RESOURCE ALLOCATION IN OVERLAY WIRELESS SENSOR NETWORKS USING MULTIAGENT REINFORCEMENT LEARNING

SAJEE SINGSANGA

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Telecommunication Engineering

Suranaree University of Technology

Academic Year 2010

การจัดสรรทรัพยากรในโครงข่ายตัวตรวจรู้ไร้สายแบบซ้อนทับโดยใช้วิธีการเรียนรู้แบบมัลติเอเจนท์รีอินฟอร์สเมนท์

นางสาวศจี สิงห์สง่า

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมโทรคมนาคม
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2553

RESOURCE ALLOCATION IN OVERLAY WIRELESS SENSOR NETWORKS USING MULTIAGENT REINFORCEMENT LEARNING

Suranaree University of Technology has approved this thesis submitted in partial fulfillment of the requirements for a Master's Degree.

Thesis Examining Committee

_____

(Asst. Prof. Dr. Peerapong Uthansakul)

Chairperson

_____

(Asst. Prof. Dr. Wipawee Hattagam)

Member (Thesis Advisor)

_____

(Assoc. Prof. Dr. Arthit Srikaew)

Member

_____

(Dr. Wut Dankittikul)

Acting Vice Rector for Academic Affairs

_____

(Assoc. Prof. Dr. Vorapot Khompis)

Dean of Institute of Engineering

ศจี สิงห์สง่า : การจัดสรรทรัพยากรในโครงข่ายตัวตรวจรู้ไร้สายแบบซ้อนทับ โดยใช้วิธีการเรียนรู้แบบมัลติเอเจนท์รีอินฟอร์สเมนท์ (RESOURCE ALLOCATION IN OVERLAY WIRELESS SENSOR NETWORKS USING MULTIAGENT REINFORCEMENT LEARNING) อาจารย์ที่ปรึกษา : ผู้ช่วยศาสตราจารย์ ดร.วิภาวี หัตถกรรม, 116 หน้า.

โครงข่ายตัวตรวจรู้ไร้สายได้ถูกพัฒนาและประยุกต์ใช้ในการเฝ้าระวังด้านสิ่งแวดล้อมอย่างกว้างขวาง โครงข่ายตัวตรวจรู้ไร้สายมีความสามารถในการเฝ้าระวังและเก็บข้อมูลเชิงกายภาพภายในพื้นที่เฉพาะหรือสิ่งแวดล้อมที่สนใจ อย่างไรก็ตามโครงข่ายตัวตรวจรู้ไร้สายถูกจำกัดด้วยแหล่งกำเนิดพลังงานที่มีขนาดเล็กมาก ดังนั้นความจุของแบตเตอร์รี่จึงถูกจำกัดตามไปด้วย ข้อจำกัดด้านพลังงานสามารถลดได้โดยการจัดสรรทรัพยากรร่วมกันระหว่างโครงข่ายตัวตรวจรู้ไร้สายหลายโครงข่ายกระจายตัวซ้อนทับกันอย่างเป็นอิสระภายในพื้นที่สนใจเดียวกัน โดยไม่ขัดแย้งซึ่งกันและกัน โครงข่ายเช่นนี้ถูกเรียกว่าโครงข่ายตัวตรวจรู้ไร้สายแบบซ้อนทับ โครงข่ายประเภทนี้มีคุณประโยชน์อย่างมากเช่น การเพิ่มเส้นทางการส่งข้อมูล ลดการใช้พลังงานในโครงข่าย ซึ่งสามารถยืดอายุการใช้งานเครือข่ายได้ อย่างไรก็ตามเนื่องด้วยพฤติกรรมที่เห็นแก่ตัวของตัวตรวจรู้ไร้สาย เพื่อสงวนพลังงานของตน การแลกเปลี่ยนทรัพยากรระหว่างเครือข่ายอาจไม่เกิดขึ้น ดังนั้นงานวิจัยนี้มุ่งเน้นไปการกำหนดกลยุทธ์การส่งเพกเกตในโครงข่ายที่ก่อประโยชน์กับทุกโครงข่ายภายใต้ทรัพยากรที่มีจำกัดในโครงข่ายตัวตรวจรู้ไร้สายแบบซ้อนทับ

วัตถุประสงค์ของงานวิจัยนี้คือการนำเสนอกระบวนการเรียนรู้แบบแนชคิวและแบบพาเรโตคิวซึ่งเป็นวิธีการเรียนรู้แบบมัลติเอเจนท์รีอินฟอร์สเมนท์ประยุกต์ใช้ในเกมการส่งแพกเกตในโครงข่ายตัวตรวจรู้ไร้สายแบบซ้อนทับที่ไม่เกื้อกูลกัน วิทยานิพนธ์นี้มีองค์ความรู้หลักสามประการ องค์ความรู้ประการแรกคือ การประยุกต์ใช้กระบวนการเรียนรู้แบบแนชคิวและแบบพาเรโตคิวเพื่อกำหนดกลยุทธ์ร่วมที่ดีที่สุดสำหรับการส่งต่อเพกเกตในโครงข่ายตัวตรวจรู้ไร้สายแบบซ้อนทับที่ไม่เกื้อกูลกันด้วยการกำหนดปัญหาให้เป็นกระบวนการตัดสินใจแบบมาร์คอฟ (Markov decision process) องค์ความรู้ประการที่สองคือ การขยายกระบวนการเรียนรู้แบบเพลโตคิวจากกระบวนการตัดสินใจแบบมาคอฟไปสู่การกำหนดปัญญาโดยใช้แบบจำลองพลังงานวิทยุ องค์ความรู้ประการสุดท้าย คือ การเลือกจุดที่เหมาะสมที่สุดแบบพาเรโตในวิธีการทางทฤษฎีเกมความไม่ร่วมมือกันเพื่อประยุกต์ใช้ในเกมการส่งต่อแพตเกต

ผลการทดลองชี้ให้เห็นว่าวิธีการที่นำเสนอสามารถกำหนดกลยุทธ์ร่วมที่ดีที่สุดได้โดยการเรียนรู้แบบออนไลน์ซึ่งต่างจากวิธีการทางทฤษฎีเกมความไม่ร่วมมือซึ่งเป็นวิธีค้นหากลยุทธ์อย่างถี่ถ้วนแบบออฟไลน์ วิธีการที่นำเสนอใช้เวลาในการคำนวณเพื่อได้มาซึ่งกลยุทธ์ น้อยกว่าวิธีการทางทฤษฎีเกมความไม่ร่วมมือ ในขณะที่ได้รับผลตอบแทนที่สูงกว่า และมีความทนทานต่อการเปลี่ยนแปลงสภาพแวดล้อมได้ดีกว่า (ได้แก่ การเปลี่ยนแปลงเส้นทางการเชื่อมต่อของตัวตรวจรู้ และค่าการสูญเสียเชิงวิถีในอากาศ) ดังนั้นวิธีการที่ถูกนำเสนอในงานวิจัยนี้จึงมีความสามารถในการปรับเปลี่ยนกลยุทธ์เมื่อเกิดการเปลี่ยนแปลงของสิ่งแวดล้อมด้วยการคำนวณที่น้อยกว่าในระยะยาว

สาขาวิชาวิศวกรรมโทรคมนาคม ลายมือชื่อนักศึกษา_____

ปีการศึกษา 2553 ลายมือชื่ออาจารย์ที่ปรึกษา_____

SAJEE SINGSANGA : RESOURCE ALLOCATION IN OVERLAY WIRELESS SENSOR NETWORKS USING MULTIAGENT REINFORCEMENT LEARNING. THESIS ADVISOR : ASST. PROF. WIPAWEE HATTAGAM, Ph.D. 116 PP.

WIRELESS SENSOR NETWORKS/OVERLAY NETWORK/NON-COOPERATIVE GAME/NASH Q-LEARNING (NASHQ)/PARETO Q-LEARNING (PARETOQ)

Wireless Sensor Networks (WSNs) have been developed and extensively applied in environment monitoring. WSNs can be used to monitor and collect various physical attributes within a specific area or environment of interest. However, WSNs have limited power sources and must be extremely small, therefore their battery capacity constraints are much higher. To alleviate such limitation, multiple sensor networks can coexist independently within a region of interest without conflicting each other in order to share resources. Such networks are referred to as overlay WSNs. These networks can potentially gain certain benefits such as alternative routing paths, reduced energy consumption, thereby prolonging their network lifetime. However, selfish behaviors may exist among sensor nodes to conserve their energy. Thus, cooperation between sensor nodes belonging to different network authorities may not always be readily available. Therefore, the main focus of this research is how to determine packet forwarding strategies which are beneficial to all networks under constrained resources in overlay WSNs.

The underlying aim of this research is therefore to propose the Nash Q-learning and the Pareto Q-learning, which are multiagent reinforcement learning algorithms in a packet forwarding game in non-cooperative overlay WSNs. The contribution of this

research are three-fold. The first contribution is the application of NashQ and ParetoQ algorithms to achieve the best mutual packet forwarding strategy in non-cooperative overlay WSNs with the MDP formulation. The second contribution is the extension of the ParetoQ algorithm from the MDP formulation to the radio energy model. The final contribution centers on the incorporation of Pareto optimality with the Non-cooperative game algorithm and its application to the packet forwarding game.

The experiments show that the proposed algorithms can obtain the best mutual strategy by learning packet forwarding strategies online, as opposed to the offline exhaustive search in an existing Non-cooperative game theoretic approach. The proposed algorithms require significantly less computational time to obtain a strategy than the Non-cooperative game algorithm while achieving higher utility and higher robustness to dynamic environments (i.e., changing topology and path loss exponent) owing to its inherent online learning process. Thus, the proposed approaches are more adaptive to environmental changes yet less computationally demanding in the long run.

School of Telecommunication Engineering Student's Signature _____

Academic Year 2010 Advisor's Signature_____

ACKNOWLEDGEMENT

I am grateful to all those, who by their direct or indirect involvement have helped in the completion of this thesis.

First and foremost, I would like to express my sincere thanks to my thesis advisors, Asst. Prof. Dr. Wipawee Hattagam for her invaluable help and constant encouragement throughout the course of this research. I am most grateful for her teaching and advice, not only the research methodologies but also many other methodologies in life. I would not have achieved this far and this thesis would not have been completed without all the support that I have always received from her.

In addition, I am grateful for the lecturers in School of Telecommunication Engineering for their suggestion and all their help. I would also like to express my thanks to Prof. Dr. Ewe Hong Tat of Universiti Tunku Abdul Rahman, Malaysia, for granting me the opportunity to do research in Malaysia.

I would also like to thank Assoc. Prof. Dr. Arthit Srikaew and Asst. Prof. Dr. Peerapong Uthansakul for accepting to serve in my committee.

My sincere appreciation goes to Ms. Maneerat Tumpong and Ms. Pranitta Arthans for their valuable administrative support during the course of my dissertation.

Finally I am most grateful to my parents and my friends both in both masters and doctoral degree courses for all their support throughout the period of this research

Sajee Singsanga

TABLE OF CONTENTS

Page

ABSTRACT (THAI) I

ABSTRACT (ENGLISH) III

ACKNOWLEDGMENTS V

TABLE OF CONTENTS VI

LIST OF TABLES IX

LIST OF FIGURES X

SYMBOLS AND ABBREVIATIONS XIII

CHAPTER

I INTRODUCTION 1

1.1 Significance of problem 1

1.2 Research objective 12

1.3 Assumption 13

1.4 Scope of the research 13

1.5 Expected usefulness 14

1.6 Synopsis of thesis 14

II BACKGROUND THEORY 16

2.1 Introduction 16

2.2 Markov Decision Process Theory 18

2.2.1 Markov Property 18

TABLE OF CONTENTS (Continued)

Page

2.2.2 Markov Decision Process 19

2.3 Reinforcement learning 21

2.3.1 The value function 22

2.3.2 The optimal value function 23

2.3.3 Q-learning 24

2.4 Multiple agent Q-learning algorithm 25

2.4.1 Nash Q-learning algorithm 25

2.4.1.1 The action-value function 25

2.4.1.2 Convergence 27

2.4.2 Pareto Q-learning algorithm 28

2.4.2.1 Pareto optimality 28

2.4.2.2 Lexicographic convention 29

2.5 Summary 31

III PACKET FORWARDING IN OVERLAY WIRELESS SENSOR NETWORKS: NASH Q-LEARNING 32

3.1 Introduction 32

3.2 Learning in non-cooperative game 34

3.3 Problem formulation 38

3.4 Experiment results 42

3.5 Summary 50

TABLE OF CONTENTS (Continued)

Page

IV PACKET FORWARDING IN OVERLAY WIRELESS SENSOR NETWORKS: PARETO Q-LEARNING 51

4.1 Introduction 51

4.2 Learning in non-cooperative game 57

4.2.1 Reinforcement learning 57

4.2.2 Multi agent Q-learning 59

4.2.2.1 Nash Q-learning 60

4.2.2.2 Pareto Q-learning 61

4.3 Problem formulation 63

4.3.1 Packet forwarding game 63

4.3.1.1 Radio model 64

4.3.1.2 Action 64

4.3.1.3 Reward 65

4.3.2 MDP model 65

4.3.3 Radio energy model 67

4.3.4 ParetoQ reinforcement learning 68

4.3.5 The non-cooperative game based on Pareto optimality framework 69

TABLE OF CONTENTS (Continued)

Page

4.4 Experiment results 70

4.4.1 ParetoQ based on MDP model 72

4.4.2 ParetoQ based on radio energy model 78

4.5 Implementation 94

4.6 Summary 94

V CONCLUSION AND FUTURE WORK 96

5.1 Conclusion 96

5.1.1 Chapter 3 97

2.1.2 Chapter 4 98

5.2 Future work 99

5.2.1 WSNs with energy harvesting technology 99

5.2.2 Security in routing protocol 100

5.2.3 Extension to distributed MARL 100

5.2.4 Sensor node mobility 101

5.2.5 Study performance with raw data 101

REFERENCES 102

APENDIX A. PUBLICATION 108

BIOGRAPHY 116

LIST OF TABLES

Table **Page**

2.1 The Nash Q-learning algorithm 27

3.1 Simulation Parameter Values 43

4.1 The Pareto Q-learning algorithm 62

4.2 Simulation Parameter Values 71

LIST OF FIGURES

Figure **Page**

2.1 A MDP model 19

2.2 Diagram of agent-environment interaction in reinforcement learning 22

2.3 Three types of general-sum game 30

3.1 Effect of network size on cooperation in common sink scenario 47

3.2 Effect of network size on cooperation in separate sink scenario 48

3.3 Effect of path loss exponent in common sink scenario 49

3.4 Convergence of Nash Q-learning in common sink scenario 50

4.1 Effect of network size on cooperation in common sink scenario with the MDP model 75

4.2 Effect of network size on cooperation in separate sink scenario with the MDP model 76

4.3 Effect of path loss exponent on cooperation in common sink scenario with the MDP model 77

4.4 Effect of network size on cooperation in common sink scenario with the radio energy model 79

4.5 The successful packet delivery radio in the common sink scenario with the radio energy model 81

4.6 The energy required in the packet delivery process in the common sink scenario with the radio energy model 82

LIST OF FIGURES (Continued)

Figure **Page**

4.7 Utility value in the common sink scenario with the radio energy model 83

4.8 Number of time steps required to obtain a strategy in the common sink scenariowith the radio energy model 84

4.9 The difference in average utility obtained by both agents in the common sink scenario with the radio energy model 86

4.10 Effect of network size on cooperation in separate sink scenario with The radio energy model 87

4.11 The successful packet delivery radio in the separate sink scenario with the radio energy model 88

4.12 The energy required in the packet delivery process in the separate sink scenario with the radio energy model 89

4.13 Utility value in the separate sink scenario with the radio energy model 90

4.14 Number of time steps required to obtain a strategy in the separate sink scenariowith the radio energy model 90

4.15 The difference in average utility obtained by both agents in the separate sink scenario with the radio energy model 91

LIST OF FIGURES (Continued)

Figure **Page**

4.16 Effect of path loss exponent on cooperation in common sink scenario with the radio energy model 93

SYMBOLS AND ABBREVIATIONS

WSNs	=	Wireless sensor networks
QoS	=	Quality-of-service
BS	=	Base station
RL	=	Reinforcement learning
TD	=	Temporal difference control algorithm
MARL	=	Multi-agent reinforcement learning
NashQ	=	Nash Q-learning
ParetoQ	=	Pareto Q-learning
MDP	=	Markov decision process
t	=	Time step index
S_t	=	State of the process at time t
S	=	State space
s	=	Current state
s'	=	Next state
A	=	Action space
a	=	Action
$E[\cdot]$	=	Expectation operator
β	=	Discount factor

SYMBOLS AND ABBREVIATIONS (Continued)

$R(s,a,s')$	=	Expected reward given any current state s and an action a with anynext state s'
r	=	Reward
P	=	State transition probability metric
π	=	Policy
π^*	=	Optimal policy
$P[A]$	=	Distribution over the action space
$Q_t^{\pi}(s,a)$	=	The action-value function of a given policy π associates all state-action pairs (s,a)
R_t	=	Expected discounted return of the agent
$E^{\pi}[\cdot]$	=	Expectation operator under the policy π
$V^{\pi}(s)$	=	Value function of a state (s) under policy π
$V^*(s)$	=	Value function of a state (s) under optimal policy π^*
α	=	Learning rate
$Q^*(s,a)$	=	The action-value function of a given optimal policy π^* associates all state-action pairs (s,a)
i	=	Agent
ξ	=	The strategy profile
NE	=	Nash equilibrium
$P(a)$	=	The transition probability matrix

SYMBOLS AND ABBREVIATIONS (Continued)

h	=	High level sate
m	=	Medium level state
l	=	Low level state
$p_{ij}(a)$	=	The transition probability from state i to state j by taking action a
η	=	the set of players
τ	=	the set of strategies
μ	=	Utility function
$C_{i,TX}$	=	The transmission cost of agent i
$C_{i,RX}$	=	The reception cost of agent i
$C_i(t)$	=	A cost of agent i in time step t
LH	=	Lemke Howson method
b	=	size of the measurement packet transmitted
E_{elec}	=	The expended cost in the radio electronics
σ	=	Path loss exponent
ε_{amp}	=	Energy consumed at the output transmitter antenna for transmitting one meter
SR_i	=	Successful measurement ratio required by agent i
$g_i(t)$	=	Gain of agent i in time step t
$p_i(t)$	=	Proportion of delivered measurement packets to the sink at time t

SYMBOLS AND ABBREVIATIONS (Continued)

DD	=	The agent does not ask the other network to forward its packets and drops all packets from other network if asked for help.
DF	=	The agent does not ask the other network to forward its packets but helps the other network to forward all packets if asked for help.
AD	=	The agent asks the other network to forward its packets but drops all packets from the other network if asked for help.
AF	=	The agent asks the other network to forward its packets and helps the other network forward all packets if asked for help.
$P_{ss'}^{a}$	=	Probability of the environment transiting to a possible next state s' t given the state s and action a
$p_{xy}(a)$	=	Transition probability of agent i to state y by taking action a at state x
θ	=	Transition probabilities
ϕ	=	Transition probabilities
δ	=	Transition probabilities
B_i	=	Bottleneck of agent i
$B_i(k)$	=	Quantized remaining battery level of agent i

CHAPTER I

INTRODUCTION

This chapter introduces a background on resource allocation problems in multi-agent wireless sensor networks (WSNs) and highlights the significance of resource allocation problems using reinforcement learning. It also presents the motivation for applying reinforcement learning to achieve the best mutual policy for the agents which is the main focus of this thesis.

1.1 Significance of the problem

A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous sensory devices that can communicate with each other to perform sensing and data processing cooperatively. The overall objective of a WSN is to provide a low-cost solution to gather physical data from the environment, such as noise, pressure, light, vibration or temperature, at different locations, observation and transmit it to a base station. The most common energy storage device used in a sensor node is a battery which is suitable for a micro sensor with very low power consumption. Therefore, WSN promises unlimited potential for numerous application areas including environmental, medical, military, transportation, entertainment, crisis management, homeland defense, and smart spaces, (Stankovic et al., 2008).

Since WSNs are based on limited power sources and must be extremely small, their battery capacity constraints are much higher. Therefore, processing power,

memory and wireless communication abilities are very limited in order to reduce power consumption.

1.1.1 Overlay WSNs

Most research works consider a WSN which is controlled simply by one authority. However, in many scenarios, multiple sensor networks can coexist independently within a region of interest without conflicting each other. These networks may even be physically overlapping and their sensor nodes may be interleaved. Such networks are referred to as overlay wireless sensor networks (WSNs). In other words, overlay WSNs include a large number of nodes that are deployed in the same area which are controlled by different authorities. The networks perform different tasks and measure different data within the same area. The role of each sensor node in an overlay WSN is to send or forward its packets to its neighbor nodes through multi-hop communications, either with nodes within the same network authority or with nodes belonging to a different authority.

Overlay networks provide several advantages such as the development of a new protocol with features needed in a short duration, implementation with a small cost and the ability to provide alternative routes for the transmission of data. Moreover, the most important usage for overlay WSNs is resource sharing between different authorities which can prolong their lifetime. The main reason is because intermediate nodes from other network authorities may help shorten the data transmission distance between neighboring nodes. When two authorities share their sensor nodes with the same goal of sending packets via a shorter distance, energy consumption is lowered; the networks can save their energy and eventually prolong their lifetime. Although overlay networks have several advantages, a significant

amount of energy is also lost when sensor nodes within the overlay WSN cooperatively process and forward the data for other networks. As energy consumption is a critical issue for such networks, reducing energy consumption and prolonging the network lifetime are important targets.

1.1.2 Significance of resource allocation in overlay WSNs

Currently, many researches related to resource allocation problems in overlay WSNs have been proposed with a focus on saving energy. Murase and Shimonishi (2006) proposed a routing protocol in overlay networks, whereby end users establish routes to improve quality-of-service (QoS) by reducing delay and packet. Their results show that efficient network resource consumption is achieved and the network lifetime is prolonged. Mao et al. (2008) proposed an energy-aware coverage control protocol (EACCP) to reduce the amount of energy consumption and improve the performance in terms of sensing coverage and lifetime by minimizing the energy consumption required for overhead control, and balancing the energy load among all nodes. EACCP has been shown to adapt well to applications with heterogeneous energy capacities in the sensor networks, as well as effectively reduce the control overhead. Wang et al. (2007) proposed an energy-efficient clustering algorithm based on virtual area partition (VAP-E) which can approximately calculate the total energy of the networks. By doing so, their method can balance the load between clusters and prolong lifetime of the network by reducing energy consumption. Furthermore, other areas of research related to overlay WSN applications include fault tolerance problem (Chitnis, 2009; Han et al., 2007) where in real applications, unpredictable events, such as battery depletion and environmental impairment, may cause these sensor nodes devices to fail. In addition, a node

deployment problem is studied in (Yu et al., 2007) where an algorithm is proposed to decide how many and where overlay sensor nodes should be deployed in the network. Several factors which impact the deployment of sensor nodes are considered to analyze and evaluate their solution including radio coverage range, node traffic load, and distance between sensor nodes. Their simulation results showed that their algorithm can increase the network lifetime and reliability in the system. Li et al (2006) considered a target detection problem. The objective was to find the best way to combine and allocate resources so as to maximize the performance level of the sensor network in terms of target destruction rate and time in a battlefield application. All of these works focus on saving energy in overlay WSNs. Their results showed that two authorities cooperatively sharing resources results in reduced energy consumption and increased network performance. However, these researches only considered the resource allocation problem in a cooperative situation, meaning that, the authorities have to agree on sharing or providing a common resource in order to increase the benefit in their networks.

Because cooperative behavior between sensor nodes belonging to authorities may not always readily available, the lack of cooperation among sensor nodes makes them more vulnerable to losing certain advantages aforementioned. Such situation is referred to as a non-cooperative game. Sensor nodes not only are deployed independently without any fixed strategy, but they may also act selfishly to conserve their energy. Furthermore, there is no guarantee that node cooperation will be beneficial to both WSNs. It may even be possible that cooperation may lead to benefits for a single party alone, or no benefit to any party at all. Vaz et al. (2008) showed that cooperation between two authorities in the same area may not always be

beneficial to any network, because whether or not each authority will cooperate depends on the configuration of each network. Their results showed that there are four factors which affect node cooperation, i.e. the density of the network, the data collection rate, the path loss exponent and the routing algorithm. Hence, node cooperation between different authorities in overlay WSNs is not straight forward. It is therefore necessary to find an algorithm (s) for each authority to decide whether to cooperate with each other or not at any given condition or instance, in a non-cooperative environment.

1.1.3 The best strategy in overlay WSNs

In a non-cooperative game situation, there exist a number of decision makers, called *agents* (or *players)*, who have potentially conflicting interests. Many researches try to find an algorithm which can rationally decide to select the best strategy in multi-agent WSNs. The tools which are usually employed to select suitable strategies for sensor node in WSNs are the Non-cooperative game algorithm (Felegyhaz et al., 2005) and reinforcement learning (Sutton and Barto, 1998).

1.1.3.1 The Non-cooperative game agorithm

Recently, game theory has become a promising tool to analyze and improve the performance of sensor networks. This owes to the fact that game theory helps analyses the problem and selects the best strategy for problem solution, (Machado and Tekinay, 2008). Game theory, (Shoham and Brown, 2009), is a discipline aimed at modeling situations in which decision makers have to make specific actions that have mutual, possibly conflicting consequences. It has been used primarily in economics, in order to model competition between companies. The major advancement that has driven much of the development of game theory is Nash

equilibrium. Nash equilibrium is a collection of strategies for each of the players such that each player's strategy is the best-response to the other players' strategies.

Game theory has also been applied in WSNs. Many researches employ an existing non-cooperative game approach to solve routing resource allocation problems in WSNs. Lima et al. (2008) used game theory to determine a relay selection strategy for geographic routing in multi-hop WSNs. Their approach can retain a substantial portion of the Packet Delivery Success Ratio (PDSR) associated with a system with perfect relay selection. Zhao et al. (2008) applied a novel concept of incompletely cooperative game the MAC layer in WSNs. The incompletely cooperative game is classified as a stochastic game, where each player (i.e., sensor node) estimates the current game state based on what happened in the past timeslots. The player can predict its opponent's actions according to what has happened, take actions simultaneously, i.e., transmitting their packets, listening or sleeping. A simplified game-theoretic MAC protocol (GMAC) was presented to design a suitable equilibrium strategy based on a conditional game in WSNs. Their simulation results showed that the incompletely cooperative game is an appropriate tool to improve throughput, and decrease delay and packet-loss-rate, while increasing energy efficiency. Niyato et al. (2007) investigated the performance of different sleep and wakeup strategies in a solar-powered wireless sensor/mesh network, where a solar cell is used to charge the battery in a sensor/mesh node. An analytical queueing model was presented for solution along with the game theoretic formulation, which was used for the design and optimization of energy efficient protocols for solar-powered wireless sensor/mesh networks under quality-of-service (QoS) constraints. Agah et al. (2004) proposed a non-cooperative game framework for intrusion detection in mobile

wireless sensor networks. They defined the attacker and the sensor network as the players in the game. They also defined the payoff function between players as a function of their distance and each node's transmitter signal strength. They showed that non-cooperative game can achieve Nash equilibrium for both players as well as significantly improve the chance of detecting intrusions. Qiang et al. (2009) constructed a power control model based on non-cooperative game theory for ad hoc and sensor networks. They proposed a distributed non-cooperative algorithm for power control (DNGAPC) that satisfied QoS requirements such as the maximum network capacity, the minimum network radius and guaranteed network connectivity. Their approach outperformed the maximum energy power control algorithm (MAXPCA) and minimum energy power control algorithm (MINPCA) in terms of reachability, capacity and energy efficiency. Ma et al. (2008) also considered an intrusion detection system (IDS) in a WSN. Their approach allowed each cluster head sensor node to decide whether a service is an intrusive service by considering the probability of starting up an IDS service using a non-cooperative game theory. The method not only ensures the security of network, but also reduces the cost caused by monitoring and prolongs the lifetime of each sensor node. Although those researches used non-cooperative game theory to determine resource allocation strategies in wireless sensor networks, non-cooperative game theory for overlay WSNs remains much to be explored.

There are related works which consider non-cooperative game in overlay WSNs. In (Wu et al., 2005; Miller et al., 2005), incentive mechanisms were used to motivate cooperation between sensor nodes. Wu et al. (2005) proposed a mechanism design (MD) approach called InterSensorNet based on game theory to

handle the strategic agents that respond to incentives, which include a mechanism design for Data Collection (MDC) and the mechanism design for Data Aggregation (MDA). They assumed multiple sensor networks under the control of different authorities. Their approach can be applied in routing and aggregation problems for optimizing the power usage and lifetime of the network. Miller et al. (2005) proposed the use of tokens as incentive to encourage each sensor associated with a network authority (sponsor) to cooperate with one another. The game starts with each sponsor acknowledging an agreed-upon number of tokens, which it distributes among its sensors. When a sensor from one sponsor requests a favor from another sponsor's sensor, it offers a token in exchange for the favor. If the request is able to provide the favor, it does so in return for the token. If a sensor runs out of tokens, it can no longer request favors. So the more tokens are available to the sensors, the more favors will be performed. Their work showed that when sponsors jointly agree on the number of tokens, they can trade for favors. However, this approach will be beneficial for both sponsors only if they sign a contract before network deployment. Felegyhazi et al. (2005) applied the Non-cooperative game algorithm to describe such a situation that cooperation can exist in overlay WSNs without incentive mechanisms. They proposed a packet forwarding game model based on a non-cooperative resource allocation problem. They performed simulations to study the conditions for a cooperative equilibrium to hold in randomly generated network topology scenarios. Their results showed that the Non-cooperative game algorithm is a suitable framework to determine equilibrium strategy for such problem.

The aforementioned works showed that game theory can determine a suitable strategy for all authorities. However, the major drawback of game

theory is the exhaustive search through the space of all joint strategies to obtain a suitable strategy. The agents must search the entire strategy space over to compute the benefits of all possible actions of sensor node to achieve the final conclusion or the best strategy.

1.1.3.2 Reinforcement Learning

Reinforcement learning is a machine learning method which allows an agent to systematically learn correct behaviors through trial-and-error interactions with a dynamic environment in order to achieve a particular goal, (Sutton and Barto, 1998). Each agent has the capability to process and communicate thereby enabling it to make decisions and perform tasks in a distributed and coordinated manner to achieve a system-wide objective. As sensors in overlay WSNs are spatially distributed, constrained in resources, and can have a large number, the distributive decision-making feature of RL is inherently suitable for WSNs. RL has been employed for resource allocation in many researches, where it has shown to be more efficient than traditional resource allocation schemes.

A reinforcement learning strategy called Q-learning is an off-policy temporal difference control algorithm, which directly approximates the optimal action-value function (Q-value). Each learning agent takes actions, receives a reward, updates local information with input from the environment, and repeats the process by learning its own optimal strategy (i.e. policy). Shah and Kumar (2007) proposed the Distributed Independent Reinforcement Learning (DIRL) based on Q-learning, which enabled autonomous self-learning/adaptive applications with inherent support for efficient resource/ management in WSNs. Their results showed that their approach can achieve greater reward than other existing methods. Wang et al. (2006) presented

a novel routing scheme called the adaptive Routing (AdaR) based on Least Squares Policy Iteration (LSPI). They improved the convergence rate of a standard Q-learning while learning the optimal routing strategy in the sensor network. Their results showed that AdaR can receive higher success rates of packet arriving at the base station and cumulative rewards than the normal Q-learning. Moreover, their results suggested that their approach can make correct tradeoffs between multiple optimization goals in order to maximize network lifetime. Forster and Murphy (2007) proposed a reinforcement learning approach to route data to multiple sinks in a WSN. They devised the Feedback Routing for Optimizing Multiple Sinks (FROM) algorithm. Q-learning has been applied in FROM to identify the best routes to multiple sinks. However, these above works only used RL to solve resource allocation problems in single authority WSNs.

To cater multiple network authority frameworks existing in overlay WSNs, the concept of Multi-Agent Reinforcement Learning (MARL) can be used. MARL integrates together the single-agent RL, game theory, and direct policy search techniques. Applications of MARL are rapidly expanding and applied to several problem domains (Busoniu et al., 2008). With the success of single-agent Q-learning, Tham and Renaud (2005) employed Q-learning to solve multi-agent problems in WSN. Based on cooperative game, their results showed that Q-learning can find the best policy for each agent to improve the performance by maximizing their network lifetime. Liang et al. (2008) applied MARL in overlay WSNs to obtain a good routing protocol. A MARL based routing protocol with QoS support for WSN called *MRL-QRP* was proposed. Sensor nodes can compute routes satisfying certain QoS requirements using a distributed reinforcement learning algorithm based on

Q-learning. However, each agent in these researches obtained the best policy with respect to its own benefits without considering the joint benefit of the other agents. As a result, the policy achieved may not be the best mutual policy for all agents.

On the other hand, (Hu and Wellman, 2003) considered joint benefits of other agents in the system. They introduced a MARL algorithm called Nash Q-learning (NashQ) to extend Q-learning to the context of non-cooperative multiple agents (i.e. authorities). NashQ uses the framework of a general sum stochastic game, whereby each agent's reward depends on the joint action of all agents and the current state. The agent attempts to learn its equilibrium Q-values, which are defined by Q-values received in Nash equilibrium. Moreover, the agent not only learns to find its own optimal policy, but it also learns actions and rewards of the other agent to find the other agent's optimal strategy. Therefore, each agent acts rationally with respect to this expectation and eventually fairness can be achieved.

Song et al. (2007) proposed another algorithm for non-cooperative multiple agents called Pareto Q-learning (ParetoQ). Although agents in ParetoQ undergo learning process similar to NashQ, ParetoQ allows agents to converge to Pareto optimum (Deb, 1999) by online learning**.** Pareto optimality is a set of strategies which an agent cannot increase its utility without decreasing the utility of at least one other agent. The advantage of ParetoQ is its simplicity for finding the optimal point in each stage game. Moreover, it guarantees that every game must have at least one such optimal point under a pure strategy. Finally, for games with multiple Pareto optimal strategies, the selection of an equilibrium can follow a lexicographic convention (Song et al., 2007). Therefore, ParetoQ can rationally select a unique Pareto optimal strategy for all agents.

In this thesis, we applied multi-agent Q-learning algorithms which are NashQ and ParetoQ algorithms to a packet forwarding problem in non-cooperative overlay wireless sensor networks. The underlying objective of this thesis is to propose an algorithm to achieve the best mutual policy and improve the network performance between two different agents belonging to non-cooperative multi-agent WSNs. Both NashQ and ParetoQ algorithms were applied to decide a suitable course of action for the agents in the packet forwarding game by learning online without exhaustive search over the strategy space. This thesis also studied the equilibrium conditions of the packet forwarding strategies in an overlay WSN and proposed fair resource allocation schemes from an online learning process.

To conclude, the main contributions of this thesis are three-fold:

1) The application of NashQ and ParetoQ algorithms to achieve the best mutual packet forwarding strategy in non-cooperative overlay WSNs with the MDP formulation.

2) The extension of ParetoQ algorithm from the MDP formulation to the radio energy model.

3) The incorporation of Pareto optimality with the Non-cooperative game algorithm and its application to the packet forwarding game.

1.2 Research objectives

1.2.1 To study optimal resource allocation schemes in overlay wireless sensor networks using multi-agent reinforcement learning.

1.2.2 To study the tradeoff between the computational time to obtain a packet forwarding strategy and the network performance (the successful packet delivery ratio, the energy consumption, the average utility of both agents and fairness).

1.3 Assumptions

1.3.1 Cooperative packet forwarding is beneficial when the network is sparse or when the environment is hostile.

1.3.2 Multi-agent Q-learning provides an equilibrium more efficiently than the Non-cooperative game approach.

1.3.3 A sensor node in overlay WSNs can communicate with each other using the same underlying protocol.

1.4 Scope of the Research

The experiment was separated into two parts. In the first part, the packet forwarding problem in non-cooperative overlay WSNs was formulated as a MDP and was solved with NashQ reinforcement learning. The performance of NashQ was compared with an existing algorithm called the Non-cooperative game approach (Felegyhaz et al., 2005). To evaluate the performance, we investigated the equilibrium conditions of the packet forwarding strategies. Results from this experiment suggested that the theoretical convergence conditions for the NashQ algorithm may be relaxed. We then proceed to study another MARL mechanism that provides a pure strategy to facilitate the conjecture of the other agent's action, and is still guaranteed to converge.

The second part therefore investigated and evaluated the performance of such mechanism so called the ParetoQ algorithm. The packet forwarding problem was formulated as a MDP and solved by the ParetoQ algorithm. The experiments were

conducted and the same metrics were measured and compared with the same sensor selection scheme as in part one. Moreover, we extended the study from the MDP formulation to a more realistic scenario by employing the radio energy regime (Naruephiphat and Usaha, 2008). Under this scenario, we numerically evaluated the following performance metrics, the number of time steps required to obtain a strategy, the average successful packet delivery rate, the average energy consumption in the packet forwarding process and the average utility of the agents. Results were compared with the Non-cooperative game algorithm.

1.5 Expected Usefulness

1.5.1 To conceptually show that the multi-agent Q-learning algorithm can be applied to find the best mutual policy for packet forwarding in non-cooperative overlay WSNs.

1.5.2 To obtain an optimal and fair resource allocation strategy for non-cooperative overlay wireless sensor networks.

1.5.3 To reduce computational time for determining the best mutual strategy by learning a strategy online.

1.6 Synopsis of Thesis

The remainder of this thesis is organized as follows. **Chapter 2** presents the theoretical background which is the foundation for the contributions of this thesis. Firstly, the concept of the Markov decision process formulation is reviewed. Next, existing tools used for solving the packet forwarding problem called NashQ and ParetoQ algorithms are introduced.

Chapter 3 studied the resource allocation problem in non-cooperative overlay WSNs. The packet forwarding game was formulated as a MDP and solved by the NashQ algorithm.

Chapter 4 studied and evaluated the performance of another multi-agent Q-learning called the ParetoQ algorithm in non-cooperative overlay WSNs. The packet forwarding game was formulated as MDP and solved by the ParetoQ algorithm. Moreover, this chapter extended from the MDP formulation to a more realistic scenario by employing the radio energy model.

Finally, **chapter 5** summarizes all the findings contribution in this thesis and points out possible future research directions.

CHAPTER II

BACKGROUND THEORY

2.1 Introduction

This thesis studies the resource allocation problem in non-cooperative overlay wireless sensor networks (WSNs). Typically, overlay WSNs include a large number of nodes that are deployed in the same area which are controlled by different authorities. The networks perform different tasks and measure different data within the same area. The most important usage for overlay WSNs is resource sharing between different authorities which can prolong their lifetime. The main reason is because intermediate nodes from other network authorities may help shorten the data transmission distance between neighboring nodes. When any authorities share their sensor nodes with the same goal of sending packets via a shorter distance, energy consumption is lowered; the networks can save their energy and eventually prolong their lifetime. However, cooperative behavior between sensor nodes belonging to authorities may not always be readily available because sensor nodes may act selfishly to conserve their energy. Furthermore, there is no guarantee that node cooperation will be beneficial to all WSNs. Therefore, it is necessary to find an algorithm (s) for each authority to decide whether to cooperate with each other or not in a non-cooperative problem in overlay WSNs.

This thesis proposed the application of reinforcement learning (RL) to address the issue of non-cooperative resource allocation problem in overlay WSNs. Reinforcement learning (Sutton and Barto, 1998) is a machine learning scheme to

provide a framework in which an agent learn the optimal policy based on the agents' past experiences without full information about the model of the environment. RL relies on the assumption that the dynamics of the system satisfies a Markov Decision Process (MDP). RL has been used to encourage cooperation between sensor nodes in a WSN where all sensor nodes belong to a single network entity (Pandana et al., 2005; Shahl et al., 2008). To cater multiple network authority frameworks such as overlay WSNs, the concept of multi-agent reinforcement learning (MARL) can be used, which integrates together RL, game theory, and direct policy search techniques.

Multi-agent systems differ from single-agent systems in that there are many different agents that are supposed to learn a task and that all of the agents' actions affect the environment. Thus, the optimal policy does not rely on only one agent, but rather it conditions on all agents. There are works which directly applied Q-learning to multi-agent systems where an individual agent maximizes its own benefit. By doing so, it neglects the presence of the other agents. As a result, this may well lead to suboptimal decisions. Therefore, an individual agent should take account of the effect of joint actions as a more suitable strategy for multi-agent system. The desired convergence in multi-agent system is based on an equilibrium strategy profile (Sen et al., 2008) i.e., a collection of strategies of the agents, rather than optimal strategies for an individual agent since all of the agents' actions affect the environment. To solve resource allocation in multi-agent systems, (Hu and Wellman, 2003); (Song et al., 2007) extended Q-learning to a non-cooperative multiple agent framework with guaranteed convergence. Hu and Wellman (2003) considered joint benefits of other agents in the system by introducing a MARL algorithm called Nash Q-learning (NashQ). In this algorithm, the agent attempts to learn its Nash equilibrium Q-values

online. Song et al. (2007) proposed another algorithm for non-cooperative multiple agents called Pareto Q-learning (ParetoQ), which allows agents to converge to Pareto optimum (Deb, 1999) by online learning**.** In this thesis, both algorithms were used to determine the best mutual policy in overlay WSNs.

Therefore, this chapter serves as an introductory to the fundamental theory of reinforcement learning which is the basis of the contribution of this thesis. The next section provides a theoretical background on Markov decision process (MDP) theory. A description of reinforcement learning is given in section 2.3. Section 2.4 presents the multi-agent Q-learning and a summary is presented in the final section.

2.2 Markov decision process theory

A Markov decision process (MDP) is a model of a decision-maker interacting synchronously with the environment. If the decision-maker sees the environment's true state, it is referred as a *completely observable Markov decision process*. The foundation of Markov decision process is presented as follows.

2.2.1 Markov property

The Markov property states that anything that has happened so far can be summarized by the current state. Thus, the probability of being in the next state at time t+1 based on the past history of state changes can be defined simply as the conditional probability based on the current state at time t,

$$P(S_{t+1} = s_{t+1} \mid S_t = s_t, ..., S_0 = s_0) = P(S_{t+1} = s_{t+1} \mid S_t = s_t). \tag{2.1}$$

This equation is referred to as the Markov property. A state refers to information on the environment that may be useful in making a decision. If the state

has the Markov property, then the environment's state at time t+1 depends only on the state representation at time t.

2.2.2 Markov Decision Process

A MDP is a discrete-time random decision process defined by a set of states, actions and the one-step dynamics of environment. Given any state s and action a, the probability of occurrence of each possible next state s' is

$$P(s' \mid s, a) = P(S_{t+1} = s' \mid S_t = s, a_t = a). \tag{2.2}$$

This equation is called the state transition probability. Similarly, given any current state and action, s and a, together with any next state, s', the expected value of the incurred reward is

$$R(s, a, s') = E[r_{t+1} \mid S_t = s, a = a, S_{t+1} = s'], \tag{2.3}$$

where $E[\cdot]$ is the expectation operator and r_{t+1} is the reward received at time t +1. Equation (2.2) and (2.3), completely specify the most important aspects of the dynamics of the MDP. A MDP model can be shown in Fig. 2.1.

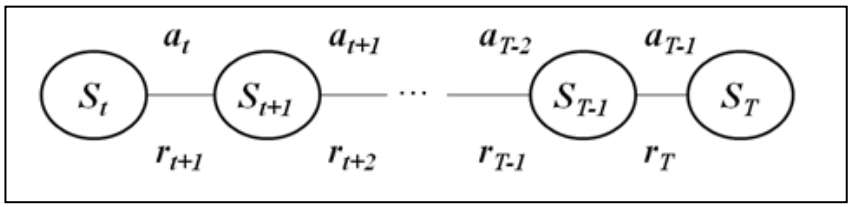

Figure 2.1 A MDP model.

A tuple (*S, A, P, R*) can describe the MDP characteristics, where *S* denotes the set of states, *A* denotes the set of possible actions, *P* is the state transition probability matrix. *R* is a function of the reward expected from the environment as a result of taking action $a \in A$. Let $P(s' \mid s, a) \in P$ be the state transition model that explains the probability of transiting to the next state $s' \in S$ after an agent takes action $a \in A$ at the current state $s \in S$.

The objective of solving a MDP is to find a policy, π, defined as a mapping of the state space to the action space, $\pi : S \to P[A]$, where *P[A]* is the distribution over the action space. The action-value function $Q_t^{\pi}(s,a)$ of a given policy π associates all state-action pairs (s,a) with an expected reward for performing action *a* in state *s* at time step *t* and following π thereafter;

$$\begin{aligned} Q^{\pi}(s,a) &= E^{\pi}[R_t \mid s_t = s, a_t = a] \\ &= E^{\pi}[\sum_{k=0}^{\infty} \beta r_{t+k+1} \mid s_t = s, a_t = a] \end{aligned} \tag{2.4}$$

where $R_t = r_{t+1} + \beta r_{t+2} + \beta^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \beta^k r_{t+k+1}$ is the expected discounted return of the agent, β is the discount factor and $E^{\pi}[\cdot]$ is the expectation operator under policy π.

The objective of MDP is to find a policy to select actions at a given state such that the long term average reward is maximized. To achieve this, particularly in scenarios where the dynamics of the environment is difficult to model (such as in WSNs), a technique called reinforcement learning can be used to solve MDPs.

2.3 Reinforcement learning

Reinforcement learning (RL) is a computational approach which identifies how a system in a dynamic environment can learn to choose optimal actions to achieve a particular goal. The learner is not taught which action to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trial-and-error interactions with its environment (Sutton and Barto, 1998).

In RL model, the learner or decision maker is called the agent. Everything outside the agent is called environment. It uses a formal framework defining the interaction between a learning agent and its environment in terms of states (s_t), actions (a_t) and rewards (r_t). The agent selects actions and the environment responds to those actions. Furthermore, the environment also feed backs to the agent rewards, as a consequence of the action selection at a given state, which the agent tries to maximize over time. More specifically, the agent and environment interact with each other in a sequence of discrete time steps. At each time step (t), the agent receives some representation of the environment's state (s_t) and selects an action (a_t). One time step later, the agent receives a numerical reward (r_{t+1}) and finds itself in a new state (s_{t+1}). Figure 2.2 shows the agent-environment interaction in reinforcement learning.

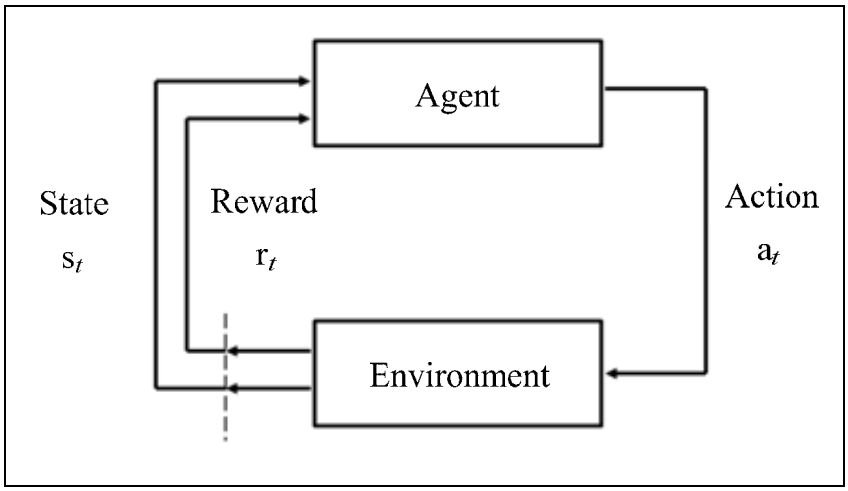

Figure 2.2 Diagram of agent-environment interaction in reinforcement learning.

2.3.1 The value function

Reinforcement learning algorithms are based on estimating value functions. A value function is the expected sum of rewards received from starting in state *s*. The value functions evaluate the performance of the decision which the learner has taken at a given state. Since the rewards received in the future by the learner depend on the actions which are taken, value functions are defined with respect to each particular policy. Therefore, we can define the value function of a state under a policy π, $V^{\pi}(s)$ is

$$V^{\pi}(s) = E^{\pi}[R_t \mid s_t = s]$$

$$= E^{\pi}[\sum_{k=0}^{\infty} \beta^k r_{t+k+1} \mid s_t = s] \qquad (2.5)$$

Similarly, the action-value function $Q_t^{\pi}(s,a)$ of a given policy π associates all state-action pairs (s,a) with an expected reward for performing action *a* in state *s* at time step *t* and following π thereafter;

$$\begin{aligned} Q^{\pi}(s,a) &= E^{\pi}[R_t \mid s_t = s, a_t = a] \\ &= E^{\pi}[\sum_{k=0}^{\infty} \beta r_{t+k+1} \mid s_t = s, a_t = a] \end{aligned} \tag{2.6}$$

2.3.2 The optimal value function

The aim of solving a MDP is to find an optimal policy that achieves the maximum reward over the long run. The optimal state-value function, denoted as $V^*(s)$,would therefore be state value function over all possible policies, at state *s* that is the maximum.

$$V^*(s) = \max_{\pi} V^{\pi}(s), \tag{2.7}$$

for all $s \in S$.

Optimal policies also share the same optimal action-value function, denoted $Q^*(s)$, and defined by

$$Q^*(s) = \max_{\pi} Q^{\pi}(s,a), \tag{2.8}$$

The standard solution to the problem above is through an iterative search method (Puterman, 1994) that searches for a fixed point of the following *Bellman* equation:

$$V^*(s) = \max_a \left\{ R_t + \beta \sum_{s'} P(s' \mid s, a) V^{\pi}(s') \right\}, \tag{2.9}$$

The equation (2.9) is a form of the Bellman optimality equation for $V^*(s)$. The Bellman optimality equation for $Q^*(s)$ is

$$Q^*(s) = R_t + \beta \sum_{s'} P(s' \mid s, a) \max_{a'} Q^*(s', a') \tag{2.10}$$

2.3.3 Q-learning

Q-learning (Sutton and Barto, 1998) defines a learning method within a MDP that is employed in single-agent RL systems. Q-learning is an algorithm that does not need a model of the environment and can directly approximate the optimal action-value function (Q-value) through online learning.

In Q-learning process, the agent starts with an arbitrary initial Q-value at time step 0. After executing action *a* at state *s*, the agent will receive an immediate reward *r* and then transits to a new state and updates the new Q-value with the input from the environment. The update rule at time step *t+1* of the Q-value is given by:

$$Q_{t+1}(s, a) = (1 - \alpha) Q_t(s, a) + \alpha_t [r + \beta \max_{a'} Q_t(s', a')], \tag{2.11}$$

where $\alpha_t \in [0,1)$ is the learning rate. The process is repeated so that the agent can learn its own optimal policy. Note that the Q value in equation (2.4) can converge to $Q^*(s, a)$ under the assumption that all states and actions have been visited infinitely often.

2.4 Multiple agent Q-learning algorithm

Multi-agent systems differ from single-agent systems in that there are many different agents that are supposed to learn a task and that all of the agents' actions affect the environment. Thus, the optimal policy does not rely on only one agent, but rather it conditions on all agents. There are works which directly applied Q-learning to multi-agent systems where an individual agent maximizes its own benefit. By doing so, it neglects the presence of the other agents. As a result, this may well lead to suboptimal decisions. Therefore, an individual agent should take account of the effect of joint actions as a more suitable strategy for multi-agent system.

2.4.1 Nash Q-learning algorithm

Hu and Wellman (2003) proposed the Nash Q-learning (NashQ) algorithm, by extending Q-learning to a non-cooperative situation where each agent can rationally decide its action whether it will cooperate with other agents or not by considering both its own and other agents' information as well.

2.4.1.1 The action-value function

Instead of finding an optimal action to maximize one single agent's reward as the single-agent Q-learning, NashQ seeks joint actions that yield the best possible reward for all agents. For a two-agent system, the action-value function for any agent becomes $Q^i(s, a^1, a^2)$, where $i = 1, 2$.

The objective of the agents in the NashQ algorithm is to learn their best mutual response policy, which is defined by the Q-values received from Nash equilibrium (NE). NE is not only used to decide the agent's own action policy, but also predict the other agent's action, given by $\pi^1(s'), \pi^2(s')$ where $\pi^i(s')$ is the agent i's distribution over the set of actions at state s'. NashQ then calculates a NE for

the stage game $(Q_t^1(s'), Q_t^2(s'))$ and updates its Q-values according to

$$Q_{t+1}^i(s, a^1, a^2) = (1-\alpha_t)Q_t^i(s, a^1, a^2) + \alpha_t[r_t^i + \beta NashQ_t^i(s')], \tag{2.13}$$

where

$$NashQ_t^i(s') = \pi^1(s') \cdot Q_t^i(s') \cdot \pi^2(s') \tag{2.14}$$

In order to calculate the Nash equilibrium, agent *i* must observe the other agent's immediate reward and previous actions and updates its conjectures on the other agent's Q-value, by maintaining its own update on the other agent's Q-value:

$$Q_{t+1}^j(s, a^1, a^2) = (1-\alpha_t)Q_t^j(s, a^1, a^2) + \alpha_t[r_t^j + \beta NashQ_t^j(s')], \quad j \neq i. \tag{2.15}$$

NE can be found in a pure-strategy equilibrium, where an agent is able to achieve the best response to the other agent's choice. However, not all games have pure-strategy equilibrium (Daskalakis et al., 2009). Under this circumstance, the agents need to select their strategies randomly according to some probability calculated from the Lemke-Howson method (Shoham and Brown, 2009) to achieve the NE. Such equilibrium is called mixed-strategy equilibrium.

Table 2.1 The Nash Q-learning algorithm.

Initialize:

Let $t = 0$, get the initial state s_0.

Let the learning agent be indexed by i.

For all $s \in S$ and $a^i \in A^i, i = 1,2$, let $Q_t^i(s, a^1, a^2) = 0$.

Loop

Choose action a_t^i.

Observe $r_t^1, r_t^2; a_t^1, a_t^2$, and $s_{t+1} = s'$

Update Q_{t+1}^i for i=1,2

$$Q_{t+1}^i(s, a^1, a^2) = (1-\alpha_t)Q_t^i(s, a^1, a^2) + \alpha_t[r_t^i + \beta NashQ_t^i(s')],$$

where $\alpha_t \in [0,1)$ is the learning rate, and $NashQ_t^i(s')$ is defined in (2.14)

Let $t := t + 1$.

2.4.1.2 Convergence

NashQ requires two conditions in a stage game during learning to converge (Hu and Wellman, 2003).

1) The stage games encountered during learning have a global optimal point, which is defined as a point of joint strategy in the stage game which every agent receives its highest payoff at this point, or

2) They all have a saddle point which is defined as a point of joint strategy in the stage game which is a NE point, and each agent would receive a higher payoff when at least one of the other agents deviates.

However, both the global and saddle points are hardly satisfied for these conditions because of both points may not exist in every stage game. Another limitation is that in selecting NE under a mixed strategy (when multiple NE exist), NashQ algorithm resorted to a mixed strategy selection where the Nash equilibrium probabilistically selected according to the Lemke Howson method (Shoham and Brown,

2009). They algorithm showed that convergence can still be established with such relaxed convergence conditions.

2.4.2 Pareto Q-learning algorithm

Song et al. (2007) proposed another algorithm called, Pareto Q-learning algorithm (ParetoQ). Instead of finding the best mutual policies from NE, ParetoQ obtains equilibrium from Pareto optimality.

2.4.2.1 Pareto optimality

One method for identifying the desired equilibrium point in a game is to compare strategy profiles using the concept of Pareto-optimality. To introduce this concept, let us first define Pareto-superiority.

*Pareto-superiority***, t**he strategy profile ξ is Pareto-superior to the strategy profile ξ' if for any agent *i*,

$$u_i(\xi_i, \xi_{-i}) \geq u_i(\xi_i', \xi_{-i}'). \tag{2.16}$$

In other words, the strategy profile ξ is *Pareto-superior* to the strategy profile ξ', if the utility of an agent *i* can be increased by changing from ξ' to ξ without decreasing the utility of other players.

Pareto optimal, the strategy profile ξ is a Pareto optimal if there does not exist another strategy profile that is Pareto-superior to ξ. Meaning that, in a Pareto optimal strategy profile, one cannot increase the utility of player *i* without decreasing the utility of at least one other player.

The advantage of Pareto optimality is that every game must have at least one such optimal pure strategy. Meaning that, the agents can decide with

certainty which action to take such that the best mutual policy is achieved. ParetoQ algorithm can converge if the agents choose their strategies and update their Q-value by using global optimal point, or Pareto optimal, or alternatively, a saddle point, which are encountered in stage games during learning. The Q-values in the ParetoQ algorithm can be update using the following rule: Global NE > Pareto optimal > Saddle NE. Furthermore, when the stage games encounter multiple Pareto optimal points, agents can choose their strategies and update their Q-value according to a lexicographic convention.

2.4.2.2 Lexicographic convention

It is possible that multiple Pareto optimum equilibriums exist in a game. To enable each agent to correctly predict the other agent's action, a lexicographic convention (Song et al, 2007) can be established as follows.

1) The set of agents is ordered.

2) The set of each agent's action is ordered.

3) The set of different solutions are ordering.

(Global NE > Pareto optimal > Saddle NE)

4) Both agents must agree to use the same series of convention.

The choice for an optimal joint action proceeds as follows. The first agent in the agent ordering chooses an optimal action (that corresponds to a Global NE) that appears first in its action ordering. The next agent then chooses its first optimal action in its action ordering given the first agent's choice. This procedure continues until all agents have chosen their actions. An example of lexicographic convention can be shown in Figure 2.3.

Game 1	Left	Right	Game 2	Left	Right	Game 3	Left	Right
Up	10,9	0,3	Up	5,5	0,6	Up	10,9	0,3
Down	3,0	-1,2	Down	6,0	2,2	Down	3,0	2,2

Figure 2.3 Three types of general-sum game.

In each stage game, agent 1 has two action choices: *Up* and *Down*. agent 2's action choices are *left* and *Right*.

The first game has only one Nash equilibrium, with values (10,9), which is a global NE point.

The second game also has a unique Nash equilibrium; in this case a saddle point, valued at (2,2). But there still is a Pareto dominating solution, valued at (5,5), which will be better for the both players. In this case, the strategy profile (5,5) will be selected.

The third game has two Nash equilibriums including a global optimum, (10,9), and a saddle, (2,2). In this case, the strategy profile (10,9) will be selected.

By using this convention, both agents are able to uniquely select their joint actions and the convergence of the ParetoQ algorithm is achieved. The advantage of ParetoQ algorithm over the Non-cooperative game approach is that it can converge by learning from the agents' past experiences online, rather than the offline exhaustive search. Furthermore, unlike that the NashQ algorithm, each agent in the ParetoQ algorithm can predict the other agents' actions with certainty such that the evaluation of the other agents' Q value (the information necessary for guessing the other agent's action) can also be accurately carried out.

2.5 Summary

In this chapter, overviews of the multi-agent Q-learning algorithm called NashQ and ParetoQ are given. Both algorithms were used to determine the packet forwarding strategies in non-cooperative overlay WSNs in this thesis. By considering joint actions, the agents can rationally determine the best mutual policy and receive fair benefit for all agents in overlay WSNs.

In the next chapter, a packet forwarding formulation in non-cooperative multi-agent WSNs is presented. Nash Q-learning is used to study the conditions which equilibriums can exist and its performance is evaluated under a MDP formulated environment.

CHAPTER III

PACKET FORWARDING IN OVERLAY WIRELESS SENSOR NETWORKS : NASH Q-LEARNING

3.1 Introduction

Wireless sensor networks (WSNs) coexisting independently within a region of interest without conflicting each other are called overlay WSNs. Such networks include a large number of nodes that are deployed in the same area which are controlled by different authorities. The most important usage for overlay WSNs is resource sharing between different authorities which can prolong their lifetime. The main reason is because intermediate nodes from other network authorities may help shorten the data transmission distance between neighbouring nodes. When two authorities share their sensor nodes with the same goal of sending packets via a shorter distance, energy consumption is lowered; the networks can save their energy and eventually prolong their lifetime.

Currently, many research related to resource allocation problems in overlay WSNs have been proposed with a focus on saving energy (Murase et al., 2006); (Mao et al., 2008); (Wang et al., 2007), fault tolerance problem (Chitnis et al., 2009); (Han et al., 2010), node deployment problem (Yu et al., 2007), target detection problem (Li et al., 2006). All of these works showed that two authorities cooperatively sharing resources results in reduced energy consumption and increased network performance. However, cooperative behaviour between sensor nodes belonging to authorities may not always be readily available. Sensor nodes not only are deployed independently

without any fixed strategy, but they may also act selfishly to conserve their energy. Furthermore, there is no guarantee that node cooperation will be beneficial to both WSNs. It may even be possible that cooperation may lead to benefits for a single party alone or no benefit to any party at all. In (Vaz et al., 2008) showed that cooperation between two authorities in the same area may not always be beneficial to any network, because whether or not each authority will cooperate depends on the configuration of each network. It is therefore necessary to find an algorithm (s) for each authority to decide whether to cooperate with each other or not.

Recently, game theory has become a promising tool to analyze and improve the performance of sensor networks. The major advancement that has driven much of the development of game theory is Nash equilibrium. Nash equilibrium is a collection of strategies for each of the players such that each player's strategy is the best-response to the other players' strategies. In (Felegyhazi et al., 2005), the authors addressed the problem of whether cooperation can exist in WSNs without incentive mechanisms. In (Wu and Shu, 2008) and (Miller et al., 2005), incentive mechanisms were used to motivate cooperation between sensor nodes.

In this chapter, we apply an existing algorithm called Nash Q-learning (NashQ) (Wu and Wellman, 2003) to a packet forwarding problem in non-cooperative overlay wireless sensor networks. NashQ uses the framework of a general sum stochastic game, whereby each agent's reward depends on the joint action of all agents and the current state. The agent attempts to learn its Nash equilibrium online. Moreover, the agent not only learns to find its own policy, but it also learns actions and rewards of the other agent to find the other agent's optimal strategy. Therefore, each agent acts rationally with respect to this expectation.

Therefore, the underlying aim of this chapter is to find the best mutual policy for all agents in non-cooperative overlay WSNs using Nash Q-learning (NashQ). Without full information, the agent not only learns its own policy, but it also observes the behavior of the other agent in order to evaluate their other policy for all agents. Hence, the best mutual policy is assumed to hold.

The emphasis of this chapter is focused on the following issues:

1. The formulation of packet forwarding game in non-cooperative overlay WSNs.

2. The MDP formulation for Nash Q-learning algorithm.

3. The comparison of algorithm performance between the Nash Q-learning algorithm and an existing algorithm called the Non-cooperative game algorithm.

To the best of our knowledge, NashQ has not been applied for resource allocation in overlay WSNs before. Therefore, the contribution of this chapter is two-fold: 1) the MDP formulation of the packet forwarding problem in a non-cooperative multi-agent WSN and, 2) the application of NashQ to solve for the best mutual policy in a multi-agent WSN.

This chapter is organized as follows. Learning in non-cooperative games will be described in section 3.2. Section 3.3 is dedicated to describing the problem formulation. Section 3.4 presents the experimental results and discussion. Finally, section 3.5 summarizes the entire chapter.

3.2 Learning in non-cooperative games

Reinforcement learning (RL) (Sutton and Barto, 1998) is a machine learning scheme which is used to learn the optimal action based on the agents' past experiences

without full information about prior model of the environment. RL relies on the assumption that the dynamics of the system follows a Markov Decision Process (MDP). The tuple (*S,A,P,R*) is used to describe the characteristics of MDP, where *S* is the set of states, *A* is the set of actions, *P* is the state transition probability matrix. Let $P(s' \mid s, a) \in P$ the probability of transiting to the next state s' from state *s* when an agent takes action *a*, and *R* is a function of reward expected from the environment after taking the action $a \in A$ at state $s \in S$ and transiting to s'. In MDP, the objective is to find a policy, π, defined as a mapping of the state set to the action set, $\pi : S \to P[A]$, where $P[A]$ is the distribution over the action space. The action-value function $Q_t^{\pi}(s, a)$ of a given policy π associates all state-action pairs (s,a) with an expected reward for performing action *a* in state *s* at time step *t* and following π thereafter;

$$\begin{aligned} Q^{\pi}(s, a) &= E^{\pi}[R_t \mid s_t = s, a_t = a] \\ &= E^{\pi}[\sum_{k=0}^{\infty} \beta^k r_{t+k+1} | s_t = s, a_t = a], \end{aligned} \tag{3.1}$$

where $R_t = r_{t+1} + \beta r_{t+2} + \beta^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \beta^k r_{t+k+1}$ is the expected discounted return of the agent, β is the discount factor and $E^{\pi}[\cdot]$ is the expectation operator. The objective of the agent in a MDP is to determine a policy to select actions so that the sum of the discounted rewards it receives over the future is maximized.

A well-known RL technique called Q-learning is employed to solve MDPs in single-agent systems. It is an algorithm that does not need a model of the environment

and can directly approximate the optimal action-value function (Q-value) through online learning. The agent takes action *a* at state *s*, receives a reward *r*, updates the local state with input from the environment, and repeats the process to learn its own optimal policy. Q-learning provides a simple procedure in which the agent starts with an arbitrary initial Q-value at time step 0. The update rule at time step *t+1* of Q-learning is given by:

$$Q_{t+1}(s,a) = (1-\alpha_t)Q_t(s,a) + \alpha_t[r + \beta \max_{a'} Q_t(s',a')] \quad , \tag{3.2}$$

where $\alpha_t \in [0,1)$ is the learning rate and s' is the next state that results from taking action *a* in state *s*. However, multi-agent systems differ from single-agent systems in that there are many different agents that are supposed to learn a task. The optimal policy does not rely on only one agent, but rather it depends on all agents. Several researches i.e. (Tham and Renaud, 2005); (Forster et al., 2007) employ Q-learning to solve multi-agent WSN problems. Their results show that Q-learning can improve the performance in their system based on cooperative game. However, cooperative behaviour between sensor nodes belonging to multiple agents may not always be available, particularly when sensor nodes are controlled by different agents. Hu and Wellman (2003) extended Q-learning to a non-cooperative multi-agent system where each agent can design its action whether it will cooperate with other agents or not. They proposed the Nash Q-learning (NashQ) algorithm, in which other agents' actions are taken into consideration as well. Instead of finding out which action to take to maximize their reward like the (single-agent) Q-learning, NashQ looks for joint actions that yield the best reward for all agents. The agents attempt to learn their best

mutual response policy, which is defined by the Q-values received from Nash equilibrium (NE). NE is not only used to decide the agent's own action policy, but also predict the other agent's action policy, given by $\pi^1(s'),...,\pi^n(s')$ where $\pi^i(s')$ is agent *i*'s distribution over its set of actions at state s' and *n* is the number of agents. NE can be found in a pure-strategy equilibrium, where an agent is able to find the highest utility, meaning that the agents can decide with absolute certainty which action to take so that the best mutual policy is achieved. But in general, not all games have pure-strategy Nash equilibrium (Daskalakis et al., 2009), so the agents have to decide to select their policies randomly according to some probability to achieve the best response. Such NE behaviour is called mixed-strategy Nash equilibriums. Any game is guaranteed to have a mixed-strategy NE (Nash, 1951). The Lemke-Howson method (LH), is the best known method to solve for mixed-strategy NE for two agents (Shoham and Brown, 2009). However, this method still has certain limitations in computing the mixed-strategy NE probability, because the sequence of variables entering the LH method affects the solution directly. Nevertheless, the advantage of LH method is that it is guaranteed to find at least one NE point. We thus employ the LH method in both the proposed NashQ and an existing non-cooperative game approach (Felegyhazi et al., 2005), (See Section 3.4 for details).

The convergence of NashQ is proved in (Hu and Wellman, 2003). They proved that NashQ can converge if (1) the stage games encountered during learning have a global optimal point, which is defined as a point of joint strategy in the stage game which every agent receives its highest payoff at this point; or, (2) they all have a saddle point which is defined as a point of joint strategy in the stage game which is a NE point, and each agent would receive a higher payoff when at least one of the

other agents deviates. The advantage of NashQ over other non-cooperative game theoretic approaches is that it can converge by online-learning from the agents' past experiences.

In this chapter, we thus apply NashQ algorithm in the problem of a packet forwarding problem in a non-cooperative multi-agent WSN in order to find the best mutual policy which provides the best benefit for all agents in the system.

3.3 Problem Formulation

In this section, we formulate the packet forwarding game in non-cooperative multi-agent WSNs. NashQ algorithm is then formally introduced in order to find the best mutual policy in multi-agent WSNs. In our game, the agent attempts to learn its equilibrium Q-values, which are defined by Q-values received in a NE. Moreover, the agent not only learns to find its own optimal policy, but it also learns actions and rewards of the other agent to find the other agent's optimal strategy. Therefore, each agent acts rationally with respect to this expectation and eventually the best mutual response can be achieved.

In our model, we assume that there are two different WSNs referred to as domains, co-existing in the same area. Each WSN is assumed to operate as a centralized system controlled by a cluster head acting as an agent for that particular domain. We assume that the cluster heads have no limit in energy (e.g. it may be equipped with a renewable battery) and can control the behaviour of the sensor nodes in its domain. Each agent maintains two different routing tables, one for routing within their own network and the other for coordinating paths with the other network. We assume that two sensor nodes are able to communicate with each other if they are

within transmission range. Even if they belong to a different network, interactions between the agents are ensured by the cluster head in their own network.

Each sensor node employs the radio model (Naruephiphat and Usaha, 2008) to compute the transmission and receiving cost required for transmitting a measurement packet. The reception cost of agent i is given by, $C_{i,RX}(b) = E_{elec} \times b,$ where $E_{elec} = 50nJ/bit$ is the expended cost in the radio electronics and we assume that $b = 250Kbits$ is the size of the measurement packet transmitted. The transmission cost of agent i is given by, $C_{i,TX}(b,d) = E_{elec} \times b + (\varepsilon_{amp} \times b \times d^{\sigma})$, where σ is the path loss exponent and $\varepsilon_{amp} = 10\ pJ/bit/m^{\sigma}$ is the energy consumed at the output transmitter antenna for transmitting one meter. We assumed that the agents send their *packets* to a base station (called *sink*) by either their own route or a coordinated route through the other network depending on actions selected by the agent.

The action space is defined by $A = \{DD, DF, AD, AF\}$ where the short hand notations refer to the following:

DD: The agent does not *ask* the other domain to forward packets and *drops* all packets from other domain if asked for help.

DF: The agent does not *ask* the other domain to forward packets but helps the other domain to *forward* all packets if asked for help.

AD: The agent asks the other domain to forward packets but *drops* all packets from the other domain if asked for help.

AF: The agent asks the other domain to forward packets and helps the other domain *forward* all packets if asked for help.

After taking an action, the agent *i* evaluates the proportion of delivered measurement packets to the sink at time *t* denoted by $p_i(t)$. If , $p_i(t) \geq SR_i$ where SR_i is the ratio of successfully delivered measurement packets required by agent *i*, then the action is successful and agent *i* will receive a gain at time *t*, $g_i(t) = G_i$, where $G_i >> 0$ in order to encourage either agent to send its packets to the sink. Otherwise, $g_i(t) = 0$. Agent *i* also incurs a cost in time step *t* denoted by $C_i(t) = C_{i,TX} + C_{i,RX}$, which is the total transmission and reception cost of all sensor nodes within the domain of agent *i*. The reward of agent *i* is then given by

$$r_i(t) = g_i(t) - C_i(t) . \tag{3.3}$$

We define the state space in our system as the set of all possible battery levels. In particular, the state space of each agent is divided into 3 states, i.e. high (*h*), medium (*m*) and low (*l*). The transition probability metric *P* for action *a* is defined by *P(a)*,

$$P(a) = \begin{bmatrix} p_{hh}(a) & p_{hm}(a) & p_{hl}(a) \\ p_{mh}(a) & p_{mm}(a) & p_{ml}(a) \\ p_{lh}(a) & p_{lm}(a) & p_{ll}(a) \end{bmatrix} = \begin{bmatrix} \phi & \delta & 1-\phi-\delta \\ 0 & \theta & 1-\theta \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.4}$$

where $p_{xy}(a)$ refers to the transition probability from state *x* to state *y* by taking action *a* and $\theta, \phi, \delta \in [0,1]$. Note that each row of *P(a)* must satisfy $\sum_{\forall y} p_{xy}(a) = 1$.

We divide the time into time units called time steps. We initialize $Q_0^1(s, a^1, a^2) = 0$ for all $s \in S$, $a^1 \in A^1, a^2 \in A^2$. Agent *i* attempts to learn its equilibrium Q-value, starting from an arbitrary guess. At time step *t*, agent *i* observes the current state, takes its action and observes its own reward. It then observes the action, reward at the other agent and observes the next state of both agents.

Agent *i* then calculates a NE $\pi^1(s'), \pi^2(s')$ for the stage game $(Q_t^1(s'), Q_t^2(s'))$ and updates its Q-values according to

$$Q_{t+1}^i(s, a^1, a^2) = (1 - \alpha_t) Q_t^i(s, a^1, a^2) + \alpha_t [r_t^i + \beta NashQ_t^i(s')], \tag{3.5}$$

where

$$NashQ_t^i(s') = \pi^1(s') \cdot Q_t^i(s') \cdot \pi^2(s'). \tag{3.6}$$

In order to calculate the Nash equilibrium, agent *i* must observe the other agent's immediate reward and previous actions and updates its conjectures on the other agent's Q-value, by maintaining its own update on the other agent's Q-value:

$$Q_{t+1}^j(s, a^1, a^2) = (1 - \alpha_t) Q_t^j(s, a^1, a^2) + \alpha_t [r_t^j + \beta NashQ_t^j(s')], \; j \neq i. \tag{3.7}$$

The process is repeated until the game ends when the battery level of either agent reaches “low” state, signifying a battery depletion of a sensor node in its domain.

In this chapter, we evaluated the performance of NashQ algorithm by comparing it with the Non-cooperative game algorithm in (Felegyhazi et al., 2005). The Non-cooperative game is a branch of game theory, applied exclusively to the situation which the interests of multiple agents conflict. A multi-stage game can be defined by the tuple (η,τ,μ), where η is the set of players, τ denotes the set of strategies and μ is a set of utility functions. There are two players (i.e. agents) in the game. The reward can be defined by (3.3). We define the *utility* as the cumulative rewards of the player, $\mu_i = \sum_{t=0}^{T} r_i(t)$, where T denotes the lifetime of the domain depending on the transition probability matrix in (3.4). While NashQ attempts to learn its policy though the Q-values obtained through NE in (3.5) and (3.6), the policies of the players in the Non-cooperative game algorithm are obtained by determining the NE from the utility functions.

In particular, to obtain the NE in the Non-cooperative game algorithm, the players first have to calculate utility functions for all possible actions $a^1 \in A^1, a^2 \in A^2$, and then choose their policies. Such method can be considered as an offline method. On the other hand, NashQ obtains its policy through online-learning, without having to compute the utility functions for all possible actions.

3.4 Experiment Results

In this section, we evaluate the proposed NashQ algorithm and investigate the equilibrium conditions of the packet forwarding strategies.

To implement the packet forwarding game, we consider two different WSNs co-existing in the same area of size $40x20m^2$. Each sensor node has the goal of maximizing its own packet delivery to a sink. In our simulation, we investigate two scenarios, i.e. the separate sink and the common sink scenarios. The simulation parameters are shown in Table 3.1.

Table 3.1 Simulation Parameter Values.

Parameter	Value
Number of sensors per domain	10-90
Distribution of the sensors	Uniformly random
Area size	$40x20\ m^2$
Path loss exponent, σ	2-5
Success requirement, SR_i	1
Positions of the common sink	[20,10]
Positions of the separate sink	[10,10] and[30,10]
Route selection	Minimum energy path

Simulation was performed for 100 runs under 100 randomly generated topologies. We compare the NashQ with the Non-cooperative game algorithm (Felegyhazi et al., 2005). For both NashQ and Non-cooperative game algorithm, the LH method is used to compute NE in (3.6). Three types of equilibrium can be observed as follows:

Defective equilibrium: the agent ends up its game at the move DD-DD.

Cooperative equilibrium: the agent ends up its game at the move AF-AF.

Other equilibrium: the agent ends up at other moves.

Figure 3.1a presents the proportion of simulation for each type of equilibrium by the NashQ algorithm in the common sink scenario. It can be seen that the proportion of cooperative equilibrium is higher than other types of equilibrium at low

sensor density but decreases as the density of sensor increases. This suggests that cooperation is required if the density of sensor is low. On the other hand, if the agents are provided several paths for their sensors to send packets to the sink, as in the case when sensor nodes are densely deployed in the area, cooperation between both agents is not necessary. This explains the reduction of cooperative behaviour and increase of defective equilibrium as the number of sensors per domain increases.

Figure 3.1b presents the proportion of simulation by the Non-cooperative game algorithm in the common sink scenario. Similar to NashQ, the Non-cooperative game algorithm increasingly favors the defective equilibrium as the density of sensor nodes increases. The results show that though both algorithms prefer the defective equilibrium in the common sink scenario, NashQ prefers it more than the other algorithm. For example, at 90 nodes per domain, the proportion of defective equilibrium from NashQ is 87% while that of the Non-cooperative game is only 75%. In addition, as the node density decreases, the proportion of cooperative equilibrium from NashQ is also greater than the Non-cooperative game. This suggests that NashQ tends to promote more cooperation when necessary and demotes it more than the Non-cooperative game, otherwise.

Figure 3.2a and 3.2b depict the proportion of simulation for each type of equilibrium in the separate sink scenario. With reasons similar to the common sink case, NashQ and the Non-cooperative game algorithms show that the cooperative equilibrium is gradually reduced as the network size increases. More interestingly is that the NashQ algorithm provides a more robust behaviour than the Non-cooperative game algorithm in the sense that NashQ shows a monotonic decrease of cooperative behaviour and monotonic increase of defective behaviour as the number of sensors per

domain increases. On the other hand, Non-cooperative game results are not monotonic as can be seen in Figure 3.2 b) at 50, 70 and 90 nodes per domain. It is caused by the limitation of LH algorithm when dealing with non-unique (i.e. multiple) NEs. As explained in Section III, the sequence of variables entering the LH algorithm directly affects the mixed-strategy NE probability. However, NashQ can avoid such condition whenever it is able to find the NE from either a global point or saddle point.

Figure 3.3a shows the proportion of simulation for each type of equilibrium as a function of path loss exponent in the common sink scenario by the NashQ algorithm. The higher the path loss exponent, the more difficult it is to send the packets to the sink. Hence, the figure shows that cooperative equilibrium is more beneficial than other equilibriums in a hostile environment.

Figure 3.3b shows the proportion of simulation by the Non-cooperative game algorithm, suggesting that cooperation between both agents is still the best strategy for all agents in a hostile environment in the separate sink scenario. Once again, it can be seen that NashQ obtains 20-50% more cooperative equilibrium than the Non-cooperative game as the path loss exponent increases. This is likely caused by the fact that the Non-cooperative game employs the LH method to solve for a mixed strategy NE, thereby is not as robust to non-unique NEs as NashQ which uses the global or saddle point for its NE whenever possible. NashQ thus outperforms the Non-cooperative game.

Figure 3.4 presents the convergence of the NashQ algorithm in terms of error rate in the common sink scenario. The error rate is the percentage of the absolute difference between the previous Q-values and the newly updated Q-values summed over all state-action pairs. The figure shows that the initial stage of learning has high

error rate but decreases steadily in the long run. This suggests that NashQ algorithm can be performed online starting from an arbitrary value and still converges as long as convergence conditions are met (Hu and Wellman, 2003). The Non-cooperative game algorithm, on the other hand, must be performed offline where an exhaustive search through all strategies is performed to determine the best mutual response strategy in each game. Therefore, once trained, NashQ appears to be less computational demanding and more adaptive to topological changes than the Non-cooperative game algorithm.

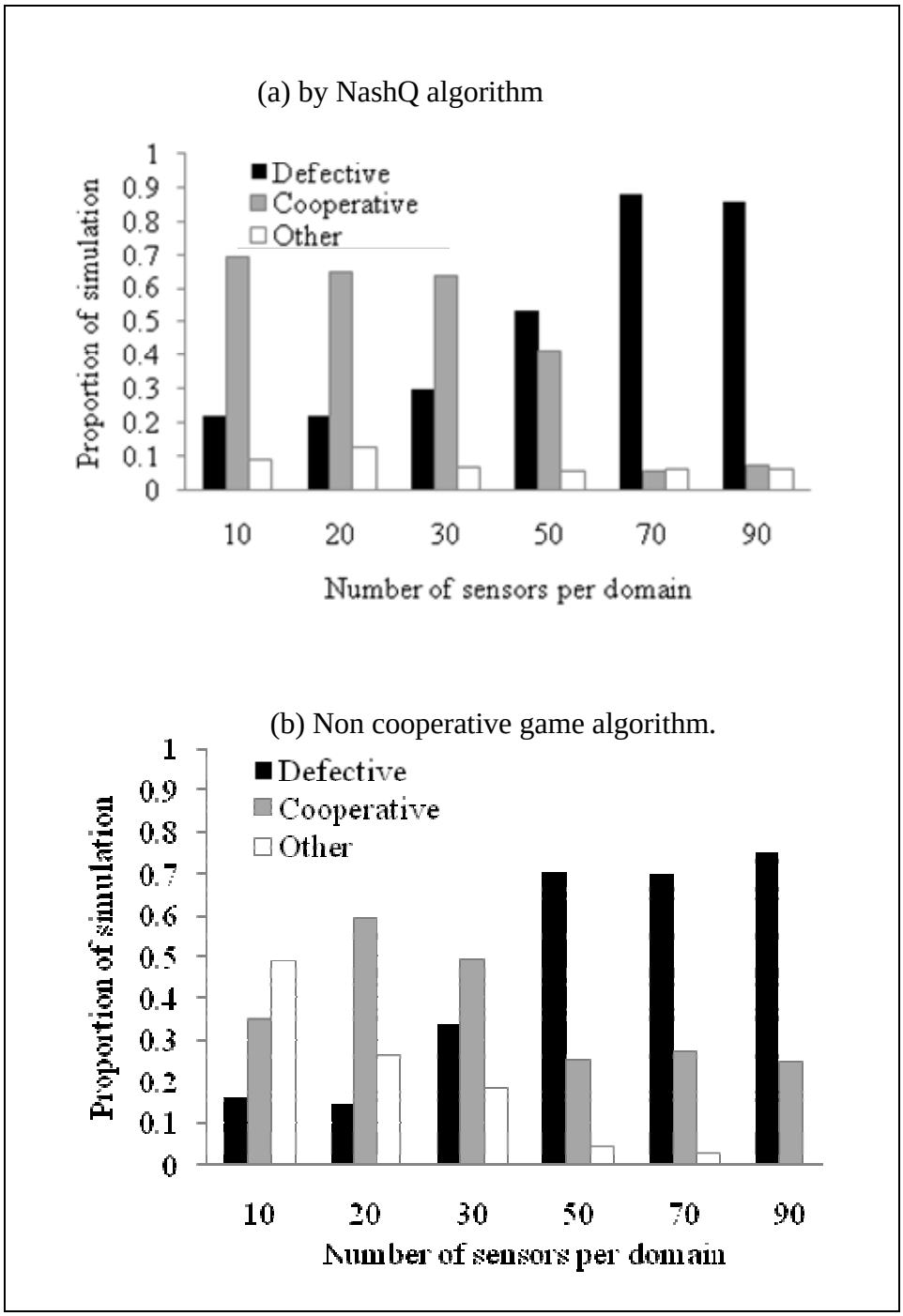

Figure 3.1 Effect of network size on cooperation in the common sink scenario

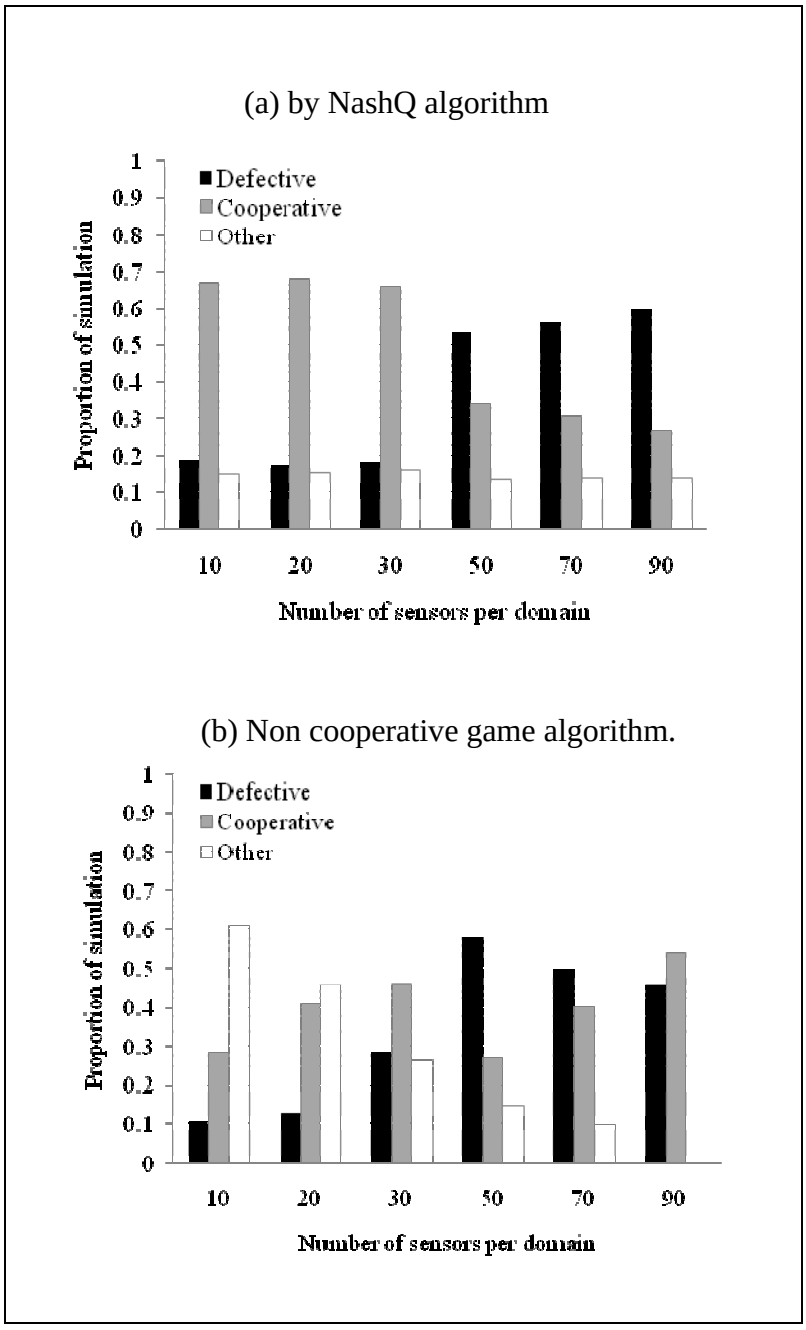

Figure 3.2 Effect of network size on cooperation in the separate sink scenario

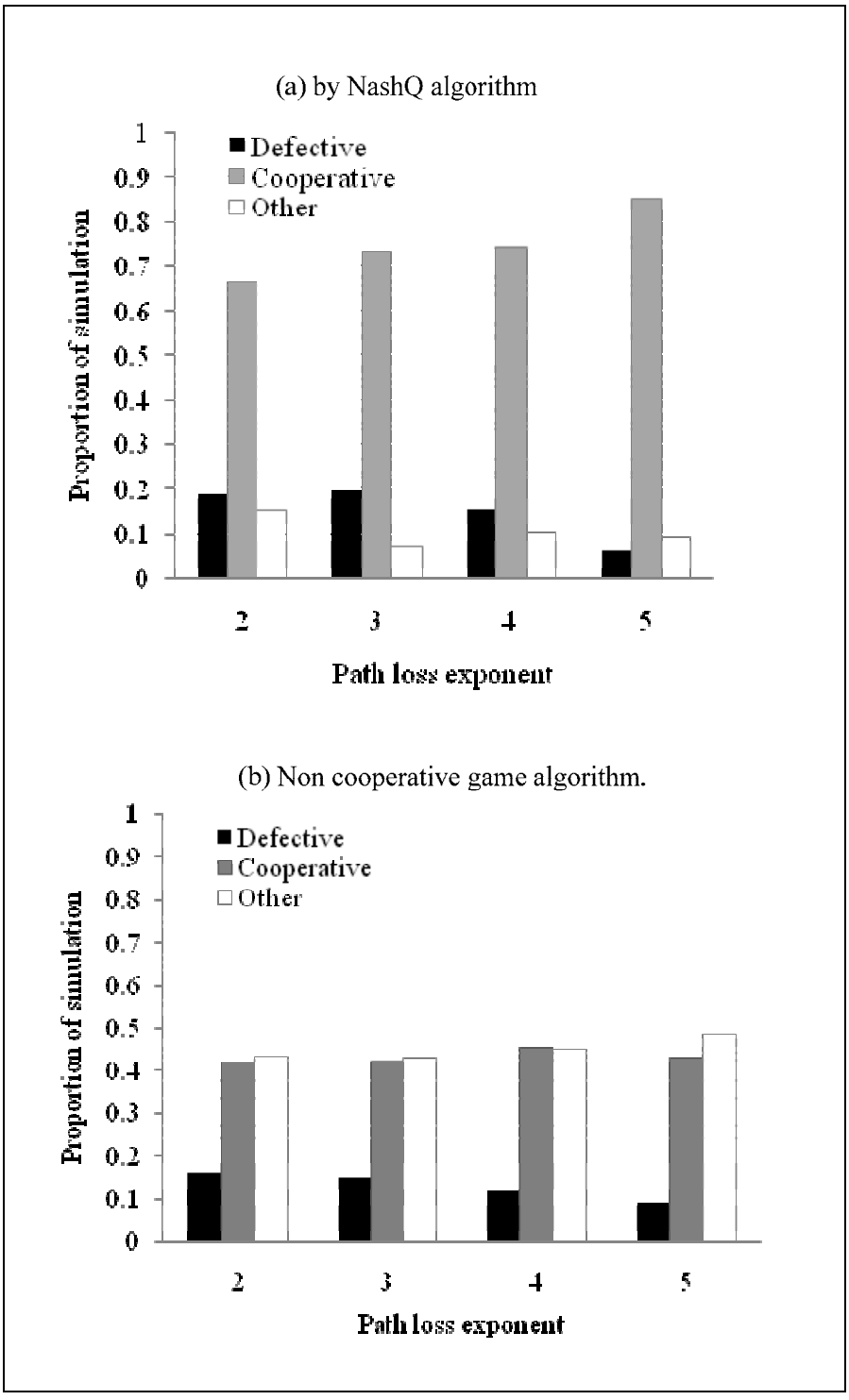

Figure 3.3 The effect of path loss exponent in the common sink scenario

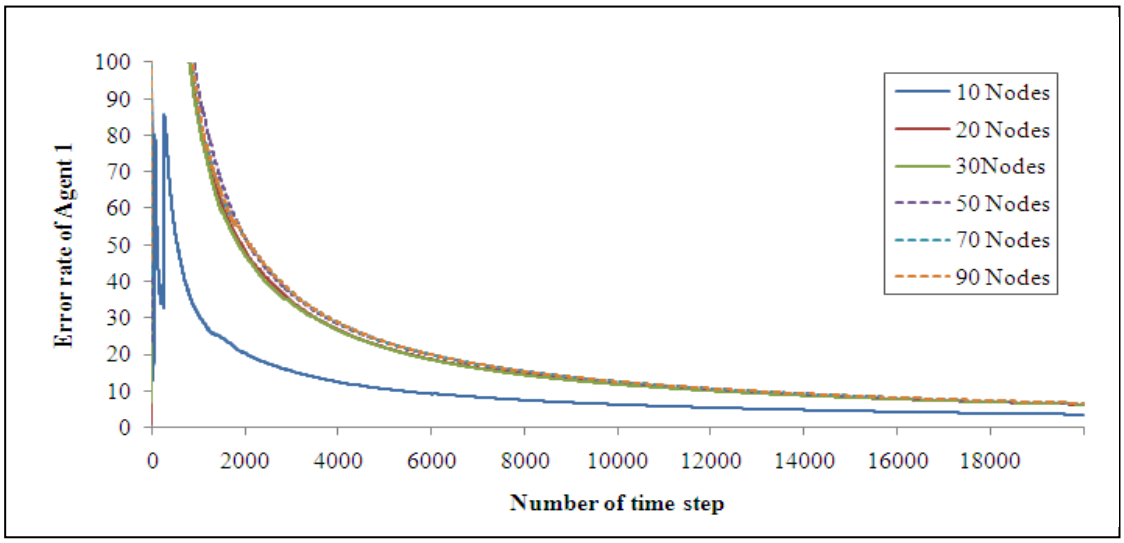

Figure 3.4 Convergence of NashQ algorithm in the common sink scenario.

3.5 Summary

In this chapter, we proposed a NashQ algorithm to determine suitable packet forwarding strategies in a multi-domain WSNs. The objective of this chapter is to conceptually show that the NashQ algorithm can be applied to promote cooperation in multi-agent in overlay WSNs. The results showed that based on Lemke-Howson method, NashQ can find the best mutual response policy for multiple agents in non-cooperative WSNs through online learning. Therefore, NashQ was more adaptive to topological changes yet less computationally intensive. In addition, NashQ promoted more cooperation between both agents in a hostile environment, and demoted more cooperation than the Non-cooperative game algorithm, when unnecessary. Furthermore, NashQ also appeared to be more robust to the non-uniqueness of Nash equilibrium when compared with the Non-cooperative game approach.

CHAPTER IV

PACKET FORWARDING IN OVERLAY WIRELESS SENSOR NETWORKS : PARETO Q-LEARNING

4.1 Introduction

Overlay wireless sensor networks (WSNs) consist of a large number of sensor nodes that are independently deployed in the same region of interest which are controlled by different network authorities without conflicting each other. These networks could potentially gain certain benefits such as alternative routing paths, reduced energy consumption) if their sensors share resources which can prolong their lifetime. Most existing works consider resource allocation problems in overlay WSNs with focus on saving energy (Murase et al., 2006); (Mao et al., 2008); (Wang et al., 2007), target detection (Li et al., 2006), node deployment (Yu et al., 2007) and fault tolerance (Chitnis et al., 2009); (Han et al., 2010). All of these works showed that resource sharing and cooperation between two agents, results in reduced energy consumption and increased their network performance. However, because of possible selfish behaviors among sensor nodes to conserve their energy, cooperative behavior between sensor nodes belonging to different network authorities may not always be readily available. Furthermore, it is possible that, under certain situations, node cooperation will not be beneficial to any WSN. Vaz et al. (2008) showed that cooperation between two different networks that are deployed in the same region may not always be beneficial to both networks. This is because whether or not each

agent will cooperate depends on the configuration of each network, network connectivity and how hostile the environment is. In this chapter, our focus is mainly on determining a resource allocation scheme for non-cooperative sensors belonging to different network authorities. The challenge of resource allocation in this scenario is thus how to achieve good packet forwarding strategies decision in dynamic environments under resource constraints in overlay WSNs.

Recently, game theory has become a common tool to analyse and improve the network performance in non-cooperative resource allocation problems. Game theory can be used to analyse the interaction and determine a set of strategies among rational agents, where each agent uses available information to decide its behaviour. The major advancement that has driven much of the development of game theory is the concept of Nash equilibrium (NE) which is used to determine a suitable and fair strategy for all agents. NE is a set of strategies for each of the players such that each player's strategy is the best-response to the other players' strategies. Many research focus on the problem of stimulating cooperation. Ref. (Wu and Shu, 2008); (Miller et al., 2005) applied game theory to overlay WSNs problems by using incentive mechanisms to motivate cooperation between sensor nodes. On the other hand, (Felegyhazi et al., 2005) applied the Non-cooperative game algorithm to describe such a situation that cooperation can exist in overlay WSNs without incentive mechanisms. They formulated a packet forwarding game into a non-cooperative resource allocation problem. The authors showed that the Non-cooperative game algorithm is a suitable framework which can determine equilibrium strategy for their problem. However, a drawback of their approach is that obtaining a strategy needs significant amount of computational time to compute the utility for all possible actions of sensor nodes.

Such computation is mainly performed offline and may not be feasible in networks in a dynamic environment. Packet forwarding strategies should not only be computationally light but also adaptive to changes in network topology, network configuration, communication channel conditions, etc. Therefore, a framework which could obtain a non-cooperative resource allocation solution which can cater dynamic environments is required.

In this chapter, we introduce the application of reinforcement learning (RL) to address the issue of non-cooperative resource allocation problem in overlay WSNs. In the context of RL framework, an agent systematically learns correct behaviours online through trial-and-error interactions with a dynamic environment in order to achieve a particular goal. There are several recent researches which employ RL to encourage cooperation between sensor nodes in WSNs (Pandana and Liu, 2005); (Seah et al., 2007); (Shah and Kumar, 2008). However, these works considered a single network where all sensor nodes belong to a single network entity. To cater multiple network authority frameworks such as overlay WSNs, the concept of multi-agent reinforcement learning (MARL) can be used, which integrates together RL, game theory, and direct policy search techniques. A standard RL method called Q-learning can be applied directly to the MARL framework (Watkins and Dayan, 1992). Applications of MARL are rapidly expanding, and a wide variety of approaches to exploit its benefits and address its challenges have been proposed and applied to several problems (Busoniu et al., 2008). Refs. (Shah and Kumar, 2007); (Tham and Renaud, 2005); (Forster et al., 2007) applied a MARL to solve resource allocation problems in WSNs. Q-learning was applied in their works to identify the best routing strategy. Their results showed that their approach can maximize their network

lifetime. Wang et al. (2007); Liang et al. (2008) improved the convergence rate of a standard Q-learning while learning the optimal routing strategy in sensor networks. Their results showed that their proposed algorithms can determine the best strategy for each agent by achieving higher success packet delivery ratio and can converge faster than the standard Q-learning method. However, these works only used MARL for resource allocation problems in WSNs which sensor nodes are cooperative and operate under a single network authority. Moreover, they assumed that each sensor node acts as an agent which can obtain the best strategy by taking its action with respect to its own benefit without concerning the joint benefit of the other agents. This is, however, inappropriate as the state of the environment changes not only as a result of the action taken by that agent, but also from actions taken by all other agents in the system. Therefore, the agent needs to consider the other agents' actions in order to predict its own optimal action.

In the previous chapter, a packet forwarding game in overlay wireless sensor networks (WSNs) based on a Markov decision process (MDP) has been formulated. The best mutual policy for two different agents belonging in the same area is solved by using Nash Q-learning (NashQ) algorithm. Recall that in NashQ algorithm, the results showed that NashQ algorithm performed well when a unique NE existed. However, when multiple NE existed, NashQ algorithm resorted to a mixed strategy selection where the Nash equilibria were probabilistically selected according to the Lemke Howson method (Shoham and Brown, 2009). Though the mixed strategy regime does not satisfy the conditions to establish convergence of NashQ algorithm, results showed that the NashQ algorithm can still converge to a solution. This suggests that theoretical convergence conditions for NashQ algorithm may be relaxed.

However, there is still a pending issue of an agent predicting the other agent's action (Hu and Wellman, 2003), in which under the mixed strategy, the probabilistic action selection results in an incorrect prediction. Therefore, the goal of this chapter is to propose the Pareto Q-learning (ParetoQ) algorithm to solve the packet forwarding problem in non-cooperative overlay WSNs to facilitate the conjecture of the other agent's action and guarantee convergence.

ParetoQ allows agents to converge to Pareto optimum (Felegyhazi et al., 2006) since every stage game is guaranteed to have at least one optimal pure strategy. Pareto optimality is a set of strategies which an agent cannot increase its utility without decreasing the utility of at least one other agent. Without full information, the agent not only learns its own action, but it also observes actions of the other agents in order to evaluate the best mutual policy for all agents. Moreover, ParetoQ algorithm can determine a strategy when there are multiple equilibriums existing in the stage game. To evaluate the performance of ParetoQ, we divide the experiment into two parts. In the first part, we formulate our packet forwarding game into a Markov Decision Process (MDP) framework in order to conceptually show that ParetoQ can be applied to obtain the best mutual policy in non-cooperative multi-agent WSNs through online learning. The second part extends the study to a more realistic scenario by replacing the MDP model with the radio energy model (Naruephiphat and Usaha, 2008) and a finer granulation of battery level.

Therefore, the underlying aim of this chapter is to select the best mutual policy for all agents in non-cooperative overlay WSNs using the ParetoQ algorithm. Without full information, the agent not only learns its own policy, but it also observes behavior

of the other agent in order to evaluate their other policy for all agents received from Pareto optimal. Hence, the best mutual policy is assumed to hold.

The emphasis of this chapter is focused on the following issues:

1. The formulation of a packet forwarding game in non-cooperative overlay WSNs.

2. The MDP formulation for the Pareto Q-learning and the Non-cooperative game algorithms.

3. The performance comparison between the Pareto Q-learning algorithm and the Non-cooperative game algorithm based on MDP formulation.

4. A formulation based on a more realistic energy model for the Pareto Q-learning and the Non-cooperative game algorithm.

5. The performance comparison between the Pareto Q-learning algorithm and the Non-cooperative game algorithm based on radio energy model.

The main contribution of this chapter is three-fold: 1) the application of ParetoQ algorithm to achieve the best mutual packet forwarding strategy in non-cooperative overlay WSNs with the MDP formulation; 2) and the extension of ParetoQ algorithm from the MDP formulation to the radio energy model; 3) the incorporation of Pareto optimality with the Non-cooperative game algorithm and its application to the packet forwarding game.

The organisation of this chapter is as follows. Section 4.2 introduces learning in non-cooperative game, followed by the problem formation in Section 4.3. Experimental results obtained from simulation and the performance comparison with an existing algorithm are shown in Section 4.4. Section 4.5 presents the implementation. Finally, a summary is given in Section 4.6.

4.2 Learning in non-cooperative game

In *Non-cooperative Game*, each agent can independently decide to interact with the other agents without any prior agreement or collaborative conditions. Therefore, it is necessary for each agent to predict actions of other agents in order to determine its own action, relative to the others. Among existing algorithms, the Non-cooperative game algorithm (Felegyhazi et al., 2005) is the most popular algorithm to provide a solution to such problem (Naserian and Tepe, 2009). The Non-cooperative game algorithm, is a branch of game theory applied exclusively to the situation where the interests of multiple agents conflict. Such algorithm consists of a multi-stage game, defined by the tuple (η,τ,μ), where η denotes the set of players, τ denotes the set of strategies (i.e. policies) and μ denotes a set of utility functions. The solution in the Non-cooperative game algorithm is based on Nash equilibrium (NE) which attains the best mutual policy for all agents in the game.

However, to determine such strategy, the Non-cooperative game needs significant amount of computational time to compute the utility for all possible actions. Therefore, the algorithm may not be practical to implement in WSNs as sensors will be deployed in an environment which changes frequently. Thus, sensors should be able to autonomously adapt their strategy according to their environmental condition.

4.2.1 Reinforcement learning

Reinforcement learning (RL) (Sutton and Barto, 1998) is a machine learning scheme in which an agent learns the optimal policy from the agents' past experiences without prior information about the model of the environment.

Convergence of RL relies on the assumption that the dynamics of environment satisfies a Markov Decision Process (MDP).

In a MDP, the tuple (*S*, *A*, *P*, *R*) is defined to describe their characteristics, where *S* denotes the set of all possible states, *A* denotes the set of all possible actions, *P* is the state transition probability matrix such that $P(s' \mid s, a) \in P$ is the probability of transiting to the next state $s' \in S$ after an agent takes action $a \in A$ at state $s \in S$. *R* is a function of the reward expected from the environment as a result of taking action $a \in A$. The objective of solving a MDP is to find a policy, π, defined as a mapping from the state space to the probability distribution, $\pi : S \to P[A]$, where $P[A]$ is the distribution over the action space. To determine the optimal policy, π^*, RL requires the knowledge of a quantification of future benefits (or returns) at a given condition called the action-value function. The action-value function of a given policy π, denoted by $Q_t^{\pi}(s,a)$, associates all state-action pairs (s,a) with an expected reward for performing action *a* in state *s* at time step *t* and following π thereafter;

$$\begin{aligned} Q^{\pi}(s,a) &= E^{\pi}[R_t \mid s_t = s, a_t = a] \\ &= E^{\pi}[\sum_{k=0}^{\infty} \beta r_{t+k+1} \mid s_t = s, a_t = a], \end{aligned} \tag{4.1}$$

where $R_t = r_{t+1} + \beta r_{t+2} + \beta^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \beta^k r_{t+k+1}$ is the expected discounted return of the agent, β is the discount factor and $E^{\pi}[\cdot]$ is the expectation operator of a given policy π. The goal of the RL agent is to determine a policy to select actions so that its expected discounted future reward is maximized.

Q-learning is one of the most popular and effective algorithm employed in RL systems (Sutton and Barto, 1998). It is an algorithm that does not need a model of the environment and can directly approximate the optimal action-value function (Q-value) through online learning. During the learning process, the agent starts with an arbitrary initial Q-value. After executing action *a* at state *s*, the agent receives an immediate reward *r* and then updates both the new state and new Q-value with input from the environment. The update rule at time step *t+1* of Q-learning is given by:

$$Q_{t+1}(s,a) = (1-\alpha_t)Q_t(s,a) + \alpha_t[r + \beta \max_{a'} Q_t(s',a')], \tag{4.2}$$

where $\alpha_t \in [0,1)$ is the learning rate. The process is repeated iteratively to learn the agent's own optimal policy. The condition for Q-learning to converge is that all states and actions must be visited infinitely often (Watkins and Davan, 1992). Q-learning is applied to solve in single-agent systems where there is only one agent which can change the state of the environment.

4.2.2 Multiple agent Q-learning

Multi-agent systems differ from single-agent systems in that there are many different agents that are supposed to learn a task and that all of the agents' actions affect the environment. Thus, the optimal policy does not rely on only one agent, but rather it conditions on all agents. There are works which directly apply Q-learning to multi-agent systems where an individual agent maximizes its own benefit. By doing so, it neglects the presence of the other agents. As a result, this may well lead to suboptimal decisions. Therefore, an individual agent should take account of the effect of joint actions as a more suitable strategy for multi-agent systems.

4.2.2.1 Nash Q-learning

Hu and Wellman (2003) proposed the NashQ algorithm for a multi-agent game framework. The authors extended Q-learning to a non-cooperative situation where each agent can rationally decide its action whether it will cooperate with other agents or not by considering both its own and other agents' information as well. Instead of finding out an optimal action to maximize their own reward like the (single-agent) Q-learning, and disregarding other agents' actions NashQ seeks joint actions that yield the best reward for all agents. The objective of the agents in the NashQ algorithm is to learn their best mutual response policy, which is defined by the Q-values received from Nash equilibrium (NE). NE is not only used to decide the agent's own action, but also predict the other agents' actions, given by $\pi^1(s')...\pi^n(s')$ where $\pi^i(s')$ is the agent *i*'s distribution over the set of actions at state s' and *n* is the number of all the agents. Considering two agent system (i.e. i=1,2), NashQ then calculates a NE for the stage game $(Q_t^1(s'), Q_t^2(s'))$ and updates its Q-values according to

$$Q_{t+1}^i(s, a^1, a^2) = (1-\alpha_t)Q_t^i(s, a^1, a^2) + \alpha_t[r_t^i + \beta NashQ_t^i(s')], \tag{4.3}$$

$$NashQ_t^i(s') = \pi^1(s') \cdot Q_t^i(s') \cdot \pi^2(s') \tag{4.4}$$

NE is able to find a *pure-strategy equilibrium*, where an agent can choose with certainty an action which achieves the best response to the other agent's choice. However, not all games have *pure-strategy equilibrium*. Under this circumstance, the agents need to select their strategies randomly according to some

probability calculated from the Lemke-Howson method (Hu and Wellman, 2003) to achieve the NE. Such equilibrium is called *mixed-strategy equilibrium.*

NashQ requires two conditions in a stage game during learning to converge, including, (1) the existence of a global optimal point, which is the joint action in the stage game whereby every agent receives the highest utility; (2) the existence of a saddle point, which is the joint action in the stage game which is a NE, and each agent would receive a higher utility when at least one of the other agents deviates. Although the authors in (Hu and Wellman, 2003) showed convergence can still be established in their framework with relaxed convergence conditions, there is no guarantee of convergence if the NashQ algorithm is to be applied in other frameworks.

4.2.2.2 Pareto Q-learning

Pareto Q-learning (ParetoQ) is another multi-agent reinforcement learning algorithm (Song et al., 2007). Instead of finding the best mutual policies by received from NE, ParetoQ seeks for Pareto optimal equilibrium.

The advantage of Pareto optimal over NE is that every game must have at least one optimal pure strategy, meaning that the agents can decide with certainty which action to take such that the best mutual policy is achieved. Furthermore, for game with multiple equilibia, actions can be selected according to a lexicographic convention (Shoham and Brown, 2009). Therefore, these features suggest that ParetoQ may achieve a more suitable mutual policy than NashQ at a given time step. The pseudocode of the ParetoQ algorithm is shown in Table 4.1.

Table 4.1 The Pareto Q-learning algorithm.

Initialize	**Loop**
Let t = 0, get the initial state $s0$. Let the learning agent be indexed by i. For all $s \in S$ and $a^i \in A^i, i = 1,2$, let $Q_t^i(s, a^1, a^2) = 0.$	Choose action a_t^i. Observe $r_t^1, r_t^2; a_t^1, a_t^2$, and $s_{t+1} = s'$ Update Q_{t+1}^i for j=1,2. $Q_{t+1}^i(s, a^1, a^2) = (1-\alpha_t)Q_t^i(s, a^1, a^2) + \alpha_t[r_t^i + \beta ParetoQ_t^i(s')]$ where $\alpha_t \in [0,1)$ is the learning rate Let $t := t + 1$.

ParetoQ algorithm can converge if the agents choose their strategies and update their Q-value by using global optimal point, or Pareto optimal, or alternatively, a saddle point, which are encountered in stage games during learning. The Q-values in the ParetoQ algorithm can be updated using the following rule: Global NE > Pareto optimal > Saddle NE. Furthermore, when the stage games encounter multiple Pareto optimal, agents choose their strategies and update their Q-value according to a lexicographic convention.

The advantage of ParetoQ algorithm over the Non-cooperative game approach is that it can converge by learning from the agents' past experiences online, rather than the offline exhaustive search. Furthermore, unlike the NashQ algorithm, each agent in the ParetoQ algorithm can predict the other agents' actions with certainty such that the evaluation of the other agents' Q values, the information necessary for guessing the other agent's actions, can also be accurately carried out. In this chapter, we thus apply ParetoQ algorithm in the packet forwarding problem in non-cooperative overlay WSNs in order to find the best mutual policy which provides the best benefit for all agents in the system.

4.3 Problem formulation

In this section, we apply the ParetoQ algorithm into the packet forwarding game where two agents co-exist in a non-cooperative overlay WSN. The ParetoQ algorithm is formally introduced in order to find the best mutual policy in multi-agent WSNs. In our game, the agent attempts to learn its equilibrium Q-values, which are defined by Q-values obtained from Pareto optimal conditions. Moreover, the agent not only learns to find its own Pareto optimal policy, but also learns actions and rewards of the other agent to find the other agent's Pareto optimal strategy. Therefore, each agent acts rationally with respect to this expectation and eventually the best mutual response can be achieved.

4.3.1 Packet forwarding game

Throughout this chapter, the energy consumption, the reward function, the action space, the energy model and the concept of the packet forwarding game from the previous chapter are used. For the sake of completeness of the chapter, the problem formulation is presented as follows.

In our model, we assume that there are two different WSNs, i.e. agent i, i=1, 2, deployed in an overlay WSN. We assume that the system operates as a centralized system whereby a representative of each network, called cluster head, controls the behaviour of all other sensor nodes in its network authority. It is assumed that the cluster heads have no limit in energy (e.g. it may be equipped with a renewable energy supply). The role of each sensor node in an overlay WSN is to send its data measurements (i.e., packets) to neighbouring nodes through multi-hop communications to a base station. We assume that two sensor nodes are able to

communicate when they are within transmission range. Even if sensor nodes belong to a different network, interactions between the agents are ensured by the cluster head in their own network. Therefore, each agent maintains two different routing tables, one for routing within their own network and the other for routing through coordinated paths with the other network.

4.3.1.1 Radio model

Energy consumption required to packet forwarding process is computed from the radio model in (Naruephiphat and Usaha, 2008). The radio model for the reception cost of agent i is given by, $C_{i,RX}(b) = E_{elec} \times b$ where E_{elec} is the expended cost in the radio electronics and we assume that b is the size of the measurement packet transmitted. Therefore, the transmission cost of agent i is given by, $C_{i,TX}(b,d) = E_{elec} \times b + (\varepsilon_{amp} \times b \times d^{\sigma})$ where σ is the path loss exponent and ε_{amp} is the energy consumed at the output transmitter antenna for transmitting one meter. We assumed that the agent sends its packet to a base station (called sink) by either its own route or a coordinated route through the other network depending on actions selected by the agent.

4.3.1.2 Action

In the Non-cooperative game, each agent can independently decide its own action whether or not to cooperate with the other agent. A set of strategies, which include all the possible joint actions available in the game, is defined by A = {DD, DF, AD, AF}, where the shorthand notations refer to the following:

DD: The agent does not ask the other network to forward its packets and drops all packets from other network if asked for help.

DF: The agent does not ask the other network to forward its packets but helps the other network to forward all packets if asked for help.

AD: The agent asks the other network to forward its packets but drops all packets from the other network if asked for help.

AF: The agent asks the other network to forward its packets and helps the other network forward all packets if asked for help.

4.3.1.3 Reward

After taking an action, agent i evaluates the proportion of delivered measurement packets to the sink at time t denoted by $p_i(t)$. If, $p_i(t) \geq SR_i$ where SR_i is the successful measurement ratio required by agent i, then the action is considered successful and agent i will receive a gain at time t, $g_i(t) = G_i \times number\ of\ delivered\ measurement\ packet$. Note that $G_i >> 0$ in order to encourage either agent to send its packets to the sink; otherwise, $g_i(t) = 0$. Agent i also incurs a cost in time step t denoted by $C_i(t) = C_{i,TX} + C_{i,RX}$,, which is the total transmission and reception cost of all sensor nodes within the network authority of agent i. The reward of agent i is then given by

$$r_i(t) = g_i(t) - C_i(t). \tag{4.5}$$

4.3.2 MDP model

In reinforcement learning, each decision which an agent makes is a function of an output from the environment, i.e., the most recent state of the environment. The environment is said to have Markov property so that the history of its state changes can be summarized into the current state. The next state of the

environment is assumed to follow randomly as a function of a given action at the current state. Given the state s and action a, the probability of the environment transiting to a possible next state s' in the next time step is given by

$$P(s' \mid s, a) = \mathrm{P}[s_{t+1} = s' \mid s_t = s, a_t = a]. \tag{4.6}$$

and consequently giving the agent a corresponding reward. This defines a discrete time stochastic control process called the Markov Decision Process (MDP). In order to conceptually investigate the performance of the ParetoQ algorithm in a MDP framework, we define the state s in our game as the battery level of the network (i.e., the battery level of the sensor node with the least residual battery) where $s \in S$. S is the state space of the environment which is divided into 3 states, i.e. high (h), medium (m) and low (l)

$$S = \{h,\ m,\ l\}. \tag{4.7}$$

where h, m, l represent the state s in (4.6). We assume that state transition for both agents in our game follows the transition probability metric for action *a*, *P(a)* is given by:

$$P(a) = \begin{bmatrix} p_{hh}(a) & p_{hm}(a) & p_{hl}(a) \\ p_{mh}(a) & p_{mm}(a) & p_{ml}(a) \\ p_{lh}(a) & p_{lm}(a) & p_{ll}(a) \end{bmatrix} = \begin{bmatrix} \phi & \delta & 1-\phi-\delta \\ 0 & \theta & 1-\theta \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.8}$$

where $p_{xy}(a)$ refers to the transition probability of agent i to state y by taking action a at state x. The parameters $\theta, \phi, \delta \in [0,1]$ and each row of *P(a)* must satisfy

$\sum_{\forall y} p_{xy}(a) = 1$. Note that θ, ϕ and δ are transition probabilities that define the network battery level remaining as a result of the agents taking action a.

We assume that the game ends when either agent again reaches “low” state after taking action a at state “low” state with probability 1, signifying a battery depletion of a sensor node in its network authority.

4.3.3 Radio energy model

In order to apply the ParetoQ to a more realistic energy state change than the MDP model in (4.6) and (4.7), we then define the state space as the set of the actual battery levels of sensor nodes in each agent. Instead of (4.6), the state is divided according to the remaining battery level after an agent has taken an action. The radio model (Naruephiphat and Usaha, 2008) is employed to calculate the energy consumption of sensor nodes activity and estimate remaining battery level of the sensor nodes.

We define the bottleneck as the lowest remaining battery level of sensor nodes within the WSN controlled by agent i given by $B_i = \min_{\forall m \in i}\{B_m\}$. B_i is then quantized into n+1 discrete values, $\{B_i(0),...,B_i(n)\}$, where $B_i(k),\ 0 \le k \le n$ denotes the quantized remaining battery level. The state space is then given by

$$S = \{s : s = B_i(k)\}. \tag{4.8}$$

Because this model refers to the actual battery level, the state transition depends on the remaining battery level, which varies with the energy consumption depending on the agent’s action. Therefore, the transition probability in (4.7) is not necessary in this

model. The game is repeated until the bottleneck of either agent reaches “0” state, signifying battery depletion of a sensor node in its domain and the game then ends.

Since the increase in state space size affects the learning rate of the MARL algorithm, the granularity of the state space division may help the agents achieve different solutions. In order to study effect of state space division, we divide the state space into two cases, 3 states (n=2), in order to compare it with the three-state MDP model in (4.6), and a more finer case with 10 states (n=9).

4.3.4 ParetoQ reinforcement learning

In our game, the time is divided into discrete time units called time steps. At the beginning, the action value functions are initialised to $Q_0^i(s, a^1, a^2) = 0,$ for all $s \in S$, $a^i \in A^i$, $i = 1, 2$. Agent i attempts to learn its equilibrium Q-value, starting from an arbitrary guess. At time step t, agent i observes the current state, takes its action and observes its own reward. It then observes the action, reward at the other agent and observes the next state of both agents.

Agent i then calculates the Pareto optimal strategy $\pi^1(s'), \pi^2(s')$ and updates its Q-values as follows

$$Q_{t+1}^i(s, a^1, a^2) = (1-\alpha_t)Q_t^i(s, a^1, a^2) + \alpha_t[r_t^i + \beta ParetoQ_t^i(s')], \tag{4.7}$$

$$ParetoQ_t^i(s') = \pi^1(s') \cdot Q_t^i(s') \cdot \pi^2(s'). \tag{4.8}$$

Recall that Pareto optimality is defined as a set of strategies which an agent cannot increase its utility without decreasing the utility of at least one other agent. In order to calculate the Pareto optimal strategy, agent i must observe the other

agent's immediate reward and previous actions and updates its conjectures on the other agent's Q-function, by maintaining its own update on the other agent's Q-function:

$$Q_{t+1}^{j}(s,a^1,a^2) = (1-\alpha_t)Q_t^j(s,a^1,a^2) + \alpha_t[r_t^j + \beta ParetoQ_t^j(s')],\ j \neq i. \tag{4.9}$$

The game is repeated until either agent runs out of battery.

In our packet forwarding game, it is possible that multiple Pareto optimum equilibriums exist in a game. To enable each agent to correctly predict the other agent's actions (and thereby update (4.9) accordingly), the lexicographic convention (Song et al., 2007) can be established as follows.

1) The set of agents is ordered.

2) The set of each agent's action is ordered.

3) The set of different solutions are ordered

(Global NE > Pareto optimal > Saddle NE)

4) Both agents must agree to use the same series of convention.

By using this convention, both agents are able to uniquely select their joint actions and the convergence of the ParetoQ algorithm is achieved.

4.3.5 The Non-cooperative game based on Pareto optimality framework

In order to evaluate the performance of the ParetoQ algorithm, we compare it in a packet forwarding problem with the Non-cooperative game algorithm (Felegyhazi et al., 2005). However, instead of obtaining the best mutual strategy received from NE as in (Felegyhazi et al., 2005), the policies of the agents in in the Non-cooperative game used in this chapter are obtained by exhaustively searching for Pareto optimal strategies through all the utility functions (and adopting the

lexicographic convention should multiple equilibriums exist). Note that such method can be viewed as an offline strategy search method. On the other hand, our proposed ParetoQ algorithm obtains its strategy though online learning, without having to compute the utility functions for all possible actions.

4.4 Experiment results

In this section, we evaluate the performance of the proposed ParetoQ algorithm and investigate the equilibrium conditions of the packet forwarding strategies in an overlay WSN. We study its performance under 2 models, i.e., the MDP model and the radio energy model.

We consider two WSNs co-existing in the same area, which are deployed in a 40x20 m in overlay WSNs. Each WSN is controlled by a cluster head acting as an agent which collects the residual energy level of the networks (state) and takes actions and observes rewards as a result of their actions. The goal of each agent is to maximize the packet delivery within its network to the sink. We investigate two scenarios, the common sink scenario where the agents share the same sink and the separate sink scenario where each agent sends packets to its own sink. Simulation results are carried out over 1000 randomly generated topologies. The simulation parameters are shown in Table 4.1.

The implementation of ParetoQ requires a two-phase learning process i.e., (1) the training phase which coarsely learns ParetoQ values and (2) the testing phase which fine-tunes the Q-values. In the ParetoQ training phase, each agent learns its packet forwarding strategies from Q-values which are initialised and iteratively updated according to (4.7)-(4.9). The training ends when the difference between the

previous and the new utility at each agent is less than 1%. The latest Q-values for each state-action pair at each agent from the training phase are used as initial Q-values of each agent in the testing phase.

We evaluated the ParetoQ algorithm by comparing it with the Non-cooperative game algorithm (Felegyhazi et al., 2005) on the Pareto optimality framework described in the previous section.

Table 4.2 Simulation parameter values.

Parameter	Value
Number of sensor per domain	10-90
Distribution of the sensors	Uniformly random
Area size	40x20m
Path loss exponent	2-5
Success requirement (SR_i)	1
Position of the common sink	[20,10]
Position of the separate sink	[10,10] and [30,10]
The size of the measurement packet transmitted (b)	250 *Kbits*
E_{elec}	50*nJ/bit*
ε_{amp}	$10\,pJ/bit/m^{\sigma}$
G_i	100
Coverage range	6 m
Route selection	Minimum energy path

To investigate situations which call for sensor node cooperation in non-cooperative WSNs, we investigate the equilibrium conditions of the packet forwarding strategies of both algorithms by distinguishing the following equilibriums:

- Defective equilibrium: the agent ends up its game at the strategy DD-DD.

- Cooperative equilibrium: the agent ends up its game at the strategy AF-AF.
- Other equilibrium**:** the agent ends up at other strategies.

4.4.1 ParetoQ based on MDP model

To evaluate the ParetoQ algorithm, we first conceptually show that the algorithm can be applied to a packet forwarding game in non-cooperative overlay WSNs by using a MDP model to characterize the energy state transition of the network.

Figure 4.1a presents the proportion of simulation for each type of equilibrium by the ParetoQ algorithm based on the MDP model in the common sink scenario. The results show that the proportion of cooperative equilibrium is higher than the other types of equilibrium when the density of sensor nodes is low. In contrast, as the density of sensor nodes increases, cooperation is reduced and defective equilibrium continually increases. This suggests that cooperation is vital in sparse networks. On the other hand, when sensor nodes are densely deployed in the area, various paths are available for their sensors to send packets to the sink, hence, cooperation between both agents is not necessary. This explains the reduction of cooperative equilibrium and increase of defective equilibrium as the network size increases.

Figure 4.1b presents the proportion of simulation for each type of equilibrium by the Non-cooperative game algorithm in the common sink scenario based on the MDP model. Similar to the ParetoQ, it can be seen that the Non cooperative game algorithm increasingly prefers the defective equilibrium as the density of sensor nodes increases yet at a less extent than the ParetoQ algorithm. For example, at 90 nodes per domain, the proportion of defective equilibrium from

ParetoQ is 78% while that of the Non-cooperative game is only 69%. The results suggest that though both algorithms favor the defective behavior in the common sink scenario, ParetoQ prefers defective equilibrium more than the other algorithm.

The proportion of simulation for each type of equilibrium in the separate sink scenario is shown in Figure 4.2a and 4.2b. The results show that both algorithms favor defective equilibrium over other equilibrium as the density of sensor nodes increases. More interestingly, the ParetoQ algorithm shows a monotonic decrease of cooperative equilibrium and monotonic increase of defective equilibrium as the number of sensors per domain increases. The Non-cooperative game results, on the other hand, are not monotonic as can be seen in Figure 4.1b and Figure 4.2b in case of 10, 20 and 30 nodes per domain. This suggests that ParetoQ algorithm can obtain a more robust behavior than the Non-cooperative game algorithm.

We then investigate sensor nodes behavior in more hostile environments. Figure 4.3a presents the proportion of simulation for each type of equilibrium as a function of path loss exponent in the common sink scenario by the ParetoQ algorithm based on the MDP model. The results show that the higher the path loss exponent, the more frequent the cooperative behavior is selected by both agents. The reason is because increasing the path loss exponent increases the difficulty in sending the packets to the sink. Hence, cooperation of both agents is more beneficial in this situation.

Figure 4.3b show the proportion of simulation by the Non-cooperative equilibrium under the MDP model. Similar to ParetoQ, the Non-cooperative game algorithm also prefers the cooperative equilibrium as the best strategy in hostile environments. While the Non-cooperative game results show a slight increase in

preference for cooperative behavior, ParetoQ tends to promote more cooperation in a more hostile environment. It can be seen that when path loss exponent is increased to 5, the proportion of cooperative equilibrium attained by the ParetoQ algorithm is 86% while that of the Non-cooperative game is only 77%.

The above results show that the ParetoQ algorithm based on the MDP model can find the best mutual response policy for multi-agents in non-cooperative overlay WSNs without having to resort to exhaustive searching as the Non cooperative game algorithm. ParetoQ also provides more robust behavior than the Non-cooperative game algorithm as shown by the monotonic behavior when the network size and path loss exponent are changed. Therefore, it also should perform well in real world application.

Hence, we then apply the ParetoQ to a more realistic scenario (i.e., by using the radio energy model) in the next experiment. We also investigate other performance metrics to evaluate the algorithm.

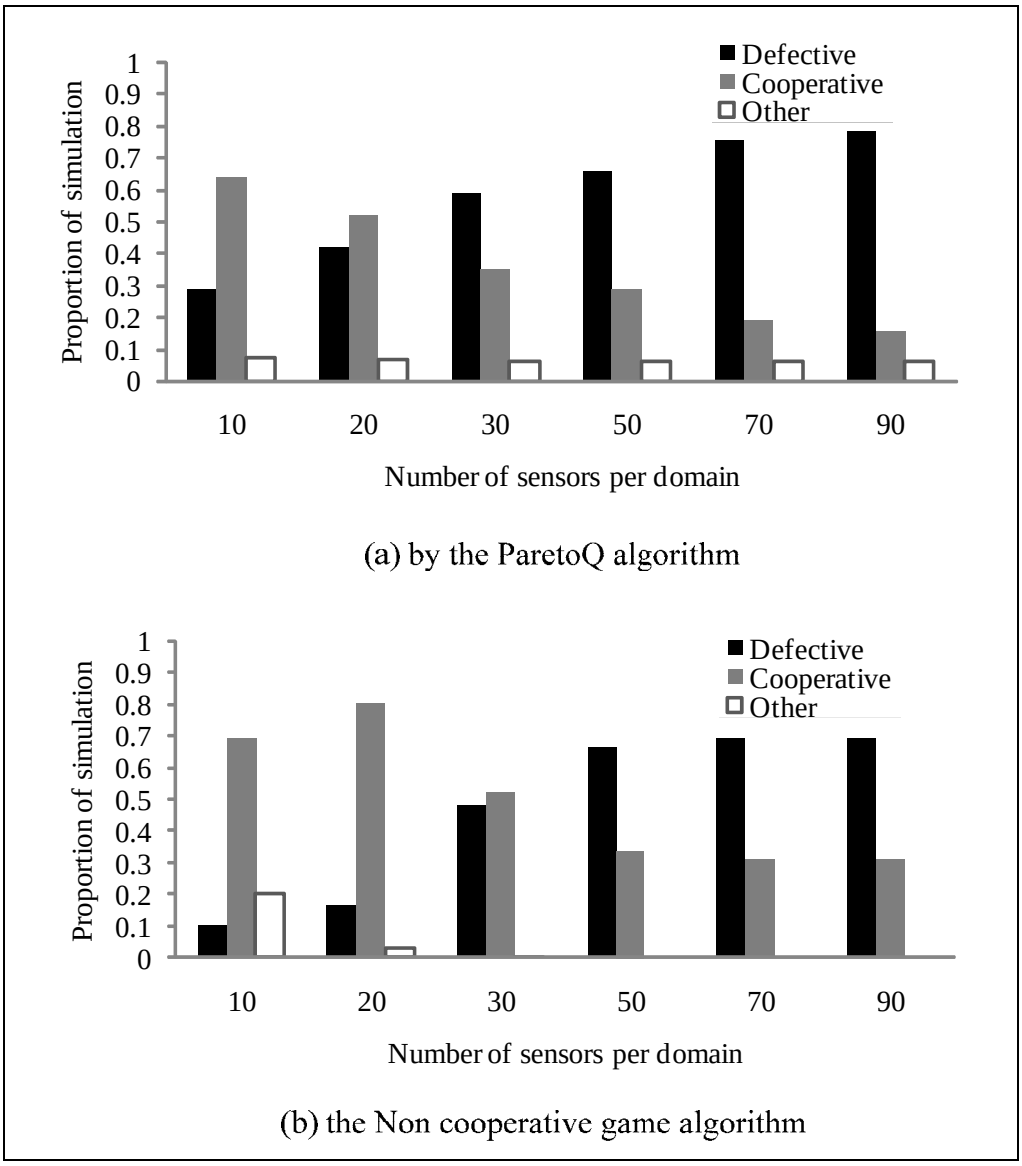

Figure 4.1 Effect of network size on cooperation in the common sink scenario with the MDP model

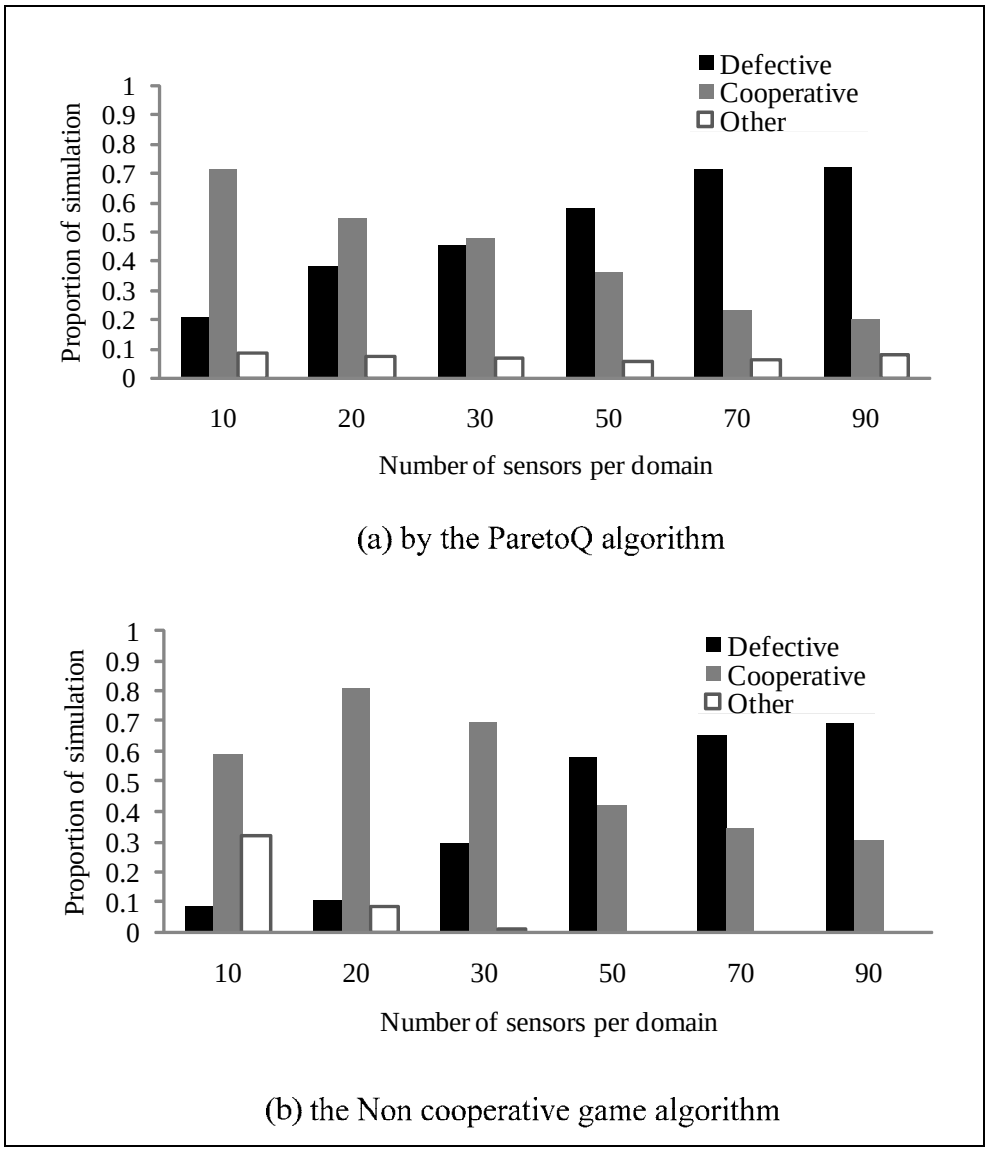

Figure 4.2 Effect of network size on cooperation in the separate sink scenario with the MDP model

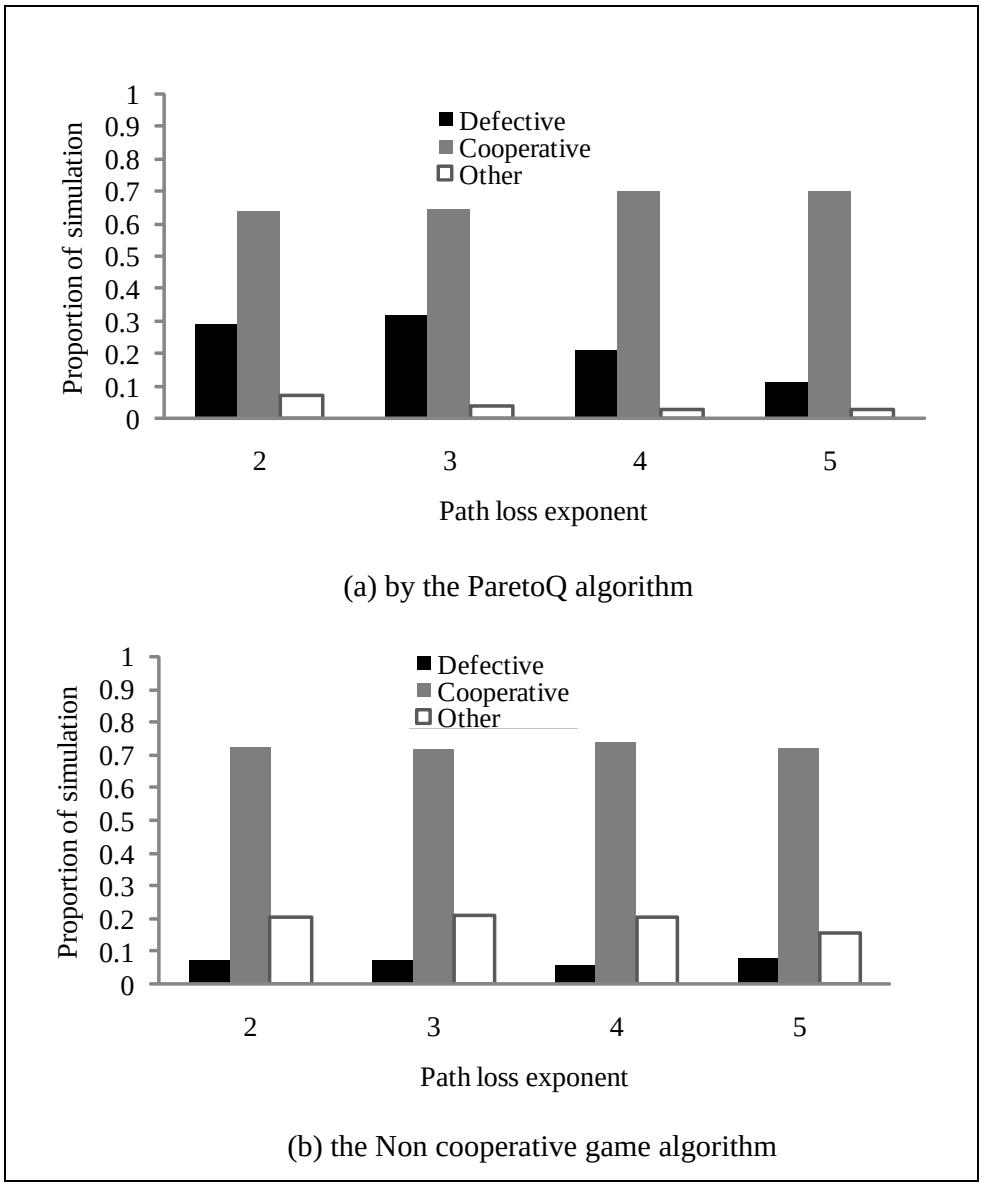

Figure 4.3 The Effect of path loss exponent in the common sink scenario with the MDP model

4.4.2 ParetoQ based on radio energy model

In this section, we evaluate the proposed ParetoQ algorithm based on a more realistic energy model which is described in Section 4.3.3. Instead of using the MDP model to characterise the energy state transition of the network, we use the radio energy model to govern the change in the remaining battery level in each sensor node in the WSNs. We divide the state space into two cases, i.e., 3 states in order to compare it with the 3 states MDP model in the previous experiment, and a finer case with 10 states.

Each sensor node is assumed to have an initial battery level of 100 J. The sensor nodes cannot recharge their batteries when depleted (except for the cluster head acting as the agent in the network). Simulation results were obtained from 1000 randomly generated topologies. The results can be observed as follows.

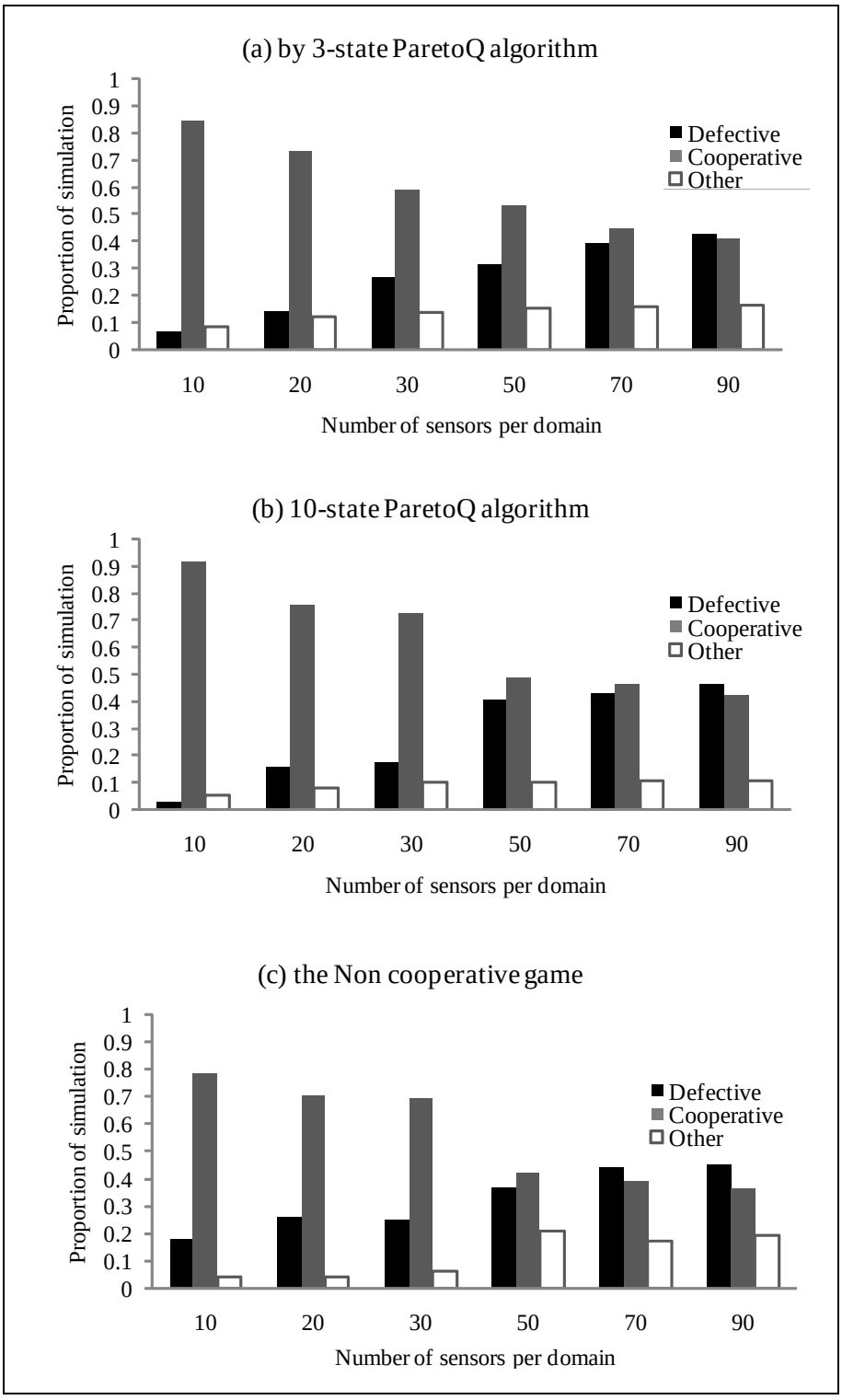

Figure 4.4 Effect of network size on cooperation in the common sink scenario with the radio energy model

Figure 4.4a and 4.4b depict the proportion of simulation for each type of equilibrium in the ParetoQ algorithm with 3-state and 10-state battery levels, respectively, in the common sink scenario with the radio energy model. The results show that the ParetoQ tends to promote cooperative equilibrium at low sensor node density and demote it when the network size increases for the same reasons given for Figure 4.1a. The results show that defective behaviour becomes more favourable for all agents as the sensor node density increases. Figure 4.4c presents the proportion of simulation by the Non-cooperative game algorithm in the common sink scenario. Similar to both cases of ParetoQ, the Non-cooperative game algorithm increasingly favors the defective equilibrium as the density of sensor nodes increases. Note that all three algorithms tend to promote cooperation between both agents when the sensor density is low in the common sink scenario. For example, in the case of 10 nodes per domain, the 10-state ParetoQ prefers it 93% of the run while the 3-state ParetoQ and the Non-cooperative game algorithm prefer it 85% and 78% of the runs, respectively. As the network size increases, all algorithms gradually change their policies to defective equilibrium. For example, at 90 nodes per domain, the 10-state ParetoQ tends to promote defective equilibrium more than other algorithms with the proportion of defective equilibrium of 49% while that of the Non-cooperative game is 45% and the 3-state ParetoQ is only 42%. The results suggest that the ParetoQ algorithm can still achieve good performance in finding the best mutual strategy for both agents when the energy consumption is governed by the radio energy model.

In addition to the equilibrium types of the packet forwarding strategies, we also considered additional performance metrics, i.e., the number of time steps required to obtain a strategy, the successful packet delivery ratio, the energy

consumption in the packet forwarding process and the average utility of an agent compared with the Non-cooperative game algorithm. Since both agents perform equally well, our experiments results are only shown for agent 1.

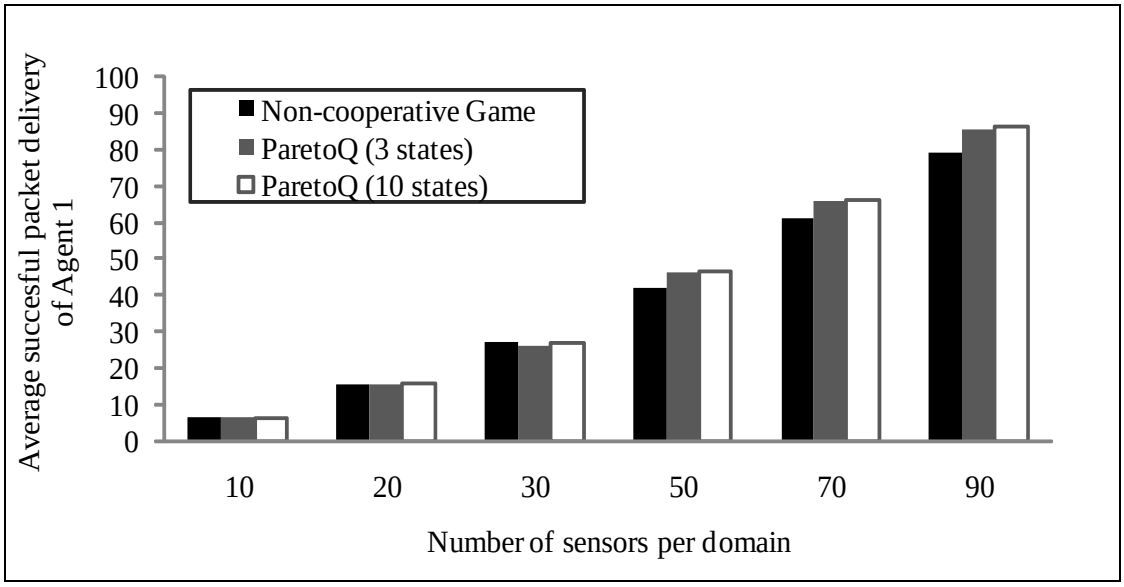

Figure 4.5 The successful packet delivery ratio in the common sink scenario with the radio energy model

Figure 4.5 presents the average successful packet delivery ratio in the common sink scenario with the radio energy model. At 10 nodes per domain, it can be seen that both the 3-state and the 10-state ParetoQ algorithms are able to deliver packets to the sink as well as the Non-cooperative game algorithm. Interestingly, when the network size increases to 50, 70 and 90 nodes per domain, both the 3-state and the 10-state ParetoQ algorithms can deliver more packets to the sink than the Non-cooperative game algorithm by 12% on average.

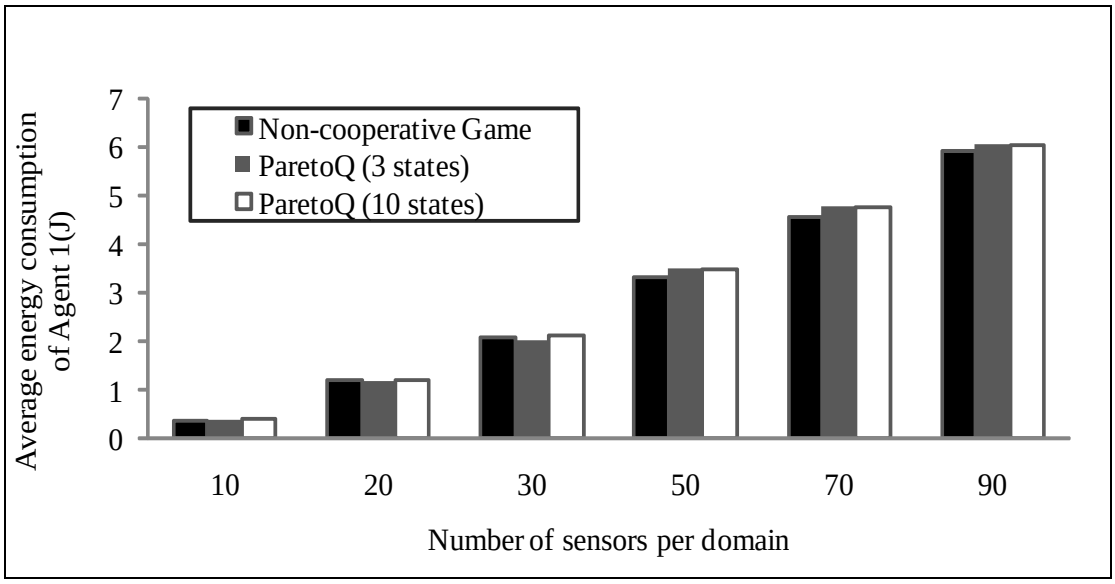

Figure 4.6 The energy required in the packet delivery process in the common sink scenario with the radio energy model.

Figure 4.6 presents the average energy consumption in the packet delivery process. The results show that when the network size increases, the 3-state and the 10-state ParetoQ algorithms require more energy consumption than the Non cooperative game algorithm. This is corroborated by the higher successful packet delivery ratio in Figure 4.5. This suggests that the increase in energy consumption in the 3-state and the 10-state ParetoQ algorithms are not an artifact of the online learning process, but by the higher number of packets successfully delivered.

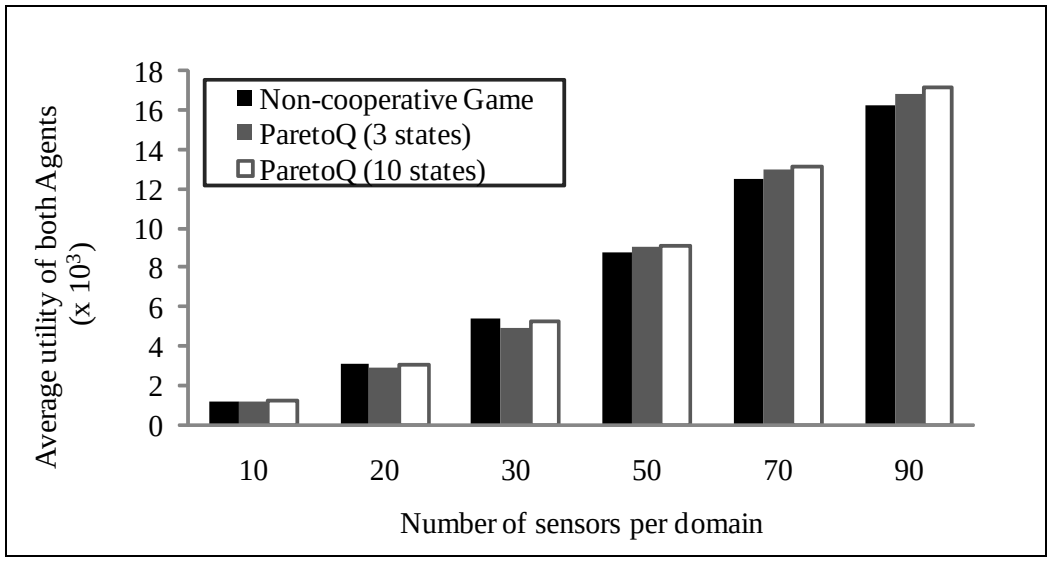

Figure 4.7 Utility values in the common sink scenario with the radio energy model

Figure 4.7 presents the average utility obtained for both agents within the common sink scenario with the radio energy model. The results show that at 30 sensor nodes per domain, the 3-state and the 10-state ParetoQ algorithms achieve less utility than the Non-cooperative game algorithm due to the lower successful packet delivery ratio. On the other hand, the 3-state and the 10-state ParetoQ algorithms achieve more utility than the Non-cooperative game algorithm as the network size increases (over 50 sensors per domain).

These results above show that the ParetoQ results can perform better than the Non-cooperative game results without the need of exhaustive search to obtain a strategy as the Non-cooperative game algorithm. More interestingly is that the proposed algorithm requires less computational time to obtain such strategy than the Non cooperative game algorithm as depicted in Figure 8. Figure 8 show that the Non cooperative game requires a significant amount of time steps to obtain a strategy. This is due to the off-line exhaustive strategy search process in the Non-cooperative

algorithm which requires information about all possible utility values from every joint strategy in order to obtain a solution. For example, at 10 nodes per domain, the density of sensor node is low, meaning that there are fewer paths between the sensor nodes and the sink. Therefore, the energy consumption required to forward the packet is also low. In order to obtain the utility for an action, it is necessary to run the game until energy is depleted (i.e., the game ends). The process is repeated for all possible joint actions. Since the energy consumed in the packet forwarding process is low, the number of time-steps encountered in the game before the game can end is high as can be observed in Figure 4.8.

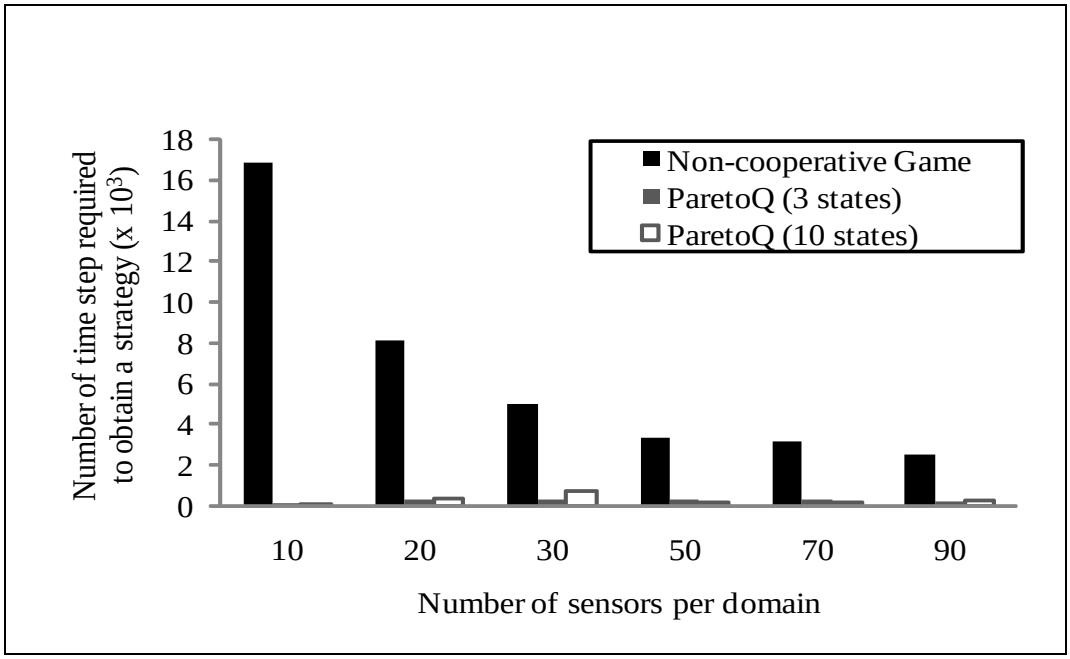

Figure 4.8 Number of time step required to obtain a strategy in common sink scenario with the radio energy model.

On the other hand, when sensor nodes are densely deployed in the area, much energy is required to execute an action. The number of time steps used in computing the utility is then reduced as the network size increases. Note that the 3-state and the 10-state ParetoQ algorithms require fewer time steps than the Non cooperative game algorithm to obtain a strategy. This is a result from the online learning process of the ParetoQ algorithm, which in each time step, the agents continue to improve their policies based on their past experience. This suggests that the online learning process of the ParetoQ algorithm for the common sink scenario can save computational time while achieving an average utility close to that of the Non cooperative game algorithm when the network size is small and outperforms the Non cooperative game algorithm as the network size grows above 50 nodes per domain (Figure 4.5, 4.7).

In order to evaluate the fairness of the best mutual strategy, we investigate the difference in average utility obtained by agent 1 and agent 2 as shown in Figure 4.9. We conjecture that the closer such difference is to zero, the fairer the strategy is for all agents. It can be seen that our proposed algorithm is closer to zero than the Non-cooperative game.

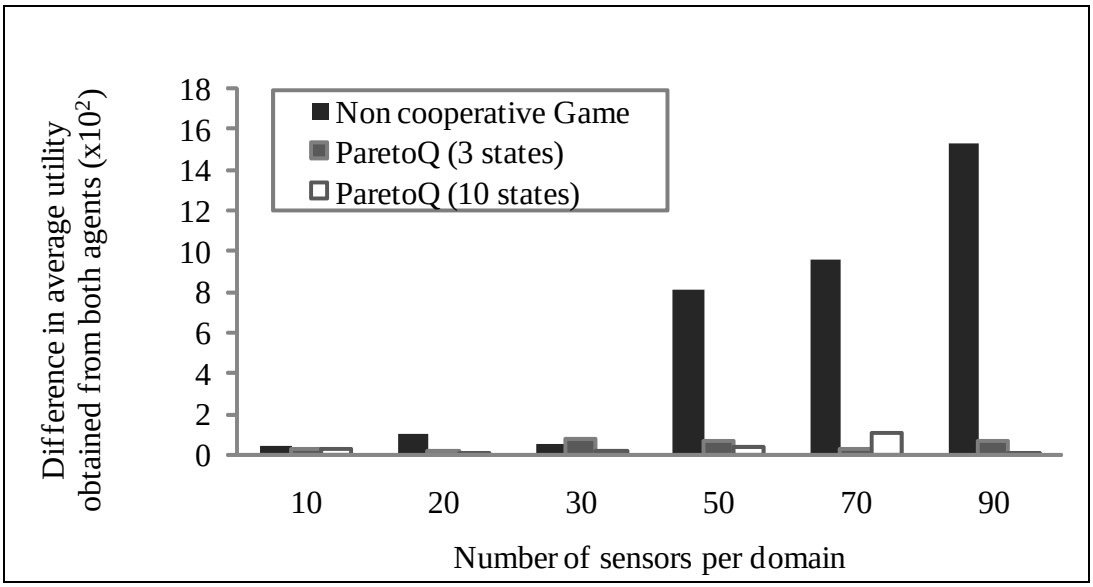

Figure 4.9 The difference in average utility obtained by both agents in common sink scenario with the radio energy model.

The performance of all algorithms in the separate sink scenario is shown in Figures 4.10a, 4.10b and 4.10c, in terms of the proportion of simulation for each type of equilibrium. The results show that all algorithms prefer cooperative equilibrium over other types of equilibrium and gradually demote it as the number of sensors per domain increases. Figure 10a and 10b depict the ParetoQ algorithm results which show a monotonic decrease of cooperative behavior and a monotonic increase of defective behavior as the number of sensors per domain increases. However, the 10-state ParetoQ algorithm promotes more cooperation between the agents than the 3 state ParetoQ algorithm. Figure 4.10c shows the Non-cooperative game algorithm results, which are non-monotonic (as can be seen at 10 nodes per domain). However, it also demotes cooperative equilibrium as the network size increases.

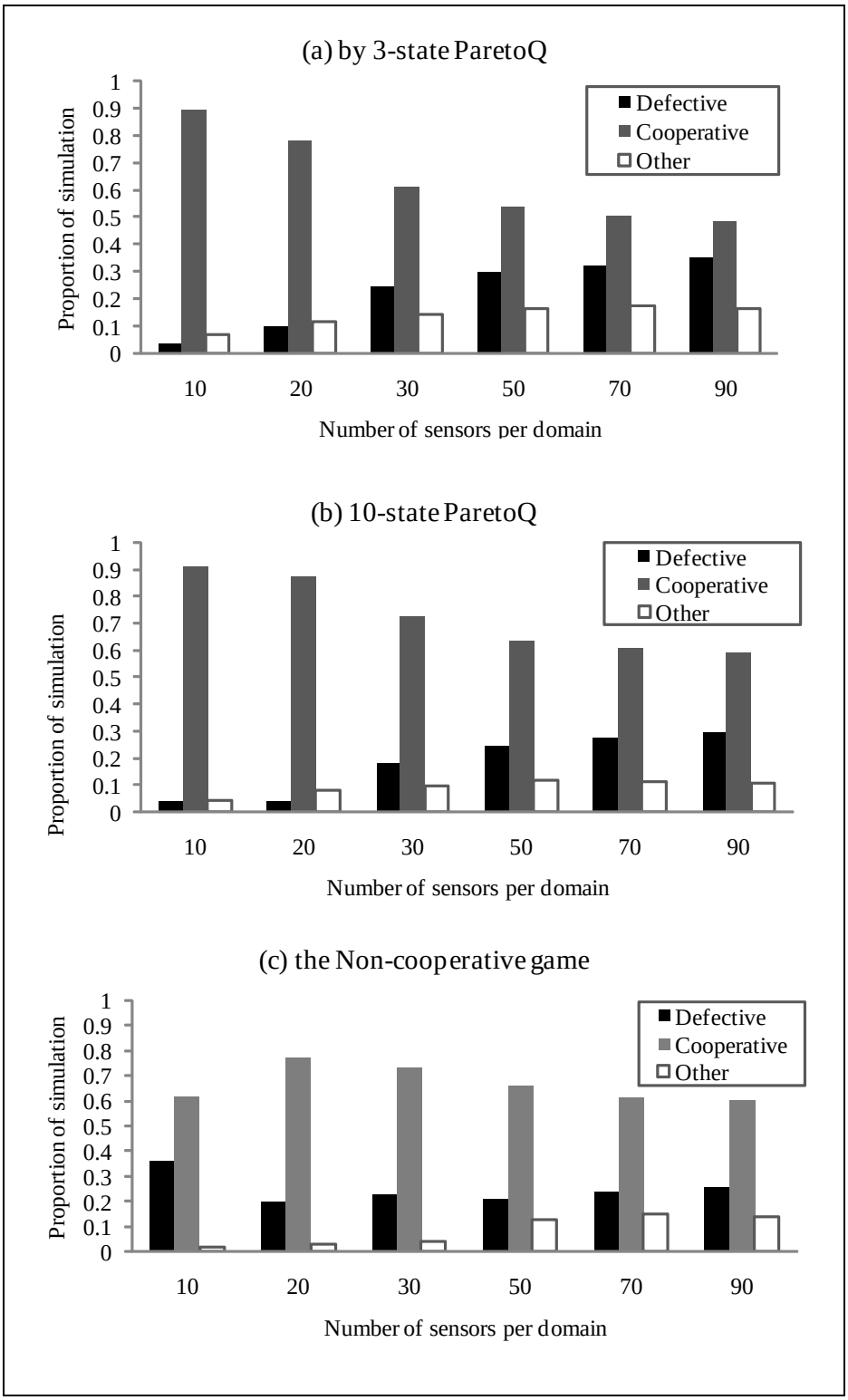

Figure 4.10 Effect of network size on cooperation in the separate sink scenario with the radio energy model

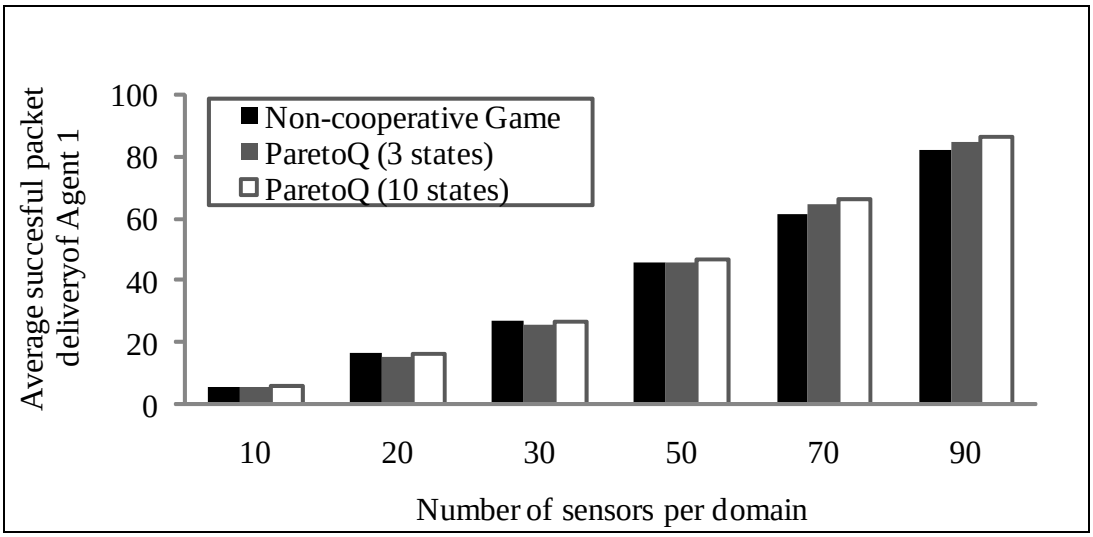

Figure4.11 The successful packet delivery ratio in the separate sink scenario with the radio energy model Non-cooperative game/ ParetoQ (3 state)/ ParetoQ(10 state)

Figure 4.11 presents the average successful packet delivery ratio in the separate sink scenario with the radio energy model. Similar to the common sink, both the 3-state and the 10-state ParetoQ algorithms can deliver more packets than the Non-cooperative game algorithm as the network size increases. However, the increase in successful packet delivery resulted in higher energy consumption than the Non-cooperative game algorithm as shown in Figure 4.12.

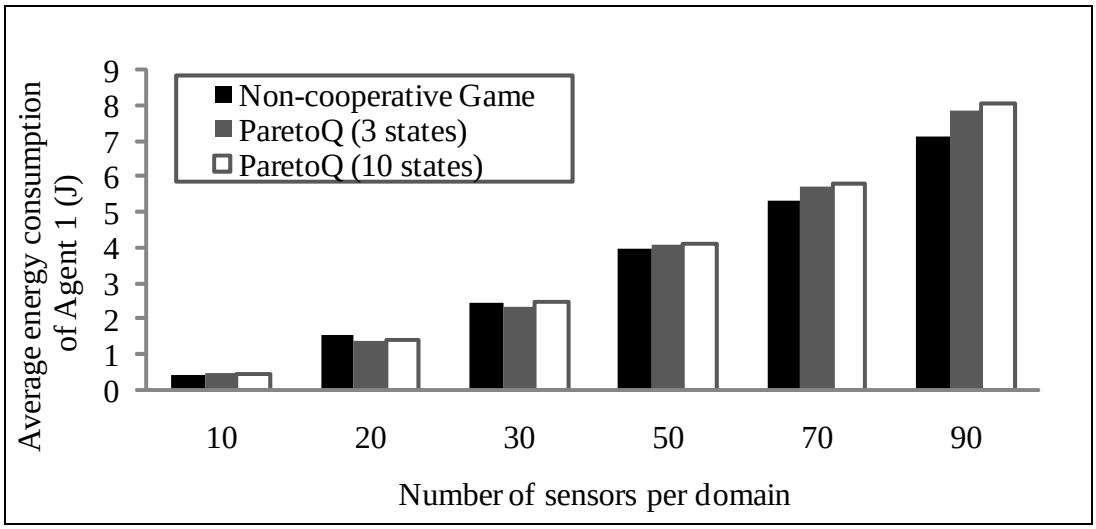

Figure 4.12 The energy required in the packet delivery process in the separate sink scenario with the radio energy model Non-cooperative game/ ParetoQ (3 state)/ ParetoQ(10 state)

Figure 4.13 presents the average utility for both agents in the domain for the separate sink scenario with the radio energy model. It can be seen that both the 3-state and the 10-state ParetoQ algorithms can achieve higher utility than the Non cooperative game algorithm as the network size increases.

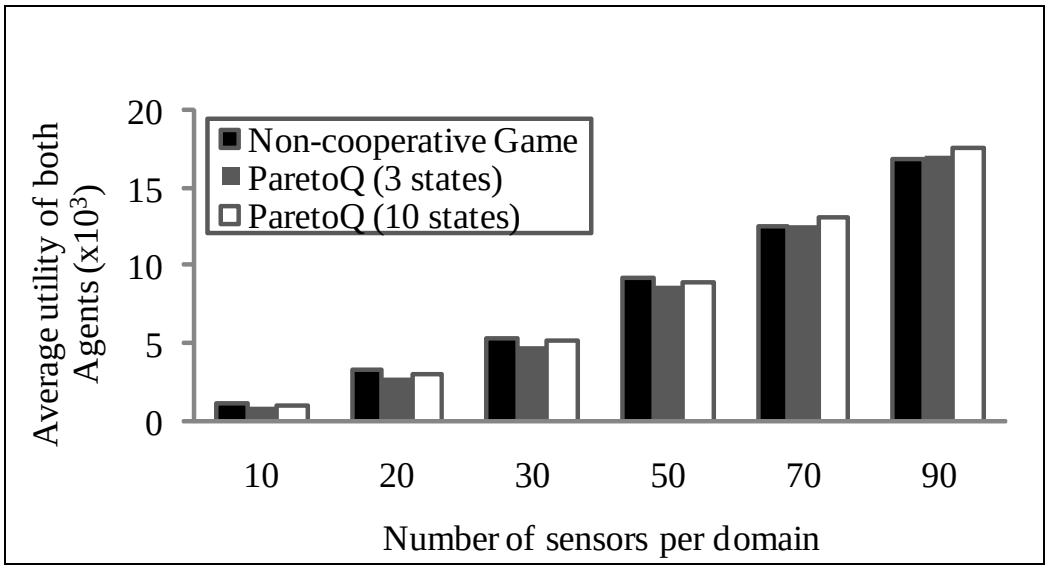

Figure 4.13 Utility value in the separate sink scenario with the radio energy model

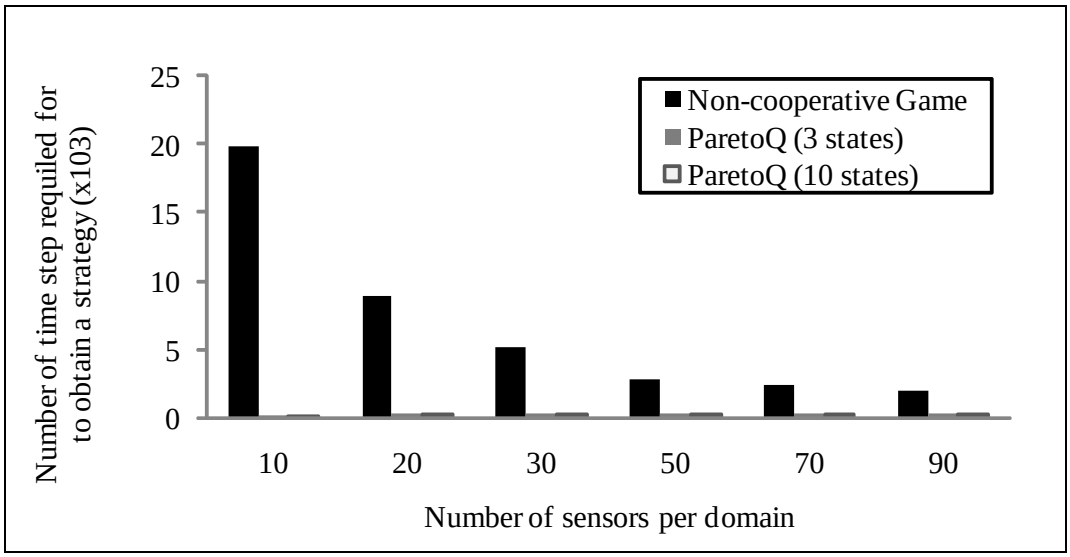

Figure 4.14 Number of time steps required to obtain a strategy in separate sink scenario with the radio energy model Non-cooperative game/ ParetoQ (3 state)/ ParetoQ(10 state)

Figure 4.14 shows the number of time steps required to obtain a strategy in the separate sink scenario with the radio energy model. The results show that the online-learning process from both the 3-state and the 10-state ParetoQ algorithms can obtain a strategy in fewer time steps than the Non-cooperative game algorithm for the same reasons given for Figure 4.8.

Figure 4.15 shows the difference in average utility obtained from both agents in the separate sink scenario with the radio energy model. It can be seen that the difference in average utility in the 3-state and the 10-state ParetoQ results are closer to zero than the Non-cooperative game algorithm. These results suggest that the ParetoQ algorithm not only performs as well as the Non-cooperative game with less computational time, but the strategy obtained is also fairer for all agents than the Non-cooperative game algorithm.

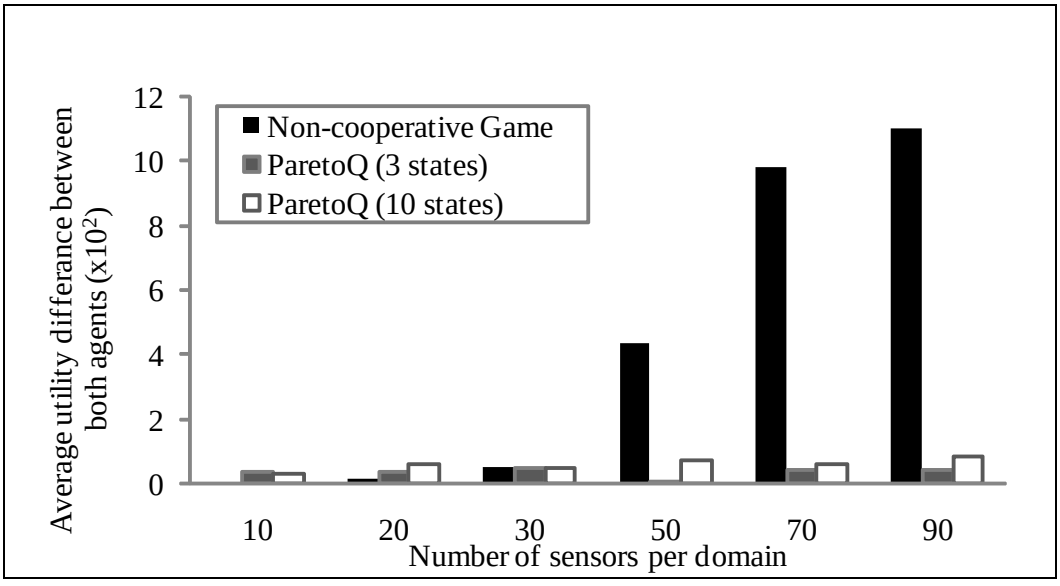

Figure 4.15 The difference in average utility obtained from both agents in separate sink scenario with the radio energy model.

Figures 4.16a, 4.16b and 4.16c present the proportion of simulation for each type of equilibrium as a function of path loss exponent in the common sink scenario for the 3-state ParetoQ, the 10-state ParetoQ and the Non-cooperative game algorithms, respectively. All algorithms show that the higher the path loss exponent, the more beneficial it is to cooperate for both agents. Furthermore, the results have shown that the 10-state ParetoQ promotes more cooperation in a hostile environment than other algorithms. For example, when the path loss exponent is 5, the 10-state ParetoQ achieves 64% cooperative equilibrium while the 3-state ParetoQ and the Non-cooperative game algorithms can attain 59% and 48%, respectively.

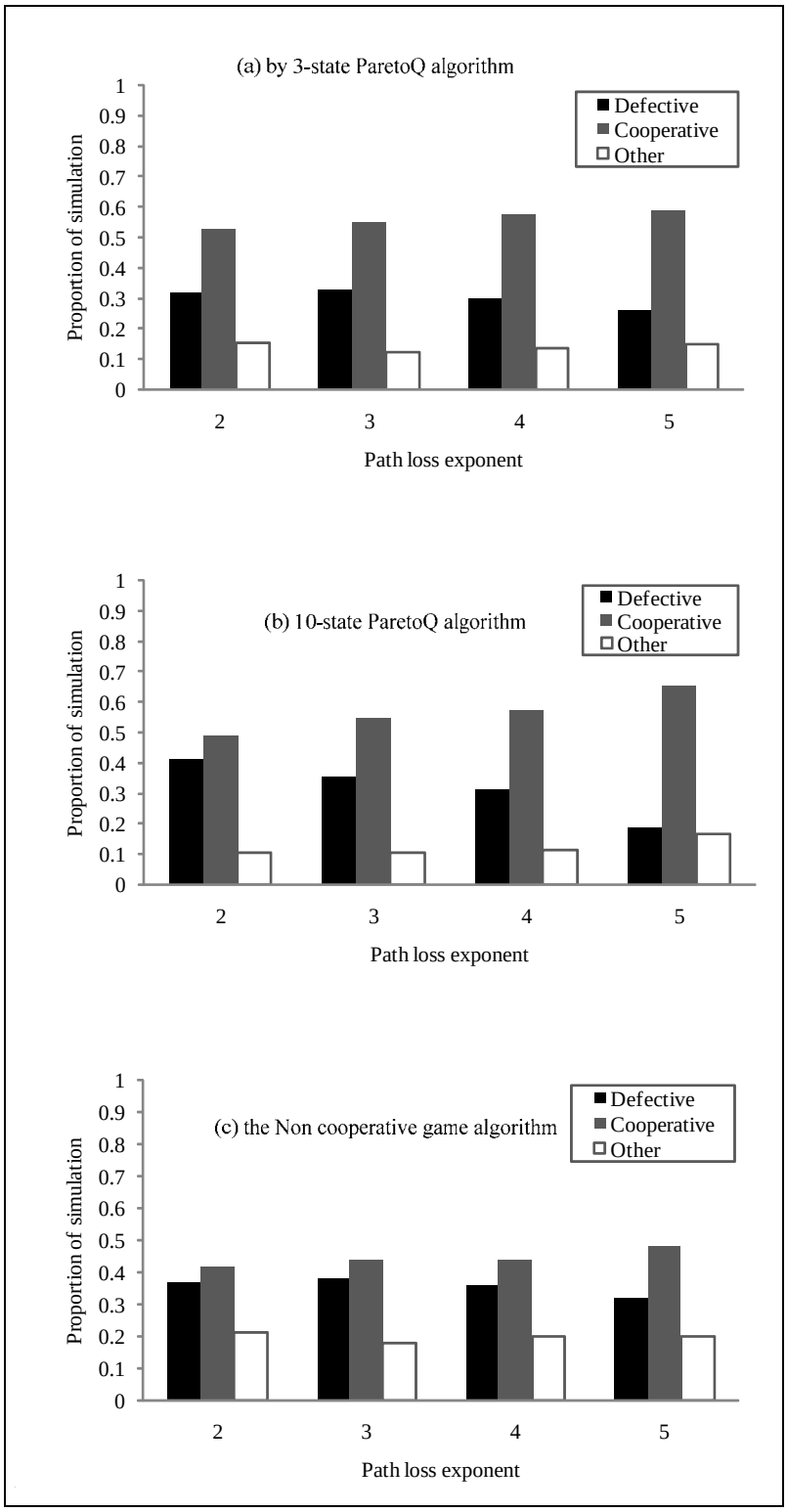

Figure 4.16 The effect of path loss exponent in the common sink scenario with the radio energy model

4.5 Implementation

The implementation of the ParetoQ algorithm requires a trained packet forwarding strategy at the cluster head of each sensor network. In terms of memory requirement for storing the Q-table, ParetoQ algorithm requires memory storage for storing all values of $Q^i(s,a)$ which has $|S_1|\times|S_2|\times|A_1|\times|A_2|$ entries at each agent. Suppose that each entry requires 8Bytes, a reasonable amount of memory of 1152 Bytes ($3^2\times4^2\times8Bytes$) and 12800 Bytes ($10^2\times4^2\times8Bytes$) are needed for the 3-state and 10-state ParetoQ algorithms, respectively. However, since each WSN is centrally controlled by an agent located at the cluster head, increasing the state or action space size will render a large amount of data stored at the cluster head which may lead to increased processing time.

4.6 Summary

In this chapter, we applied the ParetoQ algorithm to the packet forwarding game in a multi-domain WSN to determine the best mutual benefit for both agents. The contribution of this chapter is three-fold. First, we conceptually showed that the ParetoQ algorithm can be applied to promote cooperation in multi-agent overlay WSNs using the MDP model to define the state transition of the battery level in the network. Secondly, we applied the ParetoQ algorithm to a more realistic scenario using the radio energy model to evaluate the performance of the ParetoQ algorithm. Finally, we applied and evaluated the Non-cooperative game algorithm based on the Pareto optimum framework to solve for the best mutual packet forwarding policy in non-cooperative overlay WSNs.

Based on the MDP model, the results showed that the ParetoQ algorithm can find the best mutual response policy for two agents in non-cooperative WSNs through online learning. The algorithm can provide more robust behavior than the Non-cooperative game algorithm as shown by the monotonic behaviour in attaining defective and cooperative equilibriums as the network size and path loss exponent varies.

Based on the radio energy model, we investigated the effect of state division in the ParetoQ algorithm by comparing the 3-state and the 10-state ParetoQ algorithms. Results suggested that careful consideration of state space division can achieve more effective learning process and achieve higher utility from the networks in our framework. Moreover, our results suggested that even thought the ParetoQ algorithm started from randomly selecting joint strategies, it was able to learn online from its past experience to obtain the best mutual policy thereby saving time to execute a decision and improving the performance of the network without having to resort to exhaustive search as the Non-cooperative game algorithm. More interestingly, ParetoQ consistently outperformed the Non-cooperative game algorithm as the network size increased. In addition, the 10-state ParetoQ promoted the most cooperation between both agents, particularly in a hostile environment. Finally, the major advantage of online learning in the ParetoQ algorithm is that it was more adaptive to topological changes yet less computationally intensive than the Non-cooperative game.

CHAPTER V

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In overlay wireless sensor networks (WSNs), resource sharing and cooperation between the agents can prolong network lifetime. However, selfish behaviors of sensor nodes may occur in order to conserve their energy. Therefore, cooperation between sensor nodes belonging to different network authorities may not always be readily available. Furthermore, it is possible that node cooperation will not be beneficial to any WSNs. Hence, the work presented in this thesis was aimed at resource allocation algorithms to deal with non-cooperative behaviors of sensor nodes existing in overlay WSNs. The research work carried out in this thesis was divided into two parts: a MDP formulation for a packet forwarding game in overlay WSNs using Nash Q-learning (NashQ) reinforcement learning presented in Chapter 3; and a MDP formulation and a more realistic formulation based on the radio energy model for a packet forwarding game in overlay WSNs using Pareto Q-learning (ParetoQ) reinforcement learning presented in Chapter 4, respectively. The original contributions in this thesis can be summarized as follows.

5.1.1 Chapter 3

The purpose of this chapter is to conceptually show that the NashQ algorithm can be applied to promote cooperation in overlay WSNs. Two contributions were made here:

1) The MDP formulation of the packet forwarding problem in a non-cooperative multi-agent WSN.

2) The application of NashQ to solve for the best mutual policy in a multi-agent WSN.

The numerical study showed that based on the Lemke-Howson method, NashQ algorithm can find the best mutual response policy for multiple agents in non-cooperative WSNs through online learning. Therefore, NashQ algorithm was more adaptive to topological changes yet less computationally intensive than the Non-cooperative game algorithm. In addition, NashQ algorithm promoted more cooperation between both agents in a hostile environment, and demoted more cooperation than the Non-cooperative game algorithm, when it was unnecessary. Furthermore, NashQ also appeared to be more robust to the non-uniqueness of Nash equilibrium when compared with the Non-cooperative game approach. NashQ algorithm performed well when a unique NE existed. However, when multiple NE existed, NashQ algorithm resorted to a mixed strategy selection where the Nash equilibria were probabilistically selected according to the Lemke Howson method. Though the mixed strategy regime does not satisfy the conditions to establish convergence of NashQ algorithm, the results showed that the NashQ algorithm can still converge to a solution. This suggests that theoretical convergence conditions for NashQ algorithm may be relaxed. However, there was still a pending issue of an agent

predicting the other agent's action, in which under the mixed strategy, the probabilistic action selection resulted in an incorrect prediction.

5.1.2 Chapter 4

The purpose of this chapter is to facilitate the conjecture of the other agent's action, and still guarantee the convergence by applying the ParetoQ algorithm to the packet forwarding game in overlay WSNs. Three contributions were made here:

1) The application of ParetoQ algorithm to achieve the best mutual packet forwarding strategy in non-cooperative overlay WSNs with the MDP formulation.

2) The extension of ParetoQ algorithm from the MDP formulation to the radio energy model.

3) The incorporation of Pareto optimality with the Non-cooperative game algorithm and its application to the packet forwarding game.

The numerical study showed that, based on the MDP model, the ParetoQ algorithm can find the best mutual response policy for two agents in non-cooperative WSNs through online learning. The ParetoQ algorithm can provide more robust action selection than the Non-cooperative game algorithm by obtaining a monotonic increase (or decrease) equilibrium as the network size and a function of path loss exponent change.

Based on the radio energy model, we investigated the effect of state division in the ParetoQ algorithm by comparing the 3-state and the 10-state ParetoQ algorithms. Results suggested that careful consideration of state space division can achieve more effective learning process and achieve higher utility from the networks in our framework. Moreover, our results suggested that even thought the ParetoQ

algorithm started from randomly selecting joint strategies, it was able to learn online from its past experience to obtain the best mutual policy. Therefore, ParetoQ can save time in executing a decision and can improve the performance of the network without having to resort to exhaustive search as the Non-cooperative game algorithm. More interestingly, ParetoQ consistently outperformed the Non-cooperative game algorithm as the network size increased. In addition, the 10-state ParetoQ promoted the most cooperation between both agents, particularly in a hostile environment. Finally, the major advantage of online learning in the ParetoQ algorithm was that it was more adaptive to topological changes yet less computationally intensive than the Non-cooperative game.

5.2 Future work

5.2.1 WSNs with energy harvesting technology

A major limitation of sensor nodes is the limited amount of battery. It is possible for sensor nodes to use large batteries for longer lifetimes, but such nodes will have increased size, weight and cost. In recent years, energy harvesting techniques have the potential to prolong lifetime of sensor nodes in WSNs applications. An energy harvesting device can collect energy from ambient sources such as solar and wind energy thereby alleviating the need for battery replacement. The challenging issues is however to estimate the periodicity, manage the harvestable energy source and decide routing strategies which can assure the longest lifetime of the battery before the next recharge cycle.

5.2.2 Secure in routing protocol

In this thesis, we studied the packet forwarding strategy between two different WSNs belonging to the same area. Routes establish communication paths between sensor nodes and are used to forward packet from sensor nodes to a sink. Such routes, typically the shortest paths, may either belong to the sensor node's own network or a different network. Although shortest paths can reduce the amount of transmitted energy, there is no guarantee that these shortest paths are secure enough to successfully forward packets to the sink. Moreover, this thesis has not covered security issues involved when packets are falsely forwarded. Security aspects in routing procedure have not been emphasized in this thesis.

There are several WSN applications which support critical infrastructures (e.g. military, healthcare, environmental), where highly sensitive information is at risk, thus security becomes a vital issue. Security should be supported by fundamental operations in WSNs in order to promote a stable security infrastructure that will be able to handle routing effectively and efficiently.

5.2.3 Extension to distributed MARL

In this thesis, we assumed that the system operated as a centralized system whereby a representative of each network, called cluster head, controls the behavior of all other sensor nodes in its network authority. However, to enhance scalability, each sensor should be able to make its own decision whether to cooperate or not. Because sensor nodes are deployed in a large area, each sensor may have different requirement in its decision to help to forward packets or not. Thus, sensors should decide its strategy in a distributed manner. Moreover, since local information (i.e. node state and position) are locally stored in each node, sensor networks would be

more scalable in terms of reduced burden in dealing with synchronization a communication overhead.

5.2.4 Sensor node mobility

In this thesis, we focused on the situation where the location of the sensor nodes in the WSNs were fixed. However, sensors are mobile in many scenarios such as telemedicine and military applications. In such applications, sensor may frequently encounter topology changes. Therefore, routing schemes which can efficiently locate the sensor devices, establish communication paths and determine the best mutual strategy for all agents in the overlay WSN is needed.

5.2.5 Performance evaluation of testbed

The main objective of this thesis was to show that packet forwarding strategies in non-cooperative in overlay WSNs can be governed by using NashQ and ParetoQ algorithms. The packet forwarding game was simulated by Visual C++ programming to perform the online learning process and evaluate algorithms. Therefore, an important future direction is to extend the framework either to employ raw data collected from the field measurement for training the learning algorithms, or to implement the framework in an actual sensor network.

REFERENCES

Agah, A., Das, S. K., and Basu, K. (2004). A Non-cooperative Game Approach for Intrusion Detection in Sensor Networks. **Proceedings of IEEE Center for Research in Wireless Mobility and Networking.**

Busoniu, L., Babuska, R., and Schutter, B. D. (2008). A Comprehensive Survey of Multiagent Reinforcement Learning. **IEEE Transactions on Systems, MAN, and Cybernetics Part C**, vol.38, no. 2, pp. 156-172.

Chitnis, L., Dobra, A., and Ranka, S. (2009). Fault Tolerant Aggregation in Heterogeneous Sensor Networks. **Journal of Parallel and Distributed Computing**, vol. 69, no. 2, pp. 210-219.

Daskalakis, C., Goldberg P. W., and Papadimitriou, C. H. (2009). The Complexity of Computing a Nash Equilibrium. **Communications of the ACM**, vol. 52, no. 2, pp. 89-97.

Deb, K. (1999). **Multi-Objective Evolutionary Algorithms: Introducing Bias among Pareto-Optimal Solutions**. Springer-Verlag.

Felegyhaz, M., and Hubaux, J. P. (2006). Game Theory in Wireless Networks: A Tutorial. **EPFL Technical Report**.

Felegyhazi, M., Hubaux, J. P., and Buttyan, L. (2006). Nash Equilibria of Packet Forwarding Strategies in Wireless Ad Hoc Networks. **IEEE Transactions on Mobile Computing**, vol. 5, no. 5, pp. 463-476.

Felegyhazi, M., Hubaux, J. P., and Buttyan, L. (2005). Cooperative Packet Forwarding in Multi-Domain Sensor Networks. **Proceedings of IEEE International on Pervasive Computing and Communications Workshops**.

Forster, A. E., and Murphy, A. L. (2007). Exploiting Reinforcement Learning for Multiple Sink Routing in WSNs. **Proceedings of IEEE International Conference on Mobile Adhoc and Sensor Systems.**

Han, X., Cao, X., Lloyd, E. L., and Shen, C. C. (2010). Fault-Tolerant Relay Node Placement in Heterogeneous Wireless Sensor Networks. **IEEE Transactions on Mobile Computing**, vol. 9, no. 5, pp. 643-656.

Hu, J., and Wellman, M. P. (2003). Nash Q-Learning for General-Sum Stochastic Games. **Journal of Machine Learning Research**, vol. 4, no.(Dec 2003), pp. 1039-1069.

Li, M., Lul, Y., and Wee, L. (2006). Target Detection and Identification with a Heterogeneous Sensor Network by Strategic Resource Allocation and Coordination. **Proceedings of IEEE Intentional Conference on ITS Telecommunications.**

Liang, X., Balasingham, I., and Byun, S. S. (2008). A Multi-Agent Reinforcement Learning based Routing Protocol for Wireless Sensor Networks. **Proceedings of IEEE International Symposium on Wireless Communication Systems.**

Lima, C., and Abreu, G. T. F. (2008). Game-theoretical Relay Selection Strategy for Geographic Routing in Multi-hop WSNs. **Proceedings of IEEE International Conference on Navigation and Communication.**

Ma, Y., Cao, H., and Ma, J. (2008). The Intrusion Detection Method based on Game Theory in Wireless Sensor Network. **Proceedings of IEEE International Conference on Ubi-Media Computing.**

Machado, R., and Tekinay, S. (2008). A Survey of Game-Theoretic Approaches in Wireless Sensor Networks. **Journal of Computer Networks**, vol. 52, no. 16, pp. 3047-3061.

Mao, Y., Zhou, X., and Zhu, Y. (2008). An Energy-Aware Coverage Control Protocol for Wireless Sensor Networks. **Proceedings of IEEE International Conference on Information and Automation.**

Miller, D. A., Tilak, S., and Fountain, T. (2005). Token Equilibria in Sensor Networks with Multiple Sponsors. **Proceedings of IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing.**

Murase, T., Shimonishi, H,. and Murata, M. (2006). Overlay Network Technologies for QoS Control. **IEICE Transactions on Communication**, vol. E89–B, no.9, pp. 2280-2291.

Naruephiphat, W., and Usaha, W. (2008). Balancing Tradeoffs for Energy-Efficient Routing in MANETs based on Reinforcement Learning. **Proceedings of the IEEE Vehicular Technology Conference.**

Naserian, M., and Tepe, K. (2009). Game Theoretic Approach in Routing Protocol for Wireless Ad Hoc Networks. **Journal Ad Hoc Networks**, vol. 7, no.3, pp. 569-578.

Nash, J. (1951). Non-Cooperative Games. **The Annals of Mathematics**, vol. 54, no.2, pp. 286-295.

Niyato, D., Hossain, E., and Fallahi, A. (2007). Sleep and Wakeup Strategies in Solar-Powered Wireless Sensor/Mesh Networks: Performance Analysis and Optimization. **IEEE Transactions on Mobile Computing**, vol. 6, no. 2, pp. 221-236.

Pandana, C., and Liu, R. (2005). Near-Optimal Reinforcement Learning Framework for Energy-Aware Sensor Communications. **IEEE Journal on Selected Areas in Communications**, vol. 23, no. 4, pp. 788-797.

Puterman, M. L. (1994). **Markov Decision Processes: Discrete Stochastic Dynamic Programming**. Wiley-Interscience.

Qiang, S., Xianwen, Z., Niansheng, C., Zongwu, K., and Rasool, R. U. (2009). A Non-Cooperative Power Control Algorithm for Wireless Ad Hoc & Sensor Networks. **Proceedings of IEEE International Conference on Genetic and Evolutionary.**

Seah, M. W. M., Tham, C. K., Srinivasan, V., and Xin, A. (2007). Achieving Coverage through Distributed Reinforcement Learning in Wireless Sensor Networks. **Proceedings of IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing**.

Sen, S. Airiau, S., and Mukherjee, R. (2008). Towards a Pareto-Optimal Solution in General-Sum Games. **Proceedings of the IEEE Vehicular Technology Conference.**

Shah, K., and Kumar, M. (2007). Distributed Independent Reinforcement Learning (DIRL) Approach to Resource Management in Wireless Sensor Networks. **Proceedings of IEEE International Conference on Mobile Adhoc and Sensor Systems**.

Shah, K., and Kumar, M. (2008). Resource Management in Wireless Sensor Networks using Collective Intelligence. **Proceedings of IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing.**

Shoham, Y., and Brown, K. L. (2009). **Multiagent System: Algorithmic, Game-Theoretic and Logical Foundation**, Cambridge University Press.

Singsanga, S., Hattagam, W., and Ewe, H. T. (2010). Packet Forwarding in Overlay Wireless Sensor Networks using NashQ Reinforcement Learning. **Proceedings of IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing.**

Song, M. P., An, J. B., and Chen, R. (2007). A New Learning Algorithm for Cooperative Agents in General-Sum Games. **Proceedings of IEEE Conference on Machine Learning and Cybernetics**.

Stankovic, A. J. (2006). **Wireless Sensor Networks. , Chapter in Handbook of Real-Time and Embedded Systems**. CRC Press.

Sutton, R., and Barto, A. (1998). **Reinforcement Learning: An Introduction**. The MIT Press.

Tham, C. K., and Renaud, J. C. (2005). Multi-Agent Systems on Sensor Net-works: A Distributed Reinforcement Learning Approach. **Proceedings of IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing**.

Vaz, P. M., Cunha, F., Almeida, J., Loureiro, A., and Mini, R. (2008). The Problem of Cooperation Among Different Wireless Sensor Networks. **Proceedings of the International Symposium on Modelling, Analysis and Simulation of Wireless and Mobile Systems**.

Wang, P., and Wang, T. (2006). Adaptive Routing for Sensor Networks using Reinforcement Learning. **Proceedings of IEEE International Conference on Computer and Information Technology.**

Wang, R., Liu, G., and Zheng, C. (2007). A Clustering Algorithm based on Virtual Area Partition for Heterogeneous Wireless Sensor Networks. **Proceedings of IEEE International Conference on Mechatronics and Automation.**

Watkins, C.J.C.H., and Dayan, P. (1992). Q-Learning. **Journal of *Machine Learning***, vol. 8, no. 3-4, pp. 279-292.

Wu, M. Y., and Shu, W. (2005). InterSensorNet: Strategic Routing and Aggregation. **Proceedings of IEEE Global Telecommunications Conference**.

Yick, J., Mukherjee, B., and Ghosal, D. (2008). Wireless Sensor Network Survey. **Journal of Computer Networks**, vol. 52, no. 12, pp. 2292-2330.

Yu, L., Wang, N., Zhang, W., and Zheng, C. (2007). Deploying a Heterogeneous Wireless Sensor Network. **Proceedings of IEEE International Conference on Wireless Communications, Networking and Mobile Computing**.

Zhao, L., Zhang, H., and Zhang, J. (2008). Using Incompletely Cooperative Game Theory in Wireless Sensor Networks. **Proceedings of IEEE Wireless Communications and Networking Conference.**

APENDIX A

PUBLICATION

LIST OF PUBLICATION

Singsanga, S., Hattagam, W., and Ewe, H. T. (2010). Packet Forwarding in Overlay Wireless Sensor Networks using NashQ Reinforcement Learning. **Proceedings of IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing.** (ISSNIP 2010), Brisbane, Australia, December 2010, pp. 85-90.

Packet Forwarding in Overlay Wireless Sensor Networks using NashQ Reinforcement Learning

Sajee Singsanga [#1], Wipawee Hattagam [#2], Ewe Hong Tat [*3]

[#] *School of Telecommunication Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand*
[1] M5140794@g.sut.ac.th
[2] wusaha@ieee.org

[*] *Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman, Petaling Jaya, Malaysia*
[3] eweht@utar.edu.my

Abstract—This paper proposes a NASH Q-learning (NashQ) algorithm in a packet forwarding game in overlay non-cooperative multi-agent wireless sensor networks (WSNs). The objective is to achieve the best mutual response between two agents. The results show that NashQ can obtain the best mutual response by learning online, as opposed to the offline exhaustive search in an existing non-cooperative game theoretic approach. Therefore, NashQ is more adaptive to topological changes yet less computationally demanding in the long run. Furthermore, NashQ also appears to be more robust to the non-uniqueness of Nash equilibrium as results show a consistent cooperative behavior trend when compared with the existing approach.

I. Introduction

Wireless sensor networks (WSNs) coexisting independently within a region of interest without conflicting each other are called overlay WSNs. Such networks include a large number of nodes that are deployed in the same area which are controlled by different authorities. The most important usage for overlay WSNs is resource sharing between different authorities which can prolong their lifetime. The main reason is because intermediate nodes from other network authorities may help shorten the data transmission distance between neighbouring nodes. When two authorities share their sensor nodes with the same goal of sending packets via a shorter distance, energy consumption is lowered; the networks can save their energy and eventually prolong their lifetime.

Currently, many research related to resource allocation problems in overlay WSNs have been proposed with a focus on saving energy [1],[2],[3], fault tolerance problem [4], [5], node deployment problem [6], target detection problem [7]. All of these works showed that two authorities cooperatively sharing resources results in reduced energy consumption and increased network performance. However, cooperative behaviour between sensor nodes belonging to authorities may not always be readily available. Sensor nodes not only are deployed independently without any fixed strategy, but they may also act selfishly to conserve their energy. Furthermore, there is no guarantee that node cooperation will be beneficial to both WSNs. It may even be possible that cooperation may lead to benefits for a single party alone or no benefit to any party at all. In [8] showed that cooperation between two authorities in the same area may not always be beneficial to any network, because whether or not each authority will cooperate depends on the configuration of each network. It is therefore necessary to find an algorithm(s) for each authority to decide whether to cooperate with each other or not.

Recently, game theory has become a promising tool to analyze and improve the performance of sensor networks. The major advancement that has driven much of the development of game theory is Nash equilibrium. Nash equilibrium is a collection of strategies for each of the players such that each player's strategy is the best-response to the other players' strategies. In [9], the authors addressed the problem of whether cooperation can exist in WSNs without incentive mechanisms. In [10] and [11], incentive mechanisms were used to motivate cooperation between sensor nodes.

In this paper, we apply an existing algorithm called Nash Q-learning (NashQ) [12] to a packet forwarding problem in non-cooperative overlay wireless sensor networks. NashQ uses the framework of a general sum stochastic game, whereby each agent's reward depends on the joint action of all agents and the current state. The agent attempts to learn its Nash equilibrium online. Moreover, the agent not only learns to find its own policy, but it also learns actions and rewards of the other agent to find the other agent's optimal strategy. Therefore, each agent acts rationally with respect to this expectation.

To the best of our knowledge, NashQ has not been applied for resource allocation in overlay WSNs before. Therefore, the contribution of this paper is two-fold: 1) the MDP formulation of the packet forwarding problem in a non-cooperative multi-agent WSN and 2) the application of NashQ to solve for the best mutual policy in a multi-agent WSN.

II. Learning In Non-cooperative Game

Reinforcement learning (RL) [13] is a machine learning scheme which is used to learn the optimal action based on the agents' past experiences without full information about prior model of the environment. RL relies on the assumption that the dynamics of the system follows a Markov Decision

978-1-4244-7177-5/10/$26.00 © 2010 IEEE

ISSNIP 2010

Process (MDP). The tuple (S,A,P,R) is used to describe the characteristics of MDP, where S is the set of states, A is the set of actions, P is the state transition probability matrix. Let $P(s' \mid s, a) \in P$ the probability of transiting to the next state s' from state s when an agent takes action a, and R is a function of reward expected from the environment after taking the action $a \in A$ at state $s \in S$ and transiting to s'. In MDP, the objective is to find a policy, π, defined as a mapping of the state set to the action set, $\pi : S \to P[A]$, where $P[A]$ is the distribution over the action space. The action-value function $Q_t^{\pi}(s,a)$ of a given policy π associates all state-action pairs (s,a) with an expected reward for performing action a in state s at time step t and following π thereafter;

$$Q^{\pi}(s,a) = E^{\pi}\{R_t \mid s_t = s, a_t = a\} = E^{\pi}\{\sum_{k=0}^{\infty} \beta^k r_{t+k+1} | s_t=s, a_t=a\}, \quad (1)$$

where $R_t = r_{t+1} + \beta r_{t+2} + \beta^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \beta^k r_{t+k+1}$ is the expected discounted return of the agent, β is the discount factor and $E\{\cdot\}$ is the expectation operator. The objective of the agent in a MDP is to determine a policy to select actions so that the sum of the discounted rewards it receives over the future is maximized.

A well-known RL technique called Q-learning is employed to solve MDPs in single-agent systems. It is an algorithm that does not need a model of the environment and can directly approximate the optimal action-value function (Q-value) through online learning. The agent takes action a at state s, receives a reward r, updates the local state with input from the environment, and repeats the process to learn its own optimal policy. Q-learning provides a simple procedure in which the agent starts with an arbitrary initial Q-value at time step 0. The update rule at time step t+1 of Q-learning is given by:

$$Q_{t+1}(s,a) = (1-\alpha)Q_t(s,a) + \alpha[r + \beta \max_{a'} Q_t(s', a')], \quad (2)$$

where $\alpha \in [0,1]$ is the learning rate and s' is the next state that results from taking action a in state s. However, multi-agent systems differ from single-agent systems in that there are many different agents that are supposed to learn a task. The optimal policy does not rely on only one agent, but rather it depends on all agents. Several researches i.e. [14] and [15] employ Q-learning to solve multi-agent WSNs problems. Their results show that Q-learning can improve the performance in their system based on cooperative game. However, cooperative behaviour between sensor nodes belonging to multiple agents may not always be available, particularly when sensor nodes are controlled by different agents. Hu and Wellman [12] extended Q-learning to a non-cooperative multi-agent system where each agent can design its action whether it will cooperate with other agents or not. They proposed the Nash Q-learning (NashQ) algorithm, in which other agents' actions are taken into consideration as well. Instead of finding out which action to take to maximize their reward like the (single-agent) Q-learning, NashQ looks for joint actions that yield the best reward for all agents. The agents attempt to learn their best mutual response policy, which is defined by the Q-values received from Nash equilibrium (NE). NE is not only used to decide the agent's own action policy, but also predict the other agent's action policy, given by $\pi^1(s'), \ldots, \pi^n(s')$ where $\pi^i(s')$ is agent i's distribution over its set of actions at state s' and n is the number of agents. NE can be found in a pure-strategy equilibrium, where an agent is able to find the highest utility, meaning that the agents can decide with absolute certainty which action to take so that the best mutual policy is achieved. But in general, not all games have pure-strategy Nash equilibrium [16], so the agents have to decide to select their policies randomly according to some probability to achieve the best response. Such NE behaviour is called mixed-strategy Nash equilibriums. Any game is guaranteed to have a mixed-strategy NE [17]. The Lemke-Howson method (LH), is the best known method to solve for mixed-strategy NE for two agents [18]. However, this method still has certain limitations in computing the mixed-strategy NE probability, because the sequence of variables entering the LH method affects the solution directly. Nevertheless, the advantage of LH method is that it is guaranteed to find at least one NE point. We thus employ the LH method in both the proposed NashQ and an existing non-cooperative game approach [9] (see Section IV).

The convergence of NashQ is proved in [12]. They proved that NashQ can converge if 1) the stage games encountered during learning have a global optimal point, which is defined as a point of joint strategy in the stage game which every agent receives its highest payoff at this point; or, 2) they all have a saddle point which is defined as a point of joint strategy in the stage game which is a NE point, and each agent would receive a higher payoff when at least one of the other agents deviates. The advantage of NashQ over other non-cooperative game theoretic approaches is that it can converge by online-learning from the agents' past experiences.

In this paper, we thus apply NashQ algorithm in the problem of a packet forwarding problem in a non-cooperative multi-agent WSN in order to find the best mutual policy which provides the best benefit for all agents in the system.

III. Problem Formulation

In this section, we formulate the packet forwarding game in non-cooperative multi-agent WSNs. NashQ algorithm is then formally introduced in order to find the best mutual policy in multi-agent WSNs. In our game, the agent attempts to learn its equilibrium Q-values, which are defined by Q-values received in a NE. Moreover, the agent not only learns to find its own optimal policy, but it also learns actions and rewards of the other agent to find the other agent's optimal strategy. Therefore, each agent acts rationally with respect to this expectation and eventually the best mutual response can be achieved.

In our model, we assume that there are two different WSNs referred to as domains, co-existing in the same area. Each WSN is assumed to operate as a centralized system controlled by a cluster head acting as an agent for that particular domain. We assume that the cluster heads have no limit in energy (e.g. it may be equipped with a renewable battery) and can control the behaviour of the sensor nodes in its domain. Each agent maintains two different routing tables, one for routing within their own network and the other for coordinating paths with the other network. We assume that two sensor nodes are able to communicate with each other if they are within transmission range. Even if they belong to a different network, interactions between the agents are ensured by the cluster head in their own network.

Each sensor node employs the radio model [19] to compute the transmission and receiving cost required for transmitting a measurement packet. The reception cost of agent i is given by, $C_{i,RX}(b) = E_{elec} \times b$, where E_{elec} = 50 nJ/bit is the expended cost in the radio electronics and we assume that b = 250 Kbits is the size of the measurement packet transmitted. The transmission cost of agent i is given by, $C_{i,TX}(b,d) = E_{elec} \times b + (\varepsilon_{amp} \times b \times d^{\sigma})$, where σ is the path loss exponent and $\varepsilon_{amp} = 10\,pJ/bit/m^{\sigma}$ is the energy consumed at the output transmitter antenna for transmitting one meter. We assumed that the agents send their *packets* to a base station (called *sink*) by either their own route or a coordinated route through the other network depending on actions selected by the agent.

The action space is defined by $A = \{DD, DF, AD, AF\}$ where the short hand notations refer to the following:

DD: The agent does not *ask* the other domain to forward packets and *drops* all packets from other domain if asked for help.

DF: The agent does not *ask* the other domain to forward packets but helps the other domain to *forward* all packets if asked for help.

AD: The agent asks the other domain to forward packets but *drops* all packets from the other domain if asked for help.

AF: The agent asks the other domain to forward packets and helps the other domain *forward* all packets if asked for help.

After taking an action, the agent i evaluates the proportion of delivered measurement packets to the sink at time t denoted by $p_i(t)$. If, $p_i(t) \geq SR_i$ where SR_i is the ratio of successfully delivered measurement packets required by agent i, then the action is successful and agent i will receive a gain at time t, $g_i(t) = G_i$, where $G_i >> 0$ in order to encourage either agent to send its packets to the sink. Otherwise, $g_i(t) = 0$. Agent i also incurs a cost in time step t denoted by $C_i(t) = C_{i,TX} + C_{i,RX}$, which is the total transmission and reception cost of all sensor nodes within the domain of agent i. The reward of agent i is then given by

$$r_i(t) = g_i(t) - C_i(t). \tag{3}$$

We define the state space in our system as the set of all possible battery levels. In particular, the state space of each agent is divided into 3 states, i.e. high (*h*), medium (*m*) and low (*l*). The transition probability metric *P(a)* is given by:

$$P(a) = \begin{bmatrix} p_{hh}(a) & p_{hm}(a) & p_{hl}(a) \\ p_{mh}(a) & p_{mm}(a) & p_{ml}(a) \\ p_{lh}(a) & p_{lm}(a) & p_{ll}(a) \end{bmatrix} = \begin{bmatrix} \phi & \delta & 1-\phi-\delta \\ 0 & \theta & 1-\theta \\ 0 & 0 & 1 \end{bmatrix}, \tag{4}$$

where $p_{ij}(a)$ refers to the transition probability from state i to state j by taking action a and $\theta, \phi, \delta \in [0,1]$. Note that each row of $P(a)$ must add up to one since $\sum_{\forall j} p_{ij}(a) = 1$.

We divide the time into time units called time steps. We initialize $Q_0^1(s, a^1, a^2) = 0$ for all $s \in S$, $a^1 \in A^1, a^2 \in A^2$. Agent i attempts to learn its equilibrium Q-value, starting from an arbitrary guess. At time step t, agent i observes the current state, takes its action and observes its own reward. It then observes the action, reward at the other agent and observes the next state of both agents.

Agent i then calculates a NE $\pi^1(s'), \pi^2(s')$ for the stage game $(Q_t^1(s'), Q_t^2(s'))$ and updates its Q-values according to

$$\begin{aligned} Q_{t+1}^i(s, a^1, a^2) &= (1-\alpha_t) Q_t^i(s, a^1, a^2) \\ &\quad + \alpha_t [r_t^i + \beta NashQ_t^i(s')], \end{aligned} \tag{5}$$

where

$$NashQ_t^i(s') = \pi^1(s') \cdot Q_t^i(s') \cdot \pi^2(s'). \tag{6}$$

In order to calculate the Nash equilibrium, agent i must observe the other agent's immediate reward and previous actions and updates its conjectures on the other agent's Q-value, by maintaining its own update on the other agent's Q-value:

$$\begin{aligned} Q_{t+1}^j(s, a^1, a^2) &= (1-\alpha_t) Q_t^j(s, a^1, a^2) \\ &\quad + \alpha_t [r_t^j + \beta NashQ_t^j(s')], \; j \neq i \end{aligned} \tag{7}$$

The process is repeated until the game ends when the battery level of either agent reaches "low" state, signifying a battery depletion of a sensor node in its domain.

In this paper, we evaluate the performance of NashQ algorithm by comparing it with the Non-cooperative game algorithm in [9]. The Non-cooperative game is a branch of game theory, applied exclusively to the situation which the interests of multiple agents conflict. A multi-stage game can be defined by the tuple (η, τ, μ), where η is the set of players, τ denotes the set of strategies and μ is a set of utility functions. There are two players (i.e. agents) in the game. The reward can be defined by (3). We define the *utility* as the cumulative rewards of the player, $\mu_i = \sum_{t=0}^{T} r_i(t)$, where T denotes the lifetime of the domain depending on the transition

probability matrix in (4). While NashQ attempts to learn its policy though the Q-values obtained through NE in (5) and (6), the policies of the players in the Non-cooperative game algorithm are obtained by determining the NE from the utility functions.

In particular, to obtain the NE in the Non-cooperative game algorithm, the players first have to calculate utility functions for all possible actions $a^1 \in A^1, a^2 \in A^2$, and then choose their policies. Such method can be considered as an offline method. On the other hand, NashQ obtains its policy through online-learning, without having to compute the utility functions for all possible actions.

TABLE I
SIMULATION PARAMETER VALUES

Parameter	Value
Number of sensors per domain	10-90
Distribution of the sensors	Uniformly random
Area size	40x20 m^2
Path loss exponent, σ	2-5
Success requirement, SR_i	1
Positions of the common sink	[20,10]
Positions of the separate sink	[10,10] and[30,10]
Route selection	Minimum energy path
Transmission cost	$C_{i,TX}(b,d) = E_{elec} \times b + (\varepsilon_{amp} \times b \times d^{\sigma})$

IV. SIMULATION RESULTS

In this section, we evaluate the proposed NashQ algorithm and investigate the equilibrium conditions of the packet forwarding strategies.

To implement the packet forwarding game, we consider two different WSNs co-existing in the same area of size 40x20m^2. Each sensor node has the goal of maximizing its own packet delivery to a sink. In our simulation, we investigate two scenarios, i.e. the separate sink and the common sink scenarios. The simulation parameters are shown in Table 1. Simulation was performed for 100 runs under 100 randomly generated topologies. We compare the NashQ with the Non-cooperative game algorithm [9]. For both NashQ and Non-cooperative game algorithm, the LH method is used to compute NE in (6). Three types of equilibrium can be observed as follows:

Defective equilibrium: the agent ends up its game at the move DD-DD.

Cooperative equilibrium: the agent ends up its game at the move AF-AF.

Other equilibrium: the agent ends up at other moves.

Figure 1a presents the proportion of simulation for each type of equilibrium by the NashQ algorithm in the common sink scenario. It can be seen that the proportion of cooperative equilibrium is higher than other types of equilibrium at low sensor density but decreases as the density of sensor increases.

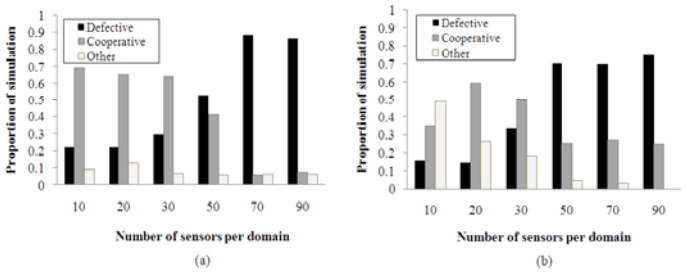

Fig. 1. Effect of network size on cooperation in the common sink scenario (a) by NashQ algorithm (b) Non cooperative game algorithm

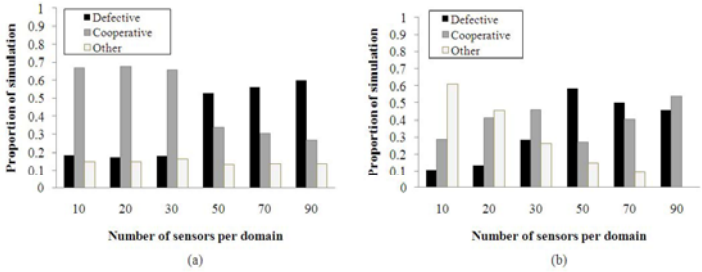

Fig. 2. Effect of network size on cooperation in the separate sink scenario (a) by NashQ algorithm (b) Non cooperative game algorithm

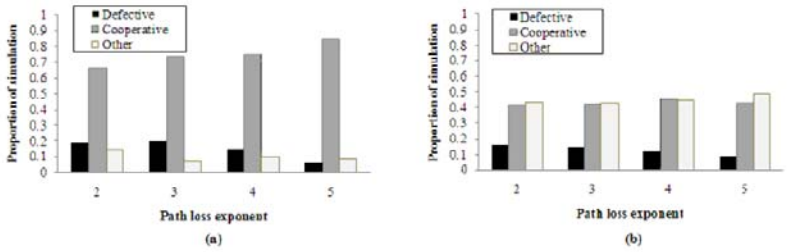

Fig. 3. The effect of path loss exponent in the common sink scenario (a) by NashQ algorithm and (b) Non-cooperative game algorithm

This suggests that cooperation is required if the density of sensor is low. On the other hand, if the agents are provided several paths for their sensors to send packets to the sink, as in the case when sensor nodes are densely deployed in the area, cooperation between both agents is not necessary. This explains the reduction of cooperative behaviour and increase of defective equilibrium as the number of sensors per domain increases.

Figure 1b presents the proportion of simulation by the Non-cooperative game algorithm in the common sink scenario. Similar to NashQ, the Non-cooperative game algorithm increasingly favors the defective equilibrium as the density of sensor nodes increases. The results show that though both algorithms prefer the defective equilibrium in the common sink scenario, NashQ prefers it more than the other algorithm. For example, at 90 nodes per domain, the proportion of defective equilibrium from NashQ is 87% while that of the Non-cooperative game is only 75%. In addition, as the node density decreases, the proportion of cooperative equilibrium from NashQ is also greater than the Non-cooperative game. This suggests that NashQ tends to promote more cooperation when necessary and demotes it more than the Non-cooperative game, otherwise.

Figure 2a and 2b depict the proportion of simulation for each type of equilibrium in the separate sink scenario. With reasons similar to the common sink case, NashQ and the Non-cooperative game algorithms show that the cooperative equilibrium is gradually reduced as the network size increases. More interestingly is that the NashQ algorithm provides a more robust behaviour than the Non-cooperative game algorithm in the sense that NashQ shows a monotonic decrease of cooperative behaviour and monotonic increase of defective behaviour as the number of sensors per domain increases. On the other hand, Non-cooperative game results are not monotonic as can be seen in Fig.2 b) at 50, 70 and 90 nodes per domain. It is caused by the limitation of LH algorithm when dealing with non-unique (i.e. multiple) NEs. As explained in Section III, the sequence of variables entering the LH algorithm directly affects the mixed-strategy NE probability. However, NashQ can avoid such condition whenever it is able to find the NE from either a global point or saddle point.

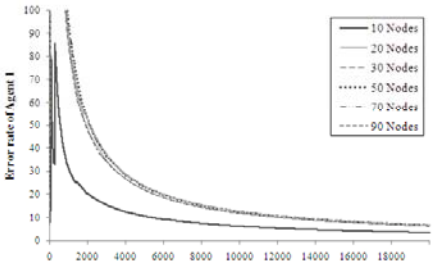

Fig. 4. Convergence of NashQ algorithm

Figure 3a shows the proportion of simulation for each type of equilibrium as a function of path loss exponent in the common sink scenario by the NashQ algorithm. The higher the path loss exponent, the more difficult it is to send the packets to the sink. Hence, the figure shows that cooperative equilibrium is more beneficial than other equilibriums in a hostile environment.

Figure 3b shows the proportion of simulation by the Non-cooperative game algorithm, suggesting that cooperation between both agents is still the best strategy for all agents in a hostile environment in the separate sink scenario. Once again, it can be seen that NashQ obtains 20-50% more cooperative equilibrium than the Non-cooperative game as the path loss exponent increases. This is likely caused by the fact that the Non-cooperative game employs the LH method to solve for a mixed strategy NE, thereby is not as robust to non-unique NEs as NashQ which uses the global or saddle point for its NE whenever possible. NashQ thus outperforms the Non-cooperative game.

Figure 4 presents the convergence of the NashQ algorithm in terms of error rate in the common sink scenario. The error rate is the percentage of the absolute difference between the previous Q-values and the newly updated Q-values summed over all state-action pairs. The figure shows that the initial stage of learning has high error rate but decreases steadily in the long run. This suggests that NashQ algorithm can be performed online starting from an arbitrary value and still converges as long as convergence conditions are met [12].

The Non-cooperative game algorithm, on the other hand, must be performed offline where an exhaustive search through all strategies is performed to determine the best mutual response strategy in each game. Therefore, once trained, NashQ appears to be less computational demanding and more adaptive to topological changes than the Non-cooperative game algorithm.

V. Conclusion

In this paper, we propose a NashQ algorithm to determine suitable packet forwarding strategies in a multi-domain WSNs. The objective of this paper is to conceptually show that the NashQ algorithm can be applied to promote cooperation in multi-agent in overlay WSNs. The results show that based on Lemke-Howson method, NashQ can find the best mutual response policy for multiple agents in non-cooperative WSNs through online learning. Therefore, NashQ is more adaptive to topological changes yet less computationally intensive. In addition, NashQ promotes more cooperation between both agents in a hostile environment, and demotes more cooperation than the Non-cooperative game algorithm, when it is unnecessary. Furthermore, NashQ also appears to be more robust to the non-uniqueness of Nash equilibrium when compared with the Non-cooperative game approach.

In the future work, we will formulate the state space based on a more realistic energy model and investigate other performance metrics such as the amount of successful delivery of the measurement packets to a sink, the energy consumption and the network lifetime.

References

[1] T. Murase , H. Shimonishi and M. Murata, "Overlay Network Technologies for QoS Control", IEICE Transactions on Communication, Vol. E89–B, No.9, pp. 2280-2291, 2006.

[2] Y. Mao, X. Zhou and Y. Zhu, "An Energy-Aware Coverage Control Protocol for Wireless Sensor Networks", Proceedings of IEEE International Conference on Information and Automation, 2008.

[3] R. Wang, G. Liu and C. Zheng, " A Clustering Algorithm based on Virtual Area Partition for Heterogeneous Wireless Sensor Networks", Proceedings of IEEE International Conference on Mechatronics and Automation, 2007.

[4] L. Chitnis, A. Dobra and S. Ranka, "Fault Tolerant Aggregation in Heterogeneous Sensor Networks", Journal of Parallel and Distributed Computing, Vol. 69, No. 2, pp. 210-219, 2009.

[5] X. Han, X. Cao, E. L. Lloyd and C.C. Shen, "Fault-Tolerant Relay Node Placement in Heterogeneous Wireless Sensor Networks", IEEE Transactions on Mobile Computing, Vol. 9, No. 5, pp. 643 – 656, 2010.

[6] L. Yu, N. Wang, W. Zhang and C. Zheng, "Deploying a Heterogeneous Wireless Sensor Network", Proceedings of IEEE International Conference on Wireless Communications, Networking and Mobile Computing, 2007.

[7] M. Li, Y. Lul and L. Wee, "Target Detection and Identification with a Heterogeneous Sensor Network by Strategic Resource Allocation and Coordination", Proceedings of IEEE Intentional Conference on ITS Telecommunications, 2006.

[8] P. Vaz de Melo , F. da Cunha, J. Almeida, A. Loureiro and R. Mini, "The Problem of Cooperation Among Different Wireless Sensor Networks", Proceedings of the International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2008.

[9] M. Felegyhazi, J. P. Hubaux and L. Buttyan, "Cooperative Packet Forwarding in Multi-Domain Sensor Networks", Proceedings of IEEE International on Pervasive Computing and Communications Workshops, 2005.

[10] M.Y. Wu and W. Shu, "InterSensorNet: Strategic Routing and Aggregation", Proceedings of IEEE Global Telecommunications Conference, 2005.

[11] D.A. Miller, S. Tilak and T. Fountain, "Token Equilibria in Sensor Networks with Multiple Sponsors", Proceedings of IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2005.

[12] J. Hu and M.P. Wellman, "Nash Q-Learning for General-Sum Stochastic Games", Journal of Machine Learning Research, Vol. 4, No.(Dec 2003), pp. 1039-1069, 2003.

[13] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, Massachusetts, 1998.

[14] C.K. Tham and J.C. Renaud, "Multi-Agent Systems on Sensor Networks: A Distributed Reinforcement Learning Approach," Proceedings of International Conference on Intelligent Sensors, Sensor Networks and Information Processing, 2005.

[15] X. Liang , I. Balasingham, S.S. Byun, "A Multi-Agent Reinforcement Learning based Routing Protocol for Wireless Sensor Networks," Proceedings of IEEE International Symposium on Wireless Communication Systems, 2008.

[16] C. Daskalakis, P.W Goldberg and C. H. Papadimitriou, "The Complexity of Computing a Nash Equilibrium," Communications of the ACM, Vol. 52, No. 2, pp. 89-97, 2009.

[17] J. Nash, "Non-Cooperative games," The Annals of Mathematics, Vol. 54, No.2, pp. 286-295, 1951.

[18] Y. Shoham and K.L. Brown, *Multi-agent System: Algorithmic, Game-Theoretic and Logical Foundation*, Cambridge University Press, Cambridge, 2009.

[19] W. Naruephiphat and W. Usaha, "Balancing Tradeoffs for Energy-Efficient Routing in MANETs based on Reinforcement Learning," Proceedings of the IEEE Vehicular Technology Conference, 2008.

BIOGRAPHY

Ms. Sajee Singsanga was born on August 25, 1985 in Nakhonratchasima province, Thailand. She finished high school education from Bunluavittayanusorn School, Nakhonratchasima province. She received her Bachelor's Degree in Engineering (Telecommunication) from Suranaree University of Technology in 2008. For her post-graduate, she continued to study with a Master's degree in the Telecommunication Engineering Program, Institute of Engineering, Suranaree University of Technology. During Master's degree education, she was a visiting researcher at Universiti Tunku Abdul Rahman, Malaysia in a topic of wireless sensor networks.