WSN, such as sensor data [118], protocol stack [120], or structure of the network [121]. Unlike traditional network management, where control is established by sending numeric commands, the context aware network management can manage a network according to any contextualised information, e.g. network structural context for topology management in [121].

To the best of our knowledge, there is no existing work on utilising user-centric AmI context for the management of WSNs, which is a research direction pursued in this thesis. Therefore, not only the AmI application, but also its underlying WSN, can be optimised according to the state of the users and their surroundings.

3.5 Context-aware Cross Layer Designs in WSNs

Many existing cross layer protocols are designed to exchange layer specific information between non-adjacent layers. However, they often ignore some important information such as context information, which can be relevant for network optimisation. In this section, a number of representative context aware cross layer designs in WSNs are reviewed.

The CIVIC routing protocol [122] for WSNs adapts the routing mechanism to the node power level and context information of a network. CIVIC is a location-based hybrid protocol, which takes advantage of both proactive and reactive routing mechanisms. It exploits the *meta* data context information from applications for the purposes of application data security and compression. By adapting its routing to the application level contexts, CIVIC can select paths based on different security and priority levels for efficient data transmissions. However, CIVIC does not specify how it structurally organises the contexts, and this may limit its applicability.

In [123, 124], the Context-Aware Clustering Hierarchy (CACH) routing protocol is proposed for WSNs. In CACH, a network is clustered based on the detected environment contexts. A cluster is formed by a group of sensor nodes with their sensor readings in a similar range. A sensor node transmits its sensor data to the CH only if the current sensor reading is different from its previous reading. Each CH aggregates the sensor data within the cluster and forwards it to the sink in one packet. CACH can improve energy efficiency by clustering the network. However, due to the clusters being formed according to sensor readings that may be non-uniform throughout the network, the cluster size may differ significantly between clusters, resulting in unbalanced traffic load and energy usage within the network.

In [9], the authors proposed a Multi-Path Multi-Priority (MPMP) transmission scheme for wireless multimedia sensor networks. The MPMP includes a Two-Phase Geographic Greedy Forwarding (TPGF) multi-path routing protocol that discovers all available node-disjoint paths between nodes, and a Context-Aware Multi-path Selection algorithm (CAMS) that determines the right number of paths from the available paths for multimedia communication in WSNs. In MPMP, the video stream is separated into audio and image sub-streams, each can be assigned with a different priority under different application scenarios. Based on the contexts deduced by a sensor node from its image and audio data, which are the surrounding brightness and noise levels of the sensor node, CAMS can assign an appropriate priority to each data stream, and select a suitable number of paths from the available paths to guarantee a certain performance during their transmission. The CAMS only consider the data value from the sensors (representing brightness and noise levels) as context information. This can cause problems such as a difficulty in distinguishing between situations with identical data value in different scenario.

An energy-efficient Context Adaptive MAC (CA-MAC) protocol for WSNs is presented in [112]. CA-MAC uses the node buffer states and the priority context of upper layer packets to amend the transmission schedule of a node by putting the node into the sleep mode whenever possible. CA-MAC stores the upper layer data packets, e.g. application data, in a node's local buffer. If the local buffer level is not over a threshold, or the packet has low priority context, CA-MAC will not contend for access to the channel. The buffer threshold value is determined by CA-MAC, which increases with decreasing hop distance of a node to the sink to improve energy efficiency. CA-MAC only considers application level packet priority as a context, and may not behave well under a long burst of high priority packets.

In [125], a generic framework for context aware routing in WSNs is proposed. The routing mechanism is based on a reactive table-driven routing approach, similar to AODV, but uses different criteria for making decisions on route selection and route request forwarding. A Multi Criteria Context-based Decision (MCCD) function is used for making such decisions. The MCCD is a multiplicative decision function that computes the utility value based on multiple weighted context criteria. The Reactive Environmental Monitoring Aware Routing (EMA) is a prototype implementation of the proposed routing framework that considers the node state, received signal strength, and hop count of a route as context criteria for making routing decisions. Reactive EMA only considers node and network level contexts, but not the context information encapsulated in the packets from the application.

Wireless Adaptive Routing Protocol version 5 (Warp-5) presented in [126] can provide route decision makings in heterogeneous networks, e.g. WSNs, by avoiding the paths affected by high network congestion and communication noise. Warp-5 allows nodes to learn the channel noise context based on transmission rates from received MAC frames. This channel noise context is further considered by an AODV-alike routing protocol during its routing discovery and data forwarding stages. Therefore, Warp-5 can allow more reliable routing paths to be constructed by reducing overall network congestion. However, Warp-5 merely supposes the

MAC level transmission rate as the only network congestion factor while other measures such as packet loss and retransmission rates of the network are not considered.

In [127], an adaptive service-oriented multipath AODV (SM-AODV) routing scheme is proposed to provide better adaptive congestion control and rate adjustment for balancing routing loads in WSNs. This scheme is a multipath version of AODV that can discover multiple routes according to link states of the intermediate nodes. Its congestion control is based on buffer level and channel load, involving the following three stages: i) congestion detection; ii) congestion control and notification; and iii) congestion cancellation and load adjustment. Therefore, SM-AODV can detect congestions and adjust loads of the multiple paths adaptively, but only according to the network context.

A cross-layer architecture to support network level context awareness in communications is proposed in [53]. This architecture can be applied to existing layered designs by integrating several key components of the architecture to the current protocol stack. This architecture categorises context according to its availability: a “local view” context represents context that can be inferred within a sensor node, while a “global view” context represents context that comes from external of the node. The structure of this architecture includes a Contextor on each layer to contextualise protocol characteristics of that layer, and a Knowledge Plane to store both the “local” and “global” view contexts in a sensor node, as shown in Figure 3.7. The Knowledge Plan also allows the Contextors of a sensor node to optimise its protocols according to the available local and global view contexts. By unifying context representation and cross layer functions, this architecture allows local and global context information to be exchanged to achieve better cross layer optimisation outcomes.

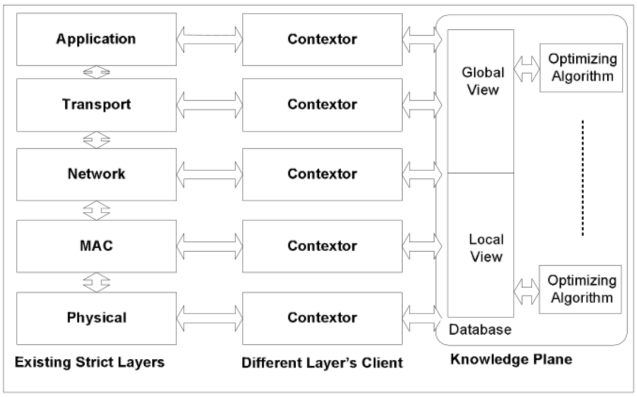


Figure 3.7 Generic context-aware cross-layer architecture. Source: [53]

Discussion

Each individual protocol layer can generate specific information to represent its operational states. Contextualising such information in a standard way can facilitate the sharing between different protocols of their operational states such as energy and network connectivity. Therefore, a layered protocol is only required to handle a single contextualisation standard instead of multiple protocol formats and structures. In this way, cross layer interactions can be enhanced, i.e. any cross layer operations can be performed according to standardised context instead of protocol specific information.

Many non-context aware cross layer designs are only using locally available information of a node. However, many context-aware cross layer designs are also using non-local or external information such as application or network context. This wider view of their environment rather than individual pieces of protocol or node level contexts can enable nodes to optimise their operations such that the entire network will benefit overall. Competing actions such as optimising one layer at the expense of another can be possibly avoided. In general, better cross layer design outcomes can be expected from the usage of both local and global context information.

Table 3-2 summarises the comparison of the discussed context-aware cross layer designs in terms of their design type, design objective, context source(s), and the global context(s) required.

Table 3-2 Comparison of context aware cross layer designs

Design	*Design type*	*Objective*	*Context source(s)*	*Type of global context(s)*
CIVIC	routing protocol	reduce network overload and improve network efficiency	node energy/network characteristic/application	application security/priority levels
CACH	routing protocol	improve network energy efficiency	sensor reading	environment states
CAMS	path selection algorithm	maximise data-gathering efficiency	sensor reading	environment brightness &noise levels
CA-MAC	MAC protocol	improve energy efficiency of WSNs	packet content/hop distance	importance of data packets
Warp-5	routing protocol	provide better routing decision	MAC- and NET- level characteristics	network channel and route states
[127]	multipath routing scheme	load balancing on paths according to adaptive congestion control and rate adjustment	channel/buffer load	node link conditions
[125]	Generic routing framework	to design context-aware routing mechanisms under different application domains	node characteristics/signal strength/connectivity	node health
[53]	cross-layer architecture	to realise network context-aware communications	network characteristics	network element-related states, neighbouring node circumstances, user-related context information

3.6 Chapter Summary

From the review of the recent literature, it is observed that context awareness can enhance the outcomes of cross layer optimisation in wireless sensor networks (WSNs). However, existing works have mainly considered node and/or network level contexts, while this thesis focuses on application level ambient intelligence (AmI) context and its utility in cross layer optimisation in wireless sensor networks. To achieve this goal, the next four chapters present the research undertaken in this thesis: network structuring for context storage; data-centric communication for context exchange; context modelling and reasoning for wireless sensor network management; and ambient intelligence context-aware cross layer optimisation in wireless sensor networks.

Chapter 4 Network Structuring and In-Network Virtual Storage Formation in WSNs

4.1 Introduction

The new generation of NAND flash memories has significantly increased the data storage capacity of WSN nodes [128]. This can make the concept of in-network storage (INS) more realistic where data users are likely to come from within, rather than external to the network [4]. For example, ambient intelligence (AmI) applications can utilise the INS approach to retrieve and process the data stored on sensing and computing devices embedded within their physical environments [1]. Under the concept of INS, selective sensor nodes are organised to provide the functionality of data storage for other nodes in the same network. Data storage can be performed by mapping the sensed data to the storage nodes using, for instance algorithms based on DHT [129] or GHT [130]. Data retrieval can be performed according to the data queries via content filtering, e.g. based on keywords, attributes, or other content-specific criteria.

Existing INS proposals for WSNs have focused on data aggregation and data content filtering mechanisms [131]. Few proposals have explored the use of sensor nodes themselves to cooperatively provide a storage service to INS users [132], e.g. AmI applications. This can eliminate the need for powerful nodes as dedicated storage devices which may not be always available, such as in homogeneous WSNs. In this thesis, we adopted the concept of virtual storage unit (VSU), which refers to a single entity that can be formed by multiple sensor nodes in co-located space sharing their resources for data storage and retrieval [133].

Often, the structure of a large WSN needs to be organised via a network structuring process. Organising the network into partitions or clusters has shown to improve the energy efficiency, scalability and load balancing of WSNs [84, 134–136]. However, how the WSN is structured in terms of the number of partitions and size of the VSUs can impact on the communication efficiency, and hence energy consumption of the network. This is the challenge to be addressed in this chapter.

There are two main contributions in this chapter. The first is an analytical model devised to determine the optimal number of partitions and VSU size that jointly minimise the total communication overhead in the network in terms of hop counts, i.e. the number of hops travelled by messages for data storage and retrieval in the network. This minimum total hop count represents the theoretical lower bound (or minimum possible amount) of communication overhead that can be achieved by a routing algorithm.

The second contribution in this chapter is a network partitioning and VSU formation algorithm. This algorithm uses node connectivity and relative node location in terms of hops to several reference nodes for structuring a network into balanced partitions, each with a VSU formed from a group of co-located nodes.

The rest of this chapter is organised as follow. Section 4.2 reviews related work. Section 4.3 describes the network model used in this research. Section 4.4 presents the proposed analytical model to find the optimal number of partitions and VSU size. Section 4.5 further presents the proposed network partitioning and VSU formation algorithm. Two variants of the algorithm will be presented. The first is based on only relative location of the nodes. The second is an improved algorithm that considers both location and energy level of the nodes. Finally, Section 4.6 concludes this chapter.

4.2 Related Work

A popular research topic in data-centric communication is data-centric routing, in which the packets are routed according to the content of the sensor data, for instance based on query interests [137, 138]. Another popular research topic is data-centric storage, which considers the data centricity characteristic as an advantage over other types of storage mechanisms. In these works, there is a temporal and spatial decoupling of the relation between the data producers and data consumers, and the procedures to store and retrieve data are based on data content [129].

Several methods have been proposed to store and retrieve data for INS systems [139–147]. Under existing data centric storage designs, there are two popular approaches: data indexing and storage mapping. In the data indexing approach, such as [141, 142], no actual data, but only *metadata* of the data, e.g. data type, location and time of creation, is transmitted and indexed by some or all the nodes in a network. The size of the metadata is relative small compared to the actual data. Therefore, it costs less energy to transmit. The actual data is only transmitted when it has been requested. Retrieving the sensor data can be performed by querying the metadata to discover on which node the data is stored.

On the other hand, in the storage mapping approach, the sensor data is transmitted and stored on particular storage nodes. A common method is using geographic hash tables [145, 146], whereby a piece of data is mapped by a hashing function to an appropriated node for storage. Data retrieval can be performed by searching the hash key to locate the node that stores the data.

Some research works have been proposed to optimise the network structure for INS, but they require additional information or steps to operate, such as knowing the node position [148], requiring geographic routing [149], or maintaining multiple copies of every single event data in a network [150].

4.3 Network Model

Given that communication is costly for sensor nodes, INS can be a suitable solution that stores the generated sensor data within the network and allow INS users, e.g. AmI applications, to decide which data is significant to retrieve. However, constrained resources, such as limited storage and energy capacity, make a single sensor node unlikely to maintain all the data alone in the network. Therefore, the nodes must cooperate together to provide the capacity of data storage. In the absence of a single dedicated resourceful device, storage and processing resources from multiple co-located sensor nodes can be harnessed to create a VSU, in a way similar to the concepts of virtual MIMO [8]. Multiple VSUs can be distributed in the WSN to cooperatively store data from sensors and process data requests from WSN applications.

The VSU design presented in this chapter uses an approach similar to storage mapping in Section 4.2, along with network structuring techniques to determine the location of VSUs. The role of a VSU is similar to a cluster head, which can achieve better energy efficiency in a network, as there is no need to send data queries all over the network to find the required data. Unlike existing designs that require absolute location information, such as GPS information, this design requires only relative location information based on the Anchor-Free Localization (AFL) algorithm [12], which expresses a node's location in terms of hops to several reference nodes.

Figure 4.1 shows the network topology considered in this work. The entire network is split into c^2 grid partitions, and a VSU is formed by a group of sensor nodes in the centre of each partition. The notations used and assumptions made are as follows:

- There are a total of $L \times L$ grids, each with a size of 10×10 m^2 and only one node in the centre;

- The network is to be split into $c \times c$ partitions, where c is a parameter to be optimised;

- Each partition has $N \times N$ sensor nodes of which $n \times n$ in the centre forms a VSU;

- Node transmission range is set to 14 m;

- Each sensor node has the role of either an ordinary node or a VSU member node.

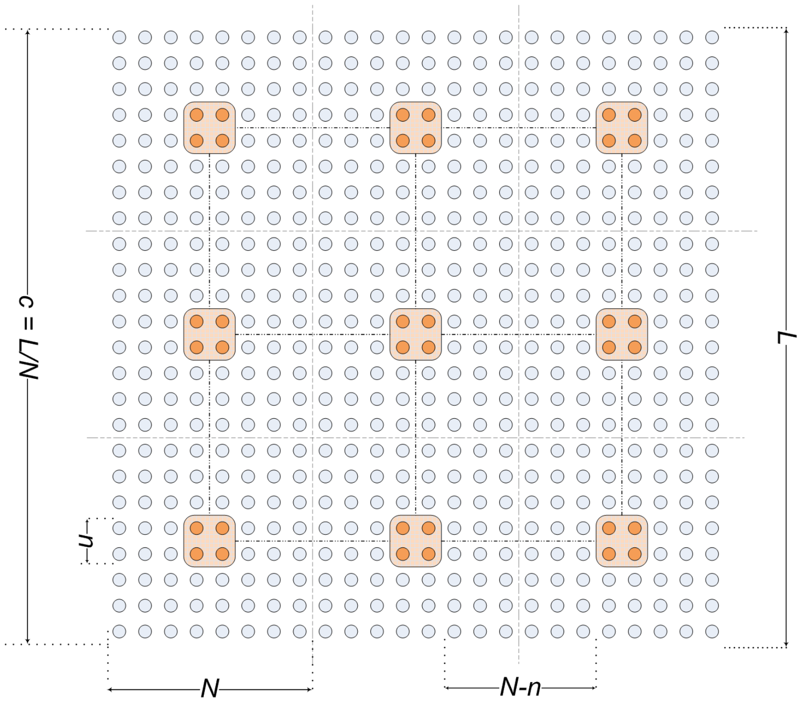


Figure 4.1 Network topology

In this network model, three types of communications can exist: node-VSU, intra-VSU, and inter-VSU communications:

- Node-VSU communications represent the communications between ordinary nodes and their nearest VSU node in the same partition. The total hop count for intra-partition communications is denoted as H_c^P.

- Intra-VSU communications refer to communications between all constituting nodes of a VSU. The total hop count for intra-VSU communications in a partition is denoted as H_n^P. With n^2 nodes in a VSU, each requiring n^2 hops of communications to all other nodes in a VSU, H_n^P can be given as:

$$H_n^P = n^4 \qquad (4\text{-}1)$$

- Inter-VSU communications occur when data exchange takes place between different VSUs, which are connected by links that constitute a minimum spanning tree [151]. The total hop count for inter-VSU communications H^N is given as:

$$H^N = c^2 \times (c^2 - 1) \times (N - n) \qquad (4\text{-}2)$$

4.4 Analytical Model

Based on the network model given in Section 4.3, an analytical model is derived to determine the optimal number of partitions and VSU size that jointly minimise the total communication overhead in the network in terms of hop counts. It should be noted that our analytical model is not advocating or limiting the network to the use of shortest path routing algorithms. In fact, any non-shortest path algorithms, such as for achieving load-balancing or energy efficiency can be used. However, the communication overhead incurred by these algorithms will not be lower than the theoretical lower bound achieved by a shortest path algorithm in a network structured according to the optimal number of partitions and VSU size derived from our model.

This analytical model derives the value of c^2 which represents the optimal number of partitions required for minimising the overall communication overhead. The total hop count of the network H_{total} can be expressed as:

$$H_{total} = H^P \times c^2 + H^N \qquad (4\text{-}3)$$

where H^P is the total hop count of a partition. H^P can be further defined as:

$$H^P = H_c^P + H_n^P \qquad (4\text{-}4)$$

With reference to Figure 4.2, the node-VSU communication hop count H_c^P can be calculated as:

$$H_c^P = \sum_{i=1}^{8} H_i \qquad (4\text{-}5)$$

where H_1 to H_8 refers to the sum of hop counts from ordinary nodes in region 1 to 8 to their nearest VSU node. Equation (4-5) can be derived based on region similarity:

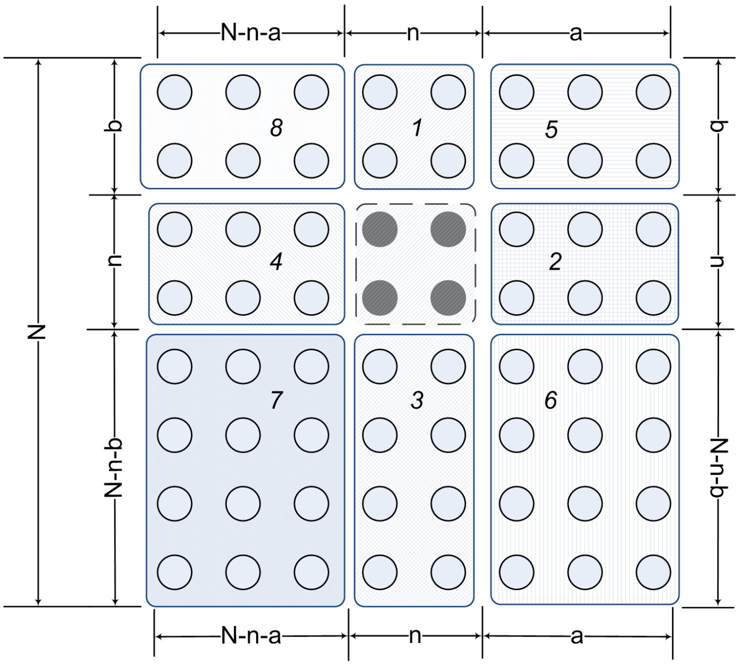


Figure 4.2 Node-VSU hop counts

$$H_c^P = \sum_{i=1}^{8} H_i = \sum_{i=1}^{4} H_i^A + \sum_{i=5}^{8} H_i^B \tag{4-6}$$

where H_i^A and H_i^B denote the hop count in each region i=1–4, and i=5–8, respectively. They can be found as follows:

$$\sum_{i=1}^{4} H_i^A = n \times \left(\sum_{j=1}^{b} j + \sum_{j=1}^{a} j + \sum_{j=1}^{N-n-b} j + \sum_{j=1}^{N-n-a} j \right) \tag{4-7}$$

$$\begin{aligned}\sum_{i=5}^{8} H_i^B = &\sum_{j=1}^{a}\sum_{k=1}^{b}(j+k) + \sum_{j=1}^{a}\sum_{k=1}^{N-n-b}(j+k) + \\ &\sum_{j=1}^{N-n-a}\sum_{k=1}^{N-n-b}(j+k) + \sum_{j=1}^{N-n-a}\sum_{k=1}^{b}(j+k)\end{aligned} \tag{4-8}$$

Therefore, the total node-VSU hop count of a partition H_c^P can be derived by substituting (4-7) and (4-8) into (4-6):

$$\begin{aligned}H_c^P = &N^3 - 2nN^2 - bN^2 - aN^2 + N^2 + \\ &n^2N + bnN + anN - nN + b^2N + a^2N\end{aligned} \tag{4-9}$$

Based on even or odd values of N and n, there are two possible locations of the VSU in a partition, as shown in Figure 4.3.

When a VSU is in the absolute centre of a partition as shown in Figure 4.3(a), the a and b values in Figure 4.2 are given by:

$$a = b = \frac{N-n}{2} \quad if\ (N-n)\ is\ even \tag{4-10}$$

When a VSU is not in the absolute centre of a partition as shown in Figure 4.3(b), the corresponding values of a and b are:

$$a = \frac{N-n+1}{2}, b = \frac{N-n-1}{2} \quad if\ (N-n)\ is\ odd \tag{4-11}$$

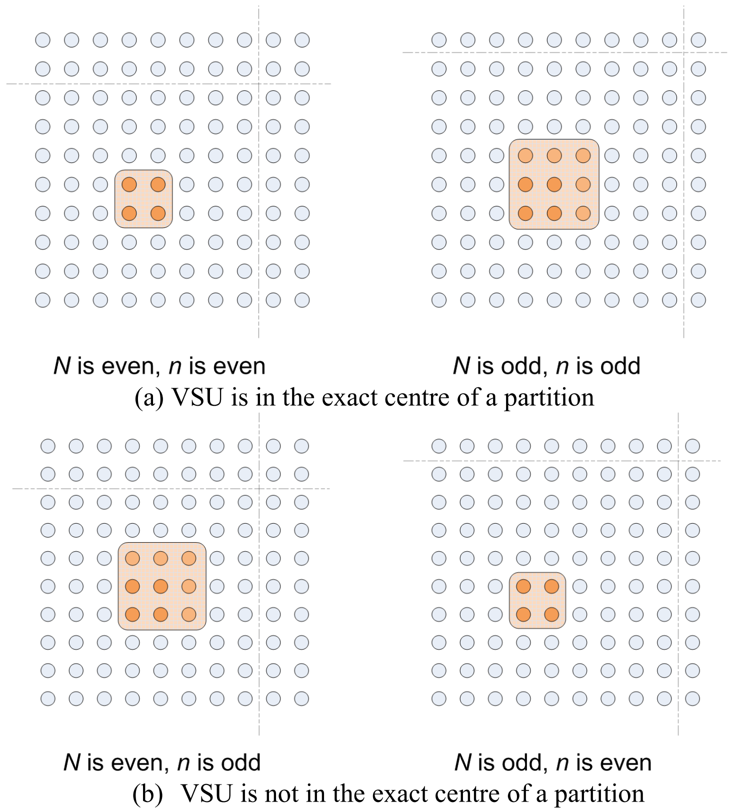


Figure 4.3 Possible locations of a VSU in a partition

Hence, the total hop count in a partition H^P can be calculated by substituting (4-10) and (4-11) into (4-9) to find H_c^P, and further substituting it into (4-4):

$$H^P = \begin{cases} \frac{N^3}{2} - nN^2 + N^2 + \frac{n^2 N}{2} - nN + n^4 & if\,|N-n|\,is\ even \\ \frac{N^3}{2} - nN^2 + N^2 + \frac{n^2 N}{2} - nN + \frac{N}{2} + n^4 & if\,|N-n|\,is\ odd \end{cases} \tag{4-12}$$

The optimal value of n can be found by:

$$\frac{\partial}{\partial n} H^P = 0 \Rightarrow n \approx \begin{cases} \sqrt[3]{\frac{N^2}{4}} & if\,|N-n|\,is\ even \\ \sqrt[3]{\frac{N^2}{4}} & if\,|N-n|\,is\ odd \end{cases} \tag{4-13}$$

Thus, the optimal n can always be obtained as a function of $\boldsymbol{N}$ as $n \approx \sqrt[3]{\frac{N^2}{4}}$, regardless of the VSU's location. With $N = \frac{L}{c}$ as shown in Figure 4.1, the total hop count of a network H_{total} in (4-3) can be found as:

$$H_{total} = L^2\delta^3 - \frac{3L^2\delta^2}{4} + L^2 + \frac{L^2\delta}{4} - cL\delta^2 + c^3 L - c^4\delta^2 - cL + c^2\delta^2 \quad when\,|N-n|\,is\ even \tag{4-14}$$

$$H_{total} = L^2\delta^3 - \frac{3L^2\delta^2}{4} + L^2 + \frac{L^2\delta}{4} - cL\delta^2 + c^3 L - c^4\delta^2 - \frac{cL}{2} + c^2\delta^2 \quad when\,|N-n|\,is\ odd \tag{4-15}$$

where $\delta = (\frac{L}{2c})^{\frac{1}{3}}$. For a network with known L^2 number of nodes, the partition size-related parameter c is the only unknown variable in (4-14) and (4-15). However, as finding the optimal c that minimises H_{total} is not analytically tractable, a numerical method is implemented in Matlab to determine the optimal c. Figure 4.4 shows the associated pseudocode.

```
Require: network length L
Ensure: optimal c for a given L
min_hops← 0
optimal_c← 0
for c, c²∈Z⁺do
   N ←L/c
   n = (N²/4)^(1/3)
   if N < n then
      break
   end if
   if |N – n| is even then
      total_hops←(4-14)
   else
      total_hops←(4-15)
   end if
   if min_hops = 0 ∨ total_hops < min_hops then
      min_hops←total_hops
      optimal_c←c
   end if
end for
c←optimal_c
return c
```

Figure 4.4 Pseudocode to determine optimal *c*

Evaluation of the Analytical Model

Numerical simulations are performed in Matlab. The network is set up according to Section 4.3. The total number of nodes in the network (L^2) is varied from 25 to 2500 in steps of one node. The results are shown in Figure 4.5 – Figure 4.7.

Using the numerical method outlined in Figure 4.4, the optimal number of partitions (c^2) is found for different total number of nodes (L^2), shown by the line with circular markers in Figure 4.5. An approximation function for c^2, as shown in (4-16), is then obtained by curve fitting the line with a power law regression model in Matlab and plotted in Figure 4.5. Equation (4-16) provides the means to find the optimal number of partitions (c^2) that minimises the total hop count (H_{total}) for any given network size, i.e. the total number of nodes (L^2). The accuracy of the approximation can be seen by the close agreement between the result obtained by numerical simulation and that given by (4-16):

$$c^2 = 0.07343 + 0.4988 \times (L^2)^{0.4814} \tag{4-16}$$

Figure 4.6 shows the total hop count H_{total} for different network sizes L^2 under the effect of a different number of partitions c^2. The c value for each network size is varied about the optimal c by ± 2.The optimal n value is used and obtained from (4-13) for each network size. The effect of a large c value for a given network size is a large number of small partitions, and vice versa. Creating too many small partitions can increase the inter-VSU hop count H^N, and consequently H_{total}. In contrast, having few but large partitions can increase H^P, and consequently again H_{total}. With an optimal c setting, the right number of partitions of the right size can be achieved to minimise H_{total} of a given network.

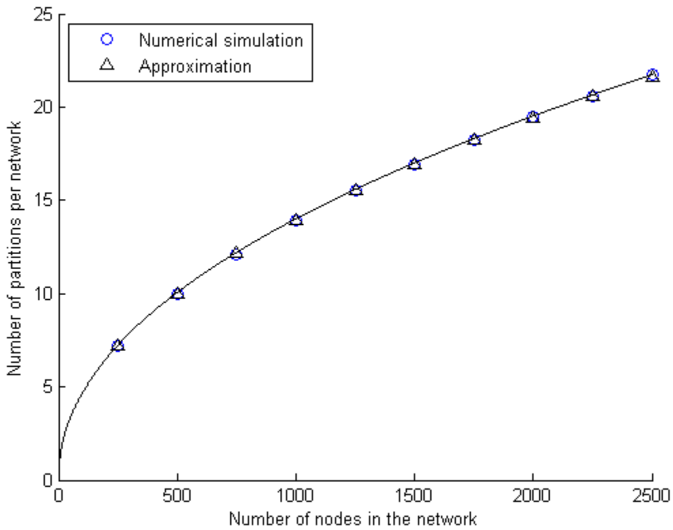


Figure 4.5 Comparison of simulated and approximated results

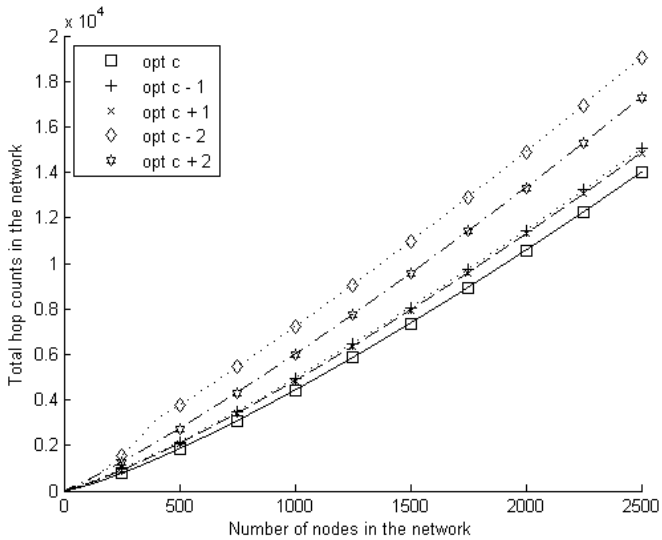


Figure 4.6 Effect of c (where c^2 = number of partitions) on H_{total} with optimal n for each network size

Figure 4.7 shows the total hop count H_{total} for different network sizes L^2, but under the effect of different n values that define the VSU size in a partition. The optimal n value is found by (4-13), which depends on N, and consequently c. The n value for each network size is similarly varied about the optimal n by ± 2. The optimal c value is used and obtained from

(4-16) for each network size. The effect of a large (small) *n* value for a given partition size is a large (small) VSU that can increase (decrease) intra-VSU hop count H_n^P but decrease (increase) node-VSU hop count H_c^P. With an optimal *n* setting, the right amount of H_n^P and H_c^P can be achieved to minimise H_{total}, as shown in Figure 4.7.

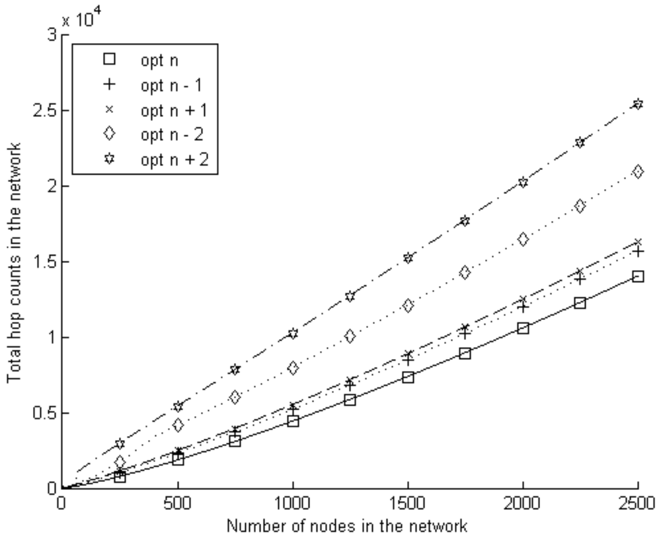
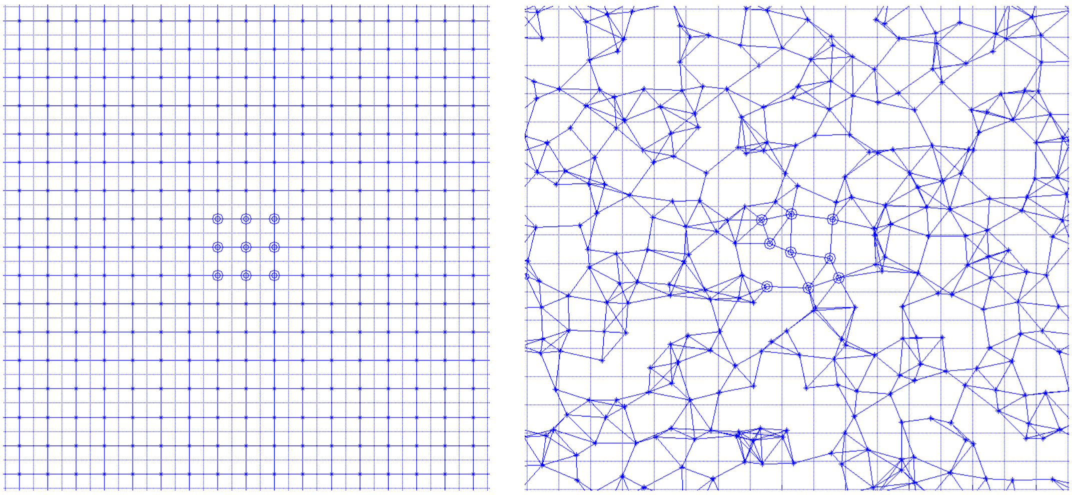


Figure 4.7 Effect of *n* (where n^2 = VSU size) on H_{total} with optimal *c* for each network size

To relax the requirement for nodes to be positioned in a rigid grid layout, we allow the location of each node to be varied by a random amount around its original position. Similar to [152], if the original position of a node is (*X*, *Y*), its new position will be $((1+\varepsilon\times x/2)\times X, (1+\varepsilon'\times y/2)\times Y)$, where ε and ε' are two random variables uniformly distributed between $\pm R$, where $R \in \{0, 0.5, 1\}$ denotes the amount of position randomness, while *x* and *y* denote the fixed size of a grid in X-, and Y-direction, respectively. A set of simulations is then performed to investigate the effect of randomness of node positions on the accuracy of the optimal formula given by (4-16).

Figure 4.8 presents two snapshots of a partition with different *R* values. Table 4-1 presents the total network hop counts under different *L*, *c*, and *R* values. The result shows that up to *L* = 32, which represents a network of 1024 nodes, the analytical model can still offer an optimal result, i.e. the derived number of partitions and VSU size can minimise the total hop counts in the network. However, beyond *L* = 32, the derived number of partitions and VSU size may not result in minimum total hop counts. For example, when *L* = 35 and *R* = 50%, the total network hop count based on the 'optimal' *c* setting is higher than the one based on the optimal *c*+2 setting as shown in Table 4-1.



(a) when R = 0 (b) when R = 1

Figure 4.8 Node randomness

Table 4-1 Total network hops with different *L*, *c*, and randomness *R* values

L	R = 0%			R = 50%			R = 100%		
	A	*B*	*C*	*A*	*B*	*C*	*A*	*B*	*C*
10	216	247	777	232	329	784	257	436	762
15	621	861	934	653	942	811	688	914	1348
20	1072	1440	2123	1148	1493	1529	1124	1743	2336
25	2308	2626	3197	2487	3011	3506	2420	3394	4501
30	2762	4028	6317	2687	4018	6103	3013	4453	6707
32	2921	3644	12046	3104	3502	10731	3824	4053	11127
35	6943	8355	7355	**6552**	9949	**6510**	7750	11084	7898

A: opt *c*, *B*: opt *c* + 1, *C*: opt *c* + 2

4.5 Network Partitioning and VSU Formation Algorithm

This section presents the proposed algorithm for network partitioning and VSU formation. Firstly, the methodology for computing the distance between a node pair in the network based on their AFL information is described. Next, two variants of the proposed algorithm are presented: one that only considers node location information, and another that considers both node location and node energy level information. This is followed by a performance evaluation and result analysis of the proposed algorithm.

4.5.1 Network Distance and Node Localisation

An undirected graph $G = (V,E)$ represents the communication graph of a network, where V is the vertex set for all sensor nodes in the network and E is the edge set for the direct communications between two vertices, i.e. direct communication between a pair of neighbour nodes. u denotes a particular node u in vertex set V where $u \in V$. $< u,v >$ denotes as a pair of nodes u and v where $\{ u,v \} \in V$. $r(u)$ denotes as the transmission range of a node u. $d_E(u,v)$ and $\ell_1(u,v)$ denote the Euclidean distance, and taxicab distance, respectively, between $< u,v >$. A network is set up according to the network model described in Section 4.3.

4.5.1.1 Taxicab Geometry in Wireless Sensor Networks

Data transmissions in WSNs may involve multiple hops. However, after a WSN has been deployed, a multi-hop path in the physical field may not be always a straight line path. In this scenario, the distance between a node pair can be described as a Taxicab Distance [153].

In the taxicab metric, the distance between two vectors on the Euclidean plane is the sum of all paths on the vertical and horizontal segments that connect those two vectors. This can present connectivity of sensor nodes in WSNs, as shown in Figure 4.9. In this case, there is no direct link between nodes A and D, and communications must go through some relay nodes, i.e. nodes B and C. The distance between nodes A and D, $d_E(A,D)$, can only be presented as:

$$d_E(A,D) = d_E(A,B) + d_E(B,C) + d_E(C,D)$$

Definition 4-1. A communication graph of a network is the result of a set of sensor nodes in a network such that a direct communication can be made between two sensor nodes if and only if their Euclidean distance is no more than both of their transmission ranges.

Remark. Without loss of generality, the mechanism described in this chapter is designed for static WSNs. It assumes that all the nodes cost a similar amount of energy to transmit the same amount of data, and have the similar transmission range.

Definition 4-2. The taxicab distance of two neighbouring nodes that are within the direct transmission range of each other, is equal to 1 hop.

$$\ell_1(u,v) = 1 \quad iff : r(u) \le d(u,v), r(v) \le d(u,v)$$

Remark. A taxicab distance of 1 hop depends on transmission power of the devices. Due to interference of radio frequency signal, its range accuracy may not be precisely enough for the sub-meter range scenarios. Therefore, this work is designed for the systems that can tolerate with moderate accuracy on the distance measurement, such as AmI.

Definition 4-3. A path denoted as $\overrightarrow{u,v}$ between a pair of nodes $< u,v >$ is a set of nodes in a sequence $\sigma(u,v) \subset V$, and $\sigma(u,v) \setminus \{ u,v \}$ defines the set of relay nodes in path $\overrightarrow{u,v}$. The number of nodes of a path $\overrightarrow{u,v}$ is denoted as $| \sigma(u,v) | = \aleph_0$.

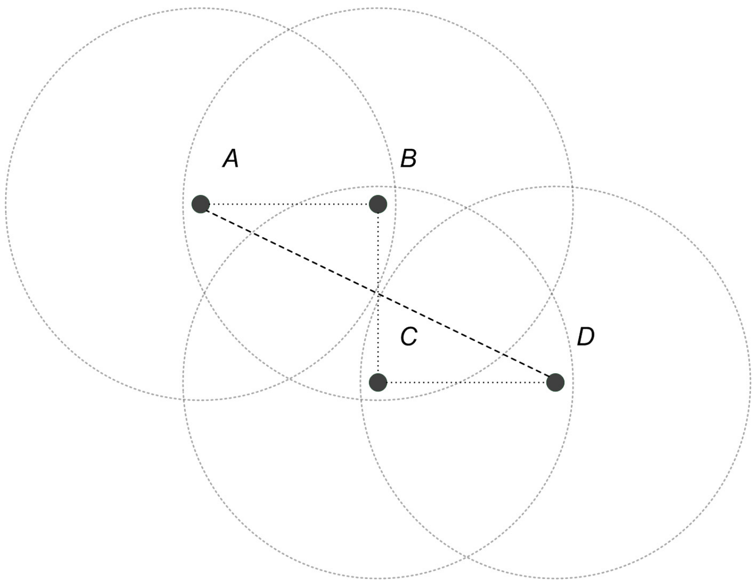


Figure 4.9 Node connectivity

Lemma 4-1. For a given path $\overrightarrow{u,v}$, the taxicab distance in terms of hops is given as $\ell_1(u,v) = | \sigma(u,v) | - 1$.

Proof: A path $\overrightarrow{u,v}$ is a joint sequence of the direct connected paths from the set *E* to establish connectivity between $< u,v >$. The set of the nodes σ for the path $\overrightarrow{u,v}$ can be defined as:

$$\sigma(u,v) = u \cup x_1 \cup \ldots \cup x_j \cup v$$

where $\{ u, x_1, \ldots, x_j, v \} \in V$. The nodes x_1 to x_j are the relay nodes of the path $\overrightarrow{u,v}$. The taxicab distance $\ell_1(u,v)$ of the path $\overrightarrow{u,v}$ is calculated as

$$\ell_1(u,v) = \ell_1(u,x_1) + \ell_1(x_1,x_2) + \ldots + \ell_1(x_j,v) = |\sigma(u,v)| - 1$$

■

Remark. In this chapter, a path $\overrightarrow{u,v}$ represents the shortest path between $< u,v >$. As the communication graph is undirected, it is also assumed that $|\sigma(u,v)| = |\sigma(v,u)|$, i.e. $\ell_1(u,v) = \ell_1(v,u)$, which means the taxicab distance for both paths $\overrightarrow{u,v}$ and $\overrightarrow{v,u}$ is equivalent. However, it is also possible to have $\exists\sigma(u,v)\Delta\,\sigma(v,u) \neq \emptyset$, which indicates that path $\overrightarrow{u,v}$ and $\overrightarrow{v,u}$ can involve different relay nodes, i.e. some nodes are the relay nodes of one path but not the other.

4.5.1.2 Taxicab Distance Calculation from Node Location

In this section, the distance between a node pair is calculated according to AFL information of the nodes. AFL is a decentralised localisation mechanism to determine the relative location of a sensor node in a network. There are 5 reference nodes, of which four are located near the four corners of the network and one is located at the centre of the network. All other nodes can locate themselves based on the number of hops to the reference nodes.

In AFL, all nodes in a network can know their relative locations in terms of hops to the four corner nodes. The relative location of a node u in a network is denoted as $T_\Xi{}^u(\xi_A, \xi_B, \xi_C, \xi_D)$, where A, B, C, and D are the four corner nodes. An individual hop distance to a corner node is denoted as $T_\Xi{}^u(\xi_i)$, where $\{ \xi_i ; i \in \{ A,B,C,D \} \}$ is the taxicab distance in terms of hops to a particular reference corner node.

Figure 4.10, as an example, shows how a node can locate itself in the network based on the AFL information. The node E is a node that can be located anywhere in the network. Under the taxicab geometry, multiple paths can co-exist between a node pair. In this example, there can be 3 paths between $< E,B >$. Each path of $\overrightarrow{E,B}$ can involve different relay nodes. However, all paths can have the same taxicab distance between $< E,B >$ according to Lemma 4-1.

As all nodes in a network can know their relative locations, it is possible for each node to estimate how close it is to the centre of the network.

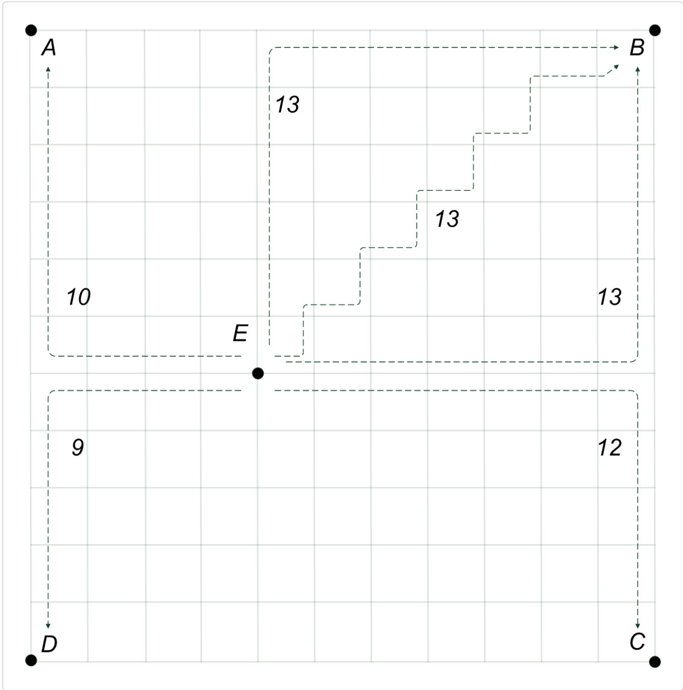


Figure 4.10 The location of the node *E*

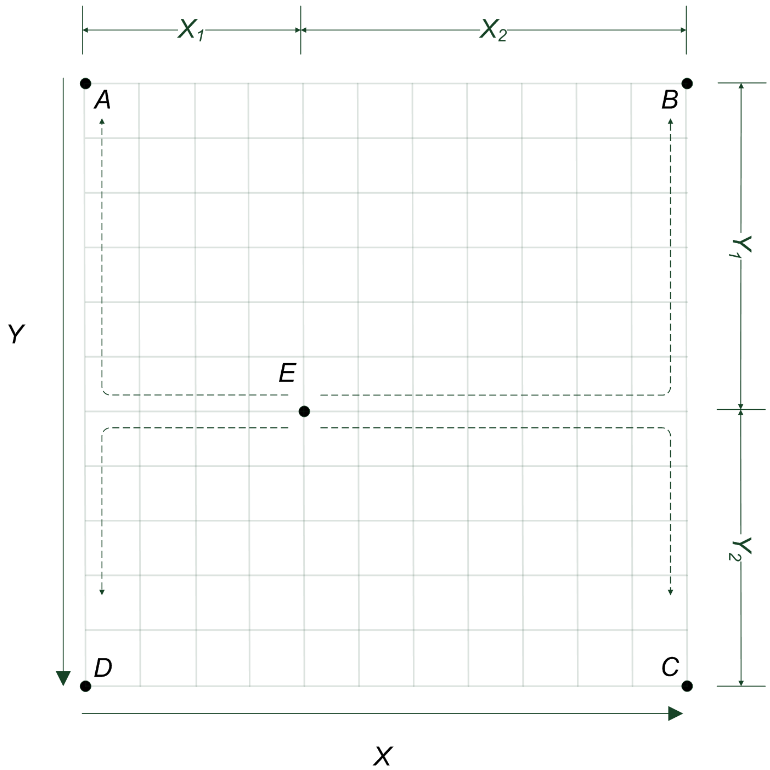


Figure 4.11 Localising a node in the network

Lemma 4-2. If a node u knows its relative location $T_{\Xi}^{u}(\xi_A, \xi_B, \xi_C, \xi_D)$ in a network, the node u can know its relative node position denoted as X^u and Y^u at the X and Y directions of the network respectively.

Proof: As shown in Figure 4.11, the relative node position at the X direction X^E for the node E is given as:

$$X^E = T_{\Xi}^{E}(A) - T_{\Xi}^{E}(B) = \ell_1(E,A) - \ell_1(E,B) \tag{4-17}$$

where

$$\ell_1(E,A) = X_1 + Y_1$$
$$\ell_1(E,B) = X_2 + Y_1$$

Therefore, (4-17) can be calculated as:

$$X^E = X_1 + Y_1 - (X_2 + Y_1) = X_1 - X_2 \tag{4-18}$$

Equation (4-18) is also true if the reference corner nodes C and D are chosen:

$$\begin{aligned} X^E &= T_{\Xi}^{E}(D) - T_{\Xi}^{E}(C) = \ell_1(E,D) - \ell_1(E,C) \\ &= X_1 + Y_2 - (X_2 + Y_2) \\ &= X_1 - X_2 \end{aligned}$$

Similar to the node location at X direction X^E for node E, the relative node position at Y direction Y^E for E can be calculated as:

$$\begin{aligned} Y^E &= T_{\Xi}^{E}(A) - T_{\Xi}^{E}(D) = \ell_1(E,A) - \ell_1(E,D) \\ &= X_1 + Y_1 - (X_1 + Y_2) = Y_1 - Y_2 \\ Y^E &= T_{\Xi}^{E}(B) - T_{\Xi}^{E}(C) = \ell_1(E,B) - \ell_1(E,C) \\ &= X_2 + Y_1 - (X_2 + Y_2) = Y_1 - Y_2 \end{aligned} \tag{4-19}$$

■

Lemma 4-3. For a grid network, if X^u and Y^u are known by a node u, then the number of nodes on the edges of the network can be determined by u.

Remark. Let L denote the number of boundary nodes on an edge of a network, and the taxicab distance in terms of hops for the edge is given as $\ell_1 = L - 1$ according to Lemma 4-1. As a grid network, it is true that $\ell_1(A,B) = \ell_1(D,C) = \ell_1(A,D) = \ell_1(B,C) = L - 1$.

Proof: A grid network can present each edge of a network as:

$$\begin{cases} X_1 + X_2 = L - 1 \\ Y_1 + Y_2 = L - 1 \end{cases} \tag{4-20}$$

Rearranging (4-20) gives:

$$\begin{cases} X_1 = L - 1 - X_2 \\ Y_1 = L - 1 - Y_2 \end{cases} \tag{4-21}$$

For node u, the taxicab distance in terms of hops to each of the four reference corner nodes is a set of linear equations, which are presented as

$$\begin{cases} X_1 + Y_1 = \xi_A \\ X_2 + Y_1 = \xi_B \\ X_2 + Y_2 = \xi_C \\ X_1 + Y_2 = \xi_D \end{cases} \tag{4-22}$$

Substituting (4-21) into (4-22) gives

$$\Rightarrow \begin{cases} L - 1 - X_2 + Y_1 = \xi_A \\ X_2 + Y_1 = \xi_B \\ X_2 + Y_2 = \xi_C \\ L - 1 - X_2 + Y_2 = \xi_D \end{cases}$$

$$\Rightarrow \begin{cases} L - 1 + 2 \times Y_1 = \xi_A + \xi_B \\ L - 1 + 2 \times Y_2 = \xi_C + \xi_D \end{cases}$$

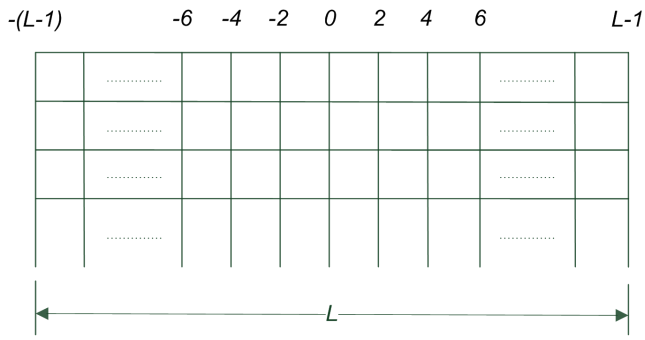
$$\Rightarrow 2 \times L - 2 + 2 \times (Y_1 + Y_2) = \xi_A + \xi_B + \xi_C + \xi_D$$

$$\Rightarrow 2 \times L - 2 + 2 \times (L - 1) = \xi_A + \xi_B + \xi_C + \xi_D$$

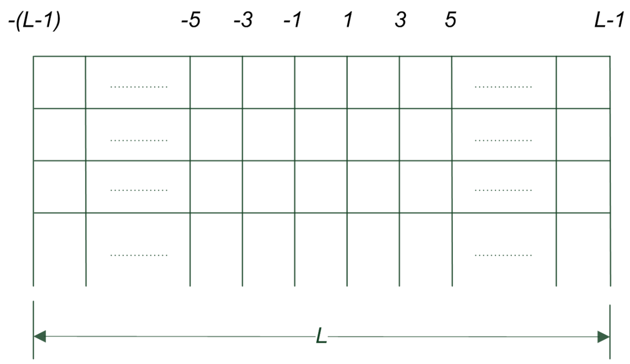
$$\Rightarrow L = \frac{\xi_A + \xi_B + \xi_C + \xi_D}{4} + 1 \tag{4-23}$$

■

Lemma 4-2 can be used by node u to estimate its relative node position X^u and Y^u at both X and Y directions. Different parity can exist for L, i.e. L can be either an even or odd value. Therefore, there will be different X^u and Y^u results at both X and Y directions, as shown in Figure 4.12 and Figure 4.13.

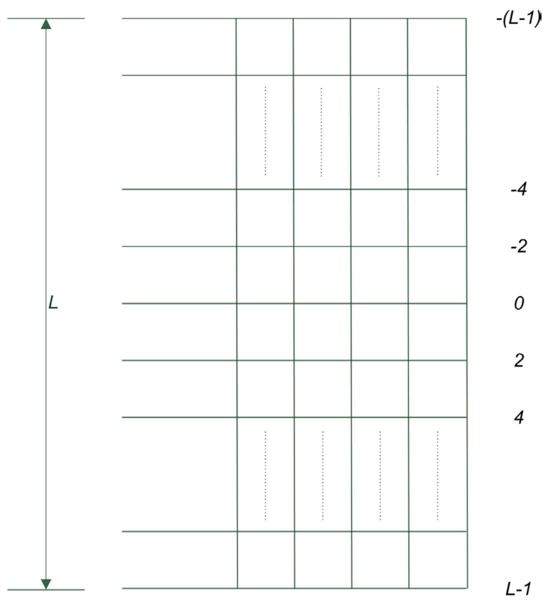


(a) X^u if L is odd

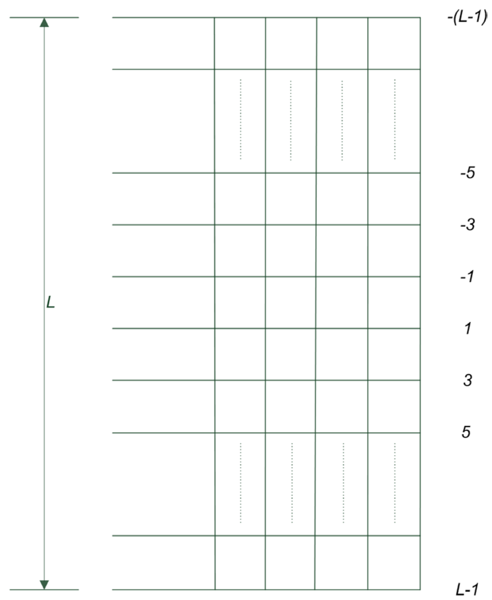


(b) X^u if L is even

Figure 4.12 Relative node position *X* at the *X* direction



(a) Y^u if L is odd



(b) Y^u if L is even

Figure 4.13 Relative node position *Y* at the *Y* direction

Algorithm 4-1. Approximate node position

Require: Node relative position X^u and Y^u of u
Ensure: How close a node u is to the centre at the X and Y directions

for i^u, $i \in \{X, Y\}$ **do**
  if $| i^u | \to 0$ **then**
    node is *close* to the centre at i^u direction
  else
    node is *away* from the centre at i^u direction
  end if
  if $i^u > 0$ **then**
    node is on the positive side of i^u direction
  else if $i^u < 0$ **then**
    node is on the negative side of i^u direction
  else
    node is on the centre of i^u direction
  end if
end for

According to Algorithm 4-1, a node u can estimate how close it is to the centre at X and Y directions according to the relative node position X^u and Y^u. The node u is closer to the centre at X and Y directions when the absolute value of X^u and Y^u moves towards to 0, and vice versa. The plus and minus sign of X^u and Y^u value can be used to indicate an approximate zone of the node u is located in the network, as shown in Figure 4.14. Even without properly partitioning a network, it is still possible for the node u to estimate where it is in the network according to its relative node positions X^u and Y^u.

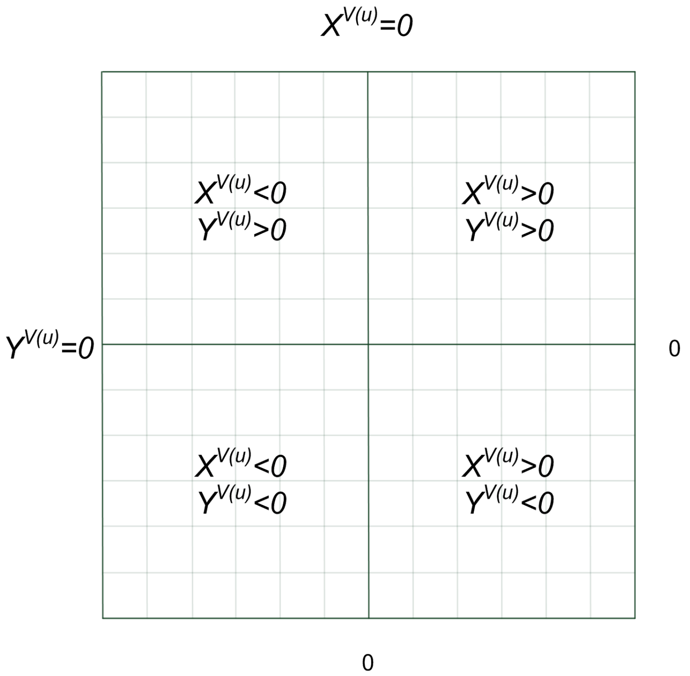


Figure 4.14 Zone approximation in the network

4.5.2 The Algorithm

The algorithm for network partitioning and VSU formation includes the following three steps:

- Partitioning the network based on relative node positions X^u and Y^u;
- Electing a centroid node on each partition;
- Formation of a single VSU around the centroid node of each partition.

4.5.2.1 Relative Location based Partitioning

This section presents the first variant of our algorithm that is based on only relative node location information in terms of hops to several reference nodes. After the completion of AFL, a node u can determine its relative position X^u and Y^u at both X and Y directions, and the boundary size of the network from Lemma 4-2 and Lemma 4-3. Algorithm 4-2 defines how a node, e.g. node u, can determine which partition it belongs to. The node u has to determine its node index, denoted as I^u, at both X and Y directions, which is explained by Algorithm 4-3; this is followed by steps to determine the node partition ID, denoted as P^u, in Algorithm 4-4, which also demonstrates how node u can determine its location, denoted as R^u, in the partition it belongs to at both X and Y directions.

Algorithm 4-4(a) and Algorithm 4-4(b) are proposed due to the different parity combinations of X^u, Y^u and c. For simplification, this chapter is using P_X^u and R_X^u at the X direction for the node u, as an example, with different parity combinations to illustrate the determination of the corresponding values.

Algorithm 4-2. Determine the partition ID P^u for node u

Require: Node relative position X^u and Y^u of u
Ensure: Determine the partition ID P^u of u

1: **for** i^u, $i \in \{X, Y\}$ **do**
2: Calculate Node Index I_i^u at the i^u direction (Algorithm 4-3)
3: Determine partition ID P_i^u at the i^u direction (Algorithm 4-4)
4: **end**

Algorithm 4-3. Calculate the node index I^u for node u

Require: Node relative position X^u and Y^u of u
Ensure: Calculate the node index I^u of u

for i^u, $i \in \{X, Y\}$ **do**
if i^u is even **then**
$I_i^u \leftarrow i^u/2$
else
if $i^u > 0$ **then**
$I_i^u \leftarrow \lceil i^u/2 \rceil$
else
$I_i^u \leftarrow \lfloor i^u/2 \rfloor$
end if
end if
end for

Algorithm 4-4. Find the node partition ID P^u and its location R^u in the partition for node u

Require: Node relative position X^u and Y^u of u
Ensure: Determine P^u and its location R^u in the partition for u

for i^u, $i \in \{X, Y\}$ **do**
if i^u is even **then**
Algorithm 4-4(a)
else
Algorithm 4-4(b)
end if
end for

Algorithm 4-4(a). Find the node partition ID P^u and its location R^u in the partition for node u (when X^u or Y^u is even)

Require: Node relative position X^u and Y^u of u, N and c
Ensure: Determine P^u and its location R^u in the partition for u

for i^u, $i \in \{X, Y\}$ **do**
$j \leftarrow |I_i^u| \pmod N$
$k \leftarrow \lceil N/2 \rceil$
if $I_i^u / N \geq 0$ **then**
$P_i^u \leftarrow \lceil I_i^u / N \rceil$
else
$P_i^u \leftarrow \lfloor I_i^u / N \rfloor$
end if
if c is even **then**
if $i^u \geq 0$ **then**
if $P_i^u = 0$ **then**
$P_i^u \leftarrow 1$
end if
if $c \times N \geq L$ **then**
$R_i^u \leftarrow |j - (k - 1)|$
else
$R_i^u \leftarrow |j - k|$
end if
else
if $j = 0$ **then**
$j \leftarrow N$
end if
$R_i^u \leftarrow |j - k|$
end if
else
if $j \geq k$ **then**
$R_i^u \leftarrow N - j$
else
$R_i^u \leftarrow j$
end if
if $j = 0 \vee (j \geq k \wedge c \times N \geq L) \vee (j > k \wedge c \times N < L)$ **then**
if $P_i^u \geq 0$ **then**
$P_i^u \leftarrow P_i^u + 1$
else
$P_i^u \leftarrow P_i^u - 1$
end
end if
if $P_i^u = -1$ **then**
$P_i^u \leftarrow 1$
end if
end if:
end for

Algorithm 4-4(b). Find the node partition ID P^u and its location R^u in the partition for node u (when X^u or Y^u is odd)

Require: Node relative position X^u and Y^u of *u, N* and *c*
Ensure: Determine P^u and its location R^u in the partition for *u*

```
for i^u, i ∈ { X,Y } do
   j ← | I_i^u | (mod N)
   k ← ⌈ N / 2 ⌉
   if I_i^u / N ≥ 0 then
      P_i^u ← ⌈ I_i^u / N ⌉
   else
      P_i^u ← ⌊ I_i^u / N ⌋
   end if
   if j = 0 then
      j ← N
   end if
   if c is even then
      R_i^u ← | j − k |
      if |P_i^u | ≥ c/2 then
         R_i^u ← R_i^u +1
         if P_i^u > 0 then
            P_i^u ← P_i^u -1
         else
            P_i^u ← P_i^u +1
         end if
      end if
   else
      if j > k then
         R_i^u ← N − j
         if P_i^u ≥ 0 then
            P_i^u ← P_i^u + 1
         else
            P_i^u ← P_i^u - 1
         end if
      else
         R_i^u ← j
         if P_i^u = -1 then
            R_i^u ← R_i^u - 1
            P_i^u ← 1
         end if
      end if
   end if
end for
```

The following are four examples with different X^u and *c* combinations to explain Algorithm 4-2 to Algorithm 4-4.

a) X^u is even and c is even

Following values are set: $c \leftarrow 2$, $L \leftarrow 9$, and $N \leftarrow 5,$

- Obtain the relative node position X^u by (4-18)
- Calculate the node index I_X^u from Algorithm 4-3
- Determine j, k, P_X^u, and R_X^u from Algorithm 4-4(a)

Table 4-2 X^u is even and c is even (c=2)

u	1	2	3	4	5	6	7	8	9
X^u	-8	-6	-4	-2	0	2	4	6	8
I_X^u	-4	-3	-2	-1	0	1	2	3	4
j	4	3	2	1	0	1	2	3	4
k	3	3	3	3	3	3	3	3	3
$P_X^{u)}$	-1	-1	-1	-1	0	1	1	1	1
$\boldsymbol{R_X^u}$	**1**	**0**	**1**	**2**	**2**	**1**	**0**	**1**	**2**
$\boldsymbol{P_X^u}$	**-1**	**-1**	**-1**	**-1**	**1**	**1**	**1**	**1**	**1**

b) *X^u is even and c is odd*

Following values are set: $c \leftarrow 3$, $L \leftarrow 9$, and $N \leftarrow 3$,

- Obtain the relative node position X^u by (4-18)
- Calculate the node index $I_X{}^u$ from Algorithm 4-3
- Determine j, k, $P_X{}^u$, and $R_X{}^u$ from Algorithm 4-4(a)

Table 4-3 X^u is even and c is odd (c=3)

u	1	2	3	4	5	6	7	8	9
X^u	-8	-6	-4	-2	0	2	4	6	8
$I_X{}^u$	-4	-3	-2	-1	0	1	2	3	4
j	1	0	2	1	0	1	2	0	1
k	2	2	2	2	2	2	2	2	2
$P_X{}^u$	-2	-1	-1	-1	0	1	1	1	2
$R_X{}^u$	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**
$P_X{}^u$	**-2**	**-2**	**-2**	**1**	**1**	**1**	**2**	**2**	**2**

c) *X^u is odd and c is even*

Following values are set: $c \leftarrow 4$, $L \leftarrow 12$, and $N \leftarrow 3$,

- Obtain the relative node position X^u by (4-18)
- Calculate the node index $I_X{}^u$ from Algorithm 4-3
- Determine j, k, $P_X{}^u$, and $R_X{}^u$ from Algorithm 4-4(b)

Table 4-4 X^u is odd and c is even (c=4)

u	1	2	3	4	5	6	7	8	9	10	11	12
X^u	-11	-9	-7	-5	-3	-1	1	3	5	7	9	11
$I_X{}^u$	-6	-5	-4	-3	-2	-1	1	2	3	4	5	6
j	0	2	1	0	2	1	1	2	0	1	2	0
k	2	2	2	2	2	2	2	2	2	2	2	2
$P_X{}^u$	**-2**	**-2**	**-2**	**-1**	**-1**	**-1**	**1**	**1**	**1**	**2**	**2**	**2**
j	3	2	1	3	2	1	1	2	3	1	2	3
$R_X{}^u$	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**

d) *X^u is odd and c is odd*

Following values are set: $c \leftarrow 3$, $L \leftarrow 12$, and $N \leftarrow 4$,

- Obtain the relative node position X^u by (4-18)

- Calculate the node index I_X^u from Algorithm 4-3
- Determine j, k, P_X^u, and R_X^u from Algorithm 4-4(b)

Table 4-5 X^u is odd and c is odd (c=3)

u	1	2	3	4	5	6	7	8	9	10	11	12
X^u	-11	-9	-7	-5	-3	-1	1	3	5	7	9	11
I_X^u	-6	-5	-4	-3	-2	-1	1	2	3	4	5	6
j	2	1	0	3	2	1	1	2	3	0	1	2
k	2	2	2	2	2	2	2	2	2	2	2	2
P_X^u	-2	-2	-1	-1	-1	-1	1	1	1	1	2	2
j	2	1	4	3	2	1	1	2	3	4	1	2
R_X^u	2	1	0	1	2	1	1	2	1	0	1	2
R_X^u	**2**	**1**	**0**	**1**	**1**	**0**	**1**	**2**	**1**	**0**	**1**	**2**
P_X^u	**-2**	**-2**	**-2**	**-2**	**1**	**1**	**1**	**1**	**2**	**2**	**2**	**2**

Under some parity combinations, the partitions may have different sizes. For instance, in *a) X^u is even and c is even*, the partition size for P_X=-1 is 4 while the partition size for P_X=1 is 5. This may mean that a few partitions are larger than the rest, but the same number of partitions in a network is still maintained.

The complete set of procedures to partition a network is presented as:

- Each node can obtain its T_Ξ values to the 4 reference corner nodes in the network according to the AFL algorithm;
- Each node can calculate its partition ID, R_X and R_Y, at both of the X and Y direction with Algorithm 4-4;
- Each node can calculate its combined relative location, denoted as R_T, in Algorithm 4-5;
- Each node only broadcasts its R_X, R_Y, P_X, and P_Y to its one hop neighbour nodes during the partitioning phase in Algorithm 4-6.

Algorithm 4-5. Find combined node relative location

Require: R_X^u and R_Y^u of u
Ensure: Calculate combined node relative location R_T^u of u

1: **return** $R_T^u = R_X^u + R_Y^u$

Algorithm 4-6. Network Partitioning and Centroid Node Election

```
for each u do
  Schedule a random one-time one-hop broadcasting message containing node ID, T_Ξ^E, P_i^u, R_i^u, i ∈ { X,Y }
  set the partition centroid node to itself
end for
if u receives a broadcast message from v then
  if P_i^u ≠ P_i^v then
    discard the message - not in same partition
    end the algorithm
  else
    calculate R_T^u for u with Algorithm 4-5
    calculate R_T^v for v from the received message with Algorithm 4-5
    if R_T^u ≤ R_T^v then
      discard the message – v is not close to centre of partition
      end the algorithm
    else
      u updates its partition centroid node as v
      u cancels its scheduled broadcast message
      broadcast the received message of v to u's 1-hop neighbours
    end if
  end if
end if
if scheduled broadcast message is not cancelled then
  broadcast the 1-hop message
end
```

The combined node relative position R_T from Algorithm 4-5 can show how close a node is to the centre of the partition it belongs to. A lower R_T value means the node is close to the centre of the partition. For INS, the data storage should be located near the centre of each partition in order to reduce the overall communications required in terms of hops for all other nodes to communicate with. Algorithm 4-6 can allow every node to know which partition it belongs to, as well as the centroid node of the partition, after exchanging the information with their one-hop neighbour nodes.

4.5.2.2 Energy and Relative Location based Partitioning

The first variant of our algorithm presented in Section 4.5.2.1 only considers node location information to find the centroid node of a partition. However, if the partitioning procedure is repeated periodically, the same nodes will always be selected as the centroid nodes, and therefore are always used to form the VSU in each partition. This can result in highly unbalanced energy dissipation between the VSU and non-VSU nodes, which reduces the overall network lifetime.

The aim of the location based algorithm is to minimise the overall communication required in terms of hops between the nodes and the VSU of a partition. If the location information is the only factor to decide the centroid node, the role as the centroid can only be switched once the existing centroid node is dead (depleted of energy). Therefore, the nodes always start to die from the partition centre, as shown in Figure 4.15.

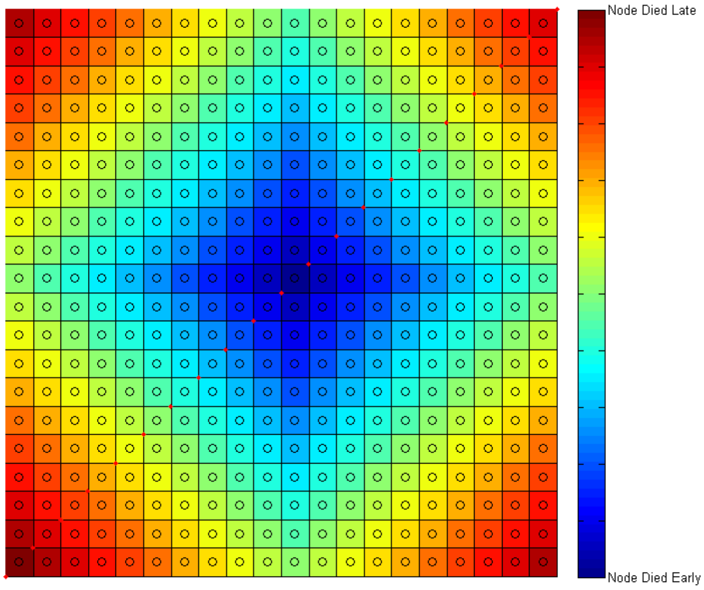


Figure 4.15 Node lifetime distribution of location-based algorithm

To better balance energy dissipation between the nodes in a partition, which can extend the lifetime of both nodes and the network, the node energy level will be taken into consideration during the centroid node selection phase.

The previous location-based algorithm is thus modified to consider a weighted combination of location and energy level of the nodes during the centroid node election phase. In this way, not only the node closest to the centre of a partition but also the node with more residual energy should be considered as a centroid node.

The weight function of a node is given as:

$$W_{total} = \alpha \times W_{location} + (1-\alpha) \times W_{energy} \tag{4-24}$$

where W_{total} is the total weighted value of a node u, $W_{location}$ is the weight value for R_T^{μ} of u, and W_{energy} is the weight value for the node's energy. α is the weight factor of $W_{location}$. $W_{location}$ and W_{energy} are defined as:

$$W_{energy} = \frac{node\ residual\ energy}{node\ initial\ energy} \tag{4-25}$$

$$W_{location} = \begin{cases} \dfrac{1}{R_X^{V(u)} + R_Y^{V(u)}} & if \quad R_X^{V(u)} + R_Y^{V(u)} < 0 \\ 1 & if \quad R_X^{V(u)} + R_Y^{V(u)} = 0 \end{cases} \tag{4-26}$$

Both W_{energy} and $W_{location}$ are in the range between 0 and 1. For W_{energy}, the more residual energy a node has, the closer will be its value to 1. Similarly for $W_{location}$, a node near the centre of a partition will have its value is close to 1, and vice versa.

For simplicity, in this thesis, we assume both location and energy level are equally important by assigning their weight factors with the same values:

$$\alpha = 0.5 \tag{4-27}$$

To implement this modified algorithm, only line 11 of Algorithm 4-6 needs to be amended to:

$$if\ W_{total}^{u} of\ node\ u \leq W_{total}^{v}\ of\ node\ v\ from\ the\ message\ then$$

In Algorithm 4-6, this weighted value is attached to each of the one-hop broadcast messages. The rest of the algorithm will remain the same. Therefore, during the centroid node selection in each partition, this weighted value can be used to decide which node should be selected as the centroid node. The VSU is then formed around the centroid node in each partition. The centroid node can be seen as a coordinator for the VSU, i.e. managing the operations of the VSU nodes, such as storage allocation, query redirections, and data offloading (for load balancing) among the VSU nodes so that data can be retrieved more efficiently. Due to the additional transmission and processing tasks, the centroid node can consume more energy than the other VSU nodes. Therefore, it is imperative to consider the energy level in addition to location information during the centroid node selection.

4.5.2.3 Virtual Storage Unit Formation

The formation of the VSU in a partition can commence immediately once the centroid node in the partition is determined. As the nodes can calculate their relative locations R_X and R_X in the partition at both X and Y directions, a node can become a VSU member node if it is close to the centroid node in the partition. Every time a node receives an update from the centroid node in its partition via one hop broadcasting in Algorithm 4-6, the node itself can decide if it will become a VSU member node according to its location, as described in Algorithm 4-7.

After the completion of the VSU formation process, every node now can know:

- which partition it belongs to through P_X and P_Y;
- its node status either as a normal node or storage node;
- which data storage node it can communicate with (if it is a normal node).

Algorithm 4-7. Virtual storage unit formation

Require: Received location information I_i^v, P_i^v, $i \in \{X, Y\}$ for a centroid node v by u
Ensure: Check the VSU status of u

if node u receives new location information updates for the centroid node v **then**
    if n is even **then**
        $n_a \leftarrow n/2 - 1$
        $n_b \leftarrow n/2$
    else
        $n_a \leftarrow (n-1)/2$
        $n_b \leftarrow (n-1)/2$
    end
    for I_i^u, $i \in \{X, Y\}$ **do**
        if $P_i^u \neq P_i^v$ **then**
            discard the message – not in the same partition
            end the algorithm
        end
        if $I_i^u \geq 0$ **then**
            if $I_i^u \geq I_i^v - n_a \wedge I_i^u \leq I_i^v + n_b$ **then**
                node u is in the VSU range at the i direction
            end if
        else
            if $I_i^u \geq P_i^v - n_b \wedge I_i^u \leq P_i^v + n_a$ **then**
                node u is in the VSU range at the i direction
            end if
        end if
    end for
    if node u is in the VSU range at the X direction $\wedge$ node u is in the VSU range at the Y direction **then**
        node u is a node included for the VSU in the partition
        node u announces itself as a data storage node around the centroid node v
    else
        node u is a normal ordinary node
    end if
end if

4.5.3 Performance Evaluation

4.5.3.1 Simulation Setup

A total of 10×10 nodes are setup as described in Section 4.3. All sensor nodes have the same transmission range and initial energy of 0.5 joules at the beginning of each simulation run. All algorithms evaluated are implemented in the OPNET Modeler[3] and the results shown are averaged over 20 simulation runs.

Similar to [154], the energy consumption of a node is calculated as:

[3] http://www.opnet.com/solutions/network_rd/modeler.html

$$energy = current \times voltage \times time \tag{4-28}$$

where *energy* is the energy cost of a node in joules during the different states (e.g. transmission, reception, or idle state), *current* is the current draw rate, *voltage* is the supply voltage, and *time* is the time duration for the data communication.

The communication duration is calculated by:

$$time = \frac{amount\ \ of\ \ data\ communicated\ \ in\ \ bits}{data\ \ rate\ \ in\ bits\ \ per\ \ \sec ond} \tag{4-29}$$

In addition, the following settings are applied:

- MAC protocol is an implementation of IEEE 802.15.4 unslotted CSMA/CA algorithm;
- Network routing protocol is an implementation of the AODV algorithm;
- Transmission and electrical parameters are set according to MICAz's specifications [155]:
  - Data rate is set to 250 kbps at 2.4 GHz;
  - Current draw rate is set to 17.4 mA at 0 dBm for data transmission, 19.7 mA for data reception, and 20 μA in idle state;
  - Voltage supply is set to 3 V.
- Partitioning process is repeated once every 3000 seconds (50 minutes);
- Packet size for control messages transmitted during partitioning and VSU formation process is set to 54 bits;
- Packet size for data transmitted from normal nodes to VSU is set to 200 bits;
- Every node transmits data to VSU at intervals of 100 seconds if not otherwise stated;
- The simulation ends when the last node in the network dies (depleted of energy).

4.5.3.2 Impact of Weight Factor Settings

This section evaluates the impact of different weight factor settings for α as in (4-27) on the performance of the proposed partitioning and VSU formation algorithm. The following three settings are evaluated and the result is shown in Figure 4.16.

- Location only setting: $\alpha=1$; (first variant of algorithm)
- Location and energy weighted setting: $\alpha=0.5$; (second variant of algorithm)
- Energy only setting: $\alpha=0$.

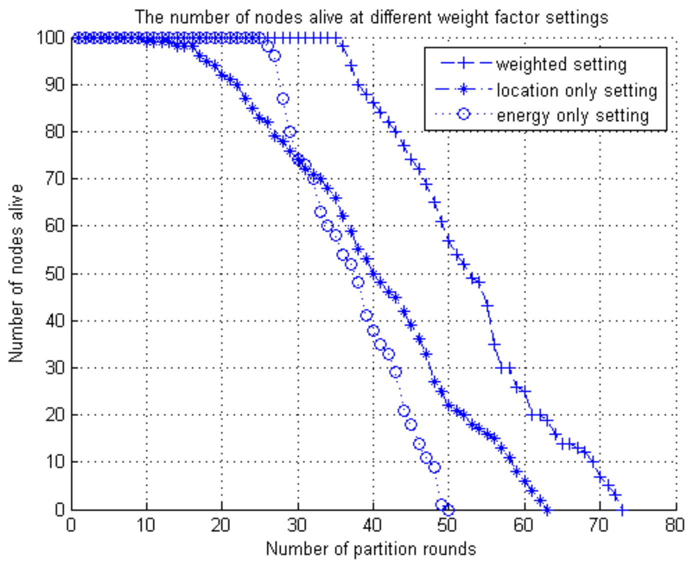


Figure 4.16 The network lifetime at the different weight settings

Figure 4.16 shows the number of nodes alive at each partitioning round. It illustrates the energy efficiency of the algorithm under the three weight factor settings by examining the overall network lifetime. The result shows that under the weighted setting, which considers both energy and location information, the network lifetime, in terms of the number of partition rounds executed before the death of the first node (99 nodes still alive) and that of the last node (0 node alive) occurred, is longer than both location-only and energy-only settings.

For location-only setting, which represents the first variant of our algorithm, the same centroid node is always selected according to its relative location R_T, and it will always be used if it is still alive at each partition round. Furthermore, the same nodes around the centroid node are also used for forming the VSU. Therefore, these same nodes are always consuming more energy than other nodes. This unbalances energy dissipation among the nodes in the network, and causes the centroid and VSU member nodes to be depleted of energy early.

On the other hand, for the energy-only setting, the role of a centroid node can be rotated among all the nodes in the network, which could balance energy dissipation but may result in longer communication hop-distances for some nodes. Overall, additional energy is required for relay nodes to forward data to the VSUs. This can be seen in Figure 4.16, where once nodes started dying, they vanished at a faster speed (steeper gradient of the fall in number of nodes alive) due to all nodes being low on energy.

For the location and energy weighted setting, it is shown to have the longest network lifetime. This can be seen from its latter occurrence of the first node death at round 36, as compared to round 11 for the location-only setting, and round 25 for the energy-only setting. Similarly, this setting has a latter occurrence of the death of all nodes at round 73, as compared to round 63 for the location-only setting, and round 50 for the energy-only setting.

It should be noted that the above three settings for weight factor α (0, 0.5, and 1) are only used to examine the impacts of $W_{location}$ and W_{energy} during the partitioning process. The optimal α setting for best network-lifetime performance (not necessary α=0.5) may need to be determined for specific application scenarios before they are deployed.

4.5.3.3 Comparisons with the alternative algorithms

In this section, the weighted partitioning algorithm proposed in this chapter is compared with two alternative algorithms: E-LEACH [87] and KOCA [88]. The reason for selecting these algorithms is that the number of partitions/clusters formed in each partition round is similar, i.e. all algorithms can generate 4 partitions on average. For the purpose of examining the efficiency of the partitioning/clustering process, no data is transmitted from the normal nodes to the cluster heads/storage nodes during the time between the completion of the partitioning/clustering and the end of the current round/commencement of the next round. Communications are only made for control messages sent during the partitioning/clustering phase.

The following simulation settings have been made for KOCA and E-LEACH:

- Packet size for messages transmitted during clustering phase of KOCA is set to 24 bits;

- Packet size for messages transmitted during clustering phase of E-LEACH is set to 10 bits;
- Probability value *p* of KOCA is set to 0.04, which defines on average 4 partitions will be formed in a network of 100 nodes during each clustering round;
- The *k* parameter of KOCA is set to 8 hops, which is the minimum value to ensure that every node can join a cluster for a network with 100 nodes;
- Probability of a node as a cluster head in E-LEACH is set to 0.04, which defines the top 4 nodes with highest residual energy will be selected as cluster heads. Therefore, 4 clusters will be created at each round;
- As in [87] for E-LEACH, the transmit power of a cluster head, and normal node, is set to 17.7 dBm (59 mW), and 0 dBm (1 mW), respectively.

Table 4-6 presents a comparison of the total energy cost during the partitioning/clustering process per node per round among the three algorithms. It shows that the proposed algorithm under weighted setting can achieve higher energy efficiency than KOCA and E-LEACH. As only control messages are transmitted, the lower energy consumption reflects a lower communication overhead, which will leave more energy for data transmission. The higher energy cost for E-LEACH is due to the higher transmission power required by its cluster heads, which increases the average energy cost during the clustering phase for broadcasting cluster head announcements. The proposed algorithm has no such requirement, i.e. all nodes utilises identical communication power setting. KOCA consumes more energy cost due to all clustering messages requiring to be broadcast over 8 hops which increase the average energy consumed by nodes for message forwarding. In contrast, the proposed algorithm limits the broadcasting range to 1 hop and only requires the message to be forwarded within a partition instead of the network. Therefore, the communication energy cost is reduced.

Table 4-6 Energy cost per node per partitioning/clustering round

Algorithm	Energy Cost (joules)
E-LEACH	0.01092
KOCA	0.01087
Proposed	0.003524

Figure 4.17 shows the cumulative control bits transmitted per node during each partitioning round. The proposed algorithm incurs the least amount of control bits transmission during the partitioning phase for the reason given above. Although E-LEACH incurs significantly less transmissions as compared to KOCA, it still transmits more control bits than the proposed algorithm. This is due to the fact that communications are required for discovering the top 4 nodes with the highest residual energy to become the cluster heads for the next clustering round.

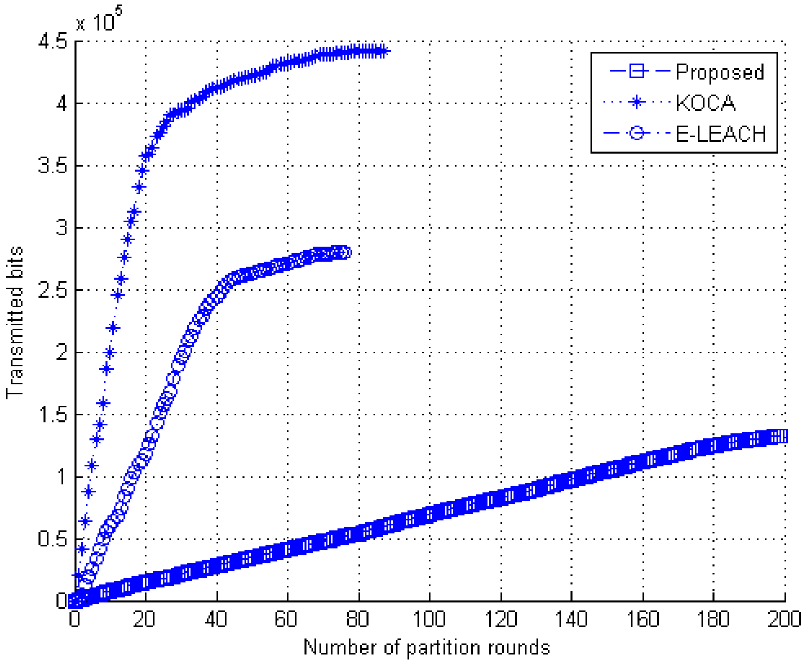


Figure 4.17 The cumulative control bits transmitted per node

Figure 4.18 presents the average network lifetime of the three algorithms. It shows that the proposed algorithm can extend the network lifetime of the other algorithms by approximately 2.5 times, which is consistent with the energy and bit transmission results shown in Table 4-6, and Figure 4.17, respectively. E-LEACH exhibits a slower rate of nodes dying than KOCA initially. This is because E-LEACH allows the cluster heads to be rotated among nodes with the highest residual energy at each round. However, the higher transmission power required for cluster head makes many nodes that have acted as cluster head to become depleted of energy much earlier as compared to KOCA.

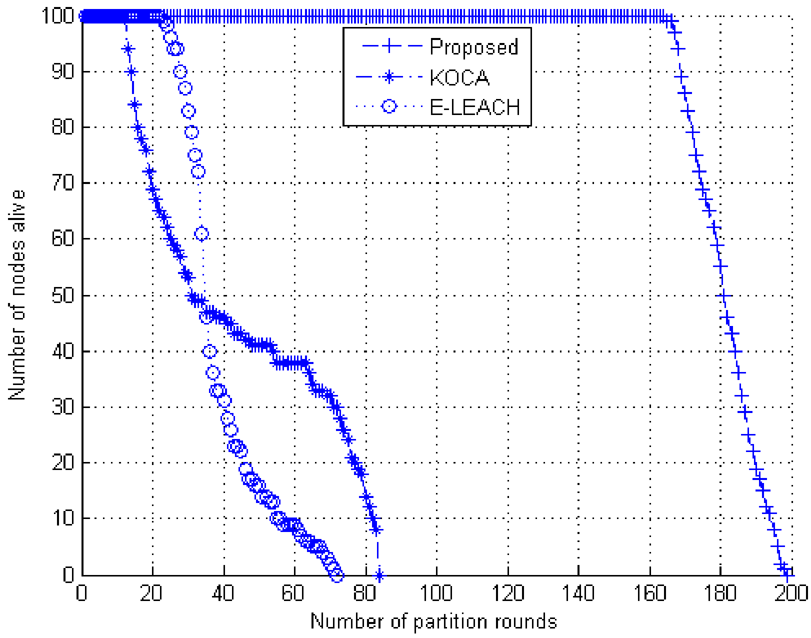


Figure 4.18 Network lifetime of different algorithms

Figure 4.19 shows the standard deviation of the partition/cluster size, which reflects the degree of balance of the partitions/clusters generated by each algorithm. A lower standard deviation means that the algorithm can form more consistently sized partitions/clusters from the number of nodes alive, which in turn can lead to more balanced traffic loads and energy consumption among nodes in the network. The result shows the proposed algorithm can achieve more balanced partitions than KOCA and E-LEACH.

E-LEACH has a tendency to create unbalanced clusters because the criterion of the cluster head selection procedure is only based on node energy. If the top 4 nodes with the highest residual energy are not evenly spaced, the resulting cluster sizes can vary significantly. KOCA has a similar issue as clusters are formed according to some probability *p*. If nodes probabilistically chosen as cluster heads are unevenly spaced, the resulting clusters formed will also be unbalanced.

Figure 4.20 shows the amount of time required for partitioning/clustering by each algorithm. The time is calculated as the duration from the first node initialising the partitioning/clustering process to the last node finishing the process at each round. This measures the time efficiency of the algorithms. The result shows that the proposed algorithm uses the least amount of time to complete the partitioning process. KOCA requires the longest time due to its requiring the cluster heads to broadcast their announcements over 8 hops, which can congest the network and cause packet delay. In contrast, our algorithm only broadcasts and forwards the 1-hop message when necessary, and cancels any scheduled broadcasts if they are not by the centroid node in a partition. Overall, this has improved the time efficiency of the partitioning process.

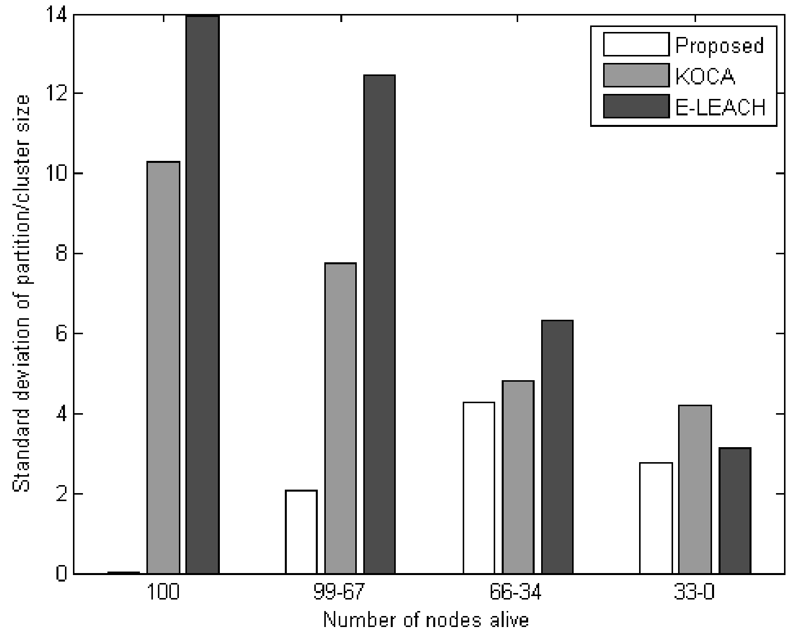


Figure 4.19 Standard deviation of partition/cluster size under different number of nodes alive

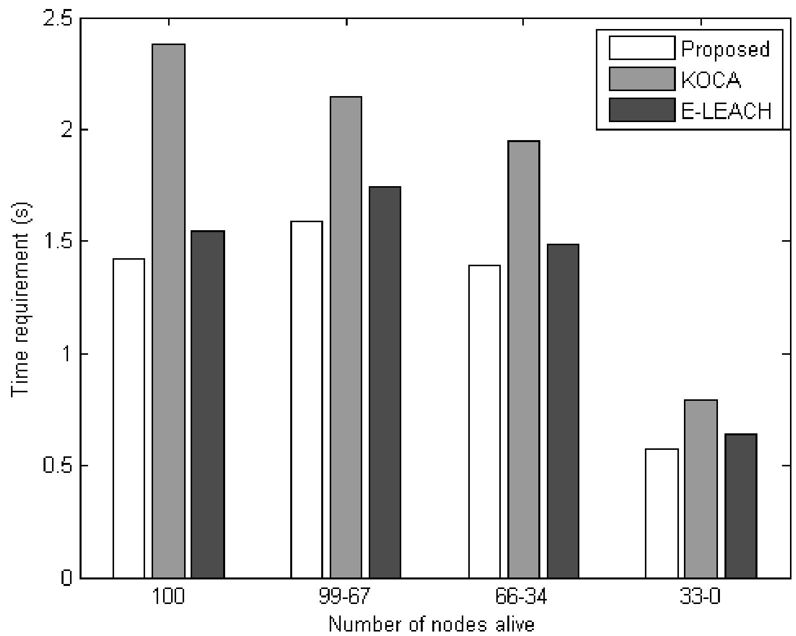


Figure 4.20 Time requirement for partitioning/clustering by each algorithm

4.6 Conclusion

This chapter presented two research contributions of this thesis. The first is an analytical model that provides a means to determine the optimal number of partitions and virtual storage unit (VSU) size that jointly minimise the total communication overhead in the network in terms of hop counts. Both the derivation and evaluation of the analytical model are presented in detail. While the proposed model is derived under the assumption of nodes deployed in a grid topology, results have shown that the model can still yield optimal parameters even when nodes are positioned with a substantial level of randomness.

The second contribution presented is a network partitioning and virtual storage unit formation algorithm, which comes in two variants: one that considers only relative node location information, and one that considers both node energy level and relative location information. The results show that by incorporating both energy and location information through a weight factor setting, the network lifetime under the proposed algorithm can be extended. The proposed algorithm is also evaluated against two other well-known algorithms: k-hop overlapping clustering algorithm (KOCA) and energy-LEACH (E-LEACH), and the results show that it can outperform them in terms of energy, communication, and time efficiency, network lifetime, and balanceness of the partition.

The virtual storage units presented in this chapter are intended to function in place of dedicated brokers in conventional publish/subscribe (pub/sub) systems. The next chapter will present our designs for a Virtual Broker-based pub/sub mechanism for wireless sensor networks (WSNs).

Chapter 5 In-Network Storage for Virtual Broker-based Publish/Subscribe in Wireless Sensor Networks

5.1 Introduction

Data centricity is a key characteristic that distinguishes wireless sensor networks (WSNs) from other types of wireless networks. In traditional WSNs, sensors detect environmental attributes or events, which are transmitted to and stored on dedicated base stations or sinks at the network edge for further processing. There are several ways for sensor devices to transmit their data to the sink, either through direct transmission or indirect transmission via cluster head, intermediate nodes, or through a backbone [87, 156, 157].

The common use of centralised sink devices for many WSN applications can be attributed to the physical hardware limitation of sensor devices. Early generation sensor devices were low in processing power as well as storage capacity [158]. With limited hardware capabilities, a sensor device can only act as an attribute monitor or event detector in its surrounding environment, and has to transmit any data generated to the more powerful base station or sink for storage and processing.

With advances in data storage technologies, it becomes possible for embedded sensor devices to cooperatively store and maintain data within the WSN for later retrieval by in-network users such as humans who interact with their environments which have been smartified by embedded intelligence.

As a follow-up work from the previous chapter, which proposed a partitioning and virtual storage unit (VSU) formation mechanism in WSNs, this chapter presents an in-network storage scheme for a virtual broker (VB) based pub/sub system in WSNs. The virtual broker nodes (VBNs) are VSU member nodes that cooperatively store data published by sensors and handle subscription requests for data from in-network users. The rest of this chapter is organised as follows. Section 5.2 overviews the key concepts of pub/sub and in-network storage. Section 5.3 presents the design of the proposed storage scheme in detail. Section 5.4 evaluates the proposed scheme and discusses the results. Finally, Section 5.5 concludes the chapter with some directions for future work.

5.2 Related Work

5.2.1 Publish/Subscribe Mechanism

Network communication is traditionally based on an address centric mechanism. In order to communicate with and obtain data from some data source in a network, the network or MAC address of this source node must be known. Using this address, a data request is then delivered to the node, which in turn replies with the requested data over the network. Under this procedure, the node address is more critical than the actual data from the node as it is impossible to retrieve any data if no address is given, although data is what one usually most cares about. On the other hand, communication based on a data-centric mechanism such as the pub/sub mechanism adopts an opposite approach [3]. The pub/sub is a promising communication paradigm for such applications given the data centric nature of its operations, and data centricity is a key characteristic of the WSNs [3]. Here, one does not need to know the address of the data source in order to obtain the data. Under this approach, data sources (e.g. sensors) can *publish* their data to brokers for storage, and data users can *subscribe* to these brokers to receive specific pieces of data. Thus, publishers and subscribers are not tightly coupled in time and space, i.e. each does not require the other to be "on" at the same time nor require to have a connected path between them, in order to send or receive data. In conventional pub/sub systems, brokers are often dedicated devices with high processing power and storage capacity. Ref. [159] presented the concept of a virtual broker (VB), defined as an entity that provides the functions of a broker, not through using a single dedicated resourceful device, but through resources harnessed from a cooperative group of ordinary sensors, which we refer as virtual broker nodes (VBNs). A VB is also referred to as a VSU in Chapter 4 (as a generic storage unit without any specific use, such as a pub/sub broker in this chapter).

As mentioned in Chapter 4, the formation of a VB is initiated by the centroid node of the partition, which is an ordinary sensor found to be the most centrally located node in terms of its hop count to other sensors in the partition by our network partitioning algorithm. This centroid node then selects an optimal number of other sensors to join itself as VBNs based on minimising the communication load within and outside of the VB. In this chapter, the VBNs cooperatively provide in-network storage of data from publishing sensors as well as handle the subscriptions of data from in-network users.

5.2.2 In-Network Storage

With recent advances in embedded hardware, the processing power of a sensor device has increased to tens or even hundreds of mega hertz, and its storage capacity increased to as high as 32 MB, such as the Sun Spot [160] and Intel Mote 2 [161]. This made it possible to store and process data within WSN, which will be useful for applications where users of the data are likely to come from within the network. For instance, ambient intelligence (AmI) applications where users interact daily with their environments which have been augmented with embedded sensing and computing devices [1]. The hardware improvements can enable sensor devices to perform more than just sensing and transmission. With concepts of cooperative systems [162], more powerful functions can be added to in-network storage design, such as Virtual MIMO [8], and achieved through resource sharing and cooperation between sensor devices.

In-network storage, data are generated, processed, and stored on one or multiple storage nodes within the network. In literature, many distributed in-network storage designs were based on the DHT (distributed hash tables) approach [142, 163, 164]. Similar to hash tables, DHT provides a lookup service for an item associated with a given key, but the mappings from keys to items is distributed over multiple nodes. It uses a simple operator *lookup(key)* to return the identity of the node responsible for storing an item with that key. This operator can be used for both storing and retrieving a data item to/from the identified node. By using DHT-like algorithms, data storage and retrieval can be quite efficient. However, DHT only provides exact-match lookups and one hash function will be needed for each hash index type. For more flexible lookups, more than one hash index is required, and this may increase storage and communication complexity in the system.

5.3 Design Specifications

This chapter proposes an in-network storage scheme for the virtual broker based pub/sub system as presented in [159]. The design of our in-network storage scheme covers data storage and retrieval operations on VBNs, as well as the communication and storage load balancing between VBNs. As a benchmark for comparison, this chapter also implemented a storage scheme based on DHT. The DHT approach was adopted as benchmark due to its popular use by in-network storage schemes as mentioned in Section 5.2.2.

5.3.1 System Definition

In this chapter, the following assumptions are made:

- The WSN is composed of identical sensor nodes with same processing, storage, and energy capacity;
- Each sensor node is a member of one, and only one of many partitions. Each partition has one and only one VB formed by two or more co-located ordinary sensor nodes;
- Every sensor node takes only one of two available roles: as ordinary sensor node or VBN. The latter is an ordinary sensor node that also performs the function of a broker in the same partition. For each partition, the number of VBNs is usually far less than the number of ordinary nodes;
- All sensor nodes, including VBNs, can publish (store) and subscribe (retrieve) data to/from VBNs in the same partition;
- All ordinary sensor nodes treat all VBNs in the same partition as one entity, and only require to communicate with one VBN either directly or indirectly via intermediate nodes, in order to fully access all data in a VB;
- All sensor nodes publish data periodically to a VB.

5.3.2 Data structure for Pub/Sub mechanism

As a data centric approach, the data has to be structurally stored in order to be systematically searched by the broker that receives a subscription request for the data. To be generic in our design so that it can be used in a wide range of applications, every published data can have multiple index values. An index value for a piece of data is defined as *a single and short keyword or description that can lead to finding the actual data stored on different nodes*. Different types of sensors will generate different types of data, such as temperature, light, and humidity data. Using a temperature sensor as an example, its temperature data can be associated with three index values of types: Category, NodeID, and Timestamp, as shown in Figure 5.1(a). With multiple index values, there can be more than one way of retrieving this data using any of the index values and their combinations. As shown in Figure 5.1(b) there are 7 possibilities to obtain the same temperature information.

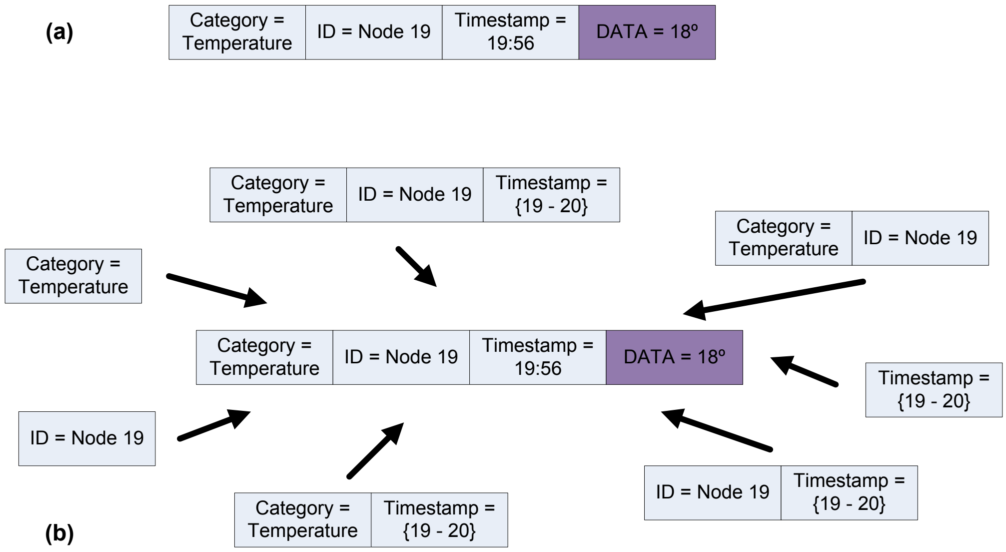


Figure 5.1 Data indexing under data centric approach. (a) multiple indexes to a single data; (b) all possible index combinations for a single piece of data.

Theorem 5-1

For *n* number of index values for a single data, where *n* is a natural number, there are $2^n - 1$ possibilities of looking up for this data.

Proof

Let *S* denotes the set of *n* index values, and let *k* denotes the number of distinct index values in a subset of *S*. The number of *k*-combinations (whose order is unimportant) of *S* is therefore a binomial coefficient. The sum of all possible *k*-combinations is given as:

$$\sum \binom{n}{k} = 2^n - 1, \quad 0 < k \leq n$$

■

With more indexes added for a single piece of data, there will be more possible index combinations for that data, and hence increase complexity of the data look-up. One possible solution is to store multiple copies of the data and associate each with one possible index combination, so that any index combination can be used to look up for the data. However, such design is storage inefficient as more storage space is required for any given piece of data.

Thus, for the storage schemes implemented in this chapter, only one copy of each published data will be stored in the VB. If a subscription request arrives at a VBN that does not hold the data, the request may be broadcast to other nearby VBNs. Given the typical small number of VBNs involved, the resulting broadcast overhead is not anticipated to be significant. The next two sub-sections present the implementation of a typical in-network storage scheme based on DHT, and our proposed scheme with storage load balancing.

5.3.3 DHT based Storage Design for VB

Here, the DHT approach is used for storing published data on VBNs, and intra-VB broadcasting is used for data subscribing, i.e. dissemination of subscription request, between VBNs. Figure 5.2(a) shows an example of a partition with ordinary sensor nodes, VBNs, and in-network user nodes. The VBNs are normally located in the centre of the partition and in proximity of each other. Before any data can be published, each VBN has to be assigned with a non-overlapping range of DHT keyspace, so that a given published data can be stored on a VBN if the hashed key value for that data is within the assigned range of the VBN, as shown in Figure 5.2(b). Otherwise, it forwards the data to the VBN with a keyspace that includes the hashed key value.

Before an ordinary sensor node publishes data to a VBN, the data is hashed into a key value using the index values of the data as inputs to a hash function, and only one hashed key value is generated for that data. As shown in Figure 5.2(c), an ordinary sensor node publishes data with a hashed key value to only the VBN closest to itself. On receiving, the VBN first checks if the hashed key value of the data is within the range of its DHT keyspace, and if so, it stores the data and if not, it forwards the data to the VBN responsible for the hashed key.

To subscribe to a piece of data, an in-network user has to firstly specify the index values that it will use in its subscription request. For example, a subscription can be as broad as subscribing to data that matches Category = {Temperature}, or as specific as Category = {Temperature}, NodeID = {19}, Timestamp = {19 – 20}. If multiple hash functions are used to represent every possible index combination, multiple copies of the same data may be stored in the VB. Under this approach, communication cost for subscribing will reduce, but the cost for publishing will increase since the data (usually of larger size than a subscription) can be retransmitted multiples times within the VB for storage on different nodes. This can increase the overall communication overhead. Thus, in this design, only one hash function is used to allow storage of only one copy of each published data in the VB as mentioned. The hash function used is a lightweight 32-bit integer hash function developed by Jenkins [165]. A VBN that does not hold data requested by a subscription (with full or partial index values) performs a one-time intra-VB broadcasting as shown in Figure 5.2(d) and Figure 5.2(e). Other VBNs on receiving this request may forward any matched data to the original VBN, which in turn transmits a reply with the data to the in-network user as shown in Figure 5.2(f) and Figure 5.2(g).

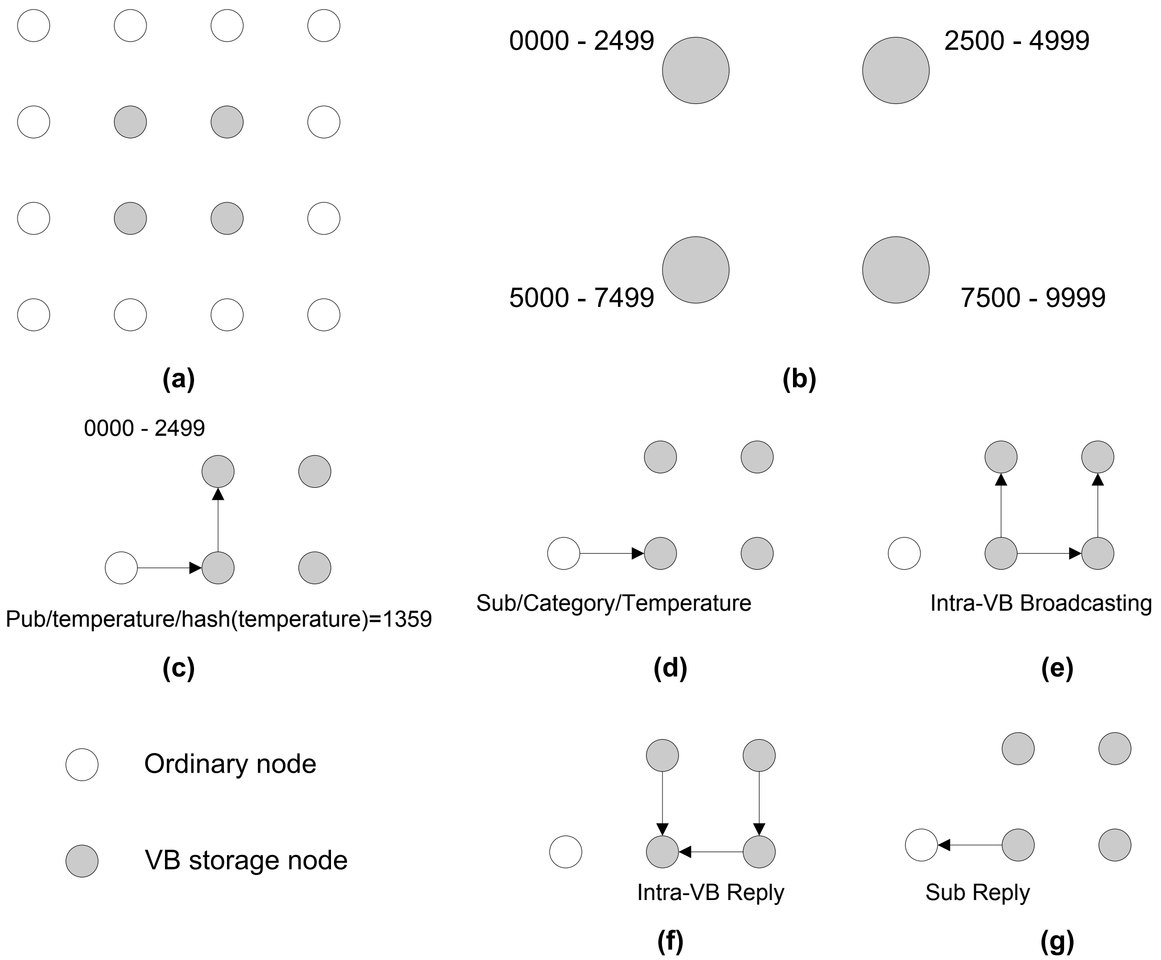


Figure 5.2 DHT-based VB design. (a) a single WSN partition; (b) DHT keyspace of each VBN; (c) data publishing; (d) data subscribing; (e) intra-VB broadcasting; (f) intra-VB reply; (g) subscription reply.

5.3.4 Balanced storage design for VB

Under ideal conditions, the DHT based storage scheme should hash all published data items into key values that are uniformly distributed over the keyspace, thereby distributing the storage load evenly among the VBNs. However, in reality, the hashed key values produced by the hash function may not be always uniformly distributed, resulting in overloading some VBNs and under utilising the storage space of some other VBNs. Furthermore, when a published data arrives at a VBN that is not responsible for its storage, the data has to be forwarded by the VBN to the one responsible. This can increase the cost of publishing.

In order to balance the storage over multiple VBNs as well as reducing the cost of intra-VB data forwarding, this chapter proposes a new storage approach for VBNs. Under the proposed approach, each VBN follows one simple rule: do not forward the received data unless there exists a better VBN (e.g. with more storage space) than itself for holding this data.

As shown in Figure 5.3(a), each data from an ordinary sensor node is received by and stored only on one VBN to which it is closest. There is no forwarding of data by the VBN to other VBNs unless its storage is full. With multiple index values defined by original sensor nodes for their published data, a subscription (with multiple index values defined by an in-network user) arriving at a VBN that does not hold that data, can be processed based on the same intra-VB broadcasting approach shown in Figure 5.2(d)–2(g).

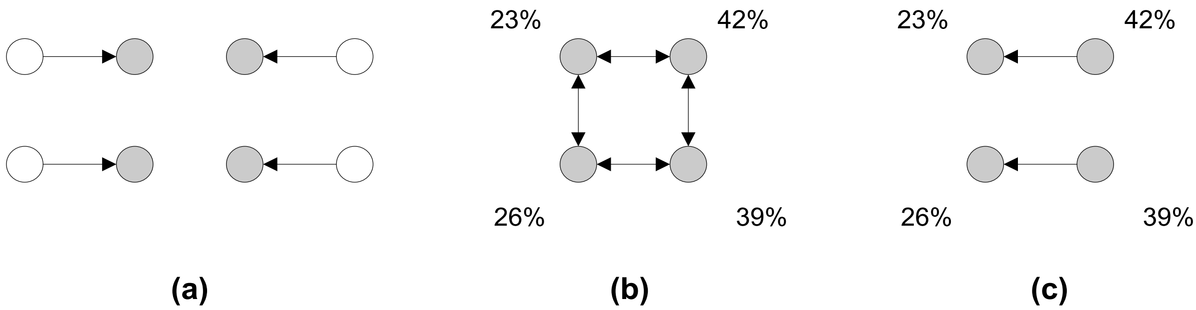


Figure 5.3 Balanced storage-based VB design. (a) publishing only to the first VB storage node; (b) all VB storage exchange storage level information with one-hop neighbour VB storage nodes; (c) VB storage nodes with lower storage level receive published data from other VB storage nodes with higher storage levels.

Moreover, each VBN maintains a record of its own storage level, defined as the ratio of the node's currently used to total storage capacity, which it updates from time to time. This information is broadcast periodically by every VBN to only its one-hop neighbours, as shown in Figure 5.3(b). On receiving, each VBN compares its own storage level with that of its neighbours. If more than one neighbouring VBNs are found to have a storage level lower than itself by at least the size of a published data record, one or more currently stored records can be offloaded to the one with the lowest storage level, as shown in Figure 5.3(c). The following presents the pseudo code of this storage load balancing process:

Algorithm 5-1. Storage load balancing

Require: storage level *Sa* and *Sb* for two VB storage node A and B, total storage size *Tn* of node A, and average published data size *Ds*
Ensure: number of published data to be exchanged from VB storage node A to storage node B

```
for each A in total VB storage nodes do
  broadcast Sa to one-hop surrounding VB storage nodes
  receiving storage levels from other VB storage nodes
  select lowest Sb from all received storage levels
  if (Sa-Sb)*Tn/(2*Ds) > 1 then
    forward ⌊(Sa-Sb)*Tn/(2*Ds)⌋ numbers of published data from storage node A to node B
  end if
end for
```

In Algorithm 5-1, $(Sa\text{-}Sb)/2$ indicates half of the difference of the storage level, in terms of percentage, between two VB storage node A and B. $(Sa\text{-}Sb)*Tn/2$ represents total size, in terms of bits, of the half storage difference. $(Sa\text{-}Sb)*Tn/(2*Ds)$ calculates the average number of published data packets based on the storage difference. In this balanced storage design, only complete (not fractional) packets can be offloaded between different VB storage nodes.

5.4 Evaluation

Both the DHT-based scheme and the proposed balanced storage (BS) scheme have been implemented and evaluated in OPNET. The schemes are evaluated for a single partition of 16 sensor nodes (publishers) and 4 in-network user nodes (subscribers) with 2 × 2 sensor nodes in the centre selected as VBNs as shown in Figure 5.2(a). The sensor nodes are static, while the in-network user nodes move according to the random waypoint model at a speed of 1 m/s with no pause time. The partition has a dimension of 50m × 50m and all nodes have a transmission range of about 15 m.

Each sensor node starts to publish at some random time after which it publishes at regular time intervals to its nearest VBN. Similarly, each user node starts to subscribe at some random time after which it subscribes at regular time intervals to its nearest VBN. The packet size of each published data and subscription request is 56 bytes, and 40 bytes, respectively, based on the designed data structure. In the evaluations, the *pub* time interval (τ_p) and *sub* time interval (τ_s) are varied between 1–5 seconds. IEEE 802.15.4 and AODV are the MAC and routing protocols used, respectively. The simulation time is 180 seconds and each result is the average over five simulation runs.

Each of the schemes is evaluated based on three metrics: i) storage load balancing, which measures the difference between the maximum and minimum storage levels between VBNs, also known as storage range; ii) intra-VB communication, which measures the communication cost incurred by VBNs for storage on receiving published data from sensor nodes, as well as the communication cost incurred by VBNs for data retrieval on receiving subscription requests from in-network users. For BS scheme, there is also a communication cost for balancing of storage load; iii) energy cost, which measures the energy consumed by VBNs due to communication during data storage and retrieval. The energy is calculated by the product of the length of time that a node spends on transmission and reception, and the voltage and current drawn during these times based on MICAz specifications [155].

5.4.1 Storage Load Balancing

Figure 5.4 shows how the storage range of both schemes evolved over time with publishing time intervals of 1, 3, and 5 seconds. For BS scheme, with storage level information exchanged between every 3 seconds, it is observed that the scheme has a much better storage range performance as compared to its DHT counterpart, which suffers increasing storage load imbalance with increased publishing rates (shorter publishing time intervals). This is due to some non-uniform occurrence of the keys (data), which when hashed to a keyspace that is uniformly distributed between VBNs, can result in those assigned with more "popular" key values to store a higher proportion of the data records.

To study the scalability performance of both VB storage mechanisms, two larger networks consisting of 64 nodes (48 publishers with 16 VB storage nodes) and 144 nodes (108 publishers with 36 VB storage nodes) are further considered. These network sizes are chosen such that all three networks maintains a publisher-to-VB storage node ratio of 3:1. All other network settings remains unchanged. Table 5-1 shows the storage range results at end of the simulations (i.e. after 180 seconds).

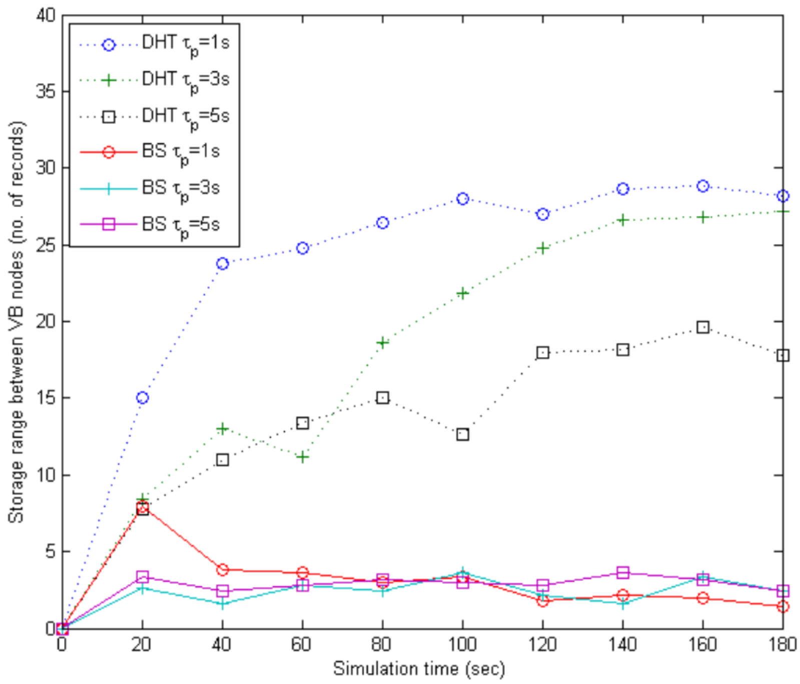


Figure 5.4 Time evolution of the storage range between VB nodes

Table 5-1 Storage range between VB nodes at end of the simulation (after 180 seconds)

Storage Scheme	DHT			BS		
τ_p (sec)	1	3	5	1	3	5
16 nodes	28.4	27.8	17.6	1.6	2.6	2.4
64 nodes	36.8	33.2	27.4	6.6	7.4	6.4
144 nodes	61.8	53.2	45.6	16	14.2	11.4

It is observed that increasing network size increases the variance of the storage range as well. By fitting these results to a power-law function (similar to the method used in obtaining Eq. (4-3)), we derive approximation functions for each scheme to estimate the storage range R for any arbitrary network size s under the operating settings defined in Chapter 5.4:

Table 5-2 Derived approximation functions for storage range estimation

Storage Scheme	τ_p (sec)	Approximation Function
DHT	1	$R = 0.01231 \times s^{1.597} + 27.37$
	3	$R = 0.002903 \times s^{1.83} + 27.34$
	5	$R = 0.1239 \times s^{1.109} + 14.92$
BS	1	$R = 0.05978 \times s^{1.121} + 0.2607$
	3	$R = 0.2138 \times s^{0.8383} + 0.415$
	5	$R = 0.3244 \times s^{0.7155} + 0.04182$

5.4.2 Communication Cost

Figure 5.5(a) shows the communication cost incurred within the VB in terms of total number of packets and bits transmitted due to the storage of published data as publishing time interval increases from 1 to 5 seconds. It also shows the packet type distribution, i.e. data or control packet. For BS scheme, the control packets related to storage are the periodic one-hop broadcast by each VBN about its storage level. These are small packets each with a size of 6 bytes. For DHT-based scheme, no control packet is used for storage. Only data packets are retransmitted to the VBNs responsible for storing them.

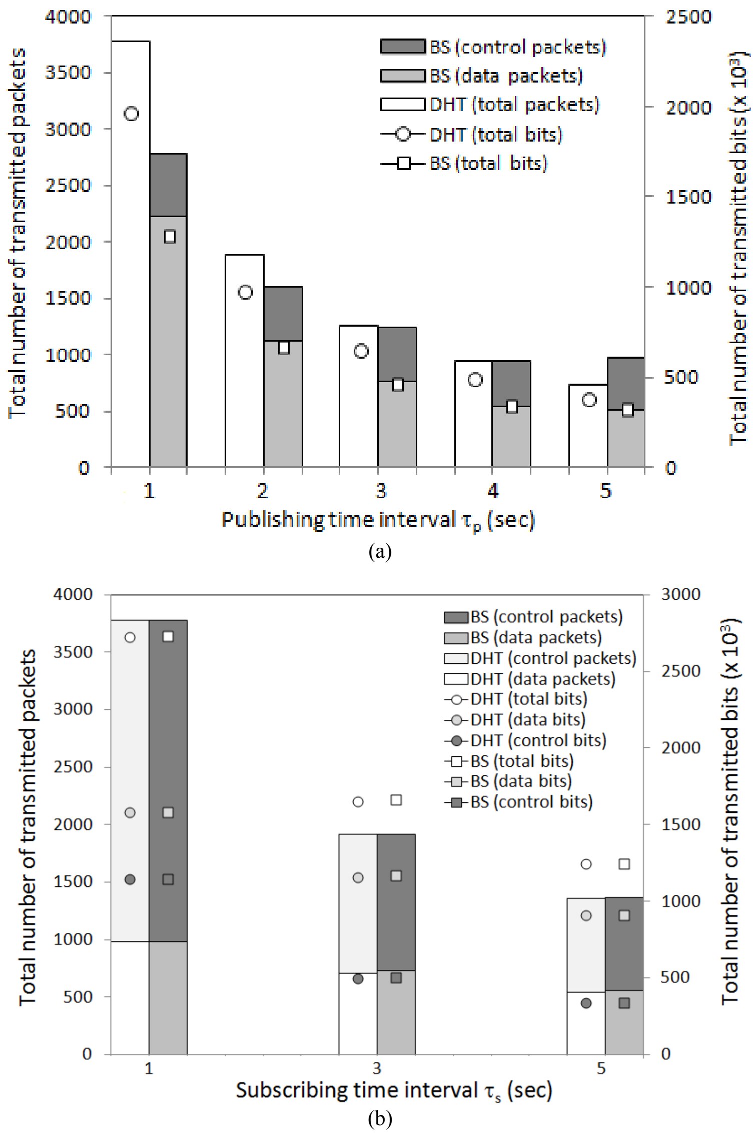


Figure 5.5 Communication cost incurred by VBNs for: (a) data storage on receiving published data from sensor nodes; (b) data retrieval on receiving subscription requests from in-network users.

From the result, it is noted firstly that the communication cost of both schemes expectedly decreases as the publishing time interval increases. It is also observed that the BS scheme consistently transmitted less total number of packets than its DHT counterpart until τ_p = 4s. For τ_p > 4s, it still generated less total number of bits than the DHT scheme even with a higher total number of packets. This is shown in Table 5-3 when the publishing time interval is extended to 90 and 180 seconds, which represents the case of only two data packets, and one data packet, published respectively, by each sensor node, given a simulation period of 180 seconds. This is because its proportion of larger-sized data packets (56 bytes as compared to 6 bytes for control packets) is less than that of the DHT scheme, resulting in an overall saving in number of bits transmitted.

Table 5-3 Total number of transmitted packets and bits by VBNs for data storage with τ_p = 90 and 180 seconds

τ_p (sec)	90	180	90	180
	total number of transmitted packets		total number of transmitted bits ($\times 10^3$)	
DHT	74	18	19.585	9.244
BS	76	38	16.147	8.405

Figure 5.5(b) shows the communication cost incurred within the VB in terms of total number of packets and bits transmitted due to the retrieval of data for a subscribing time interval of 1, 3, and 5 seconds. It also shows the distribution of data and control packets and bits. For both schemes, the control packets related to retrieval are the broadcast of subscription requests by VBNs. Since both BS and DHT-based schemes have used the same approach to handle subscription requests, their costs and type distribution are similar. In general, the cost of communication decreases as subscribing time interval increases as anticipated. It is also noted that of the total transmitted packets, the proportion of control packets is greater than the data packets. However, due to the smaller-sized control packets (40 bytes as compared to 56 bytes for data packets), the proportion of control bits in total bits transmitted is smaller than the data bits.

5.4.3 Energy Cost

The corresponding energy cost due to communication for data storage and retrieval is shown in Table 5-4 and Table 5-5, respectively. The energy cost is shown per VBN for a publishing and subscribing time intervals of 1, 3, and 5 seconds. Due to frequent intra-VB forwarding of relatively large-sized data packets between VBNs during storage, the energy consumed by each VBN under the DHT scheme is found to be significantly higher than the BS scheme. On the other hand, the energy consumed during retrieval in both schemes are comparable, as expected, since both have incurred a similar cost of communication during retrieval.

Table 5-4 Average energy consumed per VBN due to communication for data storage (E_s) with τ_p = 1, 3, and 5 seconds

τ_p (sec)	1	3	5
DHT E_s (joules)	0.548	0.186	0.116
BS E_s (joules)	0.173	0.08	0.067

Table 5-5 Average energy consumed per VBN due to communication for data retrieval (E_r) with τ_p = 1, 3, and 5 seconds

τ_p (sec)	1	3	5
DHT E_r (joules)	0.457	0.301	0.239
BS E_r (joules)	0.464	0.303	0.224

5.5 Conclusion

This chapter presents an in-network data storage scheme with dynamic storage load balancing for virtual broker (VB) based pub/sub systems in wireless sensor networks (WSNs). An alternative scheme based on distributed hash table (DHT) was also adapted for data storage by the virtual broker nodes and used as a benchmark for comparison in our evaluations. From the results, it was observed that the proposed scheme was effective in balancing the storage range between virtual broker nodes, incurred a lower amount of intra-virtual broker communication, and as a result was more energy efficient than the distributed hash table-based approach. This virtual broker based pub/sub communication and storage mechanisms are designed to serve as the underlying information infrastructure to the proposed cross layer optimisation design that will be described in Chapter 7.

Chapter 6 An Ontology-based Context Model for WSN Management

6.1 Introduction

Existing context aware designs have focused mostly on the application level aspects, *i.e.* on the provisioning of automated services to end users under particular environment or user situations. The underlying network states and conditions which can impact on the application performance, however, are often ignored [120].

There exist designs for implementing context aware services and applications for ambient intelligence (AmI) in wireless sensor networks (WSNs), but there have been hardly any attempts to bring context awareness down to the underlying networks. AmI allows user-centric services to be performed based on contextual information. Unfortunately, the network itself has been left out of this “context aware” paradigm, and network optimisation designs are not yet considering this context aware approach. Optimisation techniques can be difficult to apply in the network as heterogeneous devices operate over hierarchical networks and every network device of different types can operate with its own standards and mechanisms. Not only do modifications have to be made to devices of one type to understand and interact with devices of other types, but additional communication and processing by the devices is also often required to make this happen. By contextualising information such as network states and conditions to be shared or exchanged between heterogeneous devices, interactions between such devices can be made easier and the application of network optimisation techniques involving heterogeneous devices can be facilitated [115].

The motivation of this chapter is to provide a context model for modelling different situations and conditions of a WSN based context aware system, i.e. AmI, with the aim of optimising the underlying network using contextualised information. The networks can react autonomously to the contextual information, executing appropriate tasks and actions according to the system contexts. In this chapter, an ontology-based context model is proposed for facilitating context sharing between heterogeneous devices and network optimisations exploiting context awareness in WSNs. The remainder of the chapter is organised as follows: Section 6.2 overviews some related works for WSNs. It also introduces the general concepts of ontology based context modelling. Section 6.3 defines the context classes that will be used by the proposed context model. Section 6.4 presents in detail the proposed ontology-based context model for WSNs. Section 6.5 illustrates with an application scenario how the proposed model can be applied to model the context used in a context-aware system designed for WSNs. Finally, Section 6.6 concludes the chapter.

6.2 Background and Related Work

Due to the nature of a WSN environment—i.e., large scale, dynamic, distributed, and resource constrained—it is challenging to perform network management functions such as configuration management, performance monitoring, and resource allocation. With heterogeneous devices, it is difficult to find a standard or unified solution for managing the entire network system. As WSN network management is expected to evolve towards the distributed and more automated management paradigm, promising enablers of such automation—context awareness and ontologies—are introduced in the next paragraph.

A number of context-aware designs for WSNs have been proposed, which aim mostly at addressing, through context awareness, one specific network-related issue, such as medium access control (MAC) [112], network routing [122, 166, 167], and network formation (clustering) [168]. In [9], the authors attempted to adapt protocols at multiple layers to improve the transmission of video sensors through context-aware cross-layer optimisation. Unlike these conventional layered architectures, a non-layered protocol architecture is proposed in [169] where protocols can be modularised and dynamically composed of functional blocks according to context information. At the system level, context has also been used for management of device energy [170] and storage allocation [113] in WSNs.

There exist few ontology-based context models specifically for WSNs. In [118], a centralised model for context management in WSN is proposed. In this model, all contexts are organised in an ontology structure and stored on a relational database of a base station that functions as sensor sink in the WSN. Context querying and reasoning processes are also performed on the base station. In [119], a scheduling approach for distributed description logic (DL) reasoning is proposed. Unlike in [118], where context reasoning is performed centrally at a base station, here the reasoning tasks can be offloaded to several sensor nodes in the WSN based on context parameters such as sensor node resource and network characteristics.

6.3 Context Classification

6.3.1 Defining the Context

Existing context definitions and models are not appropriate to be applied directly to many network scenarios since they target mainly the application or system levels, and do not encompass contexts about network elements and events. In this thesis, we modifies and extends the context definition in [70] to define context for a WSN based context aware system as follows:

> **Definition 6-1**: *Context for a WSN based context aware system is any shareable knowledge to represent situations or conditions from different parts of the system. The range of contexts can include, but not limited to, device context, network contexts, system contexts, and environment contexts.*

The device context provides knowledge about local device conditions, such as the energy state, storage level, and services provided by a node. The network context represents network wide situations and states, such as network topology, overall transmission capacity, or path qualities in the network. The system context represents status of a WSN based context aware system, such as the current executing tasks of an application or the state of the WSN performance, which can be shared with the underlying network. The environment context provides knowledge for a network to understand the changes of its environmental properties or attributes, such as the occurrence of a fire incident or detection of hazardous objects. Other types of context may also be included according to the system specifications. For example, user activity and user preference can be modelled under application context if the context aware system has a user-centric design.

6.3.2 Local and Global Context

Generally, all contexts of a WSN based context aware system can belong to the category of either local or global contexts. We refer to the entire contextual information of a WSN based system as Context Resource, as shown in Figure 6.1. The local context is the context that can be deduced locally within a single node device, and represents the conditions and states of the device as well as its constituent components, such as the state of the energy resource or the services carried by a node. On the other hand, the global context refers to contexts that cannot be deduced locally, but require the exchange of local contexts between multiple nodes. The global context may include the system and network level contexts, such as the network-wide conditions, or the user contexts generated at the system application level of the system.

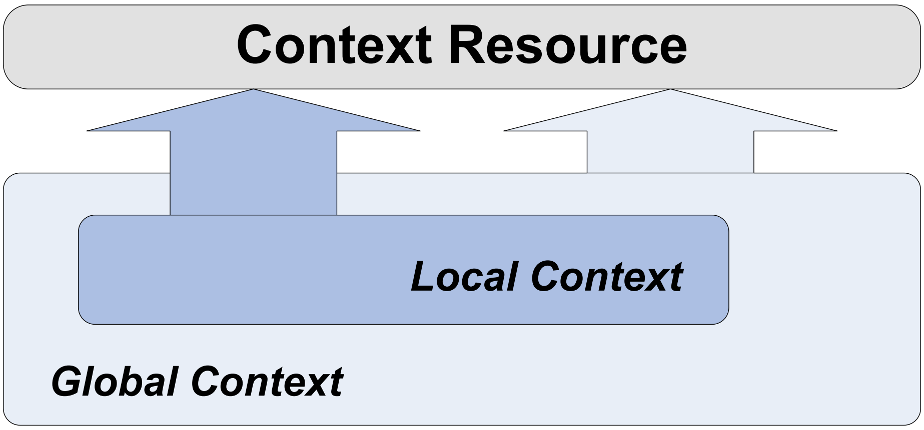


Figure 6.1 Local and global context

Hence, a single node is aware of not only the contexts about itself, but also about other nodes, the underlying network and the context aware system based on which context-aware management of WSN in the context aware system can be performed.

6.3.3 High and Low Context

Although contextual information can be classified according to their scope, i.e., as local or global context, they can also be organised into levels, *i.e.*, as high or low level contexts. Depending on the scope of contextual information, high level contexts can represent the status of a node, network, or system as a coherent entity, while low level contexts can describe the status about the elementary component parts of a node, network, or system. The relationships between local/global and high/low level contexts can be summarised as shown in Figure 6.2.

	Local Context	Global Context
High Level Context	Device (node) level context	IoT system and network level context
Low Level Context	Sub-device (component) level context	System and network constituent level context

Figure 6.2 Relationship between local/global and high/low level context

Low level local contexts refer to context about individual sub-components of a device, such as protocol states or sensor readings. Multiple low level local contexts can then be used to infer a high level local context, which represents the aggregated state of a device.

On the other hand, high level global contexts can cover the overall status of a context aware system and its underlying network, such as the overall QoS of the network, or the general performance of an application service provided by the context aware system. Low level global contexts can describe the status of those constituent parts of the system and network, such as the status of a routing path, the resource allocation states for data processing, or the environmental state from aggregated sensor readings, which are used to deduce high level global context.

For any context-aware network management scheme, when possible, it is essential to use high level contexts. Not only can high level contexts be inferred from multiple low level contexts, it is also possible to deduce low level contexts from a given high level context. For example, a single node can be constituted of multiple subsystems, such as the energy resource module, transceiver module, data storage module, and microcontroller module. Each subsystem is controlled and monitored by the operating system (OS) or firmware of the node. A high level local context such as “node is a cluster head” can be decomposed into corresponding low level local contexts for each subsystem, e.g., “fast energy depletion” for the energy resource module, “high inbound data traffic” for the transceiver module, “high memory load” for the storage subsystem, and “high CPU usage” for the microcontroller module. Thus, it is possible to make each single module aware of the status of other modules by deducing from high level contexts directly without having to exchange low level local contexts between them.

It will be even more necessary to use high level contexts when dealing with global contexts. Transmitting a piece of high level global context to be shared by all nodes in the network should incur less communication overhead than transmitting multiple low level global contexts. As communication can consume the majority of the overall energy resource for a node device [18], the amount of unnecessary communications should always be minimised by utilising high level global contexts. In addition, data storage and processing requirements for the nodes can also be minimised, as the amount of information to be shared and processed from a single high level global context should be less than that from multiple low level global contexts.

6.4 Proposed Ontology-Based Context Model

The proposed context ontology model provides vocabularies to represent context knowledge about network related situations and states of the context aware systems. The model is designed to facilitate context exchange and the understanding of such exchanged contexts between heterogeneous nodes of a context aware system to enable optimal context-aware management of the system’s underlying WSNs. Each node holds one instance of the model, which can use expression axioms to deduce the corresponding context knowledge according to the information exchanged at different levels and scopes.

This model is designed as a hierarchical structure of context classes where each class characterises the contextual information of one or more constituent parts of the context aware system. The bottom level of the model is raw information directly inherited from device components and system entities. The upper levels are the proposed context ontology to model and present inferred contexts for constituent parts of the system.

Here, raw information refers to any information, typically in numeric format, acquired directly from hardware and software components of a single node. In addition, raw information can be exchanged between nodes to infer low level global contexts, e.g., nodes in a given area can exchange raw sensor readings to derive the environmental context of their surroundings.

Normally, low level contexts can be derived by comparing numeric values of raw information with predefined thresholds. As a simple example, the "HIGH"/"LOW" energy state of a node is a low level context derived by comparing the level of residual energy on a node with a threshold value representing half of its full battery capacity. In addition, probabilistic frameworks such as Hidden Markov Models (HMM) [171, 172] can be applied when deriving high level contexts, which are not sensed directly but inferred from lower level contexts, and thus have a certain level of uncertainty, depending on the accuracy of the lower level contexts used [75]. For instance, if the derived movement and location of a certain node A are believed to be mostly (but not 100%) true, rather than to infer that "A is leaving the network", an inference that takes into account of the level of uncertainty such as "With high probability, A is leaving the network", can be more appropriate to describe the condition of the event.

The structure of the proposed context ontology model is shown in Figure 6.3. The *Context Resource* class is the root entity, which has two direct descendant classes: *Local Context* and *Global Context* classes. The following sections describe the proposed model according to their scopes and levels.

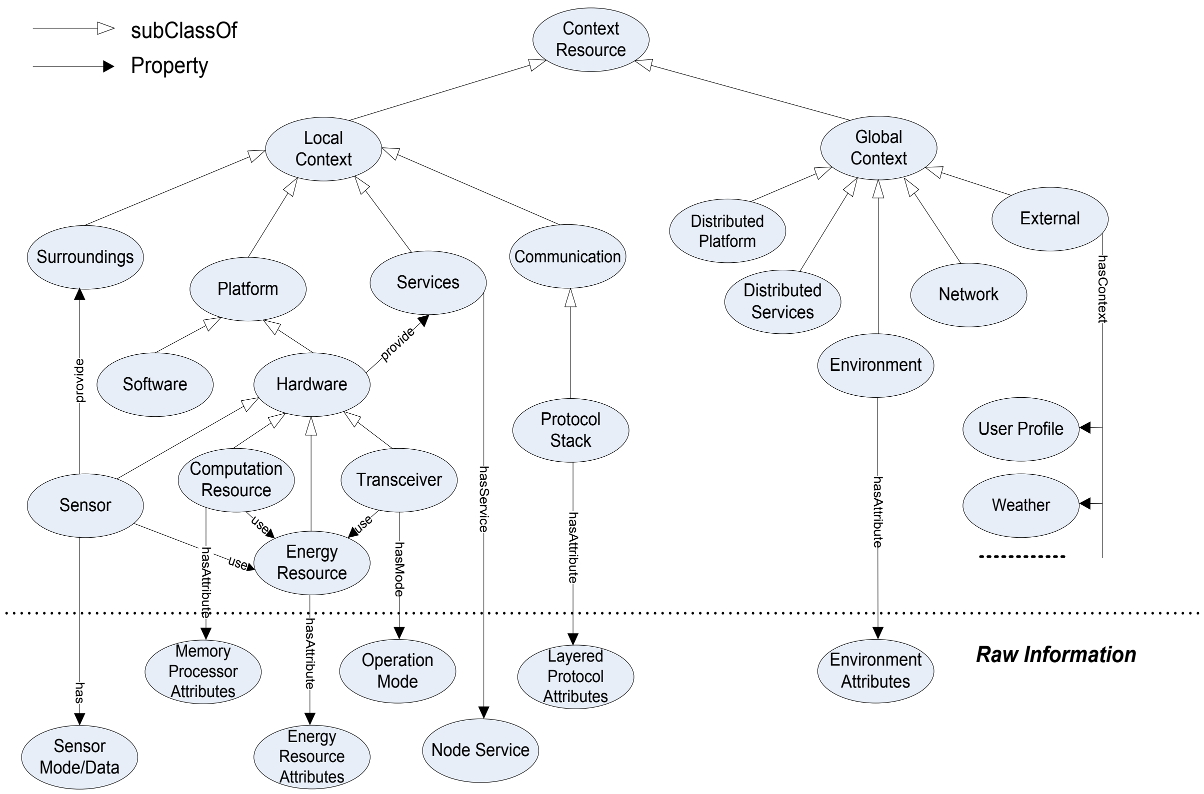


Figure 6.3 Context ontology structure for the context aware systems

6.4.1 Local Context

On the left side of this model shows the context ontology of the Local Context class, which describes the contextual information of a single node. It is a direct super class to four context classes, namely *Platform*, *Services*, *Surroundings*, and *Communication*.

6.4.1.1 Platform Context

The *Platform* context class provides a high-level description of a node's platform running state or capability based on context of its constituting hardware and software entities. The platform context can be utilised by a node to self-determine, or by other nodes (if provided with the context) to determine whether it could undertake a certain role or task, such as serving as a cluster head or performing data aggregation in a WSN.

A. Hardware Context

The *Hardware* context class describes the general resource or performance state of a node's hardware platform, which consists of four hardware components: sensor, transceiver, computation resource and energy resource, each with its own context class.

- *Sensor*: this context class can describe the operation mode of the sensing unit or basic context about its surroundings as deduced from its raw sensor data.
- *Transceiver*: this context class mainly describes the operation mode of the transceiver, e.g., transmit, receive, idle, sleep, or off. The duration and frequency for which the transceiver operates in each mode directly impact the amount of energy that it consumes. Other communication attributes such as channel conditions and bit rate shall be described by the *Communication* context class.
- *Computation Resource*: this context class describes the state of the processing and storage resources, e.g., CPU, memory or buffer storage, of the hardware platform. Such contexts can be particularly useful to support in-network mechanisms such as in-network video processing [9] and data storage [113].
- *Energy Resource*: this context class describes the state of the energy resources of a node, which can be a battery, an energy harvesting device (e.g., solar cell), or other energy module. It is defined to provide energy-related context of a node, such as its

residual energy level, energy consumption rate, or the energy generation rate of its harvesting hardware.

B. Software Context

The *Software* context class describes the state of the local OS and programs executing in a node. This can include the program configurations, performance of code executions, and other context that can be useful for WSN management. For instance, the code execution performance of a node, such as the time it takes by a program to process every 100 bytes of inbound data, can be used to determine whether the node can function as a distributed in-network processing node in the WSN.

6.4.1.2 Services Context

The *Services* context class describes the service roles or tasks that a node can perform. A single node may provide multiple services or carry out multiple tasks at the same time, e.g., a node can be a host to a software agent while acting as a data provider with its built-in sensing component. This node may also function as a cluster head in a hierarchical network, or a relaying node in a multi-hop path. By making the services context available within a node or to other nodes, internal or external functional entities may adapt accordingly to achieve better overall performance. For example, internal functional entities of a node located 1-hop away from a cluster head can adapt to increase the communication capacity in respond to the node's service context of being a frequent relaying node for inbound data to the cluster head.

6.4.1.3 Surrounding Context

The *Surroundings* context class describes the state of a node's surroundings as monitored by its built-in sensor. It is deduced from a time sequence of local sensor contexts, each of which only represents the state at the time when the raw sensor data is taken. For instance, from 10 consecutive sensor contexts provided by a built-in proximity sensor for human detection, of which eight are 'detected' and two are 'not detected', the context of the node's surrounding area over that time span could be inferred as 'highly active'.

6.4.1.4 Communication Context

The *Communication* context class provides a high-level description about the state of a node's communication with other nodes. The state can be in terms of the general quality, efficiency, security, frequency, availability, or pattern of communication. It is deduced based on the low level contexts from the node's communication protocol stack.

A. Protocol Context

The *Protocol Stack* context class describes the state of each protocol layer in a node's protocol stack. The physical layer context may express characteristics such as signal quality, channel conditions, interference, and spectrum availability. The medium access control (MAC) layer context may describe the availability, quality, and utilisation of the links to the node's direct neighbours, frame collision, and fairness of channel access. The network layer context may capture properties such as the quality, efficiency, and security of a node's multi-hop path to other nodes, traffic distribution pattern, and group membership if the node participates in group communication. The transport layer context may provide knowledge about end-to-end reliability of connection between the node and other nodes, or the occurrence of congestion along its path of communication. Finally, the application layer may present the application's context of use, e.g., involving real-time or non-real time transmission, indoor or outdoor environment, mobile or static scenario, local area or wide area deployment, cooperative or non-cooperative nodes, *etc*. which can be utilised to infer the performance, efficiency, or security requirements of the node's communication for the application.

6.4.2 Global Context

On the right side of this model shows the context ontology of the *Global Context* class, which describes the contextual information of the context aware system based on local contexts exchanged between nodes and other external contexts. It is a direct super class to five context classes, namely *Distributed Platform*, *Distributed Services*, *Environment*, *Network*, and *External*.

6.4.2.1 Distributed Platform Context

The *Distributed Platform* context class provides a high-level description of a system's distributed platform running state or capability based on exchanged *Platform* context between nodes in the system. Each node of the distributed platform can adapt to this knowledge to improve the system performance. For example, in a distributed in-network storage system, this context can be used by a node to become aware of and adapt to the storage and computation resource levels of other nodes in order to balance the data storage and processing loads among them.

6.4.2.2 Distributed Services Context

The *Distributed Services* context class describes the service roles or tasks that can be performed by multiple nodes in a system based on exchanged *Services* contexts. This knowledge can be applied to assist in the selection of nodes to undertake certain networking roles or tasks. For example, in a cluster-based WSN, this context can be used by a departing cluster head to select the best node to take over its role without re-clustering the network.

6.4.2.3 Environment Context

The *Environment context* class describes the state of a system's physical environment based on exchanged *Surroundings* contexts. It provides nodes with a wider view of the event occurrences in their environment than is possible with only local *Surroundings* context. In turn, nodes can utilise this knowledge to make more informed networking decisions. For example, in an event detection WSN, nodes detecting an event occurrence will transmit data about the event to a sink. By adapting their routing decisions to event contexts, packet congestion in the network can be avoided by routing data through nodes which have not detected any events.

6.4.2.4 Network Context

The *Network* context class provides a high-level description about the state of a system's network based on exchanged *Communication* contexts as well as contexts from any deployed network management station (NMS) for WSN, e.g., [173]. The heightened awareness of the network state can bring about more effective solutions to address problems, particularly those due to inherent constraints (e.g., resource constraints) and vulnerabilities (e.g., open distributed nature) of WSN. This may consequently give rise to new solutions such as network-state aware resource scheduling or intrusion detection techniques.

6.4.2.5 External Context

The *External* context class represents any context originated from a source external to the system. This may include user related contexts of context aware applications such as user's profile, preferences, and activity schedule, or contexts derived from weather forecast data, indoor or outdoor map information, which can be useful for WSN management.

6.5 Scenario Analysis

In this section, a context-aware multi-path selection (CAMS) algorithm for video streaming in wireless multimedia sensor networks [9] is selected as a use case of our proposed ontology model. This section illustrates how this proposed ontology model can be applied to an existing design. In this algorithm, a sensor node can generate video streams of its surrounding environment from its physical onboard sensor components comprising of an image camera and microphone. Thus, each single video stream can be decomposed into two sub-streams—image and audio streams, and transmitted over multiple node-disjoint paths simultaneously. The CAMS algorithm can choose the right number of paths for transmitting each stream so that the overall throughput is maximised. The CAMS prioritises the transmissions and the available routing paths according to the stream content, and end-to-end delay of the path, respectively. The aim is to transmit high-priority content over low delay paths whenever possible.

The original CAMS algorithm does not explicitly consider the issue of heterogeneous nodes. However, in this analysis, we consider a network composed of heterogeneous video sensors of different resolutions. As a result, differences between video sensors in their end-to-end delay requirements can be expected. The end-to-end delay requirement of a high resolution video sensor will be more stringent than a low resolution video sensor as more information bits will be transmitted for a given image or audio frame, i.e., more bits per image pixel or digitised sound sample.

Based on the proposed ontology structure, the context model for CAMS as proposed in [9] is shown on the left side of Figure 6.4, which involves only local contexts, as the algorithm does not perform any exchange of priority related information. The right side of Figure 6.4 shows the context model for CAMS that has been extended to utilise global context (to be explained later). The associated syntaxes used are defined in Table 6-1. The local context resource, CAMS priority, is constituted of *Content priority* and *Delay priority*, which can be seen as corresponding to the *Surroundings* context, and *Communication* context, respectively, of the context ontology structure shown in Figure 6.3.

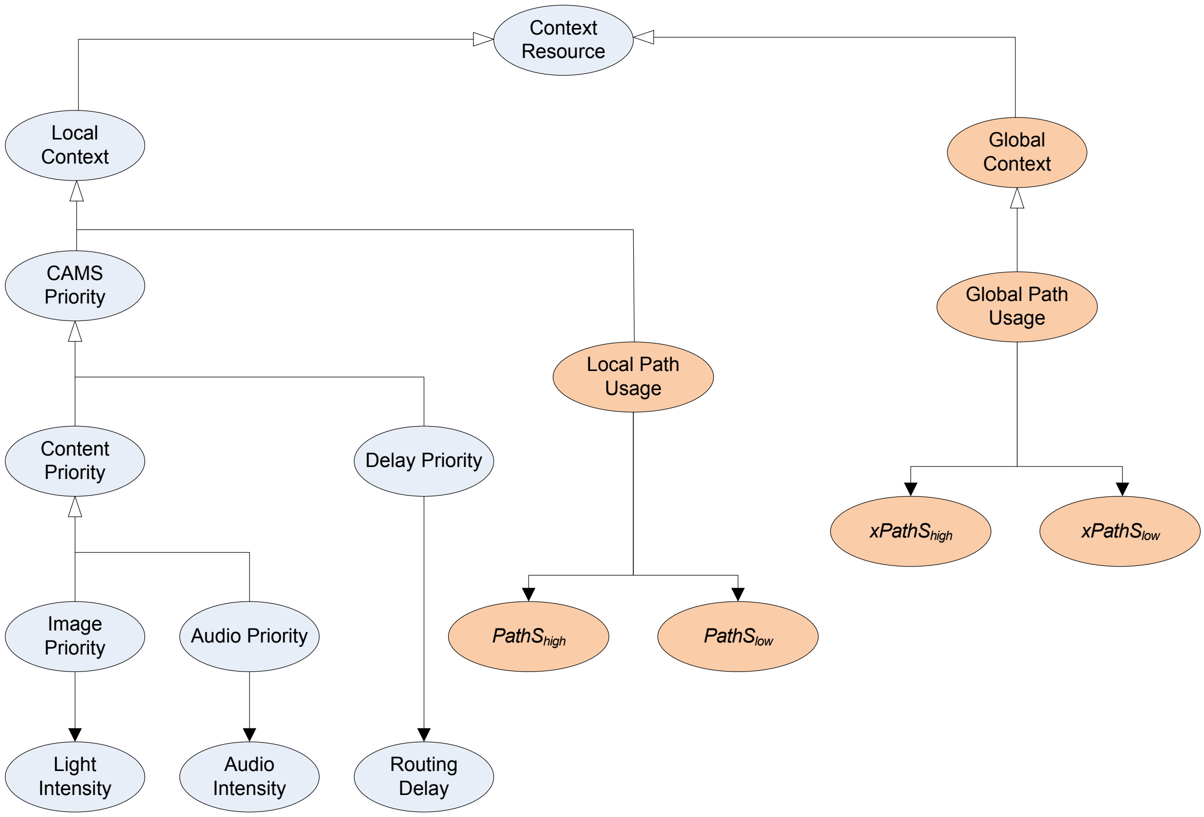


Figure 6.4 Context ontology model for Context-Aware Multi-Path Selection (CAMS)

In CAMS, a video stream can be presented in Description Logic [174] as:

$$Video \equiv Image \sqcap Audio$$

which expresses that a single video stream is composed of an image stream and an audio stream, each being a sequence of image frames, and audio frames, respectively. A video source node has to decide the priority of each outbound image and audio frame based on their importance. Two qualitative context values for frame importance, *High_Priority* and *Low_Priority*, can be assigned by the video source nodes. A high priority image frame is defined as:

$$High\_Priority.Image \equiv (>I_{brightness}\ BrightnessLevel.Image) \sqcap (\leq I_{loudness}\ LoudnessLevel.Audio \sqcup High\_Priority'.Image)$$

where an image frame is assigned a high priority if its brightness (*BrightnessLevel*) is higher than predefined brightness threshold $I_{brightness}$, and, either the loudness (*LoudnessLevel*) of the associated audio frame is equal or lower than predefined loudness threshold $I_{loudness}$, or the priority of the immediate previous image frame is high as denoted by Boolean parameter *High_Priority'.Image*. Similarly, a high priority audio frame is defined as:

$$High\_Priority.Audio \equiv (>I_{loudness}\ LoudnessLevel.Audio) \sqcap (\leq I_{brightness}\ BrightnessLevel.Image \sqcup High\_Priority'.Audio)$$

Table 6-1 Syntax definitions for CAMS algorithm

Syntax	Definition
BrightnessLevel	Brightness of the split image frame
LoudnessLevel	Loudness of the split audio frame
$I_{brightness}$	Brightness threshold for deciding the frame priority
$I_{loudness}$	Loudness threshold for deciding the frame priority
Path	A single routing path between a source and destination
PathS	Set of *Path* between a source and destination
$Delay_{path}$	End-to-end delay of a path
$DelayS_{path}$	Set of $Delay_{path}$ for each available path in *PathS*
$T_{high\text{-}priority\_max}$	Maximum time for end-to-end transmission of a high-priority frame
$T_{low\text{-}priority\_max}$	Maximum time for end-to-end transmission of a low-priority frame
$PathS_{high}$	Set of available paths for high-priority frame transmission
$PathS_{low}$	Set of available paths for low-priority frame transmission
N	Total number of available paths in *PathS*
M_{high}	Number of paths in $PathS_{high}$
M_{low}	Number of paths in $PathS_{low}$
$xPathS_{high}$	Set of exchanged available paths for high-priority frame transmission
$xPathS_{low}$	Set of exchanged available paths for low-priority frame transmission

where *High_Priority'.Audio* is the equivalent Boolean parameter denoting the priority of the immediate previous audio frame. Both *High_Priority'.Image* and *High_Priority'.Audio* are initialised to False and updated to True or False according to the respective high or low priority of each transmitted image and audio frame.

Based on the above definitions, only one type of frame—image or audio frame—of the same split video frame can be assigned as high priority, *i.e.*, both image and audio frames cannot be assigned as high priority at the same time. In addition, when both image and audio frames are above (or below) their respective brightness and loudness thresholds, they will inherit the respective priority level assigned to their immediate previous image and audio frames (*High_Priority'.Image, High_Priority'.Audio*) in order to maintain stability of the video streaming as specified in [9].

All available node-disjoint paths between a source-destination pair can also be assigned with a qualitative context based on their transmission latencies. Two qualitative context values used are: *Guaranteed_Trans._Delay* and *Non_Guaranteed_Trans._Delay*. A path is assigned with a non-guaranteed transmission delay context (*Non_Guaranteed_Trans._Delay*) if it neither satisfies the end-to-end delay requirement of the high-priority frame nor low-priority frame:

$$Non\_Guaranteed\_Trans.\_Delay.Path \equiv (>T_{high\text{-}priority\_max}\ Delay_{path}) \sqcap (>T_{low\text{-}priority\_max}\ Delay_{path})$$

If a path satisfies the end-to-end delay requirement of either the high- or low-priority frame, the path is assigned with a guaranteed transmission delay context (*Guaranteed_Trans._Delay*):

$$Guaranteed\_Trans.\_Delay.Path \equiv (\leq T_{high\text{-}priority\_max}\ Delay_{path}) \sqcup (\leq T_{low\text{-}priority\_max}\ Delay_{path})$$

The available routing paths for high-priority frame transmission ($PathS_{high}$) between a source-destination pair is defined as a set of paths in *PathS* whose end-to-end delay is equal or less than the end-to-end delay requirement of high-priority frame ($T_{high\text{-}priority\_max}$):

$$PathS_{high} \equiv PathS \sqcap \forall DelayS_{path}. \leq T_{high\text{-}priority\_max}$$

Similarly, the available routing paths for low-priority frame transmission ($PathS_{low}$) between the same source-destination pair is defined as:

$$PathS_{low} \equiv PathS \sqcap \forall DelayS_{path}. \leq T_{low\text{-}priority\_max}$$

It should be noted that the above $T_{high\text{-}priority\_max}$ and $T_{low\text{-}priority\_max}$ should be appropriately initialised for each node's instance of the ontology model based on its video resolution. This will ensure that all frames are transmitted over paths whose end-to-end delay satisfies the end-to-end delay requirement corresponding to the priority and resolution of the frames.

The CAMS supports multi-path routing, and the relationship between the number of available paths for high-priority frame transmission (M_{high}), low-priority frame transmission (M_{low}), and total number of available paths (N) can be shown as:

$$N \geq M_{high} + M_{low}$$

$$M_{high} < M_{low}$$

which expresses that the number of paths for frame transmission (high and low priority) is bounded by the total number of available paths, and due to the more stringent delay requirement of high priority frame, *i.e.*, $T_{high\text{-}priority\_max} < T_{low\text{-}priority\_max}$, there will be fewer paths in $PathS_{high}$ (M_{high}) for high-priority frame transmission than in $PathS_{low}$ (M_{low}) for low-priority frame transmission.

The original CAMS algorithm is modified to use our proposed context model as discussed above and shown on the left side of Figure 6.4. The following shows how frames are transmitted by our modified CAMS algorithm under three case scenarios:

// Case 1: no transmission if none of the available paths meets the end-to-end delay requirement.
if (∀Non_Guaranteed_Trans._Delay.PathS)
{No_Transmission}
end if

// Case 2: if all available paths meet the end-to-end delay requirement, transmit the high-priority stream simultaneously over the paths in PathS_{high}. If there are still

unused paths remaining in PathS, transmit the low-priority stream simultaneously over these paths. Otherwise, discard the transmission of the low-priority stream.

if (∀Guaranteed_Trans._Delay.PathS)
{Transmit the High-Priority Stream (sequence of high-priority frames)
Simultaneously over M_{high} number of paths in $PathS_{high}$
if ($M_{high} < N$)
{Transmit the Low-Priority Stream (sequence of low-priority frames)
Simultaneously over ($N – M_{high}$) number of paths in ($PathS \sqcap (\neg PathS_{high})$)}
end if
}
end if

// Case 3: if only a subset of available paths meet the end-to-end delay requirement, transmit the high-priority stream simultaneously over the paths in $PathS_{high}$. If there are still unused paths remaining in ($PathS_{low} \sqcap (\neg PathS_{high})$), transmit the low-priority stream simultaneously over these paths. Otherwise, discard the transmission of the low-priority stream.

if (∃Guaranteed_Trans._Delay.PathS)
{Transmit the High-Priority Stream (sequence of high-priority frames)
Simultaneously over M_{high} number of paths in $PathS_{high}$
if ($M_{high} < M_{low}$)
{Transmit the Low-Priority Stream (sequence of low-priority frames)
Simultaneously over ($M_{low} – M_{high}$) number of paths in ($PathS_{low} \sqcap$
$(\neg PathS_{high})$)}
end if
}
end if

As mentioned earlier, the CAMS as proposed in [9] does not perform any exchange of priority related information, and therefore its selection of node-disjoint paths for frame transmissions is based only on local contexts, *i.e.*, CAMS priority. However, it is conceivable that if individual nodes can be made aware of and adapt their behaviour to not only their local context, but also the global context of other nodes, a more coherent and optimal management of the network can be achieved. To illustrate the usage of global contexts, CAMS has been extended for nodes to utilise another type of local context (local path usage), which can be shared or exchanged between nodes as global context. Therefore, a new case scenario has been designed and its corresponding context model is shown on the right side of Figure 6.4.

The scenario involves multiple pairs of source-destination nodes performing CAMS at the same time. As in previous scenarios, the *PathS* of the source node will hold the available node-disjoint paths to its destination, and these paths can be further placed into set $PathS_{high}$ or $PathS_{low}$ depending on whether they satisfy the delay requirement of the high-priority frame, or low-priority frame, respectively. To motivate the need for an improved CAMS, consider the case where multiple source-destination pairs performed CAMS only according to their local contexts, resulting in some node-disjoint paths between different communicating pairs to become 'node-joint' or overlapped as shown in Figure 6.5.

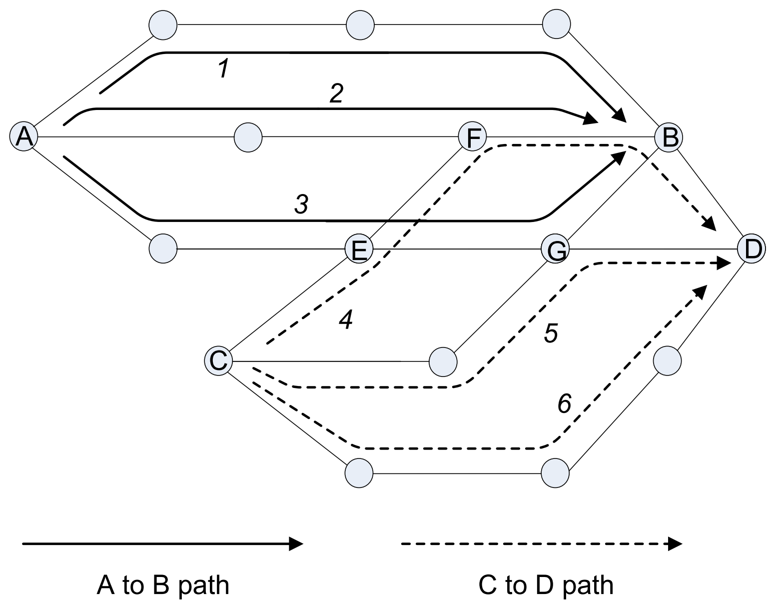


Figure 6.5 Overlapping node-disjoint paths of two communicating pairs

The common relay nodes *E*, *F*, *G*, and *B* can potentially become traffic bottlenecks for Paths 2 and 3, and Paths 4 and 5 of node pair *A–B*, and node pair *C–D*, respectively. In order to avoid the occurrence of such situations, nodes can be permitted to exchange context about their available routing paths and the CAMS can be extended to harness the knowledge of such global contexts.

Under the extended CAMS, nodes will behave as follows: After the source node determines its routing paths for high-priority ($PathS_{high}$) and low-priority ($PathS_{low}$) frame transmissions, but before it transmits any frame according to the three case scenarios, the node will share or exchange its local $PathS_{high}$ and $PathS_{low}$ information with other nodes, received upon which will be stored as $xPathS_{high}$ (exchanged $PathS_{high}$) and $xPathS_{low}$ (exchanged $PathS_{low}$). On receiving, the node can determine if any of its paths in $PathS_{high}$ and $PathS_{low}$ are 'node-joint' or overlapped with those in $xPathS_{high}$ and $xPathS_{low}$.

As shown on the right side of Figure 6.4, a new local context class *Local Path Usage* (constituted of local $PathS_{high}$ and $PathS_{low}$) and a global context class *Global Path Usage* (constituted of $xPathS_{high}$ and $xPathS_{low}$) have been introduced, which can be seen as corresponding to the *Communication* context, and *Network* context, respectively, in the context ontology structure shown in Figure 6.3.

An available routing path cannot be categorised into $PathS_{high}$ and $PathS_{low}$ at the same time. On the other hand, the same routing path may not be categorised into either $PathS_{high}$ or $PathS_{low}$ if it does not satisfy the delay requirement of either high-priority or low-priority frame. Each *Path* in $PathS_{high}$ and $PathS_{low}$ can be assigned with one of the following two qualitative context values:

$$RelayNodeShared.Path = \exists\ xPathS_{high}{:}hasRelayNode(\exists Nodeof(Path)) \sqcup \exists\ xPathS_{low}{:}hasRelayNode(\exists Nodeof(Path))$$

$$NoRelayNodeShared.Path = \exists\ xPathS_{high}{:}hasNoRelayNode(\exists Nodeof(Path)) \sqcap \exists\ xPathS_{low}{:}hasNoRelayNode(\exists Nodeof(Path))$$

which expresses that if any relay node of a *Path* in $PathS_{high}$ and $PathS_{low}$ is also a node of a path (e.g., source, destination, or relay node) in $xPathS_{high}$ and $xPathS_{low}$, this *Path* will be assigned with the state *RelayNodeShared*, otherwise it will be assigned with the state *NoRelayNodeShared.*

For a path 'marked' as having one or more shared nodes, the source node may perform a decision function to determine whether or not it should keep this path for frame transmission. The design of the decision function is often application/scenario specific, which may be based on probabilistic models, fuzzy logic, decision trees, or other reasoning mechanisms.

Figure 6.6 shows the flow of steps to handle shared paths in CAMS with global context. For each *Path* in $PathS_{high}$ 'node-joint' with other paths in $xPathS_{high}$, the node will perform a decision function to decide whether or not it should keep this *Path* locally in its $PathS_{high}$.

The same procedure is applied for each *Path* in $PathS_{low}$ 'node-joint' with other paths in $xPathS_{low}$. However, if a *Path* in $PathS_{low}$ is 'node-joint' with paths in $xPathS_{high}$, this *Path* will be removed from $PathS_{low}$ of this node. In other words, priority for using this Path is given to nodes that will be using it for high-priority frame transmission, *i.e.*, as *Path* in $PathS_{high}$. This step will be taken as well by other nodes with *Path* in their $PathS_{low}$ 'node-joint' with paths in their $xPathS_{high}$. On the other hand, if a *Path* in $PathS_{high}$ is a 'node-joint' with any paths in $xPathS_{low}$, the node will keep this *Path* in its $PathS_{high}$.

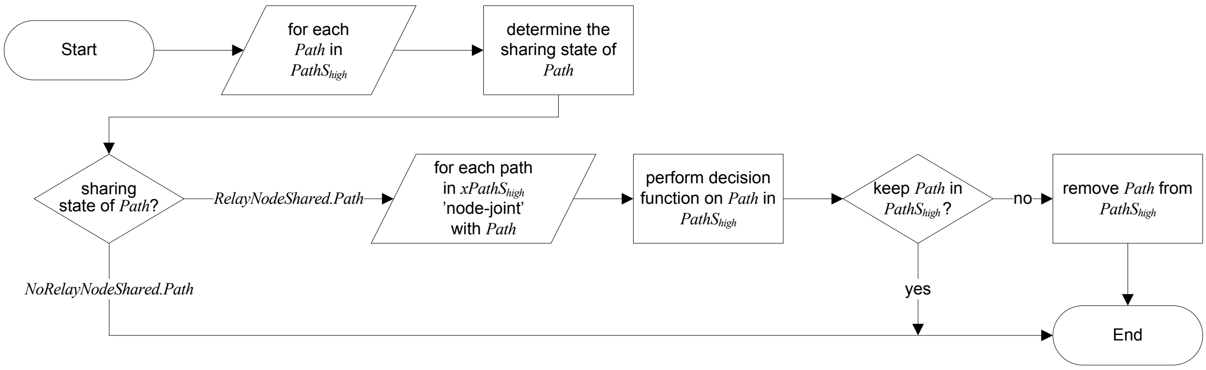


Figure 6.6 Handling of shared paths in CAMS with global context

As a formalism for ontology representation, the RDF/XML serialisation of the proposed ontology in Figure 6.4 is shown. However, XML is seen as a 'heavy' syntax for resource-constrained devices. Thus, for implementing the proposed ontology on sensor nodes, more compact XML representations such as binary XML formats should be used [175]. Another promising approach uses streaming HDT as lightweight serialisation format for RDF and Wiselib Tuplestore for storing RDF data locally on embedded devices such as sensor nodes is proposed in [176].

```
<owl:Class rdf:ID="CAMSPriority">
  <owl:Class rdf:ID="LocalContext">
    <owl:Class rdf:ID="ContentPriority">
      <rdfs:subClassOf rdf:resource="#LocalContext"/>
    </owl:Class>
    <owl:DatatypeProperty rdf:ID="PriorityState" >
      <rdfs:domain rdf:resource=" #ContentPriority"/>
      <rdfs:range rdf:resource="xsd:string"/>
    </owl: DatatypeProperty >
    <owl:ObjectProperty rdf:ID="ImagePriorityState">
```

```
    <rdfs:domain rdf:resource="#ContentPriority"/>
    <rdf:range rdf:resource="#PriorityState"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="AudioPriorityState">
    <rdfs:domain rdf:resource="#ContentPriority"/>
    <rdf:range rdf:resource="#PriorityState"/>
  </owl:ObjectProperty>
  <owl:Class rdf:ID=" DelayPriority">
    <rdfs:subClassOf rdf:resource="#LocalContext"/ >
  </owl:Class>
  <owl:ObjectProperty rdf:ID="RoutingDelay">
    <rdfs:domain rdf:resource="#DelayPriority">
    <rdf:range rdf:resource="xsd:double">
  </owl:ObjectProperty>
  <owl:Class rdf:ID=" LocalPathUsage">
    <rdfs:subClassOf rdf:resource="#LocalContext"/ >
  </owl:Class>
  <owl:ObjectProperty rdf:ID="PathS_high">
    <rdfs:domain rdf:resource="#LocalPathUsage">
    <rdf:range rdf:resource="xsd:string">
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="PathS_low">
    <rdfs:domain rdf:resource="#LocalPathUsage">
    <rdf:range rdf:resource="xsd:string">
  </owl:ObjectProperty>
 </owl:Class>
 <owl:Class rdf:ID="GlobalContext">
  <owl:Class rdf:ID="GlobalPathUsage">
    <rdfs:subClassOf rdf:resource="#GlobalContext"/ >
  </owl:Class>
  <owl:ObjectProperty rdf:ID="xPathS_high">
    <rdfs:domain rdf:resource="# GlobalPathUsage">
    <rdf:range rdf:resource="xsd:string">
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID=" xPathS_low">
    <rdfs:domain rdf:resource="# GlobalPathUsage">
    <rdf:range rdf:resource="xsd:string">
  </owl:ObjectProperty>
 </owl:Class>
</owl:Class >
```

The above use cases have illustrated how the proposed ontology-based context model can be applied to contextualise the mostly numeric data in the network, and how the contextualised data can be used beyond their sources by facilitating context sharing between network entities, all with the goal of enabling context-aware management of WSNs, which can also harness the rich context knowledge of the context aware systems. In comparison with the original CAMS, the 'contextualised' CAMS presented in this chapter, *i.e.*, CAMS using the proposed context ontology model, is more prepared to perform in the context aware environment, since all network entities share a common understanding of the network related information originated from heterogeneous sources, but contextualised using the same proposed model for a unified unambiguous interpretation. As mentioned, while there exist context ontology models for mitigating the complexity of systems operating in heterogeneous environments, most if not all of them are focused on modelling system or application level contexts, with very few or no ontology models proposed for network level contexts. Hence, to the best of our knowledge, the proposed model in this chapter is one of the first (if not the first) for context aware WSN management.

6.6 Conclusion

This chapter proposed an ontology based context model for context aware wireless sensor network (WSN) management. Unlike previous models that focus mainly on contexts at application and service levels, an ontology model is proposed for the first time that focuses on representing network related situations of a context aware system as contexts which can be shared, understood, and utilised by heterogeneous nodes for context aware management of the underlying wireless sensor networks. Under the proposed model, context knowledge from system, network and node levels can be classified as local or global, and high or low level contexts according to their scope and level of aggregation. A use case of the proposed model is presented in which a context ontology is designed for and applied to an existing context-aware algorithm for cross-layer optimisation in wireless sensor networks.

The designed ontology models the contextual information found in common wireless sensor network scenarios, and makes them shareable between different parts of a system. In the next chapter, a context aware cross layer optimisation mechanism will be presented based on this proposed ontology model for context information exchange.

Chapter 7 AmI Context-based Cross-Layer Design in WSNs

7.1 Introduction

Direct information exchange between non-adjacent protocol layers or "cross-layer" (CL) interaction can optimise network performances such as energy efficiency and delay [177]. This is particularly important for wireless sensor networks (WSNs) where sensor devices are energy-constrained and deployed for real-time monitoring applications.

In the current literature, most research on CL optimisation for WSNs has focused on interactions between lower layers of the protocol stack, i.e. physical, medium access control (MAC) and network layers [178]. There has also been research on CL optimisation that considered application requirements, e.g. the quality-of-service (QoS) requirements of multimedia applications [46].

Unlike these previous works that either were not concerned with the application layer or used the application only to define the requirements of CL optimisation, this chapter focuses on how application-derived information, i.e. the user context information derived from an ambient intelligence (AmI) application, can optimise the underlying WSN performance through CL interactions.

At the system level, an AmI system can adapt its intelligent services to user-related context information. However, its underlying WSN rarely considers AmI context information. If user context information can influence how an intelligent system responds, there is also a possibility for the underlying network to use the user context information. This could allow the network protocols becoming *smart* by adapting their functionality to user situations.

This chapter proposes a generic and customisable CL framework that utilises AmI context information from the application layer for optimising protocol performance in WSNs. This framework is sufficiently generic to customise for different AmI applications. In addition, the proposed framework is applied to two protocols: a MAC protocol and a network routing protocol, for WSNs. The backoff behaviour of a contention-based MAC protocol, and the path selection of an Ad hoc On-demand Distance Vector (AODV) based routing protocol are optimised by adapting their protocol functions in real-time to the user context information inferred from an AmI application.

The rest of the chapter is organised as follows. Section 7.2 presents the motivating scenario. Section 7.3 describes the proposed framework. Section 7.4 outlines the optimised MAC and routing protocol. Section 7.5 presents and discusses the evaluation results. Finally, Section 7.6 concludes the chapter.

7.2 Motivating Scenario

This section describes a common scenario in the AmI domain, and presents the functions of the AmI system and its underlying WSN in the scenario. In addition, this section also discusses the challenges of optimising the WSN with the available context information from the AmI system. This scenario depicts an intelligent event notification system for people in an outdoor AmI environment. Users can be notified about the occurrence of physical events in their surroundings that could be relevant to them based on their attributes, such as age and disability status, and the context of the events. This notification system can be applied to people in all age groups.

Figure 7.1 shows the scenario of an intelligent event notification system operating over a WSN deployed in an outdoor environment. A WSN is deployed in this environment where some sensor nodes are embedded into inanimate objects such as buildings and roads, while others could be on mobile objects such as cars and humans (e.g. wearable sensors). The sensor nodes can continuously monitor and detect changes in the physical properties and attributes of their environment. Each person in this environment is assumed to carry a form of smart device, e.g. smart phone, which has an intelligent software agent running on it. The intelligent agent can collect sensor data from the underlying WSN where the person is located, infer events with the collected sensor data, and notify the user about some inferred events that may affect him/her. For instance a user can receive a notification alarm from his smart device when it anticipates an incoming vehicle on the user's movement path.

The data communication and storage architecture of this notification system is presumed to be an implementation of the virtual broker (VB) based publish/subscribe (pub/sub) design proposed in Chapter 5, where the sensor nodes are data publishers and the agents on the smart devices are data subscribers for context inference. The underlying WSN is partitioned and VBs are formed according to the design proposed in Chapter 4. This scenario represents a distributed system architecture in which no centralised control is required, while the smart agents exist for individual users for the purpose of generating the notifications relevant to them. This VB-based pub/sub platform allows the intelligent notification system to have the information exchange capacity through the WSN. Therefore, the sensor data becomes available on the VBs for subscription by any intelligent agents, while a particular agent can subscribe to data for detecting events in the environment that may affect the user based on his/her states and attributes.

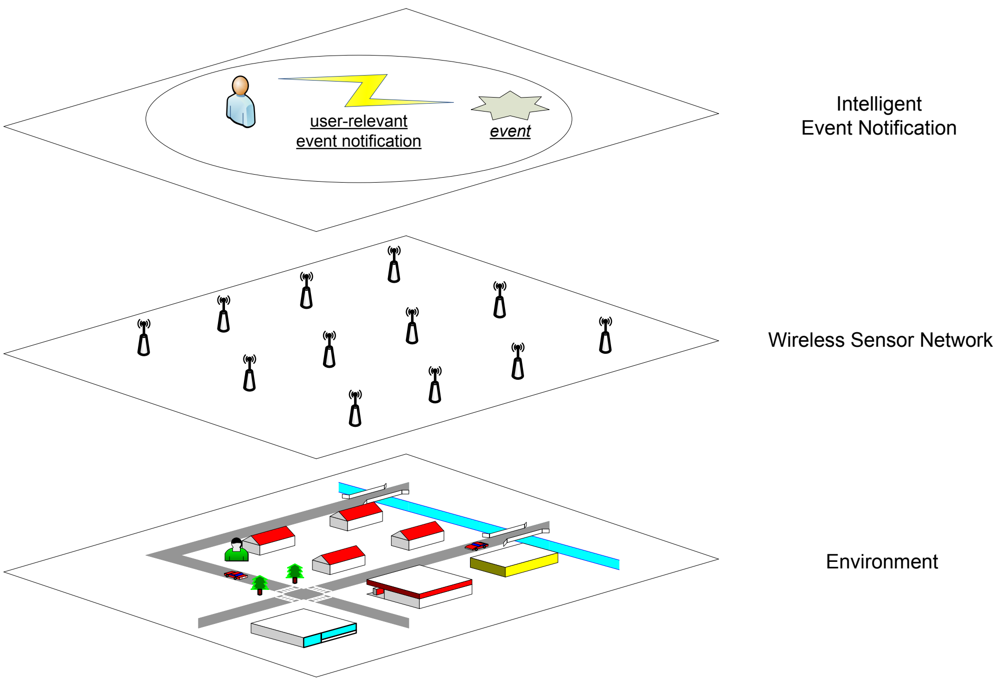


Figure 7.1 Scenario overview

Consider the case of Sam, a nine year-old boy playing basketball in his backyard. The ball rolls to the street next to the backyard and Sam runs to retrieve it. At the same time, John is driving a car that is turning into the street where Sam is located. For this notification system, the sensor nodes in the environment are publishing their data continuously to the VBs, e.g. data from the motion sensors along the street, proximity sensors on the backyard fence, and activity sensors worn by Sam. The intelligent agent on Sam's smart device is also receiving the subscribed sensor data on the VBs about the surroundings of Sam based on his location, e.g. the sensor data related to the street next to the backyard. With the collected sensor data, the agent can infer the occurrence of possible events that may affect Sam. At this time, the agent infers that a vehicle driven by John may crash into Sam. Hence, the agent declares a state of emergency and the smart device immediately alarms Sam about this dangerous situation and advises him not to cross the street until the vehicle has passed. Simultaneously, while John is driving, his intelligent agent constantly subscribes to receiving sensor data about the streets ahead of his vehicle. In this occurrence, John's smart device will notify him that someone may cross the street ahead, and suggest him to slow down and watch out for the person to avoid any potential incident.

Challenge of Using AmI Context for WSN Optimisations

The purpose of the WSN in the above AmI scenario is to collect sensor data of the monitored environments and deliver them to the intelligent agents for the inference of possible events related to particular users. A WSN node can only generate raw sensor data, which may not present any meaningful information if the data is not processed further. Therefore, communicating the sensor data from the WSN to the intelligent agents is an important part of the AmI system in order to successfully deduce and notify the user-relevant events on time.

In this scenario, the pub/sub based communication mechanism is used. Under this mechanism, data from the same sensor can be subscribed by different intelligent agents at different rates, i.e. frequency of receiving an update of the data, under different situations. For instance when Sam is detected to be approaching the street, the intelligent agent of his smart device takes the sensor data about vehicles on the street very seriously, and changes its subscription to receive an update of this sensor data more frequently. On the other hand, the same sensor data would be less important to Sam's agent if the ball had not rolled onto the street, and his agent would have just monitor this sensor data at a normal rate.

Therefore, the importance of a piece of sensor data may not be determined by the individual sensor nodes, but by an AmI system, such as Sam's intelligent agent in the above scenario. In addition, different agents may perceive the importance of a piece of sensor data differently even at the same time. For instance, Sam's agent will regard the sensor data from the street as highly important, while those of his playmates who remain in the backyard may not. Therefore, there is a need for a mechanism for the sensor nodes to optimise the communication of their data to the AmI systems in different situations.

7.3 A Generic Context Aware Cross-Layer Framework

The importance of the sensor data to particular users can be known by the sensor nodes via *reversed* pub/sub communications, where intelligent agents are context publishers that publish their inferred contexts to the VBs, and the sensor nodes are context subscribers that subscribe to particular context based on the sensor nodes' attributes such as location or sensor data type. In other words, a sensor node can subscribe to the context of an event occurring in its area and which requires its sensor data in order to be inferred. In this way, sensor nodes can become AmI context aware, and accordingly optimise the communication of their data to the intelligent agents through cross-layer interaction.

A generic CL framework that can be adapted by any context-aware systems is proposed where protocol optimisation can be achieved by allowing the inferred AmI contexts to become available to the sensor nodes through a context exchange mechanism, and allowing each node to control its transmission of any outbound data based on the data content and the inferred context of its surrounding. The framework can be implemented in firmware, and each node in the network maintains an instance of the implementation.

This section presents the architecture of the framework, including the functionality and behaviour of its constituent components. There are three parts to this framework: i) communication mechanism for network-wide AmI context exchange; ii) node architecture for node-level context handling and CL optimisation; and iii) ontology-based context modelling and reasoning mechanism for representing and inferring context within this framework.

7.3.1 Communication Mechanism

The communication mechanism of this framework is based on the data-centric publish/subscribe (pub/sub) paradigm [133]. Under this mechanism, the AmI system is a *publisher* that may publish inferred contexts, while sensor nodes are *subscribers* that may subscribe to published contexts. A VB is a virtual brokerage entity formed by a cooperative group of sensor nodes that share the responsibility of providing context storage and retrieval services. Hence, any AmI context can be disseminated to subscribing sensor nodes for making informed optimisation decisions based on situations of their monitored environment.

Figure 7.2 presents the communication model of this framework. It illustrates the bi-directional information flow: sensor data flow in one direction, and AmI context flow in opposite direction between the sensor nodes and AmI intelligent agents. The data structures for the bi-directional pub/sub communication are shown in Figure 7.3.

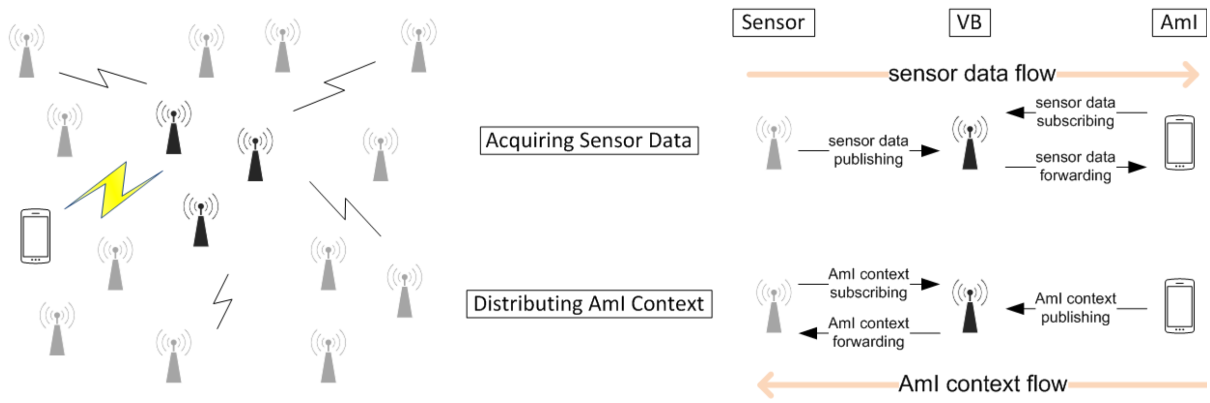


Figure 7.2 Communication model

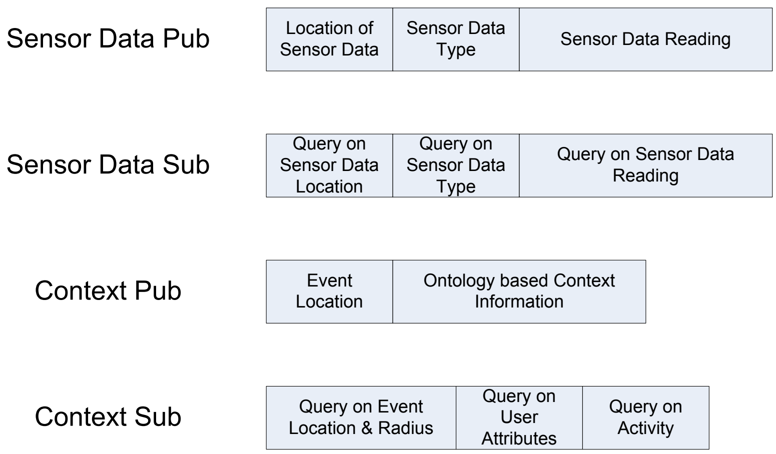


Figure 7.3 Data structures for bi-directional Pub/Sub

7.3.2 Node Architecture

Figure 7.4 presents the node architecture of the framework. It illustrates the node-level context handling and protocol adaptation for CL optimisation. This architecture has three hardware components, namely sensor element, transceiver, and storage device; and three software modules, namely Pub/Sub control, broker management, and context-aware CL optimisation.

The Pub/Sub control module allows a sensor node to perform the roles of both publisher and subscriber, i.e. not only can the sensor node publish its sensed data to the VB (for AmI agents to subscribe and generate higher-level user contexts), it can also subscribe to receive the high-level user contexts from the VB (published by AmI agents and stored on the VB). The broker management module is only used when the sensor node becomes a VB member node. It enables the sensor node to perform brokerage functions such as storing received published data and subscriptions, and forwarding matched published data to subscribers. Under this module, the data controller manages the storage of published sensor data and sensor data subscriptions, while the context storage control manages the storage of published context and context subscriptions.

The context-aware CL optimisation mechanism is the key constituent of this framework. Through context subscription, a sensor node can receive AmI context information, which is stored and later retrieved by the Context Manager for processing. The context information is modelled by an ontology. With some logic rules and a logical reasoning component, a sensor node can interpret the context and configure the protocol parameters for the desired performance.

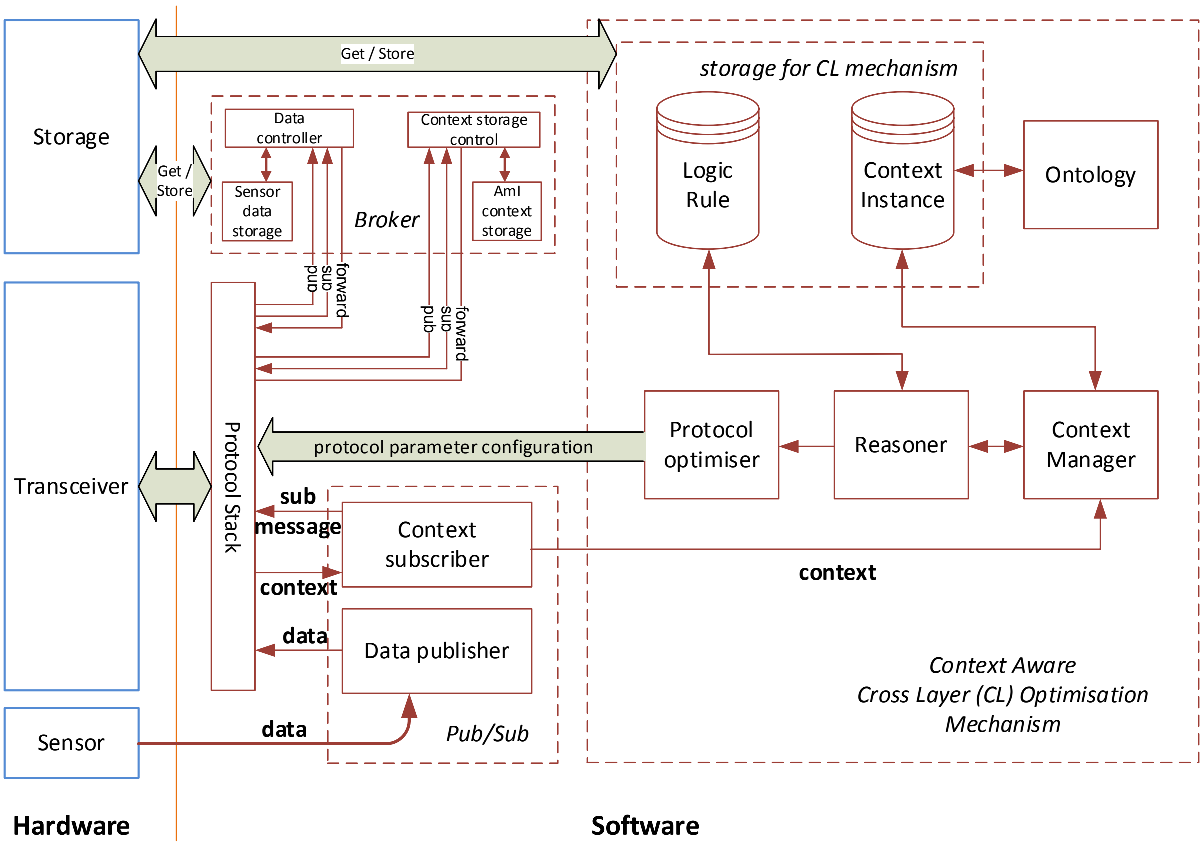


Figure 7.4 Node architecture

Both the Pub/Sub control and broker management models provide INS communication and data exchange abilities to the framework. By subscribing to AmI context, sensor nodes can obtain necessary high-level context about their users and environments. Through the Context Manager, the sensor nodes can understand the semantics of the received high-level contexts, from which actions can be performed by the protocol optimiser to achieve particular network optimisation goals.

7.3.3 Ontology-Based Context Modelling and Reasoning Scheme

Representing context information requires a modelling method to standardise and formalise information, through which a common understanding of the exchanged (global) AmI context by all sensor nodes can be achieved. In this chapter, the representation of common AmI context such as user location and activity is shown in Figure 7.5, which is based on an ontology model derived from the work of Chapter 6.

This model supports context representation by defining the ontology for context information in the scenario presented in Section 7.2. The ontology captures the basic characteristics of event attributes, personal attributes and user activity of a user-relevant event. The class *UserEvent* is the root entity. Each instance of *UserEvent* is represented for an event generated by an intelligent agent on behalf of a particular user. The class *UserEvent* has three descendant classes: *Event Attributes*, *Personal Attributes* and *Activity.* The *Event Attributes* class presents the facts of the event, which can include the location and radius of the event. The *Personal Attributes* class captures the user related facts, such as age and disability states. The *Activity* class encompasses physical activities of the user (e.g. walking and running). A representation of this Ontology can be described, for example using RDF/XML serialisation:

```
<owl:Class rdf:ID="UserEvent">
  <owl:Class rdf:ID="Event Attributes">
    <rdfs:subClassOf rdf:resource="#UserEvent"/ >
  </owl:Class> …
  <owl:Class rdf:ID="Personal Attributes">
    <rdfs:subClassOf rdf:resource="#UserEvent"/ >
  </owl:Class>
  <owl:ObjectProperty rdf:ID="age">
```

```
    <rdfs:domain rdf:resource="#Personal Attributes">
    <rdf:range rdf:resource="xsd:integer">
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="disability">
    <rdfs:domain rdf:resource="#Personal Attributes">
    <rdf:range rdf:resource="xsd:disabilitystate">
  </owl:ObjectProperty>
  <owl:Class rdf:ID="Activity">
    <rdfs:subClassOf rdf:resource="#UserEvent"/ >
  </owl:Class> …
</owl:Class>
```

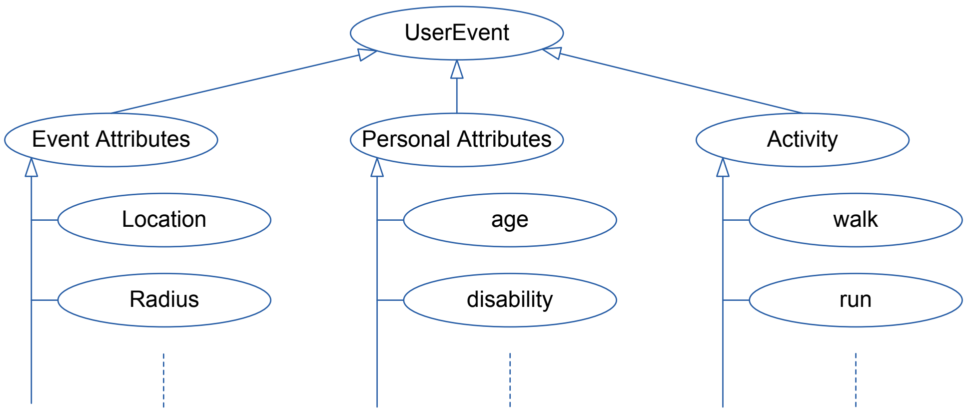


Figure 7.5 Ontology model

The context information of a user-relevant event can be represented by one instance of this ontology. It is important not to make this context model too complex as the sensor nodes are resource-constrained and do not have the capacity for complex processing. In this work, the following assumptions have been made:

- All location information is represented in the form defined by the Anchor-Free Localization (AFL) algorithm described in Chapter 4, i.e. in terms of hop counts to four corner reference nodes. A user's smart device is also capable of determining the AFL-based location of the events or users.

- To simplify the context design, personal attributes of a user are represented in terms of a high or low vulnerability state. For instance, both children and elderly are commonly considered as vulnerable individuals in AmI scenarios and therefore are represented with a high vulnerability state in the age-associated attribute. On the other hand, individuals of other ages are represented with a low vulnerability state in this attribute. Disability is another common personal attribute in AmI. People with certain disability types can be more vulnerable than those with other disabilities in a particular AmI scenario. For instance, people with vision or walking impairments are at greater risk in the outdoor scenario presented in Section 7.2. Therefore, they can be represented with a high vulnerability state in the disability-associated attribute, while those with other disabilities or otherwise healthy can be represented with a low vulnerability state in this attribute.

- Each sensor node has a simple ontology-based context module for performing simple context reasoning tasks. The objective of this reasoning is to determine importance of the sensor data to the context inference process of the intelligent agents.

A sensor node can use the following first-order logic expression to deduce its data priority context after receiving the event context information generated by the intelligent agents:

*(**Location**(Event, close to the sensor) ∨ **Radius**(Event, sensor within the event radius)) ∧ (**Age**(User, high vulnerability state) ∨ **Disability**(User, high vulnerability state)) ∧ **Activity**(Sensor data, required to infer the activity) ⊢ **SensorData**(Priority, High)*

This expression is used by each individual node to determine the priority context of its sensor data according to the attributes of the event (either the sensor node is located close to the occurrence of the event or the node is within the event radius), personal attributes of the user (age- and disability-associated vulnerability), and activity condition (the sensor data is needed to infer the activity context).

For the scenario presented in Section 7.2, a piece of sensor data is assigned to high priority when the following three characteristics are met:

- The sensor is close to or within the radius of an event;
- The age or disability attribute is in the high vulnerability state;
- The sensor data is used to infer the activity context.

7.4 Context Aware CL Optimisation on MAC and NET Layers

AmI systems require sensor data for context inference. The importance of a piece of sensor data, i.e. its usefulness to the current context inference process, can only be known by AmI. However, if sensor nodes can similarly know the importance of any sensor data at any given time through context exchange, the WSN may be optimised based on such knowledge. More specifically, situations such as the published sensor data on the VBs not matching with any data subscriptions from AmI while the required sensor data is delayed or lost due to network congestion, can be avoided. The key idea behind this context-aware CL optimisation approach is to prioritise WSN communications in AmI according to context information. Therefore, a sensor node that anticipates its data type will become important for AmI's current context inference process can assign its next data to be published with high priority, and reconfigure its protocol parameters accordingly.

In this work, two existing algorithms, Dynamic Reconfiguration MAC (DR-MAC) [11] protocol on the MAC layer, and Delay Aware AODV-Multipath (DAAM) [10] routing protocol on the NET layer, have been modified to incorporate the proposed context-aware CL optimisation framework presented in Section 7.3. The modified protocols are referred to as context-aware DR-MAC, and context-aware DAAM, respectively. Based on the discussions in Section 7.2, both context-aware protocols are based on their original mechanisms but modified to achieve better delivery performance for the high priority sensor data.

7.4.1 Context-Aware DR-MAC

The DR-MAC is a contention-based MAC protocol based on the unslotted CSMA/CA algorithm presented in Chapter 2. The original DR-MAC allows three state settings to control the number of backoffs and backoff exponential (BE) according to frame loss rate and latency, as shown in Figure 7.6. To incorporate AmI context information, DR-MAC is modified as shown in Figure 7.7.

The settings for transmitting high priority sensor data are chosen due to its having the lowest packet lost according to the results of original DR-MAC. It is important to ensure that important sensor data can arrive at its destination; otherwise it is impossible for the intelligent agents that subscribe to the sensor data to correctly perform any context inference processes. This packet loss improvement can also improve end-to-end frame latency as shown later in the evaluation section. The settings chosen for transmitting low priority sensor data is the default settings for the IEEE 802.15.4 standard.

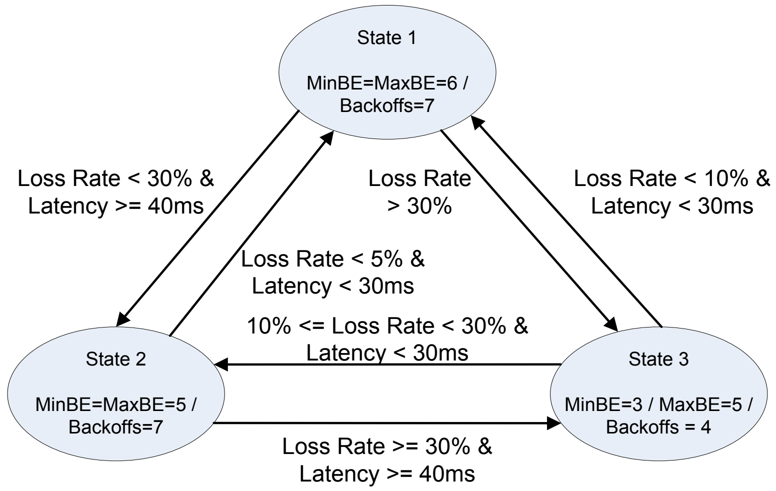


Figure 7.6 The original DR-MAC

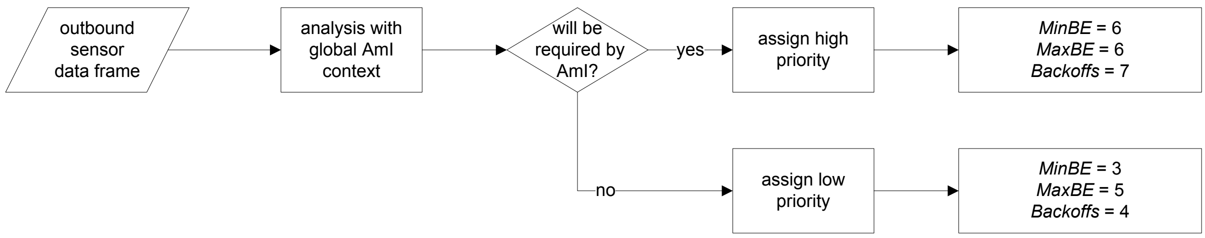


Figure 7.7 Context-aware DR-MAC

7.4.2 Context Aware DAAM

The original DAAM is a multi-path reactive routing protocol based on AODV described in Chapter 2. In DAAM, multiple node-disjoint paths can be discovered by a single route discovery procedure. In addition, DAAM modifies the original AODV routing algorithm by adding the delay information for each available path. The original AODV allows the nodes, after receiving a route reply (RREP) packet, to update their routing table entities according to the following rule:

if $((seq\_num^d_a < seq\_num^d_b)$ ***or*** $((seq\_num^d_a == seq\_num^d_b)$ ***and*** $(hop\_count^d_a > hop\_count^d_b))$
then
$seq\_num^d_a := seq\_num^d_b$
$hop\_count^d_a := hop\_count^d_b + 1$

$next\_hop^{d}_{a} := b$
endif

In AODV, when a node *a* receives a RREP packet from a one-hop neighbour node *b*, the node *a* only updates its path to the destination node *d* according to the destination sequence number ($seq\_num^{d}_{a}$ & $seq\_num^{d}_{b}$) and hop counts ($hop\_count^{d}_{a}$ & $hop\_count^{d}_{b}$) between node *a* and *b*. DAAM modifies AODV by keeping multiple node-disjoint paths for a pair of nodes. When routing the data packet for a particular application type, DAAM uses the following rule to allow the source node to select the best path for the packet transmission:

if *((($seq\_num^{d}_{a} == seq\_num^{d}_{b}$)* **and** *($route\_delay^{d}_{a} > route\_delay^{d}_{b}$)) or (($seq\_num^{d}_{a} < seq\_num^{d}_{b}$)* ***and*** *($route\_delay^{d}_{a} < request\_delay$)))*
then
$seq\_num^{d}_{a} := seq\_num^{d}_{b}$
$hop\_count^{d}_{a} := hop\_count^{d}_{b} + 1$
$next\_hop^{d}_{a} := b$
$route\_delay^{d}_{a} := route\_delay^{d}_{b}$
endif

where route delay ($route\_delay^{d}$) represents the delay to destination node *d*, and *request_delay* represents the delay requirement of an application data packet. In this chapter, DAAM has been modified to function according to the packet priority based on the user context information:

If *(outbound sensor data == high priority)*
then
send the packet through a low delay path
else
send the packet through a normal delay path
endif

In this setup, a low delay path is one whose path delay is less than a threshold. Otherwise, the path is classified as a "normal" path. This delay threshold should be application-specific and defined before the nodes are deployed.

7.5 Evaluation and Analysis

7.5.1 Simulation Parameters

This section evaluates the context-aware CL design proposed in Section 7.4. A WSN with 100 nodes distributed in a 10×10 grid topology over an area of 200×200 m^2 is simulated in OPNET. Each node has a transmission range of 20 m. A VB is formed in the centre of the network with 4 nodes. Unless specified otherwise, 25 non-VB member nodes ($N_{ami}=25$) are randomly selected to be AmI context publishers. The AmI context consists of *event attributes*, *personal attributes*, and *activity* (Figure 7.5) whose content is randomly generated during the simulation. 6 sensor data types are defined, along with 5 activities each requiring up to 3 sensor data types to be inferred. This context structure may describe a context such as '*a blind* (personalDisability) *elderly* (personalAge) *person is walking* (activity) *across the Queen Street* (eventLocation) *traffic junction area* (eventRadius)'. The sensor data type and content are randomly generated, and likewise for the AmI context. The data packet priority can only be determined by the data type and content used for inferring the generated AmI context. OPNET's uniform distribution function is used for the random generation process.

Each AmI context publisher subscribes to sensor data types needed to generate its context, which is then published at the rate of 1 frame for every 5 seconds with a frame size of 512 bits. The remaining are ordinary nodes that publish their sensor data to, and subscribe to receive AmI context from, the VB periodically. The VB forwards any matched sensor data to the AmI context publishers based on their subscription messages. The sensor data publishing settings: sensor data publishing interval (D_{freq}) and sensor data size (D_{size}) will be varied in this evaluation. Data rate is set to 250 kbps at 2.4 GHz. The current drawn in radio transmit and receive mode is set to 17.4 mA at 0 dBm, and 19.7 mA, respectively, based on MICAz's specification [155]. The AmI context publishers are mobile users who move according to a random waypoint model with a speed of 1.2 m/s and pause time of 3.6 s [179, 180].

Unless specified otherwise, IEEE 802.15.4 unslotted CSMA/CA and AODV are the default MAC protocol, and network routing protocol, used respectively, during the simulations. Delay threshold for context-aware DAAM is set to 1 second, which is the best setting obtained from several test runs. All results are the average of 10 runs over 180 seconds.

7.5.2 Performance Metrics

- *Delay*: the average time for a frame/packet containing sensor data published by a sensor node to arrive at the VB.
- *Throughput*: network capacity in bits per second based on published sensor data that successfully arrived at the VB.
- *Energy cost per delivered frame/packet*: ratio of total energy consumed by all nodes to the number of successfully delivered published sensor data frames to the VB.
- *Packet delivery ratio (PDR)*: ratio of the number of packets received by the VB to the total data packets sent by all sensor nodes.
- *Communication overhead*: number of control frames/packets sent during the sensor data publishing from the sensor nodes to the VB member nodes.

7.5.3 Simulation Results

7.5.3.1 MAC Layer Results

This section evaluates the performance of three MAC protocols, DR-MAC, context-aware DR-MAC and IEEE 802.15.4 unslotted CSMA/CA, with AODV as the common network routing protocol. The results are shown for two settings: normal traffic and high traffic. The traffic parameters are summarised in Table 7-1.

Table 7-1 Traffic parameter settings

Scenario/Parameter	Description/Value
Scenario 1	Normal traffic
D_{freq}	2 frames per second
D_{size}	512 bits
Scenario 2	High traffic
D_{freq}	10 frames per second
D_{size}	1024 bits

Figure 7.8 shows the throughput results of the three MAC protocols. It is observed that the context-aware DR-MAC can achieve 22% improvements to the original DR-MAC and 36% improvements to CSMA/CA under the normal traffic scenario. For high traffic scenario, the improvements are 22%, and 64%, respectively. The multiple backoff settings of the DR-MAC based protocols improve the frames' delivery success rate, which in turn improves their overall

throughput over that of CSMA/CA. In addition, by utilising the AmI context information, the sensor nodes can control the frame transmissions based on their priority, and thus can further improve the overall throughput.

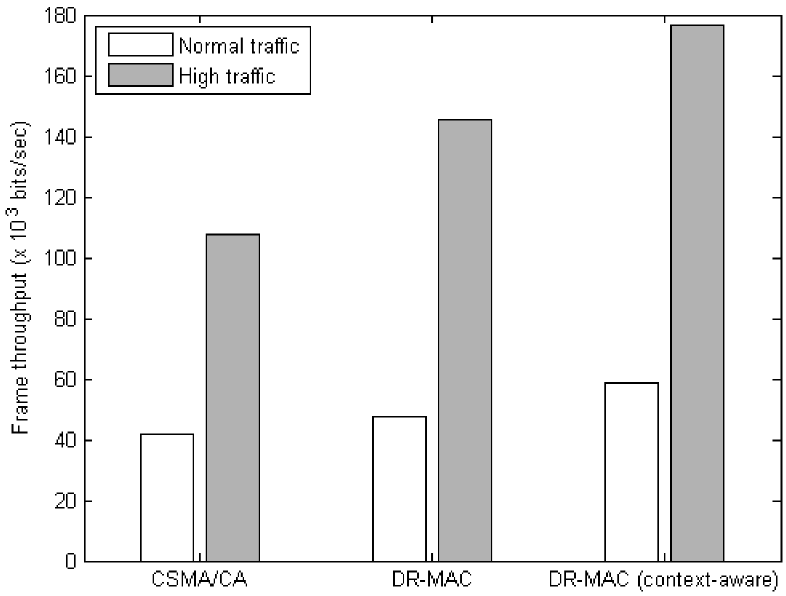


Figure 7.8 Frame throughput of the MAC protocols

Figure 7.9 compares the frame delay among the three protocols. The context-aware DR-MAC exhibits the lowest frame delay in both normal and high traffic scenarios. The frame delay is reduced by 38%, and 30%, over CSMA/CA, and original DR-MAC, respectively, under normal traffic scenario, while it is reduced by 45%, and 28%, respectively, under high traffic scenario. The DR-MAC based protocols may increase frame delay at the MAC layer for a pair of neighbour nodes. However, it could reduce the overall end-to-end frame delay. By improving the ratio of the frames being delivered between a pair of neighbour nodes, less route error packets containing the link breakage messages are issued by the routing protocol of the relay nodes. Therefore, it can reduce the need of a source node to rediscover a path and retransmit the data packet to the destination node. In turn, this can reduce the overall frame delay.

Figure 7.10 shows the average energy cost to successfully deliver a frame to the VB. Under normal traffic scenario, the context-aware DR-MAC can achieve 25%, and 10%, improvement over CSMA/CA, and the original DR-MAC, respectively. In high traffic scenario, the improvement is 35%, and 18%, respectively. By using the context information to further enhance the delivery success of the data frames, the context-aware DR-MAC incurred less energy for transmissions associated with path rediscovery and packet retransmission. Therefore, it is more energy-efficient as compared to the original DR-MAC.

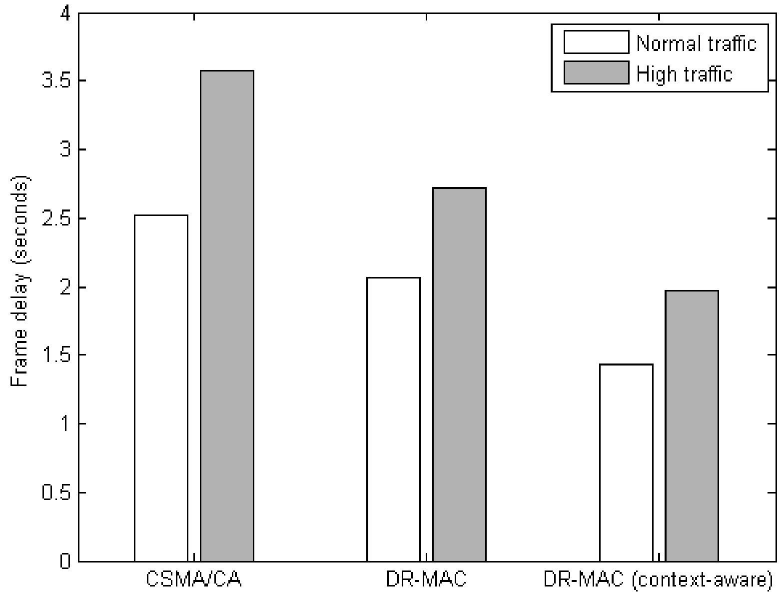


Figure 7.9 Frame delay of the MAC protocols

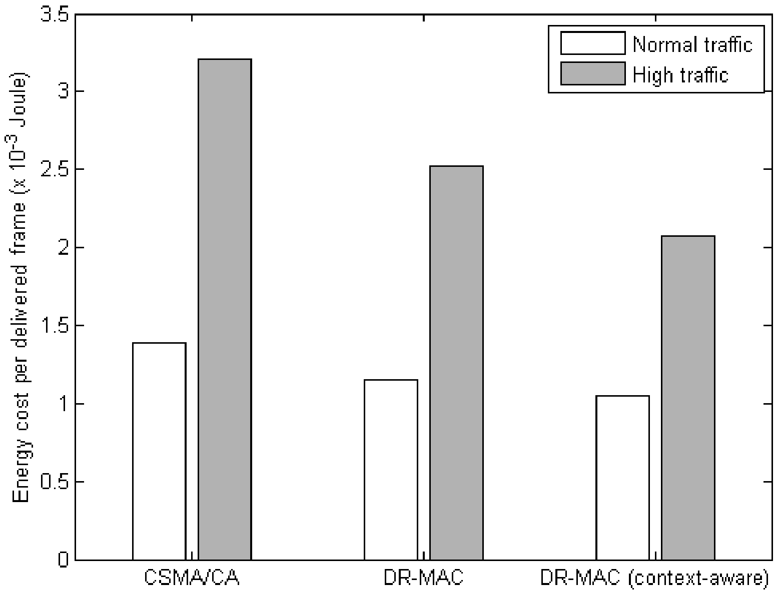


Figure 7.10 Energy cost for a successful frame delivery by the MAC protocols

7.5.3.2 NET Layer Results

This section evaluates the performance of three network routing protocols: DAAM, context-aware DAAM and AODV, with IEEE 802.15.4 unslotted CSMA/CA as the common MAC protocol. Similarly, the results are shown for two settings: normal traffic and high traffic with the settings summarised in Table 7-1.

Figure 7.11 shows the PDR of the three network routing protocols. It is observed that the context-aware DAAM can achieve 26%, and 13%, improvement to AODV and original DAAM, respectively, under normal traffic setting. The improvement in PDR under high traffic setting is 44%, and 18%, respectively. In DAAM, multiple node-disjoint paths can increase the success of the packets being delivered to the VB since any of the VB member nodes can accept the incoming sensor data packets. This avoids losses due to congestion when only a single node, e.g. a cluster head, is the destination for all sensor data packets. In addition, the context-aware DAAM makes the sensor data packets traversed through paths according to their data priority; this can further improve the packets delivery to the VB.

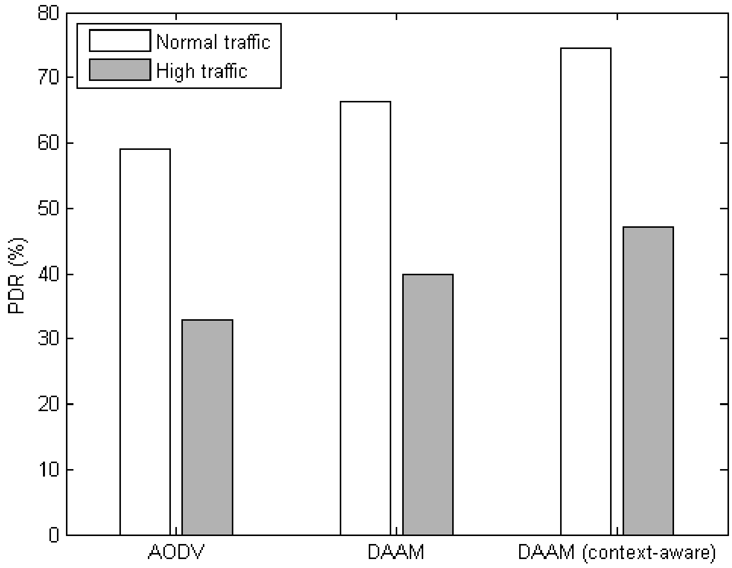


Figure 7.11 PDR of the NET protocols

Figure 7.12 compares the end-to-end delay among the three protocols. Under context-aware DAAM, the delay is reduced by 38%, and 32%, over AODV, and original DAAM, respectively, under normal traffic scenario, while it is reduced by 45%, and 35%, respectively, under high traffic scenario. The path diversity of DAAM significantly improves the delay performance of

AODV. By further combining with AmI context information, the delay can be further reduced as low latency paths are used to transmit the high priority sensor data.

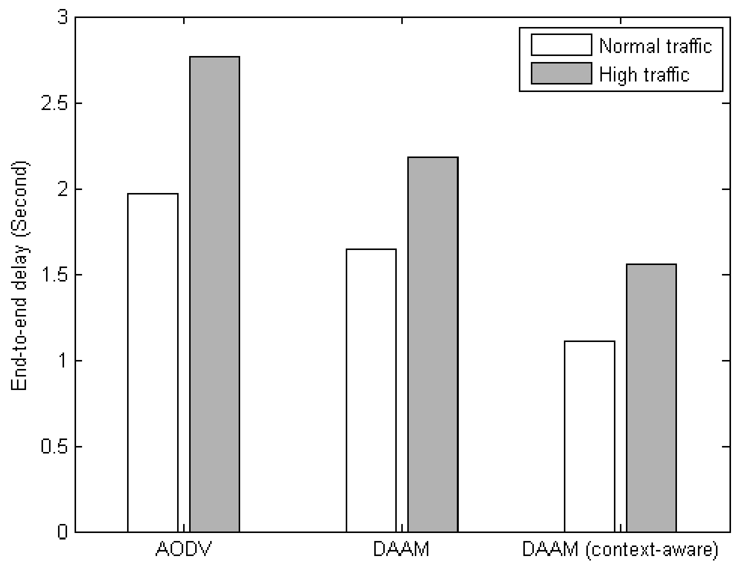


Figure 7.12 End-to-end delay of the NET protocols

Figure 7.13 shows the corresponding energy cost. It illustrates significant energy savings under high traffic scenario where the context-aware DAAM reduces the energy cost by 24%, and 15%, over AODV, and original DAAM, respectively. Compared to AODV, which only uses a single path to transmit all the data between a pair of nodes, more available paths from the DAAM based protocols allow the data to be delivered to any member nodes of a VB. This increases the PDR, as shown in Figure 7.11, which in turn improves the energy efficiency.

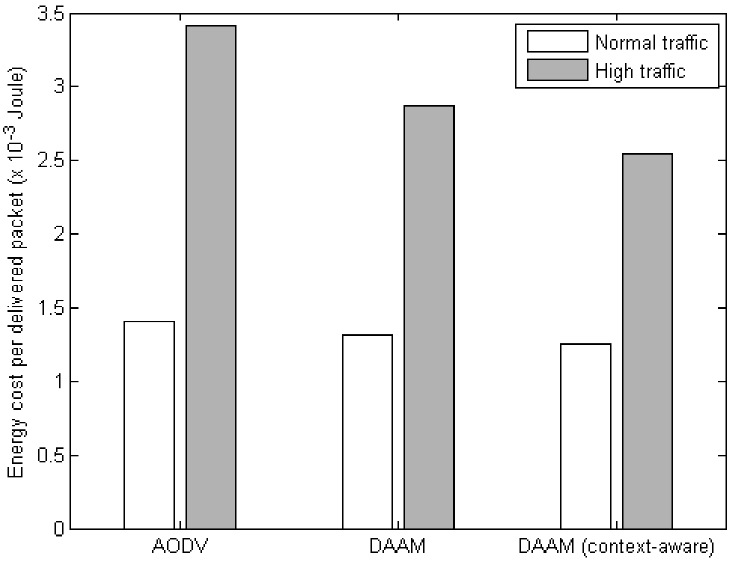


Figure 7.13 Energy cost for a successful packet delivery of the NET protocols

7.5.3.3 Results for the Different Protocol Sets

This section evaluates different protocol sets based on different combinations of the MAC and NET layers protocols. A total of 9 protocol sets are defined as shown in Table 7-2. All sets are evaluated under normal traffic scenario with parameters as outlined in Table 7-1.

Table 7-2 Protocol sets

Set	NET layer protocol	MAC layer protocol
1	AODV	CSMA/CA
2	AODV	DR-MAC
3	AODV	DR-MAC(context-aware)
4	DAAM	CSMA/CA
5	DAAM	DR-MAC
6	DAAM	DR-MAC(context-aware)
7	DAAM(context-aware)	CSMA/CA
8	DAAM(context-aware)	DR-MAC
9	DAAM(context-aware)	DR-MAC(context-aware)

Figure 7.14 shows the throughput of all the protocol sets. The largest performance differential among the protocol sets is a 64% increase in throughput by Set 9 over Set 1. Generally, it is observed that the throughput can be improved by replacing CSMA/CA with a DR-MAC based protocol. For instance, the throughput of Set 3 is improved by 36% over Set 1, 27% for Set 6 over Set 4, and 11% for Set 9 over Set 7. The less significant improvement for Set 9 over Set 7

could be due to that the context-aware DAAM has enhanced the delivery success of the sensor data packets, which leaves less room for further improvement by the context-aware DR-MAC.

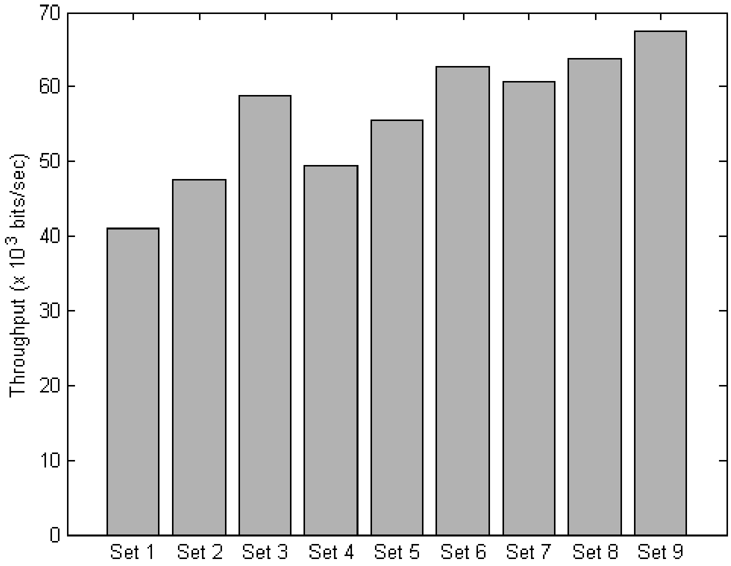


Figure 7.14 Throughput of the protocol sets

Figure 7.15 shows the PDR of all the protocol sets. It can be seen that the protocol sets with context-aware DAAM can achieve better PDR than the original DAAM under any MAC protocols, e.g. an improvement of 18% for Set 7 over Set 4, 14% for Set 8 over Set 5, and 11% for Set 9 over Set 6. There is a 61%, and 25%, improvement for Set 9 over Set 1, and Set 5, respectively. This result shows PDR can be improved by using AmI context information.

Figure 7.16 presents compares the end-to-end delay among the protocol sets. The result shows that the delay of Set 9 is reduced by 70%, and 46%, over Set 1, and Set 5, respectively. By replacing the non-context aware MAC with a context-aware version while keeping the same routing protocol, the delay can be reduced by 30% for Set 3 over Set 2, 29% for Set 6 over Set 5, and 22% for Set 9 over Set 8. By replacing the non-context aware routing with a context-aware version while keeping the same MAC protocol, the delay can be reduced by 26% for Set 7 over Set 4, 24% for Set 8 over Set 5, and 20% for Set 9 over Set 6. These results indicate that the delay can be reduced when exploiting the context information on the MAC and network layers.

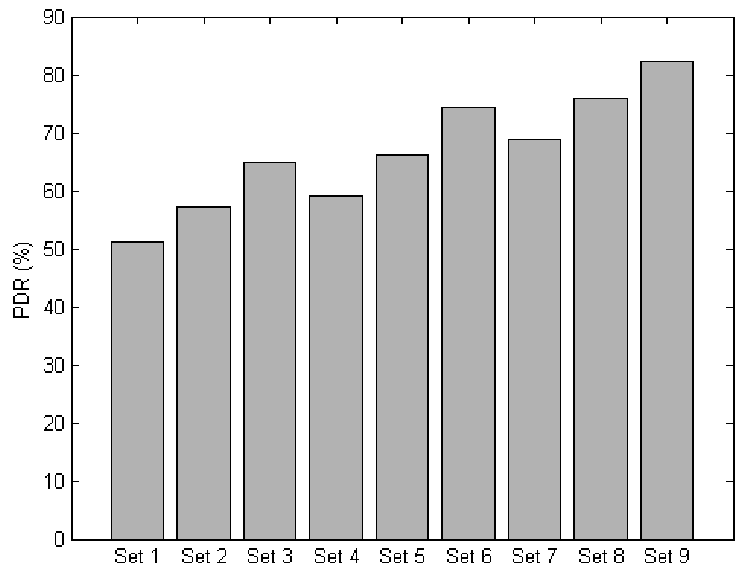


Figure 7.15 PDR of the protocol sets

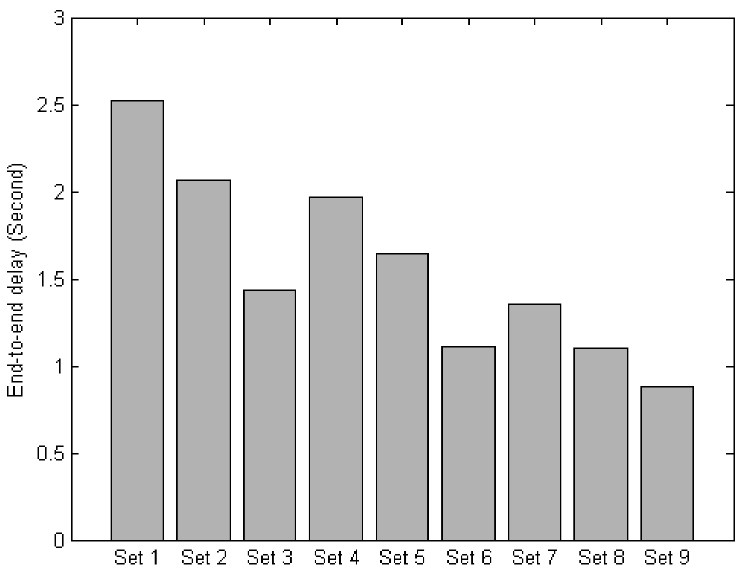


Figure 7.16 End-to-end delay of the protocol sets

Figure 7.17 shows the communication overhead in terms of the number of control frames (MAC layer ACK frames) and control packets (routing-related control packets). The result shows that the protocol sets with DAAM-based routing protocol have more overhead than those with AODV. This is because unlike AODV, a route request (RREQ) packet cannot be discarded by any relay nodes during the path discovery phase in DAAM. Therefore, there are more RREQ packets being forwarded. In addition, the destination nodes have to reply to every

RREQ packet received, which results in more route reply (RREP) packets as well. The DAAM-based protocol sets generate a similar amount of control packets. This is because with multiple available paths, the source node can always select another path, which can satisfy the priority requirement of the packet, to the destination node when the previous transmission fails. The route discovery procedure is performed and the associated routing overhead incurred only when none of the existing paths can be utilised for the transmission. In the simulations, the ACK frames are required for every frame transmission between a pair of nodes, where either a data or control packet from the NET layer is encapsulated into a frame. Therefore, the number of ACK frames is significantly higher than the total routing control packets.

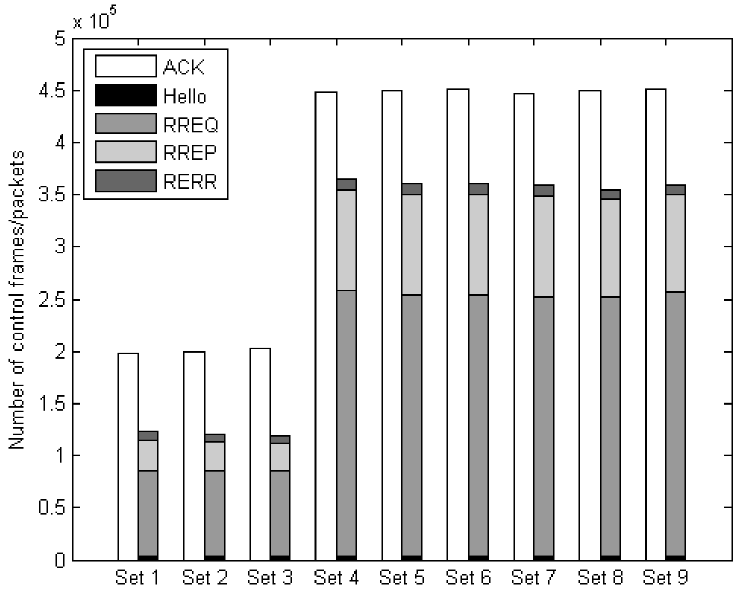


Figure 7.17 Communication overhead of the protocol sets

Figure 7.18 presents the average energy cost to successfully deliver a sensor data packet. In general, the protocol sets with DAAM-based routing cost more energy than those with AODV. This is due to the higher amount of energy used for transmitting control frames/packets. However, this higher energy use is partially offset by a larger number of sensor data packets delivered, resulting in only a moderate rise in the energy cost per packet delivered as compared to the protocol sets with AODV. The protocol sets with context-awareness are seen to achieve better energy efficiency. For instance, comparing Sets (3 and 2; 6 and 5; and 9 and 8) and Sets (7 and 4; 8 and 5; and 9 and 6) show that the energy cost per packet is reduced with context-awareness incorporated into the MAC, and network routing protocol, respectively.

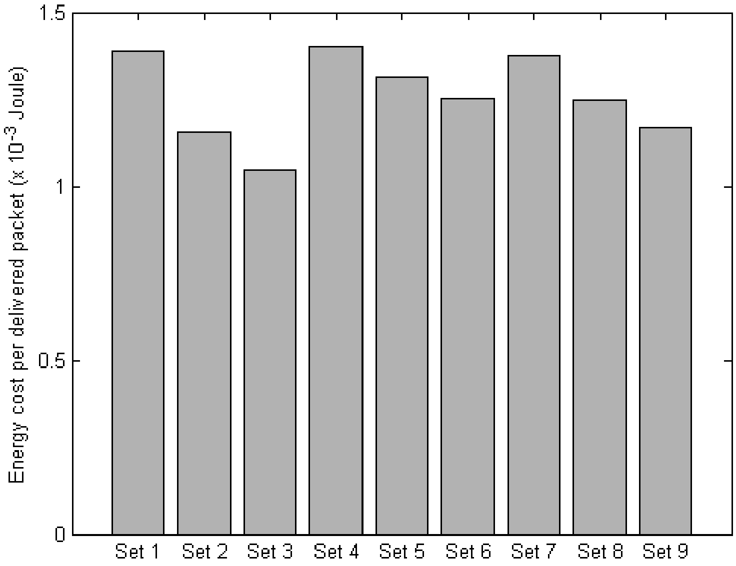


Figure 7.18 Energy cost for a successful packet delivery of the protocol sets

Table 7-3 presents the overall rankings for all the protocol sets based on the performance metrics. The protocol set with context-aware DR-MAC and context-aware DAAM (Set 9) shows the best overall result by ranking first in throughput, PDR, and delay; second in control overhead (only two ranks in this metric); and third in energy efficiency.

Table 7-3 Performance ranking

Performance metric	1st	2nd	3rd	4th	5th	6th	7th	8th	9th
Throughput	Set 9	Set 8	Set 6	Set 7	Set 3	Set 5	Set 4	Set 2	Set 1
PDR	Set 9	Set 8	Set 6	Set 7	Set 5	Set 3	Set 4	Set 2	Set 1
End-to-end delay	Set 9	Set 6	Set 8	Set 7	Set 3	Set 5	Set 4	Set 2	Set 1
Control frames/packets	Set 3; Set 2; Set 1			Set 9; Set 6; Set 8; Set 7; Set 5; Set 4					
Energy efficiency	Set 3	Set 2	Set 9	Set 8	Set 6	Set 5	Set 7	Set 1	Set 4

7.5.3.4 Parameter Effects on Context-Aware Protocol Set

This section evaluates the parameter effects of the context-aware DR-MAC and context-aware DAAM protocol set (Set 9) under different traffic settings. In this section, the number of AmI user nodes (N_{ami}), sensor data publishing frequency (D_{freq}), and sensor data packet size (D_{size}) are varied, as shown in Table 7-4.

Table 7-4 Parameter settings of the protocol set 9

Parameter	Value
N_{ami}	25, 50 nodes
D_{freq}	0.5, 1, 2, 5, 10 packets per second
D_{size}	512, 1024 bits

Figure 7.19 presents the throughput results. It shows that when D_{freq} increases, the throughput expectedly increases, as more bits are being published by sensors and delivered to VB within a given time. Similarly, when the packet size (D_{size}) doubles, the throughput increases, but only by approximately 50%. This could be due to some congestion-related packet losses in the network, but the loss is not significant enough to reduce the throughput. However, when N_{ami} increases from 25 to 50 nodes while keeping the packet size constant, throughput decreases despite an increase in the amount of sensor data forwarded to the AmI nodes, i.e. intelligent agents. This indicates that a serious congestion has occurred, and the network is more sensitive to an increase in the number of AmI nodes than an increase in the packet size.

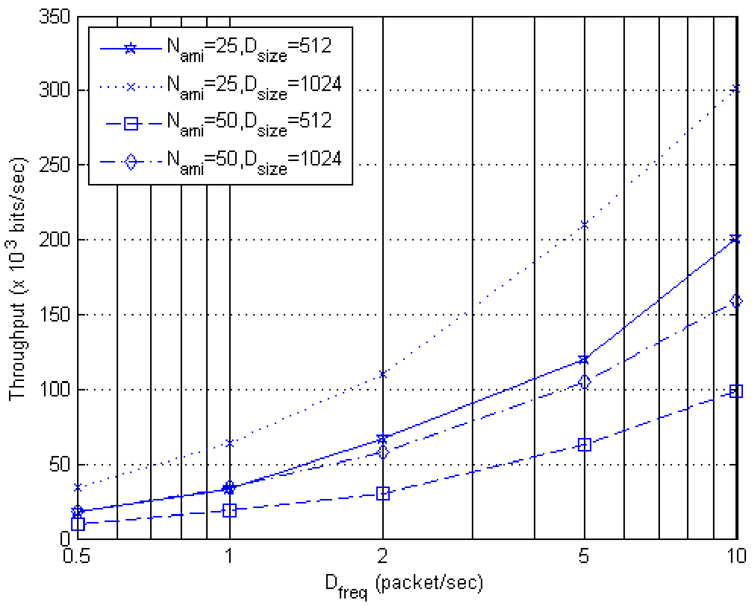


Figure 7.19 Throughput under different parameter settings

Figure 7.20 shows the PDR results. It is observed that reasonable PDR, i.e. > 50% can be achieved when the sensor data publishing frequency is less than or equal to 2 packets per second. However, at higher publishing frequency, PDR decreases to below 50% for N_{ami}=50. Similarly, doubling packet size from 512 to 1024 bits decreases the PDR. However, the PDR is not decreased proportionally by half, but up to 13%, and 20%, for N_{ami}=25, and N_{ami}=50, respectively. This may explain why the throughput still increases in Figure 7.19 when the packet size increases for a given number of AmI nodes.

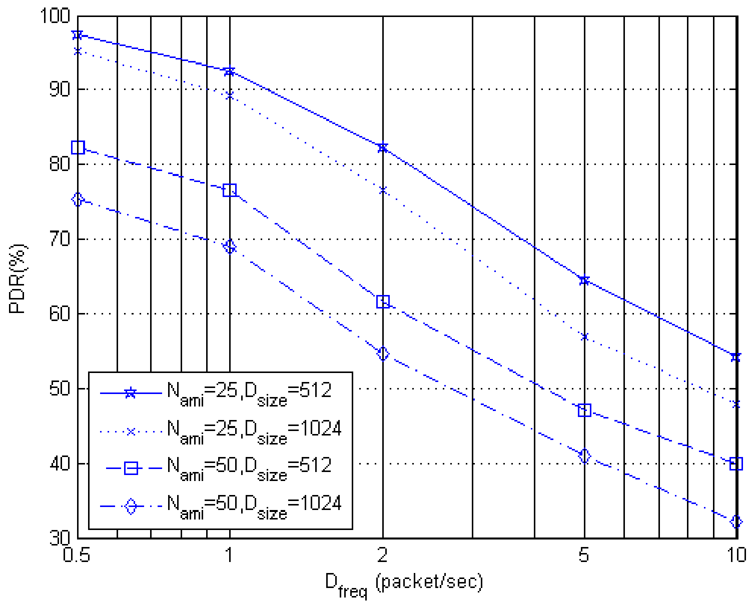


Figure 7.20 PDR under different parameter settings

Figure 7.21 shows the end-to-end delay results, which increase with the publishing frequency. As D_{freq} increases from 0.5 to 10 packets per second, the delay can increase by up to 5.6 times for N_{ami}=25, D_{size}=512, 6.1 times for N_{ami}=25, D_{size}=1024, 5.1 times for N_{ami}=50, D_{size}=512, and 3.7 times for N_{ami}=50, D_{size}=1024. Expectedly, the delay performances of the 4 settings are ordered according to the amount of sensor data bits transmitted in each setting, with the lowest delay, and highest delay, incurred by the setting N_{ami}=25, D_{size}=512, and N_{ami}=50, D_{size}=1024, respectively.

Figure 7.22 shows the energy cost per packet delivered increases as D_{freq} increases. This is because more energy is expended to transmit an increasing amount of sensor data, while less of these data can be delivered due to increasing network congestion. The result also shows that doubling the packet size from 512 to 1024 bits has a greater detrimental impact on the energy efficiency, i.e. higher energy cost per packet, than doubling the number of AmI users from 25 to 50 nodes. This may be due to more packet reception errors and subsequently more retransmissions when long packets are used.

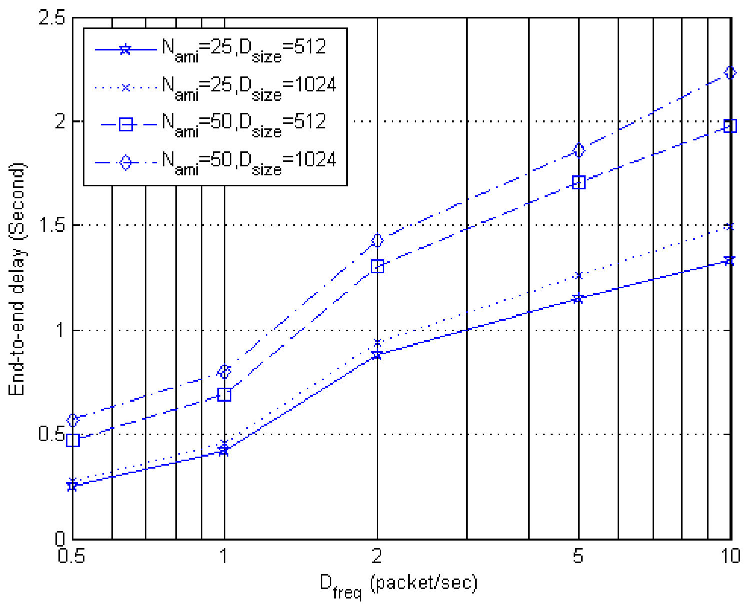


Figure 7.21 End-to-end delay under different settings

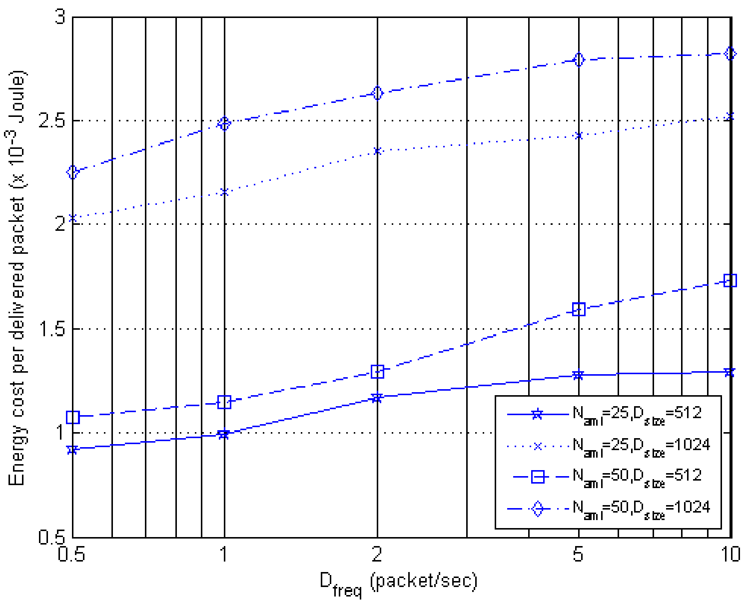


Figure 7.22 Energy Cost for a successful packet delivery under different parameter settings

7.5.3.5 Comparison between Context-Aware and non-Context Aware DR-MAC and DAAM Protocol Sets

This section evaluates the performance of the context-aware DR-MAC/DAAM protocol set (Set 9), and the original DR-MAC/DAAM protocol set without context awareness (Set 5). The results are based on the simulations of three scenarios: light, normal, and high traffic, with the parameters summarised in Table 7-5.

Table 7-5 Parameter settings for the context-aware and non-context aware DR-MAC/DAAM protocol sets

Scenario	Parameter Values	
	D_{freq}	D_{size}
Light traffic	1 frame every 2 second	512 bits
Normal traffic	2 frames per second	512 bits
High traffic	10 frames per second	1024 bits

Figure 7.23 - Figure 7.26 presents the results of the two protocol sets in terms of throughput, PDR, end-to-end delay, and energy cost per packet delivered. Clearly, context awareness can enhance the protocol performances particularly in the high traffic scenario. The protocol set 9 can achieve up to 74%, 68%, 46%, and 14% improvement over protocol set 5 in terms of throughput, PDR, end-to-end delay, and energy cost, respectively.

The improvement can be attributed to the AmI context information which is utilised: 1) for adapting the backoff behaviour of the MAC protocol to enhance the success of frame delivery between neighbouring node pairs; and 2) for prioritising packets and selecting data paths with delays corresponding to the packet priority by the network routing protocol. This results in improved throughput, PDR, and end-to-end delay. With more sensor data packets delivered, the energy efficiency is also improved, i.e. lower energy cost per packet delivered.

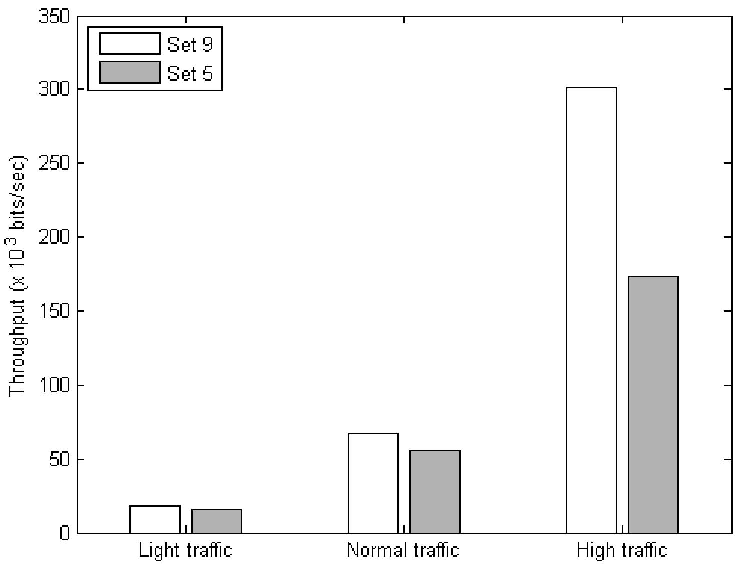


Figure 7.23 Throughput of the context-aware and non-context aware protocol sets

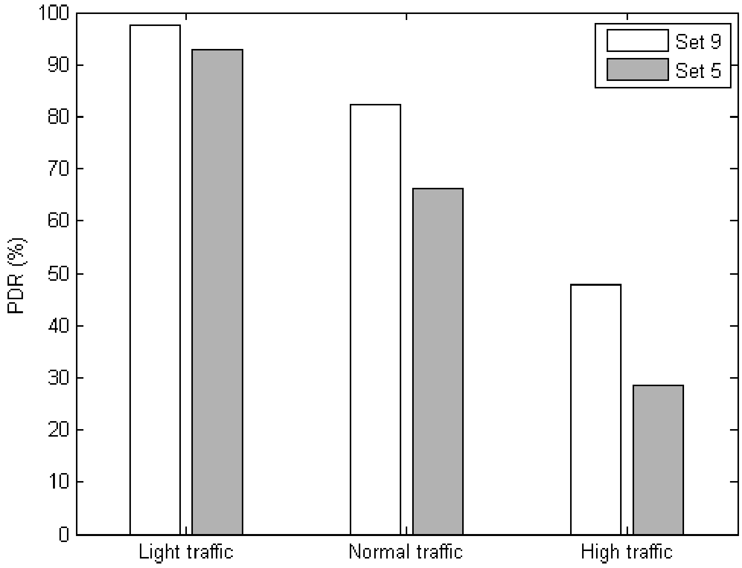


Figure 7.24 PDR of the context-aware and non-context aware protocol sets

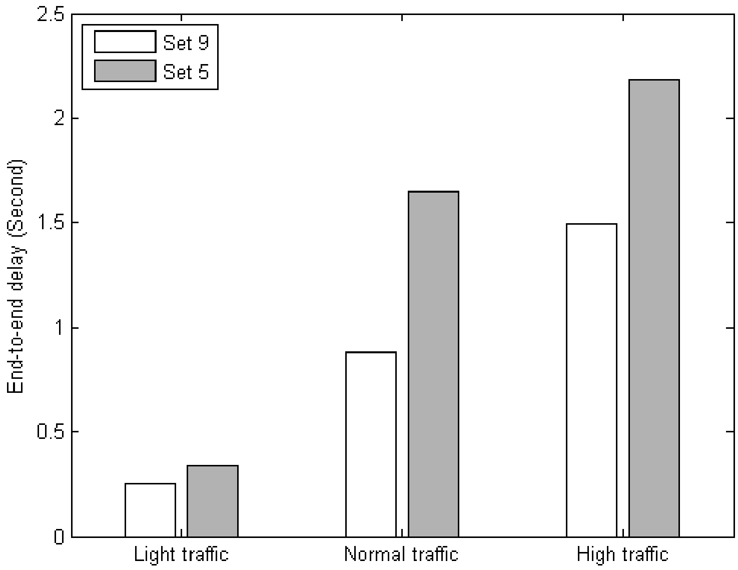


Figure 7.25 End-to-end delay of the context-aware and non-context aware protocol sets

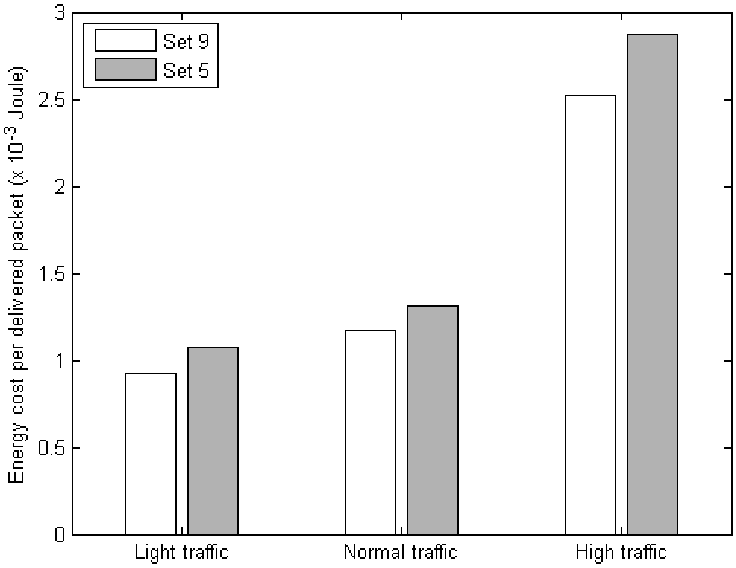


Figure 7.26 Energy cost for a successful packet delivery of the context-aware and non-context aware protocol sets

7.6 Conclusion

In this chapter, a generic cross-layer (CL) protocol optimisation framework based on the ambient intelligence (AmI) context information from the application layer, in conjunction with an ontology-based context modelling and reasoning mechanism, has been proposed. This context-aware cross-layer scheme provides wireless sensor network (WSN) nodes with the ability to gather the ambient intelligence context for the purpose of cross-layer optimisation. The framework is then implemented by two protocols on the medium access control (MAC) and network (NET) layers for joint protocol optimisations. The backoff behaviour of the medium access control protocol and path selection of the network routing protocol were modified in response to the ambient intelligence context information. It is shown that the resulting optimisation through context awareness and cross-layer interaction can yield substantial improvements in terms of throughput, packet delivery ratio (PDR), delay, and energy efficiency.

Chapter 8 Conclusion

This thesis focused on the investigation of a context-aware cross-layer optimisation approach in wireless sensor networks by using ambient intelligence context information via cross-layer interaction. The first three chapters presented the introduction, background, and literature review. The fourth chapter proposed a network structuring and in-network virtual storage formation algorithm for wireless sensor networks. This algorithm firstly structured the network into balanced partitions, and then established a virtual storage unit in each partition to provide a platform for maintaining the data from both the sensor nodes and ambient intelligence user devices in the network. Chapter Five presented the communication mechanism for storing data to and retrieving data from the virtual storage based on the publish/subscribe model. This mechanism creates the ability for sensor nodes and ambient intelligence user devices to share their sensor data, and ambient intelligence context information, respectively. An ontology-based context modelling and reasoning approach was proposed in Chapter Six. This approach allows both application and network related information to be represented as contexts, which can be shared, understood, and utilised for context-aware wireless sensor network management. Chapter Seven presented an ambient intelligence context-aware cross-layer design approach for wireless sensor networks. A generic cross layer framework for protocol optimisation in wireless sensor networks was introduced. This framework was then applied to two existing protocols on the medium access control (MAC) and network layers for joint protocol optimisation by utilising the ambient intelligence context information.

8.1 Summary of Contributions

The contributions of this thesis are summarised as follows:

- A network partitioning and in-network storage formation algorithm for data-centric communications in wireless sensor networks was proposed. The concept of virtual storage unit (VSU) was introduced, which enables a group of co-located sensor nodes to share their resources for providing data storage and retrieval functions. An analytical model was derived to obtain the optimal number of partitions and virtual storage unit size that jointly minimise the total communication overhead in the wireless sensor network.

- A data-centric communication mechanism based on the published/subscribe (pub/sub) model was designed. Virtual brokers (VBs), which are based on the concept of virtual storage unit, are used. Therefore, no dedicated special hardware are required for the brokers, and the scalability and robustness of the pub/sub can be improved, i.e. the size of the virtual brokers can be varied to suit different storage demands and virtual brokers can be formed from different sensor nodes to avoid issues such as single point of failure. In addition, a virtual broker storage scheme with dynamic load balancing is presented.

- An ontology-based context modelling approach was proposed for wireless sensor network management. Unlike many existing designs for context-aware wireless sensor network applications, this approach can model context information from both application level, i.e. ambient intelligence systems, and network level, for network management tasks. Therefore, the network mechanisms can become more aware of the context of their surrounding environments. This model was illustrated to be easily adapted by existing algorithms to respond to context information.

- A generic context-aware cross-layer framework was proposed for cross-layer protocol optimisations in wireless sensor networks using ambient intelligence context information. This framework covers the design of three core components: node architecture, which describes the functionalities for cross-layer interaction within a sensor node; communication mechanism, which defines the processes for data exchange; and an ontology context model derived for a typical ambient intelligence scenario. In addition, this framework has been applied to adapt the backoff behaviour of a MAC protocol and the path selection of a network routing protocol according to ambient intelligence context information.

8.2 Limitations of Current Work

- Ambient intelligence is still a rapidly evolving technology, with increasing number of different use cases. Although Chapter 7 provided a use case that covers the most common context categories in today's ambient intelligence scenarios, it is expected that further design of the context ontology is necessary in order to accommodate new and emerging ambient intelligence contexts and use cases.

- Another limitation is that only stationary sensor nodes are considered, although AmI users in this research could be mobile. In particular, new algorithms for virtual broker formation and storage may be needed if mobile sensor nodes are used.
- Due to both time and resource constraints, the research undertaken in this thesis is mainly analytical and simulation-based, although realistic settings are used where feasible. Real implementation may be possible when AmI technologies become more mature and practical.

8.3 Future Work

Most of the research work presented in this thesis is focused on the forthcoming challenges of cross-layer optimisation for wireless sensor networks in context-aware systems, e.g. ambient intelligence systems. The following discusses some possible future work on related topics presented in this thesis.

- The Internet of Things (IoT) is a paradigm where a massive number of networked devices can communicate over the Internet. Beyond the scope of wireless sensor networks, which mainly consists wireless sensor devices, the internet of things can encompass any devices (e.g. smart phones, smart appliances, etc.) interconnected through wired and/or wireless communications. As the internet of things is being applied to realise the vision of ambient intelligence, there could be a further work on the investigation of adapting the context-aware cross-layer approach for optimising communications for the internet of things in ambient intelligence environments.
- Wireless sensor and actuator networks (WSANs) offer not only the functionality of remote sensing but also the actuation of any controllable devices. This can greatly enhance AmI systems as more intelligent actions can be undertaken. Another possible future work is to extend the ontology model by modelling the context information related to the actuators for future context-aware wireless sensor and actuator network -based ambient intelligence systems.

- As an emerging trend in wireless sensor network research, sensor devices can now be integrated with energy harvesting capability. Hence, when energy of a sensor is no longer a constraint in wireless communications, the focus of protocol optimisations can be shifted from energy efficiency to communication performance (e.g. throughput, delay, etc).
- Recent technology developments have increased the hardware capacity of wireless sensor network nodes, e.g. more processing and storage capacity. This may eliminate the current need for ambient intelligence users to have a smart device for processing sensor data and inferring context information. Therefore, future ambient intelligence systems can be designed without this requirement and the issue of in-network cooperative context inference on sensor devices could be investigated.

List of References

[1] L. Benini, E. Farella, and C. Guiducci, “Wireless sensor networks: Enabling technology for ambient intelligence,” *Microelectronics Journal*, vol. 37, no. 12, pp. 1639–1649, 2006.

[2] T. Anker, D. Dolev, and B. Hod, “Belief Propagation in Wireless Sensor Networks - A Practical Approach,” in *Wireless Algorithms, Systems, and Applications*, Springer Berlin Heidelberg, 2008, pp. 466–479.

[3] A.-H. Jallad and T. Vladimirova, “Data-Centricity in Wireless Sensor Networks,” in *Guide to Wireless Sensor Networks*, Springer, 2009, pp. 183–204.

[4] D. Ganesan, S. Ratnasamy, H. Wang, and D. Estrin, “Coping with irregular spatio-temporal sampling in sensor networks,” *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 1, pp. 125–130, 2004.

[5] L. Van Hoesel, T. Nieberg, J. Wu, and P. J. Havinga, “Prolonging the lifetime of wireless sensor networks by cross-layer interaction,” *IEEE Wireless Communications,* vol. 11, no. 6, pp. 78–86, 2004.

[6] T. Melodia, M. C. Vuran, and D. Pompili, “The state of the art in cross-layer design for wireless sensor networks,” in *Wireless Systems and Network Architectures in Next Generation Internet*, Springer, 2006, pp. 78–92.

[7] A. Kofod-Petersen, “Challenges in case-based reasoning for context awareness in ambient intelligent systems,” in *8th European Conference on Case-Based Reasoning, Workshop Proceedings, Ölüdeniz, Fethiye, Turkey*, 2006, pp. 287–299.

[8] S. K. Jayaweera, “Virtual MIMO-based cooperative communication for energy-constrained wireless sensor networks,” *IEEE Transactions on Wireless Communications,* vol. 5, no. 5, pp. 984–989, 2006.

[9] L. Shu, Y. Zhang, Z. Yu, L. T. Yang, M. Hauswirth, and N. Xiong, “Context-aware cross-layer optimized video streaming in wireless multimedia sensor networks,” *J Supercomput*, vol. 54, no. 1, pp. 94–121, Oct. 2010.

[10] J. N. Boshoff and A. S. J. Helberg, “Improving QoS for real-time multimedia traffic in Ad-hoc Networks with delay aware multi-path routing,” in *WTS'2008*, Pomona, CA, 2008, pp. 1–8.

[11] D. Rohm, M. Goyal, W. Xie, B. Polepalli, H. Hosseini, A. Divjak, and Y. Bashir, “Dynamic Reconfiguration in Beaconless IEEE 802.15.4 Networks under Varying Traffic Loads,” in *IEEE GLOBECOM'09*, Honolulu, HI, USA, 2009, pp. 1–8.

[12] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller, “Anchor-free Distributed Localization in Sensor Networks,” in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, New York, NY, USA, 2003, pp. 340–341.

[13] D. Geer, “Users make a Beeline for ZigBee sensor technology,” *Computer*, vol. 38, no. 12, pp. 16–19, 2005.

[14] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[15] W. Dargie and C. Poellabauer, *Fundamentals of Wireless Sensor Networks: Theory and Practice*. John Wiley & Sons, 2010.

[16] S. Escolar, J. Carretero, and J. Fernandez, “Survey of the State-of-the-Art in Flash-based Sensor Nodes,” in *Flash Memories*, InTech, 2011.

[17] M. Johnson, M. Healy, P. van de Ven, M. J. Hayes, J. Nelson, T. Newe, and E. Lewis, “A comparative review of wireless sensor network mote technologies,” in *IEEE Sensors'09*, Christchurch, 2009, pp. 1439–1442.
[18] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, “How to prolong the lifetime of wireless sensor networks,” in *Mobile Ad Hoc and Pervasive Communications*, 2006, pp. 1–26.
[19] S. Roundy, D. Steingart, L. Frechette, P. Wright, and J. Rabaey, “Power Sources for Wireless Sensor Networks,” in *Wireless Sensor Networks*, Springer Berlin Heidelberg, 2004, pp. 1–17.
[20] S. Rajasegarar, C. Leckie, and M. Palaniswami, “Anomaly detection in wireless sensor networks,” *IEEE Wireless Communications,* vol. 15, no. 4, pp. 34–40, 2008.
[21] M. Cardei, M. T. Thai, Y. Li, and W. Wu, “Energy-efficient target coverage in wireless sensor networks,” in *Proceedings of IEEE INFOCOM'05.* 2005, vol. 3, pp. 1976–1984.
[22] L. Liu, “Wireless sensor and embedded active diagnosis for structural health monitoring,” ProQuest, 2006, p. 27.
[23] I. F. Akyildiz and I. H. Kasimoglu, “Wireless sensor and actor networks: research challenges,” *Ad Hoc Networks*, vol. 2, no. 4, pp. 351–367, Oct. 2004.
[24] “ISO/IEC 7498-1:1994 - Open Systems Interconnection Reference Model.” International Organization for Standardization (ISO).
[25] J. A. Gutierrez, M. Naeve, E. Callaway, M. Bourgeois, V. Mitter, and B. Heile, “IEEE 802.15.4: a developing standard for low-power low-cost wireless personal area networks,” *IEEE Network*, vol. 15, no. 5, pp. 12–19, Sep. 2001.
[26] A. Sleman and R. Moeller, “Integration of Wireless Sensor Network Services into other Home and Industrial networks - using Device Profile for Web Services (DPWS),” in *3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA'08)*, Damascus, 2008, pp. 1–5.
[27] “Approved Draft Amendment to IEEE Standard for Information technology-Telecommunications and information exchange between systems-PART 15.4:Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs): Amendment to add alternate PHY (Amendment of IEEE Std 802.15.4),” *IEEE Approved Std P802.15.4a/D7*, Jan. 2007.
[28] “ZigBee 2006 Specification.” ZigBee Alliance.
[29] “Ad Hoc On Demand Distance Vector (AODV) Routing (RFC 3561).” The Internet Engineering Task Force (IETF).
[30] M. L. Abd-El-Barr, M. A. M. Youssef, and M. M. Al-Otaibi, “Wireless sensor networks - part I: topology and design issues,” in *Canadian Conference on Electrical and Computer Engineering 2005*, Saskatoon, Canada, 2005, pp. 1165–1168.
[31] Y. Wang, “Topology control for wireless sensor networks,” in *Wireless Sensor Networks and Applications*, Springer, 2008, pp. 113–147.
[32] F. Viani, L. Lizzi, M. Donelli, D. Pregnolato, G. Oliveri, and A. Massa, “Exploitation of Parasitic Smart Antennas in Wireless Sensor Networks,” *Journal of Electromagnetic Waves and Applications*, vol. 24, no. 7, pp. 993–1003, 2010.
[33] P. Baronti, P. Pillai, V. W. C. Chook, S. Chessa, A. Gotta, and Y. F. Hu, “Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards,” *Computer Communications*, vol. 30, no. 7, pp. 1655–1695, May 2007.
[34] D. Bruneo, A. Puliafito, and M. Scarpa, “Energy control in dependable wireless sensor networks: a modelling perspective,” in *Proceedings of the Institution of*

Mechanical Engineers, Part O: Journal of Risk and Reliability, vol. 225, no. 4, pp. 424–434, Dec. 2011.
[35] S.-H. Yang, "Principle of Wireless Sensor Networks," in *Wireless Sensor Networks*, Springer London, 2014, pp. 7–47.
[36] F. L. Lewis, "Wireless sensor networks," *Smart environments: technologies, protocols, and applications*, pp. 11–46, 2004.
[37] O. Younis, M. Krunz, and S. Ramasubramanian, "Node clustering in wireless sensor networks: Recent developments and deployment challenges," *IEEE Network*, vol. 20, no. 3, pp. 20–25, 2006.
[38] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, 2005.
[39] E. G. Davis, A. Calveras, and I. Demirkol, "Improving packet delivery performance of publish/subscribe protocols in wireless sensor networks," *Sensors*, vol. 13, no. 1, pp. 648–680, 2013.
[40] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The Many Faces of Publish/Subscribe," *ACM Comput. Surv.*, vol. 35, no. 2, pp. 114–131, Jun. 2003.
[41] P. Costa, G. P. Picco, and S. Rossetto, "Publish-subscribe on sensor networks: a semi-probabilistic approach," in *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference*, Washington, USA, 2005, p. 10 pp.–332.
[42] S. Taherian and J. Bacon, "A publish/subscribe protocol for resource-awareness in wireless sensor networks," in *Proceedings of the International Workshop on Localized Algorithms and Protocols for Wireless Sensor Networks (LOCALGOS'07)*, 2007, pp. 27–38.
[43] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "A Taxonomy of Wireless Micro-sensor Network Models," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 6, no. 2, pp. 28–36, Apr. 2002.
[44] Q. Wang and M. A. Abu-Rgheff, "Cross-layer signalling for next-generation wireless systems," in *IEEE Wireless Communications and Networking (WCNC'03)*, New Orleans, LA, USA, 2003, vol. 2, pp. 1084–1089 vol.2.
[45] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *IEEE Communications Magazine*, vol. 43, no. 12, pp. 112–119, 2005.
[46] W. Su and T. L. Lim, "Cross-layer design and optimisation for wireless sensor networks," *International Journal of Sensor Networks*, vol. 6, no. 1, pp. 3–12, 2009.
[47] V. Kawadia and P. R. Kumar, "A cautionary perspective on cross-layer design," *IEEE Wireless Communications*, vol. 12, no. 1, pp. 3–11, 2005.
[48] T. Canli, F. Nait-Abdesselam, and A. Khokhar, "A Cross-Layer Optimization Approach for Efficient Data Gathering in Wireless Sensor Networks," in *IEEE International Networking and Communications Conference (INCC'08)*, Lahore, Pakistan, 2008, pp. 101–106.
[49] P. Škraba, H. Aghajan, and A. Bahai, "Cross-Layer Optimization for High Density Sensor Networks: Distributed Passive Routing Decisions," in *Ad-Hoc, Mobile, and Wireless Networks*, Springer Berlin Heidelberg, 2004, pp. 266–279.
[50] J. Haapola, Z. Shelby, C. Pomalaza-Raez, and P. Mahonen, "Cross-layer energy analysis of multihop wireless sensor networks," in *Proceeedings of the Second European Workshop on Wireless Sensor Networks*, 2005, pp. 33–44.
[51] I. Hakala and M. Tikkakoski, "From Vertical to Horizontal Architecture: A Cross-layer Implementation in a Sensor Network Node," in *Proceedings of the First*

International Conference on Integrated Internet Ad Hoc and Sensor Networks, New York, NY, USA, 2006.
[52] B. S. Manoj, R. R. Rao, and M. Zorzi, "On the Use of Higher Layer Information for Cognitive Networking," in *IEEE GLOBECOM'07*, Washington, USA, 2007, pp. 3568–3573.
[53] M. A. Razzaque, S. Dobson, and P. Nixon, "Context Awareness through Cross-Layer Network Architecture," in *Proceedings of 16th International Conference on Computer Communications and Networks (ICCCN'07)*, Honolulu, USA, 2007, pp. 1076–1081.
[54] S. A. Velastin, B. A. Boghossian, B. P. L. Lo, J. Sun, and M. A. Vicencio-Silva, "PRISMATICA: toward ambient intelligence in public transport environments," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 1, pp. 164–182, 2005.
[55] O. Karaca and R. Sokullu, "Comparative study of Cross Layer Frameworks for wireless sensor networks," in *1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace Electronic Systems Technology (Wireless VITAE'09)*, Aalborg, Denmark, 2009, pp. 896–900.
[56] H. Karl and A. Willig, *Protocols and architectures for wireless sensor networks*. Hoboken, NJ, Wiley, 2005.
[57] A. C. Roibás, "Understanding the Influence of the Users' Context in AmI," *Social Science Computer Review*, vol. 26, no. 1, pp. 103–118, Feb. 2008.
[58] C. Ramos, J. C. Augusto, and D. Shapiro, "Ambient Intelligence - the Next Step for Artificial Intelligence," *IEEE Intelligent Systems*, vol. 23, no. 2, pp. 15–18, Mar. 2008.
[59] ISTAG, "Scenarios for Ambient Intelligence in 2010." European Commission, 2001.
[60] E. H. L. Aarts and S. Marzano, *The New Everyday: Views on Ambient Intelligence*. 010 Publishers, 2003.
[61] E. Aarts and J. Encarnação, "Into Ambient Intelligence," in *True Visions: The Emergence of Ambient Intelligence*, Springer, 2006.
[62] D. J. Cook, J. C. Augusto, and V. R. Jakkula, "Ambient intelligence: Technologies, applications, and opportunities," *Pervasive and Mobile Computing*, vol. 5, no. 4, pp. 277–298, Aug. 2009.
[63] M.-R. Tazari, "An Open Distributed Framework for Adaptive User Interaction in Ambient Intelligence," in *Ambient Intelligence*, Springer Berlin Heidelberg, 2010, pp. 227–238.
[64] ISTAG, "Strategic orientations and priorities for IST in FP6." European Commission, 2002.
[65] "Philips Research - Technologies." [Online]. Available: http://www.research.philips.com/technologies/ambintel.html. [Accessed: 10-Nov-2014].
[66] M. J. Santofimia, S. E. Fahlman, F. Moya, and J. C. López, "A Common-Sense Planning Strategy for Ambient Intelligence," in *Knowledge-Based and Intelligent Information and Engineering Systems*, Springer Berlin Heidelberg, 2010, pp. 193–202.
[67] J. Euzenat, J. Pierson, and F. Ramparany, "Dynamic context management for pervasive applications," *The Knowledge Engineering Review*, vol. 23, Special Issue no. 01, pp. 21–49, 2008.
[68] P. J. Brown, J. D. Bovey, and X. Chen, "Context-aware applications: from the laboratory to the marketplace," *IEEE Personal Communications*, vol. 4, no. 5, pp. 58–64, 1997.
[69] M. Michelmann, "Context-Aware Applications, Computing in Context - Media Informatics." [Online]. Available:

http://www.medien.ifi.lmu.de/lehre/ws0607/mmi1/essays/Marcel-Michelmann. [Accessed: 09-Nov-2014].
[70] A. K. Dey, "Understanding and Using Context," *Personal Ubi Comp*, vol. 5, no. 1, pp. 4–7, Feb. 2001.
[71] R. Dapoigny and P. Barlatier, "Towards a Context Theory for Context-aware Systems," in *Proceedings of the 2007 Conference on Advances in Ambient Intelligence*, Amsterdam, Netherlands, 2007, pp. 36–55.
[72] A. Bikakis, T. Patkos, G. Antoniou, and D. Plexousakis, "A Survey of Semantics-Based Approaches for Context Reasoning in Ambient Intelligence," in *Constructing Ambient Intelligence*, Springer Berlin Heidelberg, 2008, pp. 14–23.
[73] J. Viterbo, L. Mazuel, Y. Charif, M. Endler, N. Sabouret, K. Breitman, A. E. F. Seghrouchni, and J. Briot, "Ambient intelligence: Management of distributed and heterogeneous context knowledge," *CRC Studies in Informatics Series. Chapman & Hall*, pp. 1–44, 2008.
[74] T. Strang and C. Linnhoff-Popien, "A context modeling survey," in *Workshop Proceedings*, Nottingham, UK, 2004.
[75] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni, "A survey of context modelling and reasoning techniques," *Pervasive and Mobile Computing*, vol. 6, no. 2, pp. 161–180, 2010.
[76] T. Ruiz-López, C. Rodríguez-Domínguez, M. Noguera, and M. J. Rodríguez, "A Model-Driven Approach to Requirements Engineering in Ubiquitous Systems," in *Ambient Intelligence - Software and Applications*, Springer Berlin Heidelberg, 2012, pp. 85–92.
[77] D.-M. Han and J.-H. Lim, "Design and implementation of smart home energy management systems based on zigbee," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1417–1425, Aug. 2010.
[78] F. Ramparany, R. Poortinga, M. Stikic, J. Schmalenstroer, and T. Prante, "An open context information management infrastructure - the IST-Amigo project," in *3rd IET International Conference on Intelligent Environments (IE'07)*, Ulm, Germany, 2007, vol. 2007, pp. 398–403.
[79] J. P. Martínez, A. Vosou, G. Villarreal, and M. R. De Giusti, "An ontology-based context aware system for Selective Dissemination of Information in a digital library," *Journal of Computing*, vol. 2, 2010.
[80] R. Krummenacher and T. Strang, "Ontology-Based Context Modeling," in *Workshop on Context-Aware Proactive Systems*, 2007, p. 22.
[81] L. Zhou, "Ontology learning: state of the art and open issues," *Information Technology and Management*, vol. 8, no. 3, pp. 241–252, 2007.
[82] D. Ejigu, M. Scuturici, and L. Brunie, "An ontology-based approach to context modeling and reasoning in pervasive computing," in *Proceedings of Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, White Plains, NY, USA, 2007, pp. 14–19.
[83] X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung, "Ontology based context modeling and reasoning using OWL," in *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, 2004, pp. 18–22.
[84] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer communications*, vol. 30, no. 14, pp. 2826–2841, 2007.
[85] O. Boyinbode, H. Le, and M. Takizawa, "A survey on clustering algorithms for wireless sensor networks," *International Journal of Space-Based and Situated Computing*, vol. 1, no. 2, pp. 130–136, 2011.

[86] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, 2000, p. 10–pp.
[87] F. Xiangning and S. Yulin, "Improvement on LEACH protocol of wireless sensor network," in *International Conference on Sensor Technologies and Applications (SensorComm'07)*, 2007, pp. 260–264.
[88] M. Youssef, A. Youssef, and M. Younis, "Overlapping multihop clustering for wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 12, pp. 1844–1856, 2009.
[89] L. Qing, Q. Zhu, and M. Wang, "Design of a distributed energy-efficient clustering algorithm for heterogeneous wireless sensor networks," *Computer communications*, vol. 29, no. 12, pp. 2230–2237, 2006.
[90] D. Kumar, T. C. Aseri, and R. Patel, "EEHC: Energy efficient heterogeneous clustered scheme for wireless sensor networks," *Computer Communications*, vol. 32, no. 4, pp. 662–667, 2009.
[91] S. Varma, N. Nigam, and U. Tiwary, "Base station initiated dynamic routing protocol for Heterogeneous Wireless Sensor Network using clustering," in *Fourth International Conference on Wireless Communication and Sensor Networks (WCSN'08)*, 2008, pp. 1–6.
[92] S. Lee, H. Choe, B. Park, Y. Song, and C. Kim, "LUCA: An energy-efficient unequal clustering algorithm using location information for wireless sensor networks," *Wireless Personal Communications*, vol. 56, no. 4, pp. 715–731, 2011.
[93] F. Khan, F. Bashir, and K. Nakagawa, "Dual head clustering scheme in wireless sensor networks," in *International Conference on Emerging Technologies (ICET),* 2012, pp. 1–5.
[94] Z. Ruihua, J. Zhiping, L. Xin, and H. Dongxue, "Double cluster-heads clustering algorithm for wireless sensor networks using PSO," in *6th IEEE Conference on Industrial Electronics and Applications (ICIEA'11)*, 2011, pp. 763–766.
[95] Z. Rehena, S. Roy, and N. Mukherjee, "Topology partitioning in wireless sensor networks using multiple sinks," in *14th International Conference on Computer and Information Technology (ICCIT'11),* 2011, pp. 251–256.
[96] S. Sathe, T. G. Papaioannou, H. Jeung, and K. Aberer, "A survey of model-based sensor data acquisition and management," in *Managing and Mining Sensor Data*, Springer, 2013, pp. 9–50.
[97] L. Wang, L. Chen, and D. Papadias, "Query Processing in Wireless Sensor Networks," in *Managing and Mining Sensor Data*, Springer, 2013, pp. 51–76.
[98] T. Palpanas, "Real-time data analytics in sensor networks," in *Managing and Mining Sensor Data*, Springer, 2013, pp. 173–210.
[99] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, 2000, pp. 56–67.
[100] O. Soysal and M. Demirbas, "Data spider: a resilient mobile basestation protocol for efficient data collection in wireless sensor networks," in *Distributed Computing in Sensor Systems*, Springer, 2010, pp. 393–408.
[101] A. Prajapati and S. Ganesan, "Agent based TDMA schedule for wireless sensor networks," in *IEEE International Conference on Electro/Information Technology (EIT'11)*, 2011, pp. 1–8.
[102] M. Li, D. Ganesan, and P. Shenoy, "Presto: feedback-driven data management in sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 17, no. 4, pp. 1256–1269, 2009.

[103] U. Raza, A. Camerra, A. L. Murphy, T. Palpanas, and G. P. Picco, “What does model-driven data acquisition really achieve in wireless sensor networks?” in *IEEE International Conference on Pervasive Computing and Communications (PerCom'12)*, 2012, pp. 85–94.
[104] S. Lai, J. Cao, and Y. Zheng, “PSWare: a publish/subscribe middleware supporting composite event in wireless sensor network,” in *IEEE International Conference on Pervasive Computing and Communications (PerCom'09)*, 2009, pp. 1–6.
[105] J. H. Ahnn, U. Lee, and H. J. Moon, “Geoserv: A distributed urban sensing platform,” in *11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid'11)*, 2011, pp. 164–173.
[106] F. Oldewurtel, J. Riihijarvi, K. Rerkrai, and P. Mahonen, “The RUNES architecture for reconfigurable embedded and sensor networks,” in *Third International Conference on Sensor Technologies and Applications (SENSORCOMM’09)*, 2009, pp. 109–116.
[107] D. Zhang, H. Huang, C.-F. Lai, X. Liang, Q. Zou, and M. Guo, “Survey on context-awareness in ubiquitous media,” *Multimedia tools and applications*, vol. 67, no. 1, pp. 179–211, 2013.
[108] T. Kanter, S. Pettersson, S. Forsstrom, V. Kardeby, R. Norling, J. Walters, and P. Osterberg, “Distributed context support for ubiquitous mobile awareness services,” in *Fourth International Conference on Communications and Networking in China (ChinaCOM'09)*, 2009, pp. 1–5.
[109] T. Kanter, S. Pettersson, S. Forsström, V. Kardeby, and P. Österberg, “Ubiquitous mobile awareness from sensor networks,” in *Mobile Wireless Middleware, Operating Systems, and Applications-Workshops*, 2009, pp. 147–150.
[110] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” in *ACM SIGCOMM Computer Communication Review*, 2001, vol. 31, pp. 149–160.
[111] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, “Context aware computing for the internet of things: A survey,” *IEEE Communications Surveys & Tutorials*, 2013.
[112] K. T. Kim, W. J. Choi, M. J. Whang, and H. Y. Youn, “CA-MAC: Context Adaptive MAC Protocol for Wireless Sensor Networks,” in *International Conference on Computational Science and Engineering (CSE'09)*, 2009, vol. 1, pp. 344 –349.
[113] H. Kim, J. Park, D. Seong, and J. Yoo, “A Context Aware Data-Centric Storage Scheme in Wireless Sensor Networks,” in *Multimedia, Computer Graphics and Broadcasting*, Springer Berlin Heidelberg, 2011, pp. 326–330.
[114] S. B. Kodeswaran and A. Joshi, “Content and Context Aware Networking Using Semantic Tagging,” in *Proceedings of 22nd International Conference on Data Engineering Workshops*, 2006, p. 77.
[115] P. Hu, M. Portmann, R. Robinson, and J. Indulska, “Context-aware routing in wireless mesh networks,” in *2nd ACM International Conference on Context-Awareness for Self-Managing Systems*, New York, NY, USA, 2008, pp. 16–23.
[116] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca, “A data-oriented survey of context models,” *SIGMOD Rec.*, vol. 36, no. 4, pp. 19–26, Dec. 2007.
[117] D. Griffin and D. Pesch, “Context in pervasive environments,” in *Ambient Intelligence with Microsystems*, Springer, 2009, pp. 187–203.
[118] K.-W. Lee and S.-H. Cha, “Ontology-Based Context-Aware Management for Wireless Sensor Networks,” in *Advances in Computer Science, Environment, Ecoinformatics, and Education*, Springer Berlin Heidelberg, 2011, pp. 353–358.

[119] S. Verstichel, B. Volckaert, B. Dhoedt, P. Demeester, and F. De Turck, "Context-Aware Scheduling of Distributed DL-Reasoning Tasks in Wireless Sensor Networks," *International Journal of Distributed Sensor Networks*, vol. 2011, pp. 1–24, 2011.
[120] I. Gronbek and P. K. Biswas, "Ontology-based abstractions for M2M virtual nodes and topologies," in *International Conference on Ultra Modern Telecommunications & Workshops (ICUMT'09)*, 2009, pp. 1–8.
[121] L. Frye and L. Cheng, "A Network Management System for a Heterogeneous, Multi-Tier Network," in *IEEE Global Telecommunications Conference (GLOBECOM'10)*, 2010, pp. 1–5.
[122] H. Zhou and K. Hou, "CIVIC: An Power- and Context-Aware Routing Protocol for Wireless Sensor Networks," in *International Conference on Wireless Communications, Networking and Mobile Computing (WiCom'07)*, 2007, pp. 2771 –2774.
[123] M. E. Haque, N. Matsumoto, and N. Yoshida, "Context-aware cluster-based hierarchical protocol for Wireless Sensor Networks," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 4, no. 6, pp. 379–386, Sep. 2009.
[124] M. E. Haque, N. Matsumoto, and N. Yoshida, "Exploiting Context-Awareness in Cluster-Based Wireless Sensor Networks," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 4, no. 5, pp. 677–679, 2009.
[125] B.-L. Wenning, A. Timm-Giel, and C. Görg, "A generic framework for context-aware routing and its implementation in wireless sensor networks," *ITG-Fachbericht-Mobilkommunikation-Technologien und Anwendungen*, 2009.
[126] B. Russell, M. L. Littman, and W. Trappe, "Integrating machine learning in ad hoc routing: A wireless adaptive routing protocol," *International Journal of Communication Systems*, vol. 24, no. 7, pp. 950–966, 2011.
[127] S. Li, S. Zhao, X. Wang, K. Zhang, and L. Li, "Adaptive and Secure Load-Balancing Routing Protocol for Service-Oriented Wireless Sensor Networks," *IEEE Systems Journal*, vol. 8, no. 3, pp. 858–867, Sep. 2014.
[128] G. Mathur, P. Desnoyers, P. Chukiu, D. Ganesan, and P. Shenoy, "Ultra-low power data storage for sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 5, no. 4, p. 33, 2009.
[129] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin, "Data-centric storage in sensornets," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 137–142, 2003.
[130] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "GHT: a geographic hash table for data-centric storage," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, 2002, pp. 78–87.
[131] A. T. Boloorchi and M. Samadzadeh, "Energy-efficient and secure in-network storage and retrieval for WSNs: an adaptive approach," *The Journal of Supercomputing*, vol. 65, no. 2, pp. 961–977, 2013.
[132] T. Sharma, R. Joshi, and M. Misra, "Cooperative caching for homogeneous wireless sensor networks," *International Journal of Communication Networks and Distributed Systems*, vol. 2, no. 4, pp. 424–451, 2009.
[133] Y. Liu, B.-C. Seet, and A. Al-Anbuky, "In-network storage for virtual broker-based publish/subscribe in WSNs," in *IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC'12)*, Sydney, Australia, 2012, pp. 2553–2558.
[134] D. Wei, Y. Jin, S. Vural, K. Moessner, and R. Tafazolli, "An energy-efficient clustering solution for wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 10, no. 11, pp. 3973–3983, 2011.

[135] Y. Liao, H. Qi, and W. Li, "Load-Balanced Clustering Algorithm With Distributed Self-Organization for Wireless Sensor Networks," *IEEE Sensors Journal*, vol. 13, no. 5, pp. 1498–1506, 2013.

[136] C. Sevgi and A. Koçyigit, "On determining cluster size of randomly deployed heterogeneous WSNs," *IEEE Communications Letters*, vol. 12, no. 4, pp. 232–234, 2008.

[137] Y. Zhao, Y. Chen, and S. Ratnasamy, "Load balanced and efficient hierarchical data-centric storage in sensor networks," in *5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON'08)*, 2008, pp. 560–568.

[138] R. Sarkar, X. Zhu, and J. Gao, "Double rulings for information brokerage in sensor networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 17, no. 6, pp. 1902–1915, 2009.

[139] X. Li, Y. J. Kim, R. Govindan, and W. Hong, "Multi-dimensional range queries in sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, 2003, pp. 63–75.

[140] M. Aly, K. Pruhs, and P. K. Chrysanthis, "KDDCS: A load-balanced in-network data-centric storage scheme in sensor network," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, New York, NY, USA, 2006, pp. 317–326.

[141] P. Desnoyers, D. Ganesan, and P. Shenoy, "TSAR: a two tier sensor storage architecture using interval skip graphs," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, 2005, pp. 39–50.

[142] M. Albano, S. Chessa, F. Nidito, and S. Pelagatti, "Dealing with nonuniformity in data centric storage for wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 8, pp. 1398–1406, 2011.

[143] S. Yoon and C. Shahabi, "Exploiting spatial correlation towards an energy efficient clustered aggregation technique (cag)," in *IEEE International Conference on Communications (ICC'05)*, 2005, vol. 5, pp. 3307–3313.

[144] J. Xu, X. Tang, and W.-C. Lee, "EASE: an energy-efficient in-network storage scheme for object tracking in sensor networks.," in *SECON*, 2005, pp. 396–405.

[145] S. Krishnamurthy, T. He, G. Zhou, J. A. Stankovic, and S. H. Son, "RESTORE: A real-time event correlation and storage service for sensor networks," in *Proc. of INSS*, 2006.

[146] M. Ghanem and M. Sabaei, "Data centric storage with geographic hash table for wireless sensor networks using Residue Number Systems," in *Sixth International Symposium on Telecommunications (IST'12)*, 2012, pp. 749–753.

[147] M. M. A. Mohamed and A. A. Khokhar, "Dynamic Indexing System for Spatio-temporal Queries in Wireless Sensor Networks," in *12th IEEE International Conference on Mobile Data Management (MDM'11)*, 2011, vol. 2, pp. 35–37.

[148] Y.-J. Joung and S.-H. Huang, "Tug-of-war: An adaptive and cost-optimal data storage and query mechanism in wireless sensor networks," in *4th IEEE International Conference on Distributed Computing in Sensor Systems*, Santorini Island, Greece, 2008, pp. 237–251.

[149] Á. Cuevas, M. Urueña, G. de Veciana, and A. Yadav, "STARR-DCS: Spatio-temporal adaptation of random replication for data-centric storage," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 1, p. 14, 2013.

[150] J. Ahn and B. Krishnamachari, "Fundamental scaling laws for energy-efficient storage and querying in wireless sensor networks," in *7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, Florence, Italy, 2006, pp. 334–343.

[151] R. L. Graham and P. Hell, “On the History of the Minimum Spanning Tree Problem,” *IEEE Ann. Hist. Comput.*, vol. 7, no. 1, pp. 43–57, Jan. 1985.
[152] J. Chen, H. Zhang, X. Du, B. Fang, Y. Liu, and H. Yu, “Base station location protection in wireless sensor networks: Attacks and defense.,” in *IEEE International Conference on Communications*, Ottawa, ON, Canada, 2012, pp. 554–559.
[153] E. F. Krause, *Taxicab geometry: An adventure in non-Euclidean geometry*. Courier Dover Publications, 2012.
[154] M. G. C. Torres, “Energy Consumption in Wireless Sensor Networks Using GSP,” University of Pittsburgh, 2006.
[155] MEMSIC, “MICAz Datasheet.” [Online]. Available: http://www.memsic.com. [Accessed: 10-Nov-2014].
[156] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, “Directed diffusion for wireless sensor networking,” *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 2–16, 2003.
[157] B. Zhou, A. Marshall, and T.-H. Lee, “An energy-aware virtual backbone tree for wireless sensor networks,” in *IEEE GLOBECOM’05*, 2005, vol. 1, p. 6–pp.
[158] M. Ali, “A Brief History of Sensor Networks,” *Princeton University*, 2008.
[159] Y. Liu, B.-C. Seet, and A. Al-Anbuky, “Virtual brokers for large-scale publish/subscribe in wireless sensor networks,” in *IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC'10)*, 2010, pp. 240–246.
[160] Oracle, “Sun Spot.” [Online]. Available: http://www.sunspotworld.com/. [Accessed: 10-Nov-2014].
[161] R. Adler, M. Flanigan, J. Huang, R. Kling, N. Kushalnagar, L. Nachman, C.-Y. Wan, and M. Yarvis, “Intel mote 2: an advanced platform for demanding sensor network applications,” in *Proceedings of the 3rd international conference on Embedded networked sensor systems*, 2005, pp. 298–298.
[162] A. Nosratinia, T. E. Hunter, and A. Hedayat, “Cooperative communication in wireless networks,” *IEEE Communications Magazine*, vol. 42, no. 10, pp. 74–80, 2004.
[163] A. Awad, R. Germany, and F. Dressler, “Data-centric cooperative storage in wireless sensor network,” in *2nd International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL'09)*, 2009, pp. 1–6.
[164] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, “Network coding for distributed storage systems,” *IEEE Transactions on Information Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.
[165] B. Jenkins, “Hash Functions - Algorithm Alley,” *Dr. Dobb’s Journal*, 1997.
[166] B. Koo, J. Won, S. Park, and H. Eom, “PAAR: A routing protocol for context-aware services in wireless sensor-actuator networks,” in *First Asian Himalayas International Conference on Internet (AH-ICI'09)*, 2009, pp. 1 –7.
[167] M. E. Haque, N. Matsumoto, and N. Yoshida, “Utilizing Multilayer Hierarchical Structure in Context Aware Routing Protocol for Wireless Sensor Networks,” *International Journal of Computational Science*, vol. 4, no. 1, pp. 23–37, 2010.
[168] M. ElGammal and M. Eltoweissy, “Distributed context-aware Affinity Propagation clustering in Wireless Sensor Networks,” in *6th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2010, pp. 1 –7.
[169] C. Wang, J. Liu, J. Kuang, and H. Xiang, “A Context-Aware Architecture for Wireless Sensor Networks,” in *Recent Advances in Computer Science and Information Engineering*, Springer, 2012, pp. 199–205.

[170] A. Gladisch, R. Daher, P. Lehsten, and D. Tavangarian, "Context-aware energy management for energy-self-sufficient network nodes in wireless mesh networks," in *3rd International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2011, pp. 1 –8.
[171] P. Hu, Z. Zhou, Q. Liu, and F. Li, "The HMM-based Modeling for the Energy Level Prediction in Wireless Sensor Networks," in *2nd IEEE Conference on Industrial Electronics and Applications (ICIEA'07)*, 2007, pp. 2253–2258.
[172] R. Goudarzi, B. Jedari, and M. Sabaei, "An Efficient Clustering Algorithm Using Evolutionary HMM in Wireless Sensor Networks," in *IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC'10)*, 2010, pp. 403–409.
[173] J. Shanthini and S. Vijayakumar, "Modified Simple Network Management Protocol for 6Lowpan," *Procedia Engineering*, vol. 38, pp. 1024–1029, 2012.
[174] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., *The description logic handbook: theory, implementation, and applications*. Cambridge university press, 2003.
[175] J. Iglesias, A. M. Bernardos, P. Tarrío, J. R. Casar, and H. Martín, "Design and validation of a light inference system to support embedded context reasoning," *Personal and Ubiquitous Computing*, vol. 16, no. 7, pp. 781–797, 2012.
[176] H. Hasemann, A. Kroller, and M. Pagel, "RDF Provisioning for the Internet of Things," in *3rd International Conference on the Internet of Things (IOT)*, 2012, pp. 143–150.
[177] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks*, vol. 52, no. 12, pp. 2292–2330, 2008.
[178] L. D. Mendes and J. JPC Rodrigues, "A survey on cross-layer solutions for wireless sensor networks," *Journal of Network and Computer Applications*, vol. 34, no. 2, pp. 523–534, 2011.
[179] T. S. Azevedo, R. L. Bezerra, C. A. V. Campos, and L. F. M. De Moraes, "An Analysis of Human Mobility Using Real Traces," in *IEEE Wireless Communications and Networking Conference (WCNC'09)*, 2009, pp. 1–6.
[180] B. A. A. Nunes and K. Obraczka, "On the symmetry of user mobility in wireless networks," in *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2011, pp. 1–6.