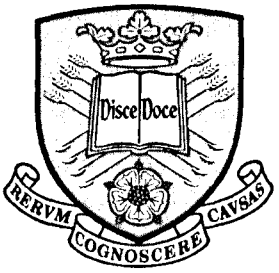

Swarm Intelligence and Its Applications to Wireless Ad Hoc and Sensor Networks

A thesis presented

by

K.Selvarajah

to

The Department of Automatic Control and Systems Engineering

in fulfillment of the requirements

for the degree of

Doctor of Philosophy

University of Sheffield

Sheffield, United Kingdom

August 2006

Abstract

Swarm intelligence, as inspired by natural biological swarms, has numerous powerful properties for distributed problem solving in complex real world applications such as optimisation and control. Swarm intelligence properties can be found in natural systems such as ants, bees and birds, whereby the collective behaviour of unsophisticated agents interact locally with their environment to explore collective problem solving without centralised control. Recent advances in wireless communication and digital electronics have instigated important changes in distributed computing.Pervasive computing environments have emerged, such as large scale communication networks and wireless ad hoc and sensor networks that are extremely dynamic and unreliable. The network management and control must be based on distributed principles where centralised approaches may not be suitable for exploiting the enormous potential of these environments. In this thesis, we focus on applying swarm intelligence to the wireless ad hoc and sensor networks optimisation and control problems.

Firstly, an analysis of the recently proposed particle swarm optimisation, which is based on the swarm intelligence techniques, is presented. Previous stability analysis of the particle swarm optimisation was restricted to the assumption that all of the parameters are non random since the theoretical analysis with the random parameters is difficult. We analyse the stability of the particle dynamics without these restrictive assumptions using Lyapunov stability and passive systems concepts. The particle swarm optimisation is then used to solve the sink node placement problem in sensor networks.

Secondly, swarm intelligence based routing methods for mobile ad hoc networks are investigated. Two protocols have been proposed based on the foraging behaviour of biological ants and implemented in the NS2 network simulator. The first protocol allows each node in the network to choose the next node for packets to be forwarded on the basis of mobility influenced routing table. Since mobility is one of the most important factors for route changes in mobile ad hoc networks, the mobility of the neighbour node using HELLO packets is predicted and then translated into a pheromone decay as found in natural biological systems. The second protocol uses the same mechanism as the first, but instead of mobility the neighbour node remaining energy level and its drain rate are used. The thesis clearly shows that swarm intelligence methods have a very useful role to play in the management and control

problems associated with wireless ad hoc and sensor networks. This thesis has given a number of example applications and has demonstrated its usefulness in improving performance over other existing methods.

Contents

List of Figures ix
List of Tables xi
Abbreviations xii

1 Introduction 1
1.1 Background 1
1.2 Contributions of the Thesis 4
1.3 Thesis Outline 6
1.4 Publications 7

2 Swarm Intelligence 8
2.1 Introduction 8
2.2 Properties of Swarm Intelligence 9
2.2.1 Positive feedback 10
2.2.2 Negative feedback 10
2.2.3 Randomness 10
2.2.4 Multiple Interaction 11
2.2.5 Stigmergy 11
2.3 Ant Algorithms 11
2.3.1 Artificial ants 12
2.3.2 Ant colony optimisation 12
2.3.3 AntNet 14
2.3.4 Ant based control 16
2.4 Analysis of Ant Algorithms 17
2.4.1 Modelling ant foraging 17
2.4.2 Analysis of ant colony optimisation 18
2.4.3 Analysis of ant routing algorithms 19
2.5 Particle Swarm Optimisation 19
2.5.1 The PSO algorithms 20
2.5.2 Modification to the PSO 21
2.5.3 Analysis of the PSO algorithms 22
2.6 Bacterial Swarm Foraging for Optimisation 23

2.6.1 The algorithm 24
2.6.2 Parameter selection 25

3 Wireless Ad Hoc and Sensor Networks 26
3.1 Introduction 26
3.2 Wireless Networks 27
3.2.1 Types of wireless networks 28
3.2.2 Enabling technologies 29
3.3 Ad Hoc Networks 30
3.3.1 Ad hoc networking issues 31
3.3.2 Ad hoc network applications 32
3.4 Ad Hoc Network Routing Protocols 32
3.4.1 Destination sequenced distance vector (DSDV) 34
3.4.2 Dynamic source routing (DSR) 35
3.4.3 Ad hoc on demand distance vector (AODV) 36
3.5 Ad Hoc Network Energy Conservation 37
3.6 Sensor Networks 38
3.6.1 Challenges in sensor networks 38
3.6.2 Sensor network applications 40
3.6.3 Routing in Sensor networks 40
3.6.4 Placement methods in sensor networks 41

4 Stability Analysis of Particle Swarm Optimisation 43
4.1 Introduction 43
4.2 Particle Swarm Optimisation 45
4.3 System Characteristics 47
4.4 Stability Analysis 51
4.5 Illustrative Examples 56
4.5.1 Nyquist plot and Circle criterion 57
4.5.2 Lyapunov function and particle trajectories 57
4.6 Conclusions 69

5 Energy Efficient Sink Node Placement in Sensor Networks 70
5.1 Introduction 70
5.2 Related Work 71
5.2.1 Optimal information extraction in energy limited wireless sensor networks 71
5.2.2 Energy aware node placement in wireless sensor networks . . . 74
5.2.3 Sink node placement methods 76
5.3 System Models 77
5.3.1 Energy Model 77
5.4 Energy Efficient Sink Node Placement 79

5.4.1 Strategy 1 81
5.4.2 Strategy 2 81
5.4.3 Strategy 3 82
5.5 Optimisation Strategies 82
5.5.1 Particle swarm optimisation 82
5.5.2 Genetic algorithm 84
5.6 Sink Node Shortest Path Problem 85
5.7 Sink Node Placement in Sensor Networks 86
5.8 Simulation Results 87
5.9 Conclusions 91

6 Swarm Intelligence Based Routing for Mobile Ad Hoc Networks 92
6.1 Introduction 92
6.2 Related Work 93
6.2.1 Ant-colony based Routing Algorithm (ARA) 93
6.2.2 Probabilistic emergent routing for mobile ad hoc networks . . 95
6.2.3 Mobile agent based routing protocol for mobile ad hoc networks 96
6.2.4 An adaptive swarm-based distributed routing algorithm 97
6.3 The SwAN Protocol 98
6.3.1 Motivation 98
6.3.2 Packet design 99
6.3.3 Protocol description 99
6.3.4 Pheromone table initialisation 100
6.3.5 Mobility information 101
6.4 Simulation Model 102
6.5 Simulation Results 103
6.5.1 Average end-to-end delay 103
6.5.2 Packet delivery ratio 106
6.5.3 Routing overhead 109
6.6 Conclusions 115

7 Swarm Intelligence Based Energy Aware Routing Algorithm 116
7.1 Introduction 116
7.2 Related Work 118
7.2.1 The minimum total transmission power routing (MTPR) . . . 118
7.2.2 The min-max battery cost routing (MMBCR) 119
7.2.3 The conditional max-min battery capacity routing (CMMBCR) 119
7.2.4 The minimum drain rate mechanism (MDR) 120
7.3 Swarm intelligence based Energy Aware Routing algorithm (SEAR) . 121
7.3.1 Protocol Description 122
7.3.2 Pheromone table initialisation 123
7.3.3 Energy level information 124

7.4 Simulation Model 125
7.4.1 Mobility and traffic model 125
7.4.2 Energy model 126
7.5 Simulation Results 126
7.6 Conclusions 129

8 Conclusions and Future Work **132**
8.1 Conclusions 132
8.2 Future Work 134

A Ns2 Simulation Scripts **136**

B Sample Scenario File **140**

C CBR Connection Pattern **147**

D Ns2 Trace Files for SwAN Protocol **153**

Bibliography **161**

List of Figures

4.1 Feedback control system representation particle dynamics 47
4.2 Maximum gain vs inertia factor for stability 56
4.3 Discrete-time Nyquist plot for inertia factor=0.8 and the limit value for its real part 58
4.4 Discrete-time Nyquist plot for inertia factor=0.2 and the limit value for its real part 58
4.5 Lyapunov function with $K = 0.04$ and $w = 0.8$ 60
4.6 Lyapunov function with $K = 1$ and $w = 0.2$ 60
4.7 Particle trajectories with $K = 0.04$ and $w = 0.8$ 61
4.8 Particle trajectories with $K = 1$ and $w = 0.2$ 61
4.9 Lyapunov function with $K = 2.5$ and $w = 0.8$ 62
4.10 Particle trajectories with $K = 2.5$ and $w = 0.8$ 63
4.11 Lyapunov function with $K = 2$ and $w = 0.2$ 63
4.12 Particle trajectories with $K = 2$ and $w = 0.2$ 64
4.13 Particle trajectories with $K = 3.5$ and $w = 0.8$ 64
4.14 Particle trajectories with $K = 3.5$ and $w = 0.9$. (a) from initial time to t=1000. (b) Zoomed trajectory in time interval [950, 1000] 65
4.15 Particle trajectories with $K = 3.8$ and $w = 0.95$ 66
4.16 Monte Carlo trials for different w values with threshold 100 68

5.1 Optimal multi path routing connection pattern by Dijkstra's algorithm. 86
5.2 Comparison of the network total power for strategy 1, strategy 2, strategy 3 and random placement 89
5.3 Comparison of the strategy 1 with different PSO parameter 90
5.4 Comparison of the strategy 1 with PSO and GA 90

6.1 Comparison of the delay as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$ 103
6.2 Comparison of the delay as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$ 104

6.3 Comparison of the delay as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$ 104
6.4 Comparison of the delay as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$ 105
6.5 Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$ 106
6.6 Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$ 107
6.7 Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$ 108
6.8 Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$ 108
6.9 Comparison of overhead packets as a function of pause time for node number 50 and maximum velocity of $20ms^{-1}$ 109
6.10 Comparison of the overhead packets as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$ 110
6.11 Comparison of overhead packets as a function of pause time for node number 50 and maximum velocity of $20ms^{-1}$ 111
6.12 Comparison of the overhead packets as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$ 112
6.13 Parallel coordinate presentation for pause time 300s and maximum velocity of $20ms^{-1}$ 113
6.14 Parallel coordinate presentation for pause time 300 and maximum velocity of $1ms^{-1}$ 114

7.1 Average number of nodes with empty battery for 50 total nodes with maximum velocity of $1ms^{-1}$ 127
7.2 Average number of nodes with empty battery for 50 total nodes with maximum velocity of $10ms^{-1}$ 127
7.3 Average number of data packets delivered to destination for 50 total nodes with maximum velocity of $1ms^{-1}$ 128
7.4 Average number of data packets delivered to destination for 50 total nodes with maximum velocity of $10ms^{-1}$ 129
7.5 Average number of control packets used for 50 total nodes with maximum velocity of $1ms^{-1}$ 130
7.6 Average number of data packets delivered to the destination for 50 total nodes with maximum velocity of $10ms^{-1}$ 131

List of Tables

4.1 Threshold and Instability count for 1000 Monte Carlo runs 68

5.1 PSO simulation parameters 87
5.2 GA simulation parameters 88
5.3 Locations, data generating rate, and initial energy for each sensor nodes 89

Abbreviations

PSO Particle swarm optimisation

DSDV Destination sequenced distance vector

DSR Dynamic source routing

AODV Ad hoc on demand routing protocols

WSN Wireless sensor networks

CPU Central processing unit

MAC Media aceess control

GPS Glopal positioning systems

QoS Quality of Services

GA Genetic algorithm

SwAN Swarm intelligence based routing algorithm for mobile ad hoc networks

ARA Ant based routing in mobile ad hoc networks

SEAR Swarm intelligence based energy aware routing

Acknowledgments

First and foremost, my deepest gratefulness goes to supervisor Dr V. Kadirkamanathan, for his invaluable guidance and support throughout my Ph.D. study. His insights and suggestions have enlightened me not only on academic thinking and technical problem solving but also on carreer choice. I feel extremly luckly to have Dr V.Kadirkamanathan as my supervisor who is always dedicated to the success of his students.

I am greateful for the support and assistance from all current and former members of the Signal Processing and Complex Systems research group. I have especially benefitted from many fruitful discussions and helps from Mike Dewar and Sean Anderson.

Finally, I would like to express my special thanks to my family. It is their love, understanding and support that enable me to accomplish this thesis.

I would like to dedicate this thesis to my loving parents ...

Chapter 1

Introduction

Ad hoc wireless networks consist of nodes interconnected by multihop wireless communication links. Unlike conventional wireless networks, ad hoc networks have no fixed infrastructure or centralised control [1]. Most of the ad hoc networks are self-organising networks where the necessary control and management are provided only by the interaction among the mobile nodes. These networks are highly attractive for future pervasive computing environments. Ad hoc networks can be rapidly deployed, reconfigured and tailored to any specific application [2]. On the other hand, swarm intelligence, as demonstrated by biological swarms in nature, has several self-organising properties and provides distributed problem solving without centralised control [3]. This thesis investigates swarm intelligence applications to wireless ad hoc and sensor networks.

1.1 Background

Recent advances in wireless communications and digital electronics have enabled rapid development in pervasive or ubiquitous computing environments. New handheld devices such as personal digital assistants (PDA), wearable computers and mobile phones enhance both information processing and accessing capabilities with mobility. Recent advances in computing and communication provide small devices, sensors and actuators increasingly with communication capabilities. These technological advances

extend our living environment to a fully pervasive computing environment. Mobile ad hoc networking is one of the most important technologies supporting pervasive computing [4].

Significant research in wireless ad hoc networks has been ongoing for nearly 30 years, also under the names of packet radio or multi-hop networks. Recently, the field has become prominent with a rapid expansion due to the advancement of inexpensive, widely available wireless devices and the interest in mobile computing. Ad hoc networks are a collection of communications devices (nodes) that wish to communicate, but have no fixed infrastructure available, and have no pre-determined organisation of available links. Individual devices are responsible for dynamically discovering which other devices they can communicate with. These networks imply not only mobility and wireless connections, but also the frequent joining and leaving of nodes, often changing interconnection patterns and the possibility of multi-hop routing among mobile nodes. Ad-hoc networks management and control techniques must be based on local knowledge, using decentralised control mechanisms, and be capable of adapting rapidly to changing network conditions [2].

Routing algorithms in mobile ad hoc networks have attracted a great deal of attention amongst the research community from the beginnings of the research in ad hoc networks until the present time. Early work focused on finding feasible routes without considering energy costs or quality of services (QoS). Later research then focused on energy efficient and QoS based routing algorithms. Many routing protocols have been proposed for ad hoc networks: ad hoc on demand distance vector (AODV) routing [5], zone routing protocol (ZRP) [6], dynamic source routing (DSR) protocol [7], cluster based routing protocol (CBRP) [8] and destination sequenced distance vector (DSDV) [9]. However, most of the existing protocols are well suited for one scenario, but not for all scenarios. For example, DSR performs well in static networks but lacks in the mobile networks but AODV performs better than DSR in mobile networks. Furthermore, routing protocols that have a good performance in small networks may not perform well in large networks where scalability issues are not considered or studied well. To achieve good routing performance in all situations, different routing strategies should be used to be adaptable in all different scenarios.

Wireless sensor networks [10] take a special role in the ad hoc networking field and offer a powerful combination of distributed sensing, computing and communication. Sensor networks provide endless opportunities, but at the same time pose great challenges of energy scarcity and limited computational capabilities. Even though sensor networks share common technical issues with wireless ad hoc networks, routing is very challenging due to sensor nodes being prone to failure and most of the sensor nodes having limited computational capabilities [10]. Several routing protocols have been proposed for wireless sensor networks [11, 12, 13, 14, 15]. Extensive research has focused on almost every layer of network protocol and energy efficient routing. One of the other main design issues for wireless sensor networks is the node placement problem. There are only a few mathematical models for analysing the fundamental performance of information routing and control in wireless sensor networks [16, 17]. Such models are necessary to understand how different parameters such as the position of nodes, number of nodes, energy levels and data rates affect the performance of the networks.

Swarm intelligence appears in certain insect species such as ants, bees and birds where the individual insects are not generally considered to be intelligent. However, the group behaviour of autonomous members give intelligent behaviour through complex interaction between members. The distributed way of problem solving approach in swarm intelligence can potentially solve numerous problems of future communication networks.

Swarm intelligence techniques have successfully been applied in telecommunication networks routing and control [18, 19]. A computer program based on ant foraging principles that routes telephone calls efficiently has been developed in [19]. When the phone calls are re-routed through the better part of its network, the process not only allows those calls to get through quickly, but also enables the congested areas to recover from the overload. The ultimate application of swarm intelligence might be on the future pervasive computing environment where communication networks are becoming increasingly diverse and heterogeneous. Swarm intelligence techniques have also been very successfully applied in the optimisations known as ant colony optimisation [18] and particle swarm optimisation [20]. Other main areas of application are

in unmanned aerial vehicles (UAV) [3] and robotics [21].

Swarm intelligence boasts a number of advantages for mobile ad hoc networks due to the use of distributed control and mobile agents. Swarm intelligence algorithms are distributed, adaptive, robust and scalable and have several self-organising properties such as positive feedback, negative feedback, randomness and multiple interaction where these components facilitate the natural systems to accomplish complex tasks with unsophisticated and simple individual members. These characteristics have resulted in the design of distributed and adaptive algorithms for self organised ad hoc networks where the need for seamless interaction of numerous heterogeneous network components (nodes) presents a great challenge.

1.2 Contributions of the Thesis

This thesis considers swarm intelligence algorithms and their applications to wireless ad hoc and sensor networks. The thesis can be divided into two parts: the first part deals with particle swarm optimisation analysis and its application to the sensor network optimisation problem; the second part considers the application of swarm intelligence to the ad hoc networks routing problem. The thesis clearly shows that swarm intelligence methods have a very useful role to play in the management and control problems associated with wireless ad hoc and sensor networks. .

The first part of this thesis deals with the analysis of the newly proposed swarm intelligence based algorithm, the so called particle swarm optimisation. It has been empirically shown that the algorithm performs well when exposed to many optimisation problems [22]. Since these type of algorithms are based on a sequence of random parameters which are usually not independent, it is difficult to theoretically analyse algorithms behaviour. Previous stability analysis of the particle swarm optimiser was restricted to the assumption that all parameters are non-random, in effect a deterministic particle swarm optimiser. This thesis analyses the stability of the particle dynamics without this restrictive assumption, using Lyapunov stability analysis and the concept of passive systems. Through this analysis sufficient conditions are derived for stability. The prediction based on this theory is that the stability of the particle

dynamics requires an increase in the maximum value of the random parameter when the inertia factor is reduced.

In addition, the thesis also considers the application of particle swarm optimisation for the wireless sensor network sink node placement problem. Wireless sensor networks consist of small battery powered devices with limited energy resources where replacement of the energy source is not feasible after deployment. Energy efficiency is therefore a vital design issue in wireless sensor networks. There has been extensive research effort on how to design protocols and algorithms to prolong network lifetime. In this thesis, three algorithms are proposed to deploy sink node in an optimal way which minimises the energy consumption of the overall wireless sensor networks. The sink node placement is formulated into a complex nonlinear programming problem. Since the problem is NP-hard in general, particle swarm optimisation is chosen to solve the the problem effectively.

The second part of this thesis proposes two different routing methods for mobile ad hoc networks. The first protocol is a Swarm intelligence based routing Algorithm for mobile ad hoc Networks (SwAN). The node mobility is the most important factor for route changes in the mobile ad hoc networks and the routing algorithm compensates the mobility of the nodes. Mapping the pheromone laying and following behaviour of biological ants, the algorithm allows each node in the network to choose the next node for information packets to be forwarded on the basis of mobility influenced pheromone table. The effectiveness of the proposed approach is demonstrated through an extensive simulation study.

The second protocol is a Swarm intelligence based Energy Aware Routing (SEAR) algorithm for mobile ad hoc networks. SEAR uses the distributed method of packet forwarding in the dynamic ad hoc networks. Most of the wireless nodes are powered by batteries where nodes can not be recharged. The node energy level should be considered in packet forwarding. Therefore the neighbour node remaining energy level and its drain rate information are predicted using HELLO packets and then related to a pheromone decay as found in natural foraging ant systems. The effectiveness of the proposed approach is demonstrated through an extensive simulation study using the NS2 network simulator.

1.3 Thesis Outline

Chapter 2 gives an introduction to the swarm intelligence and its applications to the optimisation and routing problems. Firstly, the swarm intelligence properties are described, then the proposed swarm intelligence optimisation techniques such as ant colony optimisation, particle swarm optimisation and bacterial foraging optimisation are briefly outlined. The ant based routing algorithms AntNet and Ant based control are detailed.

Chapter 3 gives an introduction to the wireless ad hoc and sensor networks. It first introduces wireless networks at general level, then the ad hoc networks and sensor networks are discussed. Ad hoc network applications and challenges are identified, followed by details of existing routing protocols. Finally, sensor networks opportunities and challenges are discussed.

Chapter 4 provides the stability analysis of the particle swarm optimisation. The shortcoming of the previous stability analysis of the particle swarm optimiser is discussed. Then, using Lyapunov stability techniques, an analysis is given of the particle dynamics without the assumption that parameters are non-random. Sufficient conditions for stability are identified and illustrative examples are given.

Chapter 5 investigates the energy efficient sink node placement in sensor networks using particle swarm optimisation. Firstly, an introduction to the WSN is presented and then the WSN mathematical modelling work in the literature is discussed. The WSN optimisation problem is formulated as a nonlinear optimisation problem and suitable optimisation techniques are detailed. The numerical results demonstrate how the algorithms produce efficient solutions for the sink node placement problem.

Chapter 6 proposes a Swarm intelligence based routing protocol for mobile Ad hoc Networks (SwAN). The existing ant based routing algorithms are described in detail. The SwAN algorithm is described and pheromone laying and following behaviour of biological ants are related in to the algorithm. Finally, a simulation model for NS2 network simulator is outlined and the effectiveness of the suggested approach is demonstrated through an extensive simulation study.

Chapter 7 provides a Swarm intelligence based Energy Aware Routing (SEAR)

protocol for ad hoc and sensor networks. Existing energy aware routing algorithms are discussed and the motivation of the SEAR protocol is described. The SEAR algorithm implemented in NS2 network simulator is detailed. Finally, simulation results are presented to demonstrate how the algorithm significantly increases the lifetime of the network.

Finally, Chapter 8 draws conclusions and proposes a number of future directions for research in swarm intelligence and applications in ad hoc and sensor networks.

1.4 Publications

The publication resulting from the research work reported in this thesis are:

"Stability Analysis of the Particle Dynamics in Particle Swarm Optimiser", V. Kadirkamanathan, K. Selvarajah, and P. J. Fleming, IEEE Transactions on Evolutionary Computation, Volume 10, Issue 3, June 2006: Pages 245–255.

"Stability Analysis for the stochastic best Particle Dynamics of a Continuous-time Particle Swarm Optimiser", K. Selvarajah, V. Kadirkamanathan, and P. J. Fleming, Proceeding of the Adaptation in Artificial and Biological Systems Conference, April 2006.

"Swarm Intelligence based routing for mobile ad hoc networks", K. Selvarajah, V.Kadirkamanathan, Proceeding of the 12th European Wireless Conference, April 2006.

"Energy efficient sink node placement in sensor networks using particle swarm optimisation", K. Selvarajah, V. Kadirkamanathan, accepted the publication in Fifth International Workshop on Ant Colony Optimisation and Swarm Intelligence, 2006.

In addition, the following submissions have also been made.

"Neighbour Aware Routing Algorithm for Mobile Ad hoc Networks" , K. Selvarajah, V. Kadirkamanathan, submitted to IEEE Transactions on Parallel and Distributed Systems, August 2006.

" Sink node placement in sensor networks using particle swarm optimisation" , K. Selvarajah, V. Kadirkamanathan, submitted to IEEE Transactions on Evolutionary Computation, August 2006.

Chapter 2

Swarm Intelligence

2.1 Introduction

Swarm intelligence (SI) is the property of a system whereby the collective behaviours of (unsophisticated) agents interacting locally with their environment cause coherent functional global patterns to emerge [3]. Swarm intelligence provides a basis with which it is possible to explore collective (or distributed) problem solving without centralised control or the provision of a global model [3].

Individual insects are not generally considered to be intelligent. However, the group behaviour of, for instance, a flock of birds, school of fish [23], swarm of bees and colony of ants [18, 3] show a connection between optimisation, engineering applications and swarm behaviour. For example, the foraging techniques of ant colonies can be applied to the shortest-path problems and other optimisation problems in the real world [3]. For a bird to participate in a flock, it must have behaviours such as collision avoidance, velocity matching and flock centreing that allow it to coordinate its movements with those of its flock mates. These techniques can be applied to solve nonlinear function optimisation problems and unmanned aerial vehicle (UAV) control problems.

All these tasks in the social insects are accomplished without centralised control. That is, individuals communicate via direct or indirect contact. The individuals involved do not have a global understanding of the tasks or solutions. Rather, these

complex behaviours emerge as a result of numerous individuals sensing and acting locally on the basis of a simple rule [24]. However, the group behaviours of social insects can serve as valuable models for problem solving strategies in the design and management of complex systems. Natural systems are self-organised, distributed, adaptive and robust which make them interesting for the perspective of complex systems [24]. This simple yet powerful approach can help solve a number of real world engineering and business issues. New methods are being developed based on swarm intelligence techniques for solving distributed problems [25]. They are based on the principles underlying the behaviour of natural systems such as ant colonies, bird flocking and fish schooling. The approach emphasises distributed solutions to problems, direct or indirect interactions among relatively simple agents, flexibility and robustness. In this chapter a overview of properties of swarm intelligence, ant algorithms, analysis ant algorithms, particle swarm optimisation and bacterial swarm optimisation are detailed.

2.2 Properties of Swarm Intelligence

Social insects possess some collective behaviour in order to perform tasks [24]. The following sections give a description of the primary mechanisms which determine the collective behaviour of the social insects during food foraging and other tasks. The main properties are identified in social insects such as self-organisation, positive feedback, negative feedback, randomness and multiple interaction.

Self-organisation is a phenomenon that exists in complex adaptive systems, including living systems and human organisational systems. This type of behaviour has been intensively studied in biology, sociology, management science and organisational theory [24]. Self-organising systems are typically comprised of a large number of commonly similar components. The most effective approach to study the self-organising systems is to first understand some basic modes of interaction among the components: positive feedback, negative feedback, randomness and multiple interaction. These components yield the natural systems to accomplish complex tasks with unsophisticated and simple individual behaviour [25].

2.2.1 Positive feedback

Positive feedback is a feedback system in which the systems responds to the perturbation in the same direction as perturbation [26]. Most self-organising systems use positive feedback this includes recruitment and reinforcement [25]. The simple example of positive feedback can be found in ant foraging behaviour where ants in the good path attract other ants to that particular path. When ant colonies are foraging, positive feedback mechanisms allow ants to find the shortest path from the food source to the nest [27]. Ants deposit a chemical substance called pheromone while searching for a food source. The other ants can identify the amount of pheromone deposited and have a natural tendency to follow the trail. This trail laying and trail following mechanism is a positive feedback in the natural foraging behaviour of ants [3].

2.2.2 Negative feedback

Negative feedback is a feedback system responds to the perturbation in the opposite direction as perturbation [26]. Negative feedback is used in many types of amplification systems to stabilise their operating characteristics. So, negative feedback is a counterbalance of positive feedback and it helps to stabilise the collective pattern.

Negative feedback can be found in many biological systems such as the baroreflex in blood pressure regulations. Many biological process in the human anatomy also use negative feedback from regulating of body temperature to the regulating of blood glucose levels. In the ant foraging mechanism negative feedback is achieved through the pheromone evaporation technique. This property assists ants to recognise less effective routes and it also helps ants to explore other new food sources [3].

2.2.3 Randomness

The term randomness is often used in statistics to signify well defined statistical properties such as lack of bias or correlation. Randomness is also referred to as ampli-

fication of random fluctuations and it is crucial to the discovery of the new solutions in the natural systems. For example, in ant foraging, ants follow the pheromone trail with some level of error. This phenomenon may seem inefficient, but ants may find new, unexploited food sources and recruit nest mates to these food sources.

2.2.4 Multiple Interaction

Multiple interaction is a key feature of the natural systems where a single individual can not find the optimal solutions; moreover an individual should be able to make use of the results of their own activities as well as the activities of others. Self-organising systems generally require a minimal density of mutually tolerant individuals. For example, ants can self-organise if individuals use of other ants pheromone information [3].

2.2.5 Stigmergy

Self-organisation in social insects requires interactions among other insects. The interaction can have direct or indirect forms. Direct interactions are, for example, antennation, mandibular contact, visual contact and chemical contact. Indirect interactions are where two individuals interact indirectly when one modifies the environment and the other responds to the new environment at a later time. This form of interaction is called stigmergy [3]. However, it does provide a general mechanism that relates individual and colony level behaviour: individual behaviour modifies the environment, which in turn modifies the behaviour of other individuals. Stigmergy in social insects shows how problems can be solved easily by replacing coordination through direct communications by indirect interaction. This simple yet powerful idea can be used to design simple agents and reduce communication among the agents.

2.3 Ant Algorithms

Swarm intelligence was originally inspired by the observation of real ant colonies. Ants have an interesting method of finding and transporting food to their nest. What

is interesting is that ants are able to discover the shortest path to a food source and share that information with other ants through stigmergy [28]. Ants achieve stigmergic communication by laying a chemical substance called pheromone. When they search for a food source, they lay a pheromone. Ant pheromone is a very strong stimulant and when an ant senses pheromone, it greatly increases the probability that the ant will follow the trail of the pheromone. The amount of pheromone that has been left on a certain path indicates the number of ants that have taken that path recently. When another ant is searching for food, it will very likely take the path marked by a stronger pheromone concentration. Ant algorithms [25] is a new heuristic algorithm which can be applied to solve different optimisation and control problem because it is versatile, robust and a population based approach [25, 29].

2.3.1 Artificial ants

Artificial ants are modelled from real ants with some additional features to solve the real world problems efficiently. Thus, artificial ants are not intended to model the real ant. The intention is to keep the artificial ants simple, but this may not be efficient as they are required to manage a high level of complexity. For example, artificial ants have a memory to remember past experience. Artificial ants deposit virtual pheromones on the path they have been on before. Other properties of self-organising systems can be modelled by the artificial ant depending on the nature of the problem to be solved. For example, randomness can be incorporated by artificial ants to perform stochastic walks on a graph, consisting of a series of stochastic steps [25].

2.3.2 Ant colony optimisation

Ant colony optimisation is a class of heuristic search algorithms that have been successfully applied to solving combinatorial optimisation problems such as the travelling salesman problem [27] and the quadratic assignment problem [30]. In ant colony based algorithms, a set of artificial ants move on the graph which represents the instance of the problem: while moving they create solutions and modify the prob-

lem representation by adding collected information. A number different optimisation problems has empirically shown the effectiveness of ant colony optimisation. Recently, a convergence proof for the ant colony optimisation algorithms was developed in [31] to proof the efficiency of ant based optimisation algorithms.

The first application of an ant colony optimisation was in response to the travelling salesman problem (TSP) as a benchmark problem. TSP is the most studied NP-hard problem in combinatorial optimisation [27]. In the TSP, one has to find a closed tour of minimal length connecting n given cities. Each city must be visited once and only once. Let d_{ij} be the distance between cities c_i and c_j. The problem can be more generally defined on a graph $G = (V, E)$, where the cities are vertices V and the connections between the cities are the edges of the graph E. The ants build solutions in parallel by visiting sequentially the cities of the graph. On each edge (i, j) of the TSP graph an artificial pheromone trail $\tau_{ij}(t)$ is maintained. The values $\tau_{ij}(t)$ are used by ants to direct the way they build tours. They are updated by means of a reinforcement procedure: once an ant has completed a tour it updates the edges it has crossed by adding a quantity of pheromone proportional to the success of the tour. More formally, at iteration t, after completing its tour $T_k(t)$, the kth ant lays a quantity of pheromone $\delta\tau_{ij}^k(t)$ on each edge (i, j) belonging to $T_k(t)$. $\delta\tau_{ij}^k(t)$ is a function of the length L_k of the tour $T_k(t)$ [27]

$$\delta\tau_{ij}^k(t) = \begin{cases} Q/L_k & \text{if edge } (i,j) \in T_k(t) \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

Where Q is an adjustable parameter. Ants build solutions using a probabilistic transition rule. The Probability $p_{ij}^k(t)$ with which an ant k in a city i at iteration t chooses the next city j to move to is a function of the following:

- For each ant, a list is maintained that contains all the cities that the ant has already visited in order to prevent cities from being visited more than once. The list grows during one tour until it is full, and is then emptied at the end of the iteration. J_i^k is the set of cities that remain to be visited by ant k when ant k is in the city i. By exploiting J_i^k an ant k can avoid visiting a city more than once.

- An heuristic measure η_{ij} of the desirability of adding edge (i, j) to the TSP is $\eta_{ij} = 1/d_{ij}$, i.e., the inverse of the distance between cities i and j.

- The amount $\tau_{ij}(t)$ of artificial pheromone on the edge connecting i and j: Formally p_{ij}^k is given by

$$p_{ij}^k = \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}(t)]^\beta}{\sum_{l \in J_i^k}^n [\tau_{il}(t)]^\alpha [\eta_{il}(t)]^\beta} \tag{2.2}$$

where α and β are two adjustable parameters that control the relative influences of the pheromone trail $\tau_{ij}(t)$ and heuristic desirability η_{ij}. The above algorithm could not perform well without pheromone evaporation. In fact, because the initial exploration of the search space is mostly random, the values of the pheromone trails in the initials phases are not very informative and it is therefore necessary that the system slowly forgets these initials values to allow the ants to move towards better solutions. Pheromone decay is implemented by introducing a coefficient of evaporation $\rho, 0 < \rho < 1$, such that

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{2.3}$$

where $\Delta\tau_{ij}(t) = \sum_{k=1}^m \Delta\tau_{ij}^k(t)$, and m is the number of ants. This update equation ensures efficient solution space exploration. The trail intensity must be allowed to decay, otherwise the algorithm will end up prematurely in a sub-optimal solution. The above algorithm [27] have shown that ant colony optimisation is an interesting novel approach to the parallel stochastic optimisation of the Travelling Salesmen Problem(TSP).

2.3.3 AntNet

In AntNet [18, 3], routing is determined by means of very complex interactions of forward and backward network exploration ants. The idea behind this sub-division of ants is to allow the backward ants to utilise the useful information gathered by the forward ants on their trip from source to destination. A routing table is built based on the probability distribution functions derived from the trip times of the routes discovered by ants.

Let s and d be the source and destination node. In the network, there are N nodes, and each node i is characterised by a routing table $R_i = [r^i_{n,d}(t)]_{k_i,N-1}$, where k_i is the number of neighbour nodes of node i. The entry $r^i_{n,d}(t)$ in the routing table of node i represents the probability at time t of choosing node n as the next node for a packet to be delivered to for the destination node d. A local estimate, denoted by $E_i = \{\mu_{i,d}, \sigma_{i,d}\}$, comprises average estimated trips times, $\mu_{i,d}$, the average time to go from node i and d, and their associated standard deviations $\sigma_{i,d}$. The routing table R_i and the local estimate E_i are updated by the ants in the ways described below [3]

- Each network node launches forward ants to a randomly selected destination at regular times intervals.
- The forward ant selects a path to the destination randomly based on the current routing table.
- The forward ants create a stack, pushing in journey times for every node as that node is reached.
- When the forward ant reaches the destination, it generates a backward ant and transfers to it all of its memory.
- The backward ant makes the same path as that of its corresponding forward ant, but in the opposite direction.
- Arriving in a node i coming from a neighbour node $i-1$, the backward ant updates the local estimate E_i and the Routing table R_i.

The local model E_i is updated from the times elapsed for the arrival in every node on the path i to d, i.e., the path followed by the forward ant starting from the current node i, is used to update the corresponding sample means and variances. The routing table R_i is changed by increasing the probability $r^i_{i-1,d}(t)$ associated with other neighbour nodes $i-1$ and the destination node d, decreasing the probabilities $r^i_{n,d}(t)$ associated with other neighbour nodes of n, $n \neq i-1$, for the same destination. The trip time $T_{i,d}$ experienced by the forward ant is used to assign the probability

increments. It gives an indication about the goodness r of the followed route because it is proportional to its length from a physical point of view and from a traffic congestion point of view.

The value stored in the mode E_i is used to score the journey times so that they can be transformed to a reinforcement signal r, where the goodness value $r \in [0, 1]$. The value r is used by the current node i as positive reinforcement for node $i-1$ the backward ants comes from. The probability $r^i_{i-1,d}$ is increased by the reinforcement values as follows.

$$r^i_{i-1,d}(t+1) = r^i_{i-1,d}(t) + r(1 - r^i_{i-1,d}(t)) \tag{2.4}$$

Probabilities $r^i_{n,d}(t)$ for destination d of the other neighbour node n receive a negative reinforcement as follows:

$$r^i_{n,d}(t+1) = r^i_{n,d}(t)(1-r), \quad n \neq i-1 \tag{2.5}$$

The network then applies these probabilities in a deterministic way, in choosing the next hop based on the highest probability in sending the packets.

The above proposed algorithm showed very good performance and robustness under all experimental conditions with respect to other existing algorithms [18].

2.3.4 Ant based control

Ant-based Control (ABC) is another successful swarm intelligence based algorithm designed for communication networks [19, 32]. This algorithm shares many key features with AntNet, but has important differences. The basic principle shared is the use of a multitude of agents interacting using stigmergy. The algorithm is adaptive and exhibits robustness under various network conditions. It also incorporates randomness in the motion of ants, which increases the chances of discovery of new routes. The routing table of every node is the same and satisfies the constraints as in the AntNet algorithm. The update philosophy of the routing table is a little different from AntNet. There is only one group of ants, which are launched from the sources

to various destinations at regular time intervals. The ants are eliminated once they reach their destination. Therefore, the probabilities of the routing tables are updated as ants visit the nodes, based on the life of the ant at the time of the visit. The life of the ant is the sum of the delays of the nodes $T = \sum_{i=1}^{n} D_i$ where the delays D_i are given by $D_i = ce^{-dS}$, c and d are design parameters, and S is the spare capacity of each node in the network. Then, a step size is defined for that node according to $\delta_r = a/T + b$, where a and b are both design parameters. This step size rule is intuitive, because it assigns a greater step size to those ants who are successful at reaching the node faster. The routing table is then updated according to [19]:

$$r^i_{i-1,s}(t+1) = \frac{r^i_{i-1,s}(t) + \delta_r}{1 + \delta_r} \tag{2.6}$$

$$r^i_{n,s}(t+1) = \frac{r^i_{n,s}(t)}{1 + \delta_r}, \quad n \neq i-1 \tag{2.7}$$

where s is the source node, i is the current node and $i-1$ the previous node. It should be noted that the ant uses and updates the routing table at the same time.

2.4 Analysis of Ant Algorithms

Ant algorithms have been successfully applied to several optimisation and routing problems. Ant algorithms are based on a sequence of random decisions made by artificial ants. It is difficult to analyse ant algorithms behaviour theoretically with random variables. The following sections briefly explain modelling ant foraging behaviour and provide an analysis of ant colony optimisation and ant routing.

2.4.1 Modelling ant foraging

Deneubourg et al. [33] developed model of ant foraging behaviour, the behaviour of which closely matches the experimental observations. Here we explain the experiment carried out by Deneubourg [33]. A food source is connected to an ant nest by a bridge with two equally long branches. When the experiment starts the ants select randomly, with equal probability, one of the branches. Nevertheless, random fluctuations cause

a few more ants to randomly select one branch. A simplified assumption is that the amount of pheromone on a branch is proportional to the number of ants that have used the branch in the past. This assumption implies that pheromone evaporation is not taken into account. In the model, the probability of choosing a branch at a certain time depends on the total amount of pheromone on the branch. U_m and L_m are the numbers of ants that have used the branches after m ants have crossed the bridge, with $U_m + L_m = m$. The probability $P_U(m)$ with which the $(m+1)$th ant chooses the one branch is

$$P_U(m) = \frac{(U_m + k)^h}{(U_m + k)^h + (L_m + k)^h} \tag{2.8}$$

while the probability of the other branch is $P_L(m) = 1 - P_U(m)$. This functional form of the probability of choosing a branch over the other was obtained from experiments on trail-following and the parameters h and k allow the model to match experimental data.

It is easy to modify the above experiment with different the length of branches. In this case, because of the same pheromone laying mechanism as in the previous situation, the shortest branch is most often selected. The first ants returning to the nest are those that took the shortest path twice, so more pheromone is present on the shortest branch than on the long branch immediately after these ants have returned, stimulating nest mates to choose the short branch. This differential length effect explains how ants ultimately choose the shortest of the two paths without using any global knowledge about their environment [34].

2.4.2 Analysis of ant colony optimisation

In recent years, a number of applications to different types of NP-hard combinatorial optimisation problems have empirically shown the effectiveness of ant colony optimisation. Convergence proof for a class of ant colony optimisation algorithms can be found in [31]. Here, the authors prove two theorems that apply to the ant colony optimisation.

First theorem states that for any small constant $\epsilon > 0$ and for a sufficiently large number of algorithm iterations t, the probability of finding an optimal solution

at least once is $P^*(t) \geq 1 - \epsilon$ and that the probability tends to 1 the number of iterations tends to infinity. The second theorem states that starting from a fixed number of iterations after the optimal solution has been found, the pheromone trails will be higher on the connections belonging to the optimal solution than on other connection. Each iteration step chooses the connection with higher pheromone trail will deterministically construct the optimal solution [31].

2.4.3 Analysis of ant routing algorithms

Researchers have used the ant colony metaphor to design distributed adaptive routing algorithms for communication networks and the effectiveness of these algorithms have been shown empirically. Theoretical analysis can be found in [35] where the authors analysed two algorithms; the first of these is based on the earlier work by Holland et. al. [19] for call routing in telephone networks. They then developed an another algorithm which is a natural multi-path routing algorithm that is applicable to data networks [35]. A convergence result was also presented for each algorithm based on probability theory.

The analysis of mobile agent based algorithm for network routing and management was developed in [36]. This work develops some preliminary analysis on the ant algorithm with regard to its population growing property and jumping behaviour. The theoretical analysis shows that the expected number of agents in a node is shown to be no more than $(1+\max_i\{|\pi_i|\})km$, where $|\pi_i|$ is the number of neighbouring hosts of the ith host, k is the number of agents generated per request and m is the average number of requests. The authors also derived condition for the expected number of agents in a node no to be more than $(1 + \max_i\{\pi_i\})$. These theoretical findings are useful when using the ant like mobile agents for network management and control.

2.5 Particle Swarm Optimisation

Particle swarm optimisation (PSO) is a parallel evolutionary computation technique developed by Kennedy and Eberhart [20], inspired by social behaviour of bird

flocking and fish schooling. Let assume that a group of birds are randomly searching for food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is. However, they know how far away the food is in each iteration of the search. What is the best strategy to find the food?. The answer is to follow the bird which is nearest to the food. This is the essence behind the idea of particle swarm optimisation.

PSO algorithm maintains a population of initial solutions called "particles". All the particles have fitness values which are evaluated by the fitness function which is to be optimised, and have velocities which direct the motion of the particles. The particles search through the problem space by following the current best particles. Some of the attractive features of the PSO includes the ease of implementation and the fact that no gradient information is required. PSO can be applied to a variety of different non linear function optimisation problems.

2.5.1 The PSO algorithms

The PSO formulation defines each particle as a potential solution to a problem in d-dimensional space with memory of its previous best position and the best position amongst all particles, in addition to a velocity component. At each iteration, the particles are combined to adjust the velocity along each dimension which in turn is used to compute the new particle position. The PSO update equations are given by,

$$v_{i,j}(t+1) = v_{i,j}(t) + c_1 r_1 (p_{i,j}(t)^{(l)} - x_{i,j}(t)) + c_2 r_2 (p_{i,j}(t)^{(g)} - x_t) \quad (2.9)$$

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \quad (2.10)$$

where $v_{i,j}(t)$ is the particle velocity at the tth iteration associated with the jth dimension of the velocity of ith particle, $x_{i,j}(t)$ is the particle position of jth dimension of particle i at the tth iteration, $p_{i,j}(t)^{(l)}$ is the best local position or the particle's best position thus far, $p_{i,j}(t)^{(g)}$ is the best global position or the best solution amongst all particles and w is the inertia factor. The algorithms use two independent random sequences $r_1 \sim \mathcal{U}(0,1)$ and $r_2 \sim \mathcal{U}(0,1)$, $\mathcal{U}(0,1)$ is a uniform distribution in the interval (0,1) and c_1, c_2 are constants. These constants are called the acceleration coefficients

and they influence the maximum size of the step that a particle can take in a single iteration.

The personal best position associated with particle i is the best position that the particle has visited so far, yielding the highest fitness value for that particle. The position yielding a smaller function value is regarded as having a higher fitness, for a minimisation task. f denotes the objective function that is being minimised. The personal best position at time t for a particle i can be calculated as follows

$$p_i(t+1)^{(l)} = \begin{cases} p_i(t)^{(l)} & \text{if} \quad f(x_i(t+1)) \geq f(p_i(t)) \\ x_i(t+1) & \text{if} f(x_i(t+1)) < f(p_i(t)) \end{cases} \tag{2.11}$$

The best global position at time t is the position yielding a smaller function value among the all population. The definition of $p_i(t)^{(g)}$ is given below.

$$p_i(t)^{(g)} = \arg\min_{x_i(t)} \{f(x_1(t)), f(x_2(t)), ...f(x_s(t))\} \tag{2.12}$$

where s is the population size used in the algorithm.

2.5.2 Modification to the PSO

Several modifications have been made for improving the performance of the PSO. These proposals usually involve changes to the PSO update equations, without changing the structure of the algorithms. This normally introduces more control parameters to the algorithm.

Inertia factor

The initial algorithm proposed by Eberhart and Kennedy was found to lack convergence. One of the earliest modification to the PSO was the introduction of the inertia factor by Shi and Eberhart [37]. The idea behind this inertia factor was to reduce the velocity for the next iteration from the previous iteration. The new modified PSO velocity equation, is given by

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_1 (p_{i,j}(t)^{(l)} - x_{i,j}(t)) + c_2 r_2 (p_{i,j}(t)^{(g)} - x_t) \tag{2.13}$$

The original PSO velocity update equation can be obtained if $w = 1$. Empirical evidence shows that for $0 < w < 1$ better convergence results for the PSO are obtained.

Constriction factor

The constriction factor model describes a way of choosing the parameter values w, c_1 and c_2 so that convergence is ensured. A modified velocity update equation with the constriction factor is as follows [38]:

$$v_{i,j}(t+1) = \chi(v_{i,j}(t) + c_1 r_1(p_{i,j}(t)^{(l)} - x_{i,j}(t)) + c_2 r_2(p_{i,j}(t)^{(g)} - x_t)) \tag{2.14}$$

where

$$\chi = \frac{2}{|2 - \rho - \sqrt{\rho^2 - 4\rho}|} \tag{2.15}$$

and $\rho = c_1 + c_2$, $\rho > 4$.

2.5.3 Analysis of the PSO algorithms

The analysis of PSO algorithm may give proper design parameter to find the solution effectively. The primary aim of PSO however is optimisation while maintaining convergence. We present an analysis of a particle in the PSO algorithm and the parameter choice as developed in [39]. The PSO update equations with inertia weight are repeated for convenience,

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_1(p_i(t)^{(l)} - x_{i,j}(t)) + c_2 r_2(p_i(t)^{(g)} - x_t) \tag{2.16}$$

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \tag{2.17}$$

The following analysis only consider a single particle and single dimension without loss of generality, so that the subscript i and j in the equations 2.16 and 2.17 can be dropped. Now, by substituting 2.16 into 2.17, the following non-homogeneous recurrence can be obtained,

$$x_{t+1} = (1 + w - \alpha^{(l)} - \alpha^{(g)})x_t - wx_{t-1} + \alpha^{(l)}y + \alpha^{(g)}\hat{y} \tag{2.18}$$

where $\alpha^{(l)} = c_1 r_1(t)$ and $\alpha^{(g)} = c_2 r_2(t)$. Here $\alpha^{(l)}$, $\alpha^{(g)}$ and w are assumed to be constants. It has been claimed that the analysis of a particle dynamics can be easily performed when largest values of $\alpha^{(l)}$ and $\alpha^{(g)}$ are used for the analysis. A condition was derived for convergence of PSO particle by solving the non-homogeneous recurrence in 2.18 as follows,

$$w > \frac{c_1 + c_2}{2} - 1 \tag{2.19}$$

Another particle swarm optimisation convergence analysis and parameter selection can be found in [40]. The deterministic version of the particle swarm optimisation algorithm is considered. This analysis uses the theory of discrete-time dynamic systems where the behaviour of the particle depends on the eigenvalues of the system matrix. Through this analysis the following set of conditions are derived for the parameter selection of the PSO algorithm.

$$w < 1 \tag{2.20}$$

$$c_1 + c_2 < 2(w + 1) \tag{2.21}$$

The condition in 2.19 is equivalent to in 2.21.

A similar analysis based on the discrete dynamical systems theory was developed in [38]. The analysis includes the constriction factor and the parameters are assumed to be constant for the analysis. The exact relationship between deterministic and random versions of the algorithms are not explained well in these analyses while success of most search algorithms are influenced by the random parameters.

2.6 Bacterial Swarm Foraging for Optimisation

Researchers consider individual and group foraging in bacteria, organisms that are much simpler than ants or birds, yet can still work together for the benefit of the group. The *E.coli* bacterium is probably the best understood microorganism and the bacterium has a guidance system that enables it to search for food and try to avoid unwanted substances. For instance, it moves away from acidic environments and towards more neutral ones. Passino et al. use the ideas from bacterial foraging to solve non linear optimisation problems. The biology and physics underlying

the chemotactic (foraging) behaviour was described in [41]. The method applied to optimisation problems is described below.

2.6.1 The algorithm

First, suppose that θ is the position of a bacterium and $J(\theta)$ represents the combined effects of attractants and repellants from their environment, with for example, $J(\theta) < 0, J(\theta) = 0$, and $J(\theta) > 0$ representing that the bacteria at location θ is in nutrient-rich, neutral, and noxious environments respectively. Basically, chemotaxis is a foraging behaviour that implements a type of optimisation where bacteria try to climb up the nutrient concentration, avoid noxious substances, and search for ways out of neutral media. Let j be the index of the chemotatic step. Let k be the index for the reproduction step and l be the index of the elimination-dispersal event. Let

$$P(j,k,l) = \{\theta^i(j,k,l)|i = 1,2,3....S\} \tag{2.22}$$

represent the position of each member in the population of the S bacteria at the j th chemotatic step, kth reproduction step, and lth elimination-dispersal event. $J(i,j,k,l)$ denote the cost at the location of the ith bacterium $\theta^i(j,k,l)$. For actual bacteria, S can be very large, but for optimisation tasks much smaller population sizes are used and kept fixed. Let N_c be the length of the lifetime of the bacteria as measured by the number of chemotatic steps they take during the life time. $C(i) >$ 0,1,2,3...,S, denotes a basic chemotatic step size that is used to define the lengths of the steps during the runs. To represent a tumble, a unit length random direction, say $\phi(j)$, is generated. After tumble,

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\phi(j) \tag{2.23}$$

so that $C(i)$ is the size of the step taken in the random direction specified by the tumble. If at $\theta^i(j+1,k,l)$ the cost $J(i,j+1,k,l)$ is better than at $\theta^i(j,k,l)$, then another step size $C(i)$ is taken in the same direction. This is continued as long as the cost is continuously reduced, but only up to a maximum number of steps, N_s. The cell tends to keep moving only if it is heading in the direction of increasing nutrients

and favourable environments. Basically, this is foraging behaviour that implements a type of optimisation where each cell try to find lower and lower values for $J(\theta)$.

2.6.2 Parameter selection

The success of the bacterial foraging optimisation algorithm requires the proper selection of a variety of parameters. First, increasing the population size of the bacteria, it is apparent that increasing the size of S can significantly increase the computational complexity of the algorithm. If the C_i are too large, then the search may tend to jump out of the search region. On the other hand, if the C_i values are too small, it takes a long time to converge. The large value for N_c will result in many chemotactic steps and more function evaluations but incurs more computational complexity. If N_c is chosen to be too small, the algorithm can easily converge to a local minimum due to premature convergence [41]. A complete stability analysis of social foraging swarms was developed in [42], where the analysis is of the continuous version of bacterial swarm optimisation [41]. The study provided conditions for collective convergence to more favourable regions.

and favourable environments. Basically, this is foraging behaviour that implements a type of optimisation where each cell try to find lower and lower values for $J(\theta)$.

2.6.2 Parameter selection

The success of the bacterial foraging optimisation algorithm requires the proper selection of a variety of parameters. First, increasing the population size of the bacteria, it is apparent that increasing the size of S can significantly increase the computational complexity of the algorithm. If the C_i are too large, then the search may tend to jump out of the search region. On the other hand, if the C_i values are too small, it takes a long time to converge. The large value for N_c will result in many chemotactic steps and more function evaluations but incurs more computational complexity. If N_c is chosen to be too small, the algorithm can easily converge to a local minimum due to premature convergence [41]. A complete stability analysis of social foraging swarms was developed in [42], where the analysis is of the continuous version of bacterial swarm optimisation [41]. The study provided conditions for collective convergence to more favourable regions.

Chapter 3

Wireless Ad Hoc and Sensor Networks

3.1 Introduction

The history of wireless networking started in the 1970s with the advent of the development of packet radio networks by the Defence Advanced Research Projects Agency (DARPA). Recent developments in wireless communication and digital electronics and high processing power available in computing and communication devices have combined to put more and better computer based applications into the hands of the population. Extensive work has been done recently in integrating these elements into traditional networks such as the Internet [2]. However, a mobile user will want to communicate in situations in which no fixed infrastructure is available, either because it may not be economically practical or physically possible to provide the necessary infrastructure or because the expediency of the situation does not permit its installation. For example, a class of students may need to interact during a lecture, friends or business associates may run into each other in an airport terminal and wish to share files, or a group of emergency rescue workers may need to be quickly deployed after an earthquake or flood. In such situations, a collection of mobile nodes with wireless network interfaces may form a temporary network without the aid of any established infrastructure or centralised administration. This type of wireless network is known

as an ad hoc network [1].

In the ad hoc networking field, wireless sensor networks [10] take a special role. A sensor network is composed of a large number of small sensor nodes, which are typically densely (and randomly) deployed inside the area in which a phenomenon is being monitored or controlled. Wireless ad hoc networking techniques also give the basis for sensor networks. However, the special constraints imposed by the unique characteristics of sensing devices, and by the application requirements, make many of the solutions designed for ad hoc networks (generally) not suitable for sensor networks. The combinations of sensors and ad hoc networks will have significant application in the pervasive computing environment and pose real challenges [43]. In this chapter a detailed overview of ad hoc networks, sensor networks and their challenges and applications is presented.

3.2 Wireless Networks

The history of modern wireless communications started in 1896 with Marconi who demonstrated wireless telegraphy by sending and receiving Morse code using high power transmitters. In 1907, the first commercial trans-Atlantic wireless service was initiated using huge ground stations. Since then the world has seen the rapid development of communication technology and other technologies which, lead to the advent of modern wireless systems. With the advent of new digital systems, the existence of wireless data communication became very common. In fact, the GSM and IS-95 standards evolved, in the 1990s, to include wireless data transmission as an integral part of their service. Finally, third-generation (3G) wireless systems, based on CDMA technologies, are being developed and deployed with data and voice communication [44].

Wireless networks offer the following: productivity, convenience and cost advantages over traditional wired networks. It provides mobile users with access to real-time information so that they can roam the network without getting disconnected. This mobility supports productivity and service opportunities not possible with wired networks [45]. Installing a wireless system can be fast and easy and can eliminate the

need to pull cable through walls and ceilings. The wireless network can be extended to places which cannot be reached by wired networks. The wireless networks offer more flexibility and adapt easily to changes in the configuration of the network. The initial investment required for wireless network hardware can be higher than the cost of wired network hardware, however overall installation expenses and life-cycle costs can be significantly lower in dynamic environments. The wireless systems can be configured in a variety of topologies to meet the needs of specific applications and installations. Configurations can be easily changed and range from peer-to-peer networks suitable for a small number of users to large infrastructure networks that enable roaming over a broad area [2].

3.2.1 Types of wireless networks

Wireless networks consist of wireless devices equipped with wireless transceivers using radio frequency to transmit data from one location to another. Today, different kinds of networks exist in practice which are wireless wide area networks (WAN), wireless local-area networks (LAN) and wireless personal area networks (PAN) [46].

Wireless local area networks use high frequency electromagnetic waves, either infrared (IR) or radio frequency (RF), to transmit information for one point to another. Wireless LANs are set up to provide wireless connectivity within a finite coverage area. Typical application areas might be a hospital (for patient care systems), a university, the airport, or a gas plant. Wireless LANs work in an unregulated part of the spectrum, so that anyone can create their own wireless LAN, either in the home or office. In principle, anyone can have complete control over where coverage is provided. Traffic from multiple users is modulated into the radio waves at the transmitter, and extracted at the receiver. Multiple radio carriers can coexist in the same physical space and at the time, without interfering with each other by transmitting at different frequencies (FDMA), in different time slots (TDMA) or using specific codes for each message (CDMA) [47].

Wireless personal area networks provide wireless connectivity over distances of up to 10m, but this range allows a computer to be connected wirelessly to a nearby

printer, or a cell phone's hands-free headset to be connected wirelessly to the cell phone. Personal area networks are slightly different to wireless LANs in one important respect. In the wireless LAN cases, networks are set up first, which devices then use. In the Personal Area Network case, there is no independent pre-existing network. The participating devices establish an ad hoc connection between the devices when they are within range, and the network is dissolved when the devices pass out of range [1].

The one used most successfully today is a wireless wide area network built on top of a wired network and thus creating reliable infrastructured wireless networks. An example of this wireless network is cellular networks. A cellular network provides cell phones or mobile devices, to use a more general term, with wireless access to the public switched telephone network (PSTN). The service coverage area of a cellular network is divided into many smaller areas, referred to as cells, each of which is served by a base station (BS). Here the base station is fixed and it is connected to the mobile telephone switching office (MTSO). With the wireless link between the BS and mobile devices, mobile devices are able to communicate with wire line phones in the PSTN [43].

3.2.2 Enabling technologies

The success of a network technology is connected to the development of networking products at a better price. A major factor in achieving this goal is the availability of appropriate networking standards. IEEE 802.11 [48] and Bluetooth are the two main standards for short-range communication.

The IEEE adapted the first wireless local area network standard, named IEEE 802.11, with the data rates up to 2 Mbps. Since then, several task groups have been created to extend the IEEE 802.11 standard. The task groups 802.11b and 802.11a have completed their work by providing two relevant extentions to the original standard which are often referred to with Wireless Fidelity (Wi-Fi). The IEEE 802.11 standard specifies a MAC layer and Physical layer for wireless networks. The Physical layer uses either a direct sequence spread spectrum or a frequency hopping spread spectrum to transmit data between nodes. The MAC layer offers two different types

of service: distributed coordination function (DCF) and point coordination function (PCF) [49].

Bluetooth technology [50] is a de-facto standard for low cost, short range radio links between Laptops, mobile phones and other portable devices. The Bluetooth special interest group (SIG) releases the Bluetooth specifications with the joint effort of many leading companies including IBM, Intel, Lucent, Microsoft, Motorola and Nokia. A Bluetooth unit integrated into a microchip enables wireless communication between portable and/or fixed electronic devices. As a low cost, low power solution and industry wide support, Blutooth wireless technology has already started to revolutionise the personal connectivity market by providing freedom from wired connections. The main strength of Bluetooth is its ability to simultaneously handle both data and voice transmissions. It is capable of supporting one data channel and up to three voice channels, or one channel supporting voice and data. This capability combined with ad hoc device connections and automatic service discovery make it a good solution for mobile devices and Internet applications. This combination enables innovative solutions such as a mobile hands-free headset for voice calls and other applications in mobile devices [50].

3.3 Ad Hoc Networks

It would be beneficial to provide a definition of the phrase ad hoc as it is used in the context of mobile wireless networks. Common definitions of this phrase are "for a specific purpose or occasion" or "for this case alone". There is no universally accepted definition for wireless ad hoc networks, but there are a few features that are shared by most descriptions of such networks. It is probable that the main difference between ad hoc networks and conventional cellular technology is the apparent lack of a centralised entity with an ad hoc network. There are no base stations or mobile switching centres in ad hoc networks. In other words, all network protocols must operate in a distributed manner.

Mobile ad hoc networks are self-organising mobile wireless networks that do not rely on a pre-existing infrastructure to communicate. Nodes of such networks have

limited transmission range and packets may need to traverse multiple nodes before reaching their destination. Lack of a fixed network and the nature of the nodes give rise to challenges such as reliable data routing, dynamic network topologies, changing environments, selfish nodes and scarce radio resources [51, 2].

3.3.1 Ad hoc networking issues

Ad hoc networking inherits the traditional problems of wireless communications and wireless networking. In addition, the multi hop nature and lack of infrastructure adds a number of characteristics, complexities, and design constraints that are specific to ad hoc networking [45, 52]. Some of the main issues in ad hoc networking will be discussed in the following section.

Mobile ad hoc networks do not depend on any established infrastructure or centralised administration. Each node operates in a distributed peer-to-peer mode. It acts as an independent router and generates independent data. Network management has to be distributed across different nodes, which creates added difficulty in fault detection and management. There is no default router available and every node acts as a router and forwards to each others packets that enable information sharing between mobile nodes. In mobile ad hoc networks, because nodes can move arbitrarily, the network topology, which is typically multi-hop, can change frequently and unpredictably, resulting in route changes, frequent network partitions, and possibly packet losses [52].

Each mobile node might have different capabilities, resulting in variability in processing power. Designing network protocols and algorithms for this heterogeneous network can be complex, requiring dynamic adaptation to the changing conditions such as power and channel conditions, traffic load and congestion. The existing management algorithms are mostly designed to work on fixed or relatively small wireless networks. Many mobile ad hoc network applications involve large networks with tens of thousands of nodes. Network management issues in a large network consisting of nodes with limited resources are not straight forward, and present many challenges that are still to be solved in areas such as addressing, routing, location management

and scalability [2, 45].

3.3.2 Ad hoc network applications

Ad-hoc networks are suited for use in situations where an infrastructure is unavailable or to deploy one is not possible. One of the main possible uses of mobile ad hoc networks is in environments where there is a need for temporary networks such as in a conference or meeting. A mobile ad hoc network can also be used in disaster recovery where an entire communication infrastructure is destroyed and restoring communication quickly is crucial. By using a mobile ad hoc network, an infrastructure could be set up instantly [2].

Early ad hoc network applications have been military oriented but non-military applications have also been growing rapidly since then. The rapid advances in ad hoc networking research have attracted the business and industrial sectors as well as other public sectors. The introduction of new technologies such as Bluetooth and IEEE 802.11 facilitates the deployment of ad hoc networks outside the military domain. Mobile ad hoc networking applications have appeared mainly in emergency services, disaster recovery and environment monitoring. Mobile ad hoc networking technology makes several other applications possible such as, in personal area networking, home networking, law enforcement operation, search and rescue operations, commerce and education [2].

3.4 Ad Hoc Network Routing Protocols

In the ad hoc networks environment, the routing is normally the distributed version of the shortest path routing. Each node in the network maintains a preferred neighbour for each destination. When a node receives a data packet, it forwards the packet to the preferred neighbour associated with the destination. Each routing table is constructed, maintained and updated to achieve the common objective of the optimal path for the data packets. The routing methods can be categorised into two primary classes, which are link state and distance vector.

The link-state approach is closer to the centralised version of the shortest path method. Each node in the network maintains the information of the network topology with the cost of each link. Each node periodically broadcasts the link cost to all other nodes using flooding. As a node receives the cost information, it updates the information and performs a shortest-path algorithm to choose its next hop for each destination.

In distance vector algorithms, every node i maintains, for each destination x, a set of distances d_{ij}^x where j ranges over the neighbour of i. Node i treats neighbour k as a next hop for a packet destined for x if d_{ik}^x is the lowest cost. The succession of next hops chosen in this manner lead to destinations along the shortest path. Each node periodically broadcasts to its neighbours in order to keep the distance estimates up-to-date. Distance-vector algorithm is the classical Distributed Bellman-Ford (DBF) algorithm [53]. It has some benefits over the link-state method such as being computationally more efficient, easier to implement and requiring less storage space. However, it is well known that this algorithm can cause the formation of both short-lived and long-lived loops. Furthermore, within an ad hoc mobile environment enforcing any such internodal coordination mechanism would be difficult due to the rapidly changing topology of the underlying routing network.

In order to facilitate communication within the network, a routing protocol is used to discover routes between nodes. The primary goal of such an ad hoc network routing protocol is correct and efficient route establishment between a pair of nodes so that messages may be delivered in a timely manner. Since the advent of DARPA packet radio networks in the early 1970s, numerous protocol have been proposed in the Internet Engineering Task Force (IETF) for execution in ad hoc mobile networks: Ad hoc On Demand distance Vector (AODV) routing [5], Zone Routing Protocol (ZRP) [6], Dynamic Source Routing (DSR) protocol [7], Cluster Based Routing Protocol (CBRP) [8] and Destination Sequenced Distance Vector (DSDV) [9]. Preliminary classification of the routing protocols can be made depending on whether they use a unicast, multicast or broadcast. Broadcast is the basic mode of operation over a wireless channel; each message transmitted on a wireless channel is generally received by all neighbours located within the hop from sender. Unicast forwarding means a one

to one communication, i.e., one source transmits data packets to a single destination. Multicast routing protocols come in to play when a node needs to send the message to multiple destinations [54].

Mobile Ad hoc Networking routing protocols are typically subdivided into two main categories: Proactive routing protocols and reactive on-demand routing protocols. proactive routing protocols attempt to maintain routing tables and updates at fixed time intervals. As the routing information is usually maintained in tables, these protocols are sometimes referred to as table-driven protocols. Reactive on demand routing protocols, on the other hand, establish the route to a destination only when there is a demand for it. The source node through the route discovery process usually initiates the route requested. Once a route has been established, it is maintained until either the destination becomes inaccessible, or until the route is no longer used, or expired [2]. The following sections briefly describe the key features of the DSDV, DSR and AODV protocols.

3.4.1 Destination sequenced distance vector (DSDV)

DSDV [9] is a hop by hop distance vector routing protocol requiring each node to periodically broadcast routing table updates. Each DSDV node maintains a routing table listing the next hop for each reachable destination. Packets are transmitted between hosts of the network by using routing which are stored at each host in the network. Each routing table, at each of the stations, lists all available destinations, and the number of hops to each. Each routing table entry is tagged with a sequence number which is originated by the destination station. To maintain the consistency of routing tables in a dynamically varying topology, each station periodically transmits updates, and transmits updates immediately when significant new information is available. Routing information is advertised by broadcasting or multicasting the packets which are transmitted periodically and incrementally as topological changes are detected – for instance, when stations move within the network. The DSDV protocol requires each mobile station to advertise, to each of its current neighbours, its own routing table.

3.4.2 Dynamic source routing (DSR)

The Dynamic source routing (DSR) [7] protocol is an on demand routing protocol which is based on the concept of source routing. Nodes in the network are required to maintain the route table that contains the source routes of which the node is aware. In other words, DSR uses hop by hop routing, with each packet to be routed carried in its header, that is, the complete ordered list of nodes through which the packet must pass. The main advantage of this kind of routing is that intermediate nodes do not need to maintain up-to-date routing information in order to route the packets to respective destinations, since the packets already contain all the nodes information. Any hops in the route table that move out of wireless transmission range of the the next or previous hop are monitored. Routing table entries are updated as new routes are learned. DSR protocol is composed of two mechanisms that work together to allow the discovery and maintenance of source routes in the ad hoc network. Route discovery is the mechanism by which a node wishing to send a packet to a destination node obtains a source route to the destination and it allows any node in the ad hoc network to dynamically discover a route to any other host in the ad hoc network. A node initiating a route discovery broadcasts a route request packet which may be received by those nodes within wireless transmission range of it. The route request reaches the destination, referred to as the target of the discovery, then the destination node launches a route reply packet to the initiating node. If the route discovery is successful, the source node receives a listing sequence of network nodes through which it may reach the destination. Route Maintenance is the mechanism by which packets' sender detects if the network topology has changed such that it can no longer use its route to the specific destination, since the nodes listed in the route have moved out of range of its transmission range. Conventional routing protocols integrate route discovery with route maintenance by continuously sending periodic routing updates. The periodic updates will eventually reflect the changes to all other routes, resulting in the addition of new routes. However, there are no periodic messages of any kind from any of the mobile nodes. Instead, while a route is in use, the route maintenance mechanism monitors the operation of the route and informs the sender of any routing

errors. Route maintenance in the DSR uses end-to-end acknowledgements rather than hop-by-hop acknowledgements.

3.4.3 Ad hoc on demand distance vector (AODV)

AODV [5] is essentially a combination of both DSR and DSDV. It borrows the basic on-demand mechanism of route discovery and route maintenance from DSR, plus the use of hop-by-hop routing, sequence numbers and periodic beacons from DSDV. AODV uses sequence numbers to ensure the freshness of routes. It is loop-free, self-starting, and scales to large numbers of mobile nodes. It is an on demand algorithm, meaning that it builds routes between nodes only as desired by source nodes. It maintains these routes as long as they are needed by the sources. When a node needs a route to some destination, it broadcasts a route request message to its neighbours, including the last known sequence number for that destination. The route request is flooded in a controlled manner through the network until it reaches a node that has a route for the destination. Each node that forwards the route request creates a reverse route for itself back to the source node. When the route request is reached via a route to the destination from the source node, that node generates a route reply that contains the number of hops necessary to reach the source and the sequence number for the destination most recently seen by the node generating the reply. Each node that participates in forwarding this reply back to the originator of the reply, creates a forward route to the destination. The state created in each node along the path from source to destination is know as a hop-by-hop state; that is, each node only remembers the next hop and not the entire route, as would be done in source routing.

As long as the route remains active, it will continue to be maintained. A route is considered active as long as there are data packets periodically travelling from the source to the destination along that path. Once the source stops sending data packets, the links will time out and eventually be deleted from the intermediate node routing tables. If a link break occurs while the route is active, the node sends a route error message to the source node to inform it of the unreachable destination.

After receiving the route error, if the source node still desires a route, it can re-initiate route discovery. In order to maintain routes, AODV normally requires that each node periodically transmits a HELLO message, with a default rate of once per second. AODV is considered as a de facto mobile ad hoc routing protocol because it gives high performance.

3.5 Ad Hoc Network Energy Conservation

Most mobile nodes rely on battery power and battery power is limited. Energy conservation represents one of the greatest constraints in designing a routing algorithm for mobile ad hoc networks [55, 56]. Power saving mechanisms at the operating system level include strategies for CPU scheduling [57] and hard-disk management [58]. However, in small mobile wireless devices, networking activities have a major impact on energy consumption. Therefore power-saving strategies in the ad hoc networks can be divided into two classes: local strategies and global strategies.

Local strategies operate inside a node and keep the network interface in a power saving mode with a minimum impact on transmit and receive operation. These mechanisms operate at the Physical and MAC layer where power-saving strategies are designed to avoid transmitting when the channel is congested. While a node transmits a packet, the other nodes within the same interference and carrier-sensing range must remain silent. Therefore, these nodes are switched-off with no impact on system behaviour. For example, in PAMAS [59] a node turns off the radio when it overhears a packet not addressed to it. A comparison of a number of MAC layer protocols from the energy efficiency standpoint, can be found in [60] and references therein.

Global strategies are utilised to maximise the network life time. These are based on the network-wide approach to power saving when a region is dense in terms of nodes, only a small number of them need to be turned on in order to forward the packets. To achieve this, a set of nodes is identified which must guarantee network connectivity while the remaining nodes can spend most of the time in the sleep state to maximise energy saving. Controlling the power of the transmitting node is the

other main direction for achieving power saving in ad hoc networks. A reduced transmission power also allows for the reuse of frequencies, which can help to increase the total throughput of the network and to minimise interference [61]. In addition, the minimum energy based routing protocols in mobile ad hoc networks are developed [62, 63, 64].

3.6 Sensor Networks

Wireless sensor networks (WSN) are capable of observing the environment, processing data and making decisions based on these observations. WSN consist of a large number of sensors, referred to as nodes. A sensor node integrates hardware and software for sensing, data processing and communication. Improvements in wireless network technology interfacing with emerging micro sensors based technology is allowing sophisticated, inexpensive, storage, processing and communication capabilities to be unobtrusively embedded into our every day life. Although sensor network technologies are not new, technological barriers of performing wireless sensor networks have been limited in the past. Some of the benefits of the newer, more advanced sensor nodes have the potential to form large scale networks. Sensor network applications can be found in a wide variety of areas including industrial, military, bio-medical and environment monitoring. The sensor on these applications may be small or large, and the networks may be wired and wireless. However, wireless networks of micro sensors probably offer the most potential in changing the world of sensing [2].

3.6.1 Challenges in sensor networks

Even though all sensor networks share common technical issues, various applications may have different challenges. Unlike traditional networks, a sensor node may not need an address, i.e., sensor network applications are focused on the data generated by sensors. Data is named by attributes and applications request data matching certain attributes values. So, the communication primitive in this system is a request. Traditional networks are designed to accommodate a wide variety of

applications, but the routers in the traditional networks differ from sensor networks where intermediate sensor nodes can perform application specific data aggregation and caching or informed forwarding of requests for data. It is necessary to find the suitable architecture to provide efficient communication between the nodes. Sensor network nodes coordinate to perform high level sensing tasks according to the application of interest. Clearly, this kind of communication can be structured in a centralised manner. Individual sensors report their data to a central node, but this centralised algorithm will not contribute much to sensor networks for several reasons such as a single point of failure, energy inefficiency and scalability [2]. Researchers proved that sensor network coordination applications are better realised using distributed algorithms, which means sensor nodes only communicate with sensors within some neighbourhood, yet the overall computation achieves a desired global objective. However, design of localised algorithms for sensor networks pose challenges in data-centric application-specific sensor networks [52]. In addition to the wireless communication problems, wireless sensor networks pose technical challenges in network discovery, network control and routing, collaborative information processing, querying, and tasking.

Topology information of the sensor networks is essential for a sensor in the network to operate properly. Each node needs to know the information of its neighbour nodes. In the immobile networks, the topology of the networks is fixed and usually known. In the case of mobile networks, since the topology of nodes change over time, methods should be provided to discover the topology changes. Generally global knowledge is not required, since each sensor node only interacts with its neighbours. In addition to knowledge of the topology, each sensor also needs to know its own location. When GPS is not feasible or too expensive, other means of location information have to be provided.

Each node in the sensor networks collaborate to collect and process data to generate application specific information. Processing data from more sensors generally results in a better performance, but, in turn, also requires more communication resources. Therefore, one needs to consider the multiple trade-offs between the performance and resource utilisation in collaborative signal and information processing

using micro sensors.

3.6.2 Sensor network applications

Recent advances in sensor network technologies have introduced more and more practical applications of wireless sensor networks and development is continuing to emerge. Military sensing, air traffic control, traffic surveillance, video surveillance, industrial and manufacturing automation, distributed robotics, environment monitoring, and building and structures monitoring are the current and potential applications of wireless sensor networks.

In the monitoring applications where specialised sensor nodes that are able to detect temperature changes and other useful information can be deployed in high risk areas of a forest to receive early warning of a forest fire. In relation to indoor surveillance, sensor networks can be used to provide security in super markets, art galleries and other facilities. Sensor networks can be used in intrusion detection and tracking where sensor nodes are deployed along the border of a battlefield to detect, classify, and track intruding personnel and vehicles.

3.6.3 Routing in Sensor networks

Routing in sensor networks is highly challenging due to several characteristics that distinguish them from wireless communication and wireless ad hoc networks. Sensor nodes have less energy and computational capabilities than nodes in ad hoc networks. Sensor nodes are prone to failures and it is not possible to build global addressing schemes. Due to such differences, many new algorithms have been proposed for WSN. These routing mechanisms have addressed the characteristics of the sensor nodes along with the application and architecture. Proposed routing protocols for the WSN can be classified as data-centric protocols [11], hierarchical protocols [12, 13], location based protocols [14] and QoS-aware protocols [15].

Data centric protocols are query based and depend on the naming of desired data, which helps in eliminating many redundant transmissions. SPIN [11] is the first data centric protocol. Subsequently, directed diffusion has been developed and

has become notably efficient in data-centric routing. The main aim of hierarchical routing is to efficiently maintain the energy consumption by involving them in multi-hop communication within a particular cluster and by performing data aggregation in order to decrease the number of transmitted messages to sink node. LEACH [12] is one of the first hierarchical routing approaches to sensor networks. LEACH assumes that all the nodes can transmit with enough power to reach the sink node if needed, the node can use power control to vary the amount of its transmit power, and nodes organise themselves into clusters, with one node acting as a cluster head. While there are advantages to using the LEACH distributed cluster formation algorithm, it offers no guarantee about the placement and/or number of cluster head nodes. This is the basis for the LEACH-centralised algorithm (LEACH-C) [12] that uses centralised clustering algorithms for clustering nodes and cluster head selection. PEGASIS [13] is an improvement of LEACH which forms chains from sensor nodes so that each node transmits and receives from its neighbour and only one node is selected from that chain to transmit to the sink node. Location based protocols require location information of sensor nodes. MECN [14] is a location based routing protocol for sensor networks and maintains its location using low power GPS. Finally, QoS-aware protocols are based on general network flow modelling and protocols that strive to meet some QoS requirements along with the routing function [15].

3.6.4 Placement methods in sensor networks

Previous research in sensor networking has focused on developing protocols and algorithms and has largely ignored the optimal node placement issues. Recently works have emerged which address the node placement problem in WSN. For example, in [65], authors proposed an algorithm for sensor placement, wherein a minimum number of sensors are deployed to provide sufficient grid coverage of the sensor nodes. The optimisation framework was probabilistic due to the uncertainty associated with sensor detections. Optimal information extraction in energy limited wireless sensor networks was proposed in [16]. Here, the authors addressed the need for a symmetric methodology by developing formal nonlinear optimisation models of static WSN that

yield fundamental performance bounds and optimal designs. Two problems were addressed, namely maximising the total information gathered subject within the energy constraints and minimising the energy usage subject to information gathering. Energy aware node placement in wireless sensor networks can be found in [17] where the authors formulated a constrained multi-variable nonlinear programming problem to determine both the locations of the sensor nodes and data transmission patterns. The sensor networks model considered by most researchers has a single static sink node located randomly in the sensor networks region [16, 17]. It is apparent that no research has been done for the sink node placement problem within a multi-hop nature.

Nodes closer to a sink node will experience heavier traffic load since they not only collect data within their sensing range, but also forward data to the sink node or to the next node. Such an unbalanced traffic load introduces an asymmetric power consumption among the sensor nodes. Hence, the node placement methods will have considerable impact on the life time of WSN. In [66], the authors investigated the energy provisioning for wireless sensor networks and considered a two-tier wireless networks. It was proposed that a relay node be deployed into the network to mitigate network geometric deficiencies and prolong the network lifetime.

Chapter 4

Stability Analysis of Particle Swarm Optimisation

4.1 Introduction

Particle swarm optimisation (PSO) is a swarm intelligence technique developed by Eberhart and Kennedy [20], inspired by social behaviour of bird flocking and fish schooling. PSO is a population based search process where individuals, referred to as particles, are candidate solutions to the optimisation problem at hand. Particles change their state by evolving in a multi-dimensional search space until an equilibrium or optimal state has been reached or until computation limitations are exceeded. PSO has been shown to be a very effective optimiser, especially in large complex search spaces [67]. Empirical evidence has accumulated that the algorithm is a useful tool for optimisation [22]. PSO has been applied to many optimisation problems in engineering [68, 69, 70, 71, 72, 73]. On the algorithmic front, extensions have been made to deal with dynamical environments and efficient exploration [74, 75]. More recently, multi-objective particle swarm optimisers have also been derived [76, 77, 78, 79]. Additional operators have been incorporated into the basic particle swarm optimisation scheme such as the selection operator in genetic algorithms [80] and a neighbourhood operator [81]. The similarity between a population of particles in swarm optimisation and a population of genotypes in genetic algorithms has resulted

in a comparison between the two [67] where the PSO shows good performance than genetic algorithm for chosen optimisation problems.

The first analysis of the simplified particles behaviour was carried out by Kennedy [82] who showed the different particle trajectories for a range of design choices for the gain through simulations. In [83], the authors showed that a particle in a simple one dimension PSO system follows a path defined by a sinusoidal wave, randomly deciding on both its amplitude and frequency. The first formal analysis of the stability properties of the algorithm was carried out in [38]. Essentially, the analysis required the simplification of the standard stochastic PSO to a deterministic dynamical system by treating the random coefficients as constants. The resulting system was a second order linear dynamical system whose stability depended on the system poles or the eigenvalues of the state matrix. A similar analysis based on the deterministic version of the PSO was also carried out in identifying regions in the parameter space that guarantees stability [84]. The issue of convergence and parameter selection was also addressed in [37, 40]. However, the authors acknowledge the limitations of their results which do not take the stochastic nature of the algorithm into account. Similar analysis on a continuous-time version of PSO have also been carried out in [85]. A Lyapunov analysis approach has also been adapted in [42] for the social foraging swarms, different to the PSO, in a continuous-time setting.

In this chapter, we provide a stability analysis of the stochastic particle dynamics. The analysis is made feasible by representing the particle dynamics as a nonlinear feedback controlled system as formulated by Lure [26, 86]. Such systems have a deterministic linear part and a nonlinear and/or time varying gain in the feedback path. It is well known that the stability of such nonlinear feedback systems cannot be determined by analysing the stability of all possible linear feedback systems resulting from the nonlinear and/or time varying gain being replaced by constant linear gain values spanning the entire range of the gain [86]. Known as *Aizerman's conjecture*, its implication is that the stability conditions derived by treating the particle dynamics as deterministic, is not valid for the stochastic case in general.

4.2 Particle Swarm Optimisation

The PSO formulation defines each particle as a potential solution to a problem in d-dimensional space with memory of its previous best position and the best position amongst all particles, in addition to a velocity component. At each iteration, the particles are combined to adjust the velocity along each dimension, which in turn is used to compute the new particle position. Since each dimension is updated independently of others and the only link between the dimensions of the problem space are introduced via the objective functions, analysis can be carried out on the one-dimensional case without loss of generality. The original version was found to lack precision in local search solution. This led to the introduction of an inertia factor in the velocity update in [37], giving rise to the commonly used form of the PSO. The particle dynamics in one dimension is given by

$$v_{t+1} = wv_t + \alpha_t^{(l)}(p^{(l)} - x_t) + \alpha_t^{(g)}(p^{(g)} - x_t), \tag{4.1}$$

$$x_{t+1} = x_t + v_{t+1}, \tag{4.2}$$

where v_t is the particle velocity at the tth iteration, x_t is the particle position at the tth iteration, $p^{(l)}$ is the personal best position or the particle's best position thus far, $p^{(g)}$ is the best global position or the best solution amongst all particles, w is the inertia factor and $\alpha_t^{(l)} \sim \mathcal{U}[0, c_1]$, $\alpha_t^{(g)} \sim \mathcal{U}[0, c_2]$ are random parameters with uniform distributions.

The following statements can be derived from the particle dynamics of (4.1):

- The system dynamics is stochastic and is of order 2.
- The system does not have an equilibrium point if $p^{(g)} \neq p^{(l)}$.
- If $p^{(g)} = p^{(l)} = p$ is time invariant, there is a unique equilibrium point at $v_* = 0$, $x_* = p$.

An equilibrium point, thus exists only for the best particle whose local best solution is the same as that of the global best solution. If asymptotic stability of the dynamics for the best particle can be guaranteed, then this particle will reach the equilibrium

point relating to the best solution is guaranteed. The analysis of the non-best particle is more challenging and is beyond the scope of this paper. Clearly, the conditions outlined for the existence of an equilibrium point does not hold true for any particle at all times in the particle swarm optimisation. There are two points to be made with regard to this. Firstly, convergence to a fixed equilibrium point requires time invariance of the best solution position. Secondly, particles stop improving their solution after a finite number of iterations so that beyond this point the conditions can be deemed to hold.

We proceed to consider the particle dynamics associated with the best particle (local best solution is the same as that of the global best solution),

$$v_{t+1} = wv_t + \alpha_t(p - x_t), \tag{4.3}$$

$$x_{t+1} = x_t + v_{t+1}, \tag{4.4}$$

where $\alpha_t = \alpha_t^{(l)} + \alpha_t^{(g)}$. The combined stochastic parameter is no longer uniformly distributed but satisfies the following inequality:

$$0 < \alpha_t < K, \tag{4.5}$$

where $K = c_1 + c_2$. Note that the use of (4.3) with p as a constant is not valid for non-best particle dynamics. The following expression used in [38], [40], for the deterministic PSO, given that

$$p = \frac{\alpha_t^{(l)} p^{(l)} + \alpha_t^{(g)} p^{(g)}}{\alpha_t} \tag{4.6}$$

is generally time varying if $p^{(g)} \neq p^{(l)}$ and if $\alpha_t^{(l)}$ and $\alpha_t^{(g)}$ are random.

The previous stability analysis [38, 40] proceeded to represent the system in state space form:

$$\begin{pmatrix} x_{t+1} \\ v_{t+1} \end{pmatrix} = \begin{pmatrix} 1 - \alpha_t & w \\ -\alpha_t & w \end{pmatrix} \begin{pmatrix} x_t \\ v_t \end{pmatrix} + \begin{pmatrix} \alpha_t \\ \alpha_t \end{pmatrix} p. \tag{4.7}$$

By treating the random variable α_t as a constant, essentially deterministic particle dynamics, the system dynamics is reduced to a simple time invariant linear second

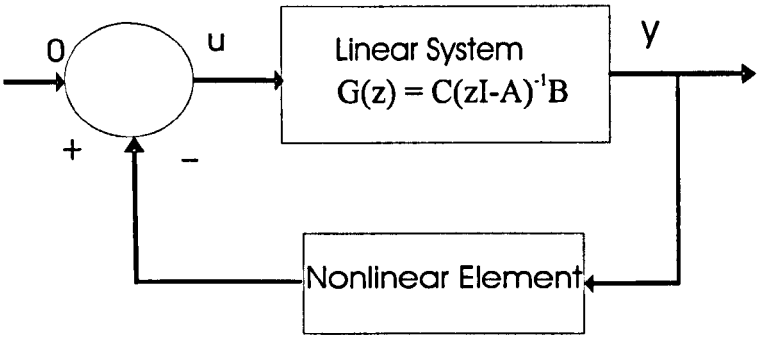

Figure 4.1: Feedback control system representation particle dynamics

order dynamic model. Stability of such a deterministic particle dynamics can be concluded based on the eigenvalues of the state matrix in (4.7), as shown in [38, 84, 40]. The conditions for convergence derived in [84, 40] in our notation are given by

$$w < 1 \quad \text{and} \tag{4.8}$$

$$K < 2(w+1). \tag{4.9}$$

We shall see in section 4.4 that the sufficient conditions for the stability of the stochastic particle dynamics differ from those given in (4.8) and (4.9).

4.3 System Characteristics

We note that the stability analysis of the particle dynamics can be mapped to the problem of absolute stability of nonlinear feedback systems, known as Lure's stability problem[86, 87]. The stochastic particle dynamics is thus represented as a feedback controlled dynamic system as shown in Figure 4.1. The feedback control system representation depicts a time invariant linear plant in the forward path and an output control with time varying gain in the feedback path. The equations governing

the dynamics in this new representation can be expressed as

$$\begin{pmatrix} x_{t+1} \\ v_{t+1} \end{pmatrix} = \begin{pmatrix} 1 & w \\ 0 & w \end{pmatrix} \begin{pmatrix} x_t \\ v_t \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u_t, \tag{4.10}$$

$$y_t = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_t \\ v_t \end{pmatrix}, \tag{4.11}$$

$$u_t = -\alpha_t(y_t - p), \tag{4.12}$$

where u_t is interpreted as the control input signal.

Under the conditions of p being time invariant, the dynamical system equation can be simplified further by introducing the state vector as follows:

$$\xi_t = \begin{pmatrix} x_t - p \\ v_t \end{pmatrix}.$$

The resulting state space representation from (4.10), (4.11), (4.12) is thus,

$$\xi_{t+1} = A\xi_t + Bu_t, \tag{4.13}$$

$$y_t = C\xi_t, \tag{4.14}$$

$$u_t = -\alpha_t y_t, \tag{4.15}$$

where the state matrix A, input matrix B and the output matrix C are given by,

$$A = \begin{pmatrix} 1 & w \\ 0 & w \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \end{pmatrix}. \tag{4.16}$$

Definition (Equilibrium[88]). *ξ_* is an equilibrium point of a dynamical system in the state space form $\xi_{t+1} = f_t(\xi_t)$ if it satisfies $\xi_* = f_t(\xi_*)$ for every $t \geq 0$.*

Remark. *For the PSO, the dynamical systems with feedback can be rewritten in the following state space representation*

$$\xi_{t+1} = (A - \alpha_t BC)\xi_t, \tag{4.17}$$

$$(A - \alpha_t BC) = \begin{pmatrix} 1 - \alpha_t & w \\ -\alpha_t & w \end{pmatrix}. \tag{4.18}$$

If $w \neq 0$, then $(A - \alpha_t BC)$ is nonsingular and hence the only solution that satisfies $\xi_* = (A - \alpha_t BC)\xi_*$ is $\xi_* = 0$. Hence, the particle dynamics specified in (4.13-4.15) has a unique equilibrium point at the origin in the ξ state space.

Remark. *If $p^{(g)} \neq p^{(l)}$, the particle converges to the line that connects its personal best and the global best particle.*

The transfer function of the linear plant is then,

$$G(z) = C(zI - A)^{-1}B = \frac{z}{(z-1)(z-w)}, \tag{4.19}$$

where z is the complex variable associated with Z-Transforms [89].

Remark. *The linear plant has poles at $z = 1$ and $z = w$ and hence is (marginally) stable if $|w| < 1$ and is unstable if $|w| \geq 1$. Poles are also the eigenvalues of A.*

For dynamical systems specified in the state space form, the following properties are of interest and are needed for the analysis in the next section.

Definition (Controllability[90]). *A system is completely controllable if the system state $x(t_f)$ at time t_f can be forced to take on any desired value by applying a control input $u(t)$ over a period of time from t_0 until t_f. Suppose n, m, l are given integers, $A \in \Re^{n\times n}$, $B \in \Re^{n\times m}$, $C \in \Re^{l\times n}$, $D \in \Re^{l\times m}$ and $x_{t+1} = Ax_t + Bu_t$, $y_t = Cx_t + Du_t$ represents the dynamics of the linear systems. Then the pair (A, B) is said to be controllable if*
Rank$[B\ AB...A^{n-1}B] = n$.

Definition (Observability[90]). *A system is completely observable if any initial state vector $x(t_0)$ can be reconstructed by examining the system output $y(t)$ over some period of time from t_0 until t_f. Suppose n, m, l are given integers $A \in \Re^{n\times n}$, $B \in \Re^{n\times m}$, $C \in \Re^{l\times n}$, $D \in \Re^{l\times m}$ and $x_{t+1} = Ax_t + Bu_t$, $y_t = Cx_t + Du_t$ represents the dynamics of the linear systems. Then the pair (C, A) is said to be observable if*
Rank$[C\ CA....CA^{n-1}]^T = n$.

State space representation of the linear part of PSO system is given by

$$\xi_{t+1} = A\xi_t + Bu_t, \tag{4.20}$$

$$y_t = C\xi_t, \tag{4.21}$$

where the state matrix A, input matrix B and the output matrix C are given by

$$A = \begin{pmatrix} 1 & w \\ 0 & w \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 \end{pmatrix}. \tag{4.22}$$

According to the controllability definition, the PSO dynamics of (4.13) gives rise to

$$(B\ AB) = \begin{pmatrix} 1 & 1+w \\ 1 & w \end{pmatrix}. \tag{4.23}$$

Rank$(B\ AB) = 2$.
Hence the linear part of the PSO system is controllable.
According to the observability definition, the PSO dynamics of (4.13) gives rise to

$$(C\ CA)^T = \begin{pmatrix} 1 & 0 \\ 1 & w \end{pmatrix}. \tag{4.24}$$

Rank$(C\ CA)^T = 2$, if $w \neq 0$.
Hence, the linear part of the PSO systems is observable, provided $w \neq 0$. The linear plant pair $\{A, B\}$ is controllable and pair $\{A, C\}$ is observable.

Remark. *The implication of complete controllability and observability of the particle dynamics is that the dynamics is always that of a second order system (not reduced to first order for example due to pole-zero cancellation). Such a condition is necessary for us to use the method of positive real lemma in the next section.*

The time varying memoryless feedback gain satisfies the so called sector condition $\alpha_t \in (0, K)$ and hence satisfies the following inequality:

$$u_t^2 + Ku_t y_t \leq 0. \tag{4.25}$$

4.4 Stability Analysis

The stability analysis is carried out using the concept of passive systems and Lyapunov stability [86]. We begin this treatment by explaining some basic concepts and their interpretations.

Definition ([86]). *The linear plant has a stable matrix A, if its eigenvalues lie strictly inside the unit circle in the Z-plane or equivalently $|\lambda_i\{A\}| < 1$ for all i. Here $\lambda_i\{\cdot\}$ represents the ith eigenvalues of A.*

Remark. *The linear plant in the feedback representation of the particle dynamics has a semi-stable A matrix with a simple pole on $|z| = 1$ when $|w| < 1$.*

Definition ([86]). *A dynamical system is said to be* passive *if there is a non-negative scalar function $V(\xi)$ with $V(0) = 0$ which satisfies*

$$V(\xi_{t+1}) - V(\xi_t) \leq y_t u_t. \tag{4.26}$$

Remark. *The equation above can be interpreted as the increase in stored energy is less than or equal to the energy input so that energy is lost in passive systems.*

Theorem (Lyapunov Stability[86]). *Let $\xi = 0$ be an equilibrium point of the system. The equilibrium point is asymptotically stable if there is a non-negative scalar function $V(\xi)$ with $V(0) = 0$ which satisfies*

$$V(\xi_{t+1}) - V(\xi_t) < 0. \tag{4.27}$$

Remark. *Lyapunov stability analysis is based on the idea that if the total energy in the system continually decreases, then the system will asymptotically reach the zero energy state associated with an equilibrium point of the system.*
A system is said to be asymptotically stable if all the states approach zero with time.

The passivity idea and the Lyapunov stability idea are combined to analyse the Lure stability problem [86] whereby if all subsystems in a feedback system are passive, then the total energy can only decrease in an autonomous system (with zero input energy).

For linear systems, the passivity property can be related to a condition in the frequency domain known as positive real transfer functions.

Definition ([86]). *The transfer function $H(z)$ of a dynamical system is said to be positive real if and only if the system is stable and*

$$\Re\left\{H(e^{j\theta})\right\} > 0$$

for every $\theta \in [0, 2\pi)$, where $\Re\{\cdot\}$ indicates the real part of its argument, $j = \sqrt{-1}$ is the imaginary number.

Remark. *The transfer function $G(z)$ representing the linear plant in the particle dynamics is not a positive real transfer function. However, a lower limit for $\Re\{G(e^{j\theta})\}$ exists and is given by,*

$$\Re\left\{G(e^{j\theta})\right\} > -\frac{(1+w)}{2(1-2|w|+w^2)} \quad \textit{for all} \quad \theta \in [0, 2\pi). \tag{4.28}$$

Proof. The transfer function of the linear part of (4.13-4.15) is given by,

$$G(z) = \frac{z}{(z-1)(z-w)} \tag{4.29}$$

The Real part of $G(e^{j\theta})$ is given by,

$$\Re\{G(e^{j\theta})\} = \frac{(cos\theta + jsin\theta)}{(cos\theta - 1 + jsin\theta)((cos\theta - w + jsin\theta))} \tag{4.30}$$

$$= -\frac{(w+1)}{2(1-2wcos\theta + w^2)} \tag{4.31}$$

This leads to the inequality,

$$\Re\{G(e^{j\theta})\} > \frac{-(1+w)}{2(1-2|w|+w^2)} \quad \text{for all} \quad \theta \in [0, 2\pi) \tag{4.32}$$

□

An important result that is necessary for the stability analysis is the discrete-time positive real lemma which links the concepts of positive real transfer functions and the existence of a Lyapunov function.

Lemma (Discrete-Time Positive Real Lemma [91],[92]). *Let $H(z) = C(zI - A)^{-1}B + D$ be a transfer function, where A is a stable matrix or a semi-stable matrix with a simple pole on $|z| = 1$, $\{A, B\}$ is controllable, and $\{A, C\}$ is observable. Then $H(z)$ is strictly positive real if and only if there exist a symmetric positive definite matrix P, matrices W and L, and a positive constant ε such that [91], [92],*

$$A^TPA - P = -L^TL, \tag{4.33}$$

$$B^TPA = C - W^TL, \tag{4.34}$$

$$D + D^T - B^TPB = W^TW. \tag{4.35}$$

Now we are ready to state the main result of this chapter which specifies the conditions that when satisfied by the design parameters w and K, guarantee the stability of the particle dynamics.

Theorem (Main Result). *Let the particle dynamics be represented by (4.13-4.15) and satisfying (4.5) with an equilibrium point at the origin. Then the origin is asymptotically stable if $|w| < 1, w \neq 0$ and*

$$K < \left(\frac{2(1 - 2|w| + w^2)}{1 + w}\right).$$

Proof. Consider the Lyapunov function

$$V(\xi) = \xi_t^TP\xi_t, \tag{4.36}$$

where P is a symmetric positive definite matrix.
The decrease in the system energy as represented by the Lyapunov function between two discrete time instants is given by

$$\Delta V_{t+1} = V(\xi_{t+1}) - V(\xi_t) \tag{4.37}$$

$$= \xi_{t+1}^TP\xi_{t+1} - \xi_t^TP\xi_t \tag{4.38}$$

$$= \xi_t^T(A^TPA - P)\xi_t - 2\alpha_t y_t B^TPA\xi_t + (\alpha_t y_t)^2 B^TPB. \tag{4.39}$$

Since $-2\alpha_t y_t(\alpha_t y_t - Ky_t) \geq 0$, if we add this component to the right-hand side of the

equation, we get

$$\begin{aligned}\Delta V_{t+1} &\leq \xi_t^T(A^TPA - P)\xi_t - 2\alpha_t y_t B^TPA\xi_t \\ &+ \alpha_t {y_t}^2 BB^T - 2\alpha_t y_t(\alpha_t y_t - Ky_t) \qquad (4.40)\\ &= \xi_t^T(A^TPA - P)\xi_t - 2\alpha_t y_t(B^TPA - KC)\xi_t \\ &- (\alpha_t y_t)^2(2 - B^TPB). \qquad (4.41)\end{aligned}$$

We can show that the right-hand side is negative by completing a square term if the following matrix equations are satisfied

$$A^TPA - P = -L^TL, \qquad (4.42)$$

$$B^TPA = KC - W^TL, \qquad (4.43)$$

$$2 - B^TPB = W^TW. \qquad (4.44)$$

Comparing these with the relationship established in the *Positive Real Lemma* above indicates that if and only if the linear system with the transfer function

$$\tilde{H}(z) = KC(zI - A)^{-1}B + 1$$

satisfies all the conditions stated in the positive real lemma, then (4.42)–(4.44) hold.

It is straightforward then to show that $\tilde{H}(z)$ satisfies the conditions in the Positive Real Lemma, if

$$|w| < 1, w \neq 0 \qquad (4.45)$$

and

$$1 + K\Re\{G(e^{j\theta})\} > 0, \qquad (4.46)$$

which then leads to

$$K < \left(\frac{2(1 - 2|w| + w^2)}{1 + w}\right). \qquad (4.47)$$

Then

$$\begin{aligned}\Delta V_{t+1} &\leq -\xi_t^T L^T L\xi_t - 2\alpha_t y_t W^T L\xi_t - (\alpha_t y_t)^2 W^T W \\ &= -(L\xi_t - \alpha_t y_t W)^T(L\xi_t - \alpha_t y_t W) \qquad (4.48)\\ &\leq 0.\end{aligned}$$

Since the difference in the Lyapunov function is non-increasing, the particle dynamics is guaranteed to be stable, according to Lyapunov stability theorem.

In fact, asymptotic stability can be guaranteed using La Salle's extension [86] to Lyapunov stability observing that when $\Delta V_{t+1} = 0$, the particle dynamics is such that at the next time point, it will be non-zero except when the particle has reached equilibrium. To see this, consider the following scenarios:
$\Delta V_{t+1} = 0$ implies that $L\xi_t - \alpha_t y_t W = 0$ which can be written as follows with substitution for $y_t = C\xi_t$,

$$(L - \alpha_t WC)\xi_t = 0. \tag{4.49}$$

If rank of $(L - \alpha_t WC) = 0$, then if any solution is to exist, it will be unique $\alpha_t = \alpha_*$. Then for any $\alpha_{t+1} \neq \alpha_*$, $\Delta V_{t+1} < 0$, given α_t is random, it can be seen that the energy will only continue to decrease barring time instants when $\Delta V_{t+1} = 0$ at which time it will temporarily stop decreasing.
If rank of $(L - \alpha_t WC)$ is rank deficient then this implies $|L - \alpha_t WC| = 0$, which gives at most a quadratic equation in α_t for constant L, W, C. Hence, at most, α_t can take only two specific values, say α_1^*, α_2^*. Since α_t is random with probability density $P(\alpha_t)$; $P_r(\alpha_t = \alpha_1^*) + P_r(\alpha_t = \alpha_2^*)$ is infinitesimally small. Hence the probability of the event that $\alpha_t = \alpha_1^*$ or $\alpha_t = \alpha_2^*$ is infinitesimally small. Therefore, the energy will stop decreasing only at infinitesimally small finite time instants implying that asymptotically zero energy state will be reached.
If rank of $(L - \alpha_t WC) = 2$, then the only solution for (4.49) is $\xi_t = 0$, implying that energy will stop decreasing only when the system reaches equilibrium.
Hence, $V_t \longrightarrow 0$ as $t \longrightarrow \infty$. □

Remark. *The equilibrium point at the origin represents the particle position reaching the minimum location p with zero velocity. Lyapunov stability results give only sufficient conditions and hence can be very conservative. Violation of the stability conditions do not imply instability – rather that stability cannot be guaranteed.*

When $w > 0$, the condition (4.47) reduces to $K < \frac{2(1-w)^2}{1+w}$, and when $w < 0$, the condition (4.47) reduces to $K < 2(1 + w)$. The sufficient stability conditions

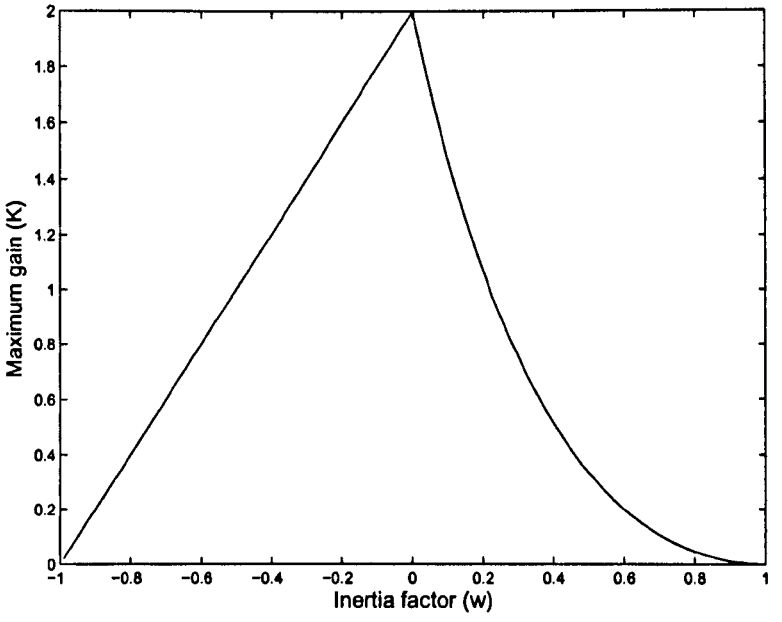

Figure 4.2: Maximum gain vs inertia factor for stability

derived in the main theorem is illustrated graphically in Figure 4.2, which shows the maximum gain for a chosen inertia factor.

Remark. *Note that the maximum gain that gives sufficient guarantees for the stability of particle dynamics decreases with the increase in inertia factor when it is positive. This is in contrast to the results derived in [84], [40] under non-random constant gain assumptions where the maximum gain increased with the inertia factor.*

4.5 Illustrative Examples

The stability analysis given in this chapter can be interpreted in the frequency domain and time domain. Through an illustrative example, we demonstrate their utility and insight.

4.5.1 Nyquist plot and Circle criterion

The main stability theorem and the proof are based on the discrete-time version of the circle criterion, which can be used as a frequency domain graphical method for stability analysis [86]. The result derived here is a special case when the lower limit for the feedback gain α_t is zero.

The circle criterion when applied to the stability of particle dynamics simply states that the Nyquist plot of the linear plant in the feedback system representation should lie to the right side of the point $-\frac{1}{K} + j0$ in the Z-plane.

For the general particle dynamics as represented in (4.7), the discrete-time Nyquist plots of $G(z)$ in (4.19) with the inertia factor (design parameter) $w = 0.8$ is given in Figure 4.3 and with $w = 0.2$ given in Figure 4.4. The Nyquist plots showing the real and imaginary parts of $G(z)$ clearly lie to the right of a limiting vertical line. The required conditions identified to satisfy positive realness in (4.46) then implies that the real value of this limiting line can be translated into a limiting condition on the gain K. The vertical lines on the figures show the limiting condition for the positive realness, which are

$$\begin{aligned} -1/K &< -22.5 \quad \text{for} \quad w = 0.8, \\ -1/K &< -0.9375 \quad \text{for} \quad w = 0.2. \end{aligned}$$

The graphical results match those obtained from the results from the main theorem as expected.

Note, however, the circle criterion can be applied to general sector conditions such as $\alpha_{\min} \leq \alpha_t \leq \alpha_{\max}$ and thus provides us flexibility in designing further parameters.

4.5.2 Lyapunov function and particle trajectories

The stability conditions derived are based on Lyapunov stability analysis and hence are overly conservative. It is therefore important to analyse the impact on the particle dynamics of the choices for the design parameters. In particular, it is of interest to analyse the case when the derived stability conditions are violated.

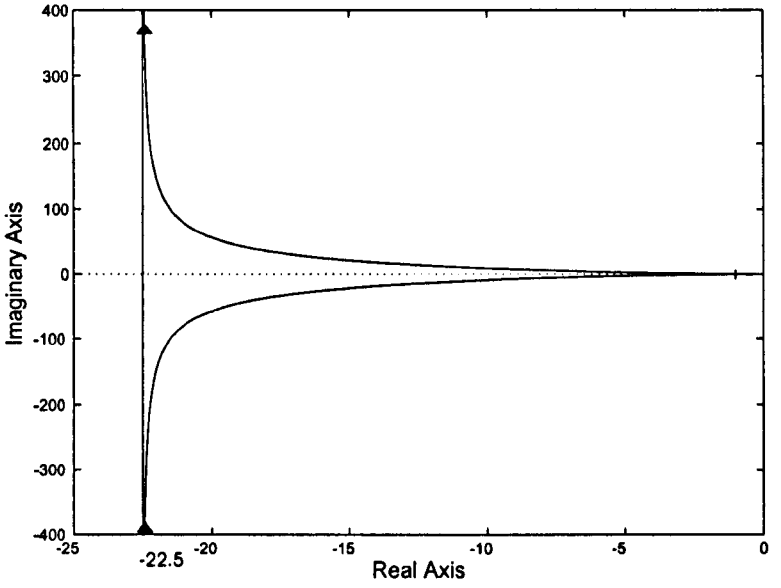

Figure 4.3: Discrete-time Nyquist plot for inertia factor=0.8 and the limit value for its real part

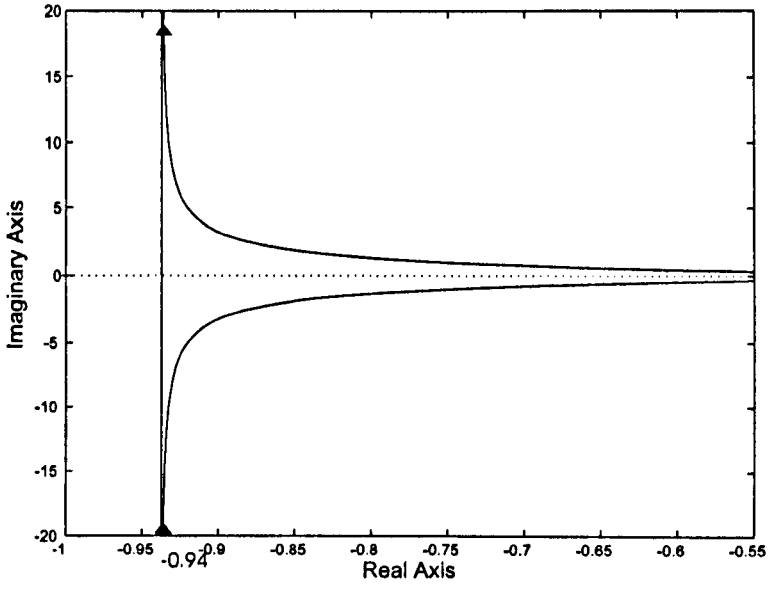

Figure 4.4: Discrete-time Nyquist plot for inertia factor=0.2 and the limit value for its real part

First, we will determine a candidate positive definite matrix P in the Lyapunov function for the chosen inertia factor w. Consider the system with $w = 0.8$ then, the system state matrix is

$$A = \begin{pmatrix} 1 & 0.8 \\ 0 & 0.8 \end{pmatrix}. \tag{4.50}$$

For this case, stability requires $K < 0.044$. A choice of $K = 0.04$ that satisfies this condition but is close to the limit is made for the analysis of this particle. However, any value satisfy this inequality for K will demonstrate the analysis. This is to ensure that while a worse case condition within limits is considered, it gives convenient rounded values for the matrices A and thus $P1$, $P2$.

By solving for P from (4.42–4.44), the solutions are given by

$$P_1 = \begin{pmatrix} 0.008 & 0.032 \\ 0.032 & 0.4372 \end{pmatrix}, \quad P_2 = \begin{pmatrix} 0.008 & 0.032 \\ 0.032 & 0.2108 \end{pmatrix}. \tag{4.51}$$

Likewise, for the system with $w = 0.2$, the state matrix is

$$A = \begin{pmatrix} 1 & 0.2 \\ 0 & 0.2 \end{pmatrix}. \tag{4.52}$$

A convenient choice to demonstrate the result is $K = 1$, which satisfies the stability guarantees of the main results. The solutions for the positive definite matrix P are given by

$$P_1' = \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.7905 \end{pmatrix}, \quad P_2' = \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.1215 \end{pmatrix}. \tag{4.53}$$

Having computed the Lyapunov function matrix for the two design choices, we can analyse how this function evolves over time. All the simulations are carried out based on equations (4.1), (4.2) and with initial conditions of $x = 1$, $v = 0$. Figures 4.5 and 4.6 show the Lyapunov energy function based on P_1 and P_1' decrease with time monotonically for the respective values of w. The trajectory of the particles for the two cases above are also given in Figures 4.7 and 4.8 demonstrating asymptotic stability of the particle dynamics.

In order to analyse the behaviour of the particle under conditions that do not guarantee stability, the evolution of the Lyapunov function determined in (4.51) was

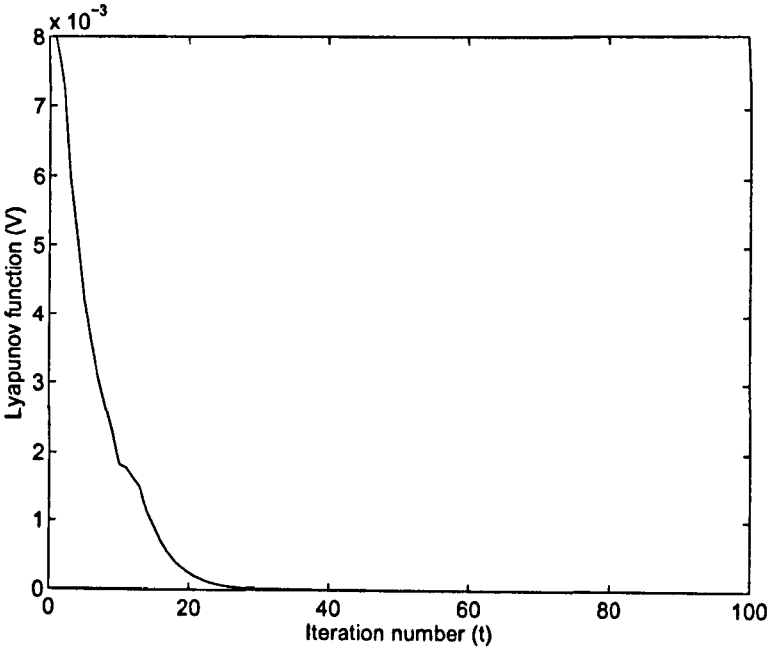

Figure 4.5: Lyapunov function with $K = 0.04$ and $w = 0.8$

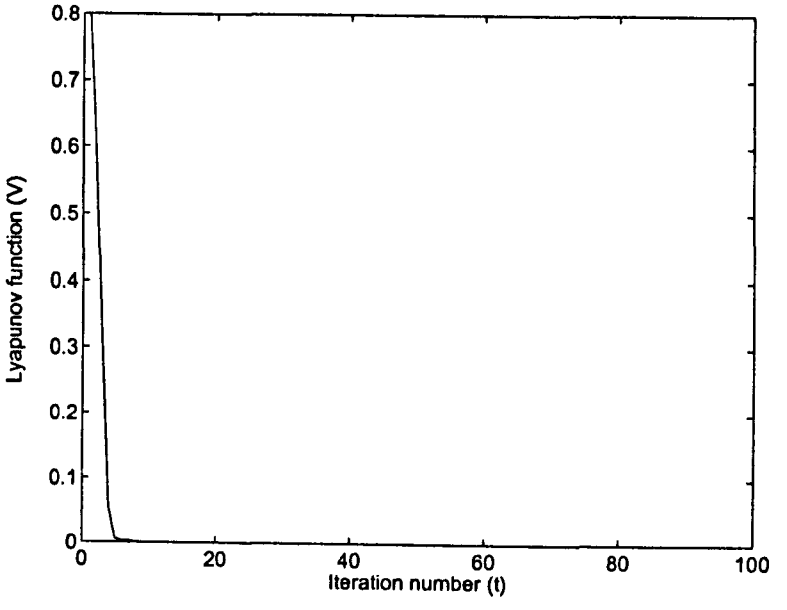

Figure 4.6: Lyapunov function with $K = 1$ and $w = 0.2$

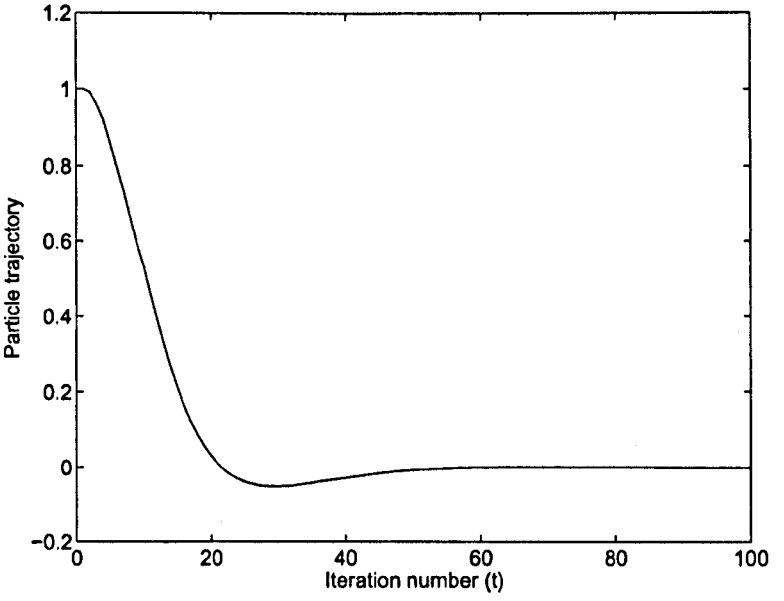

Figure 4.7: Particle trajectories with $K = 0.04$ and $w = 0.8$

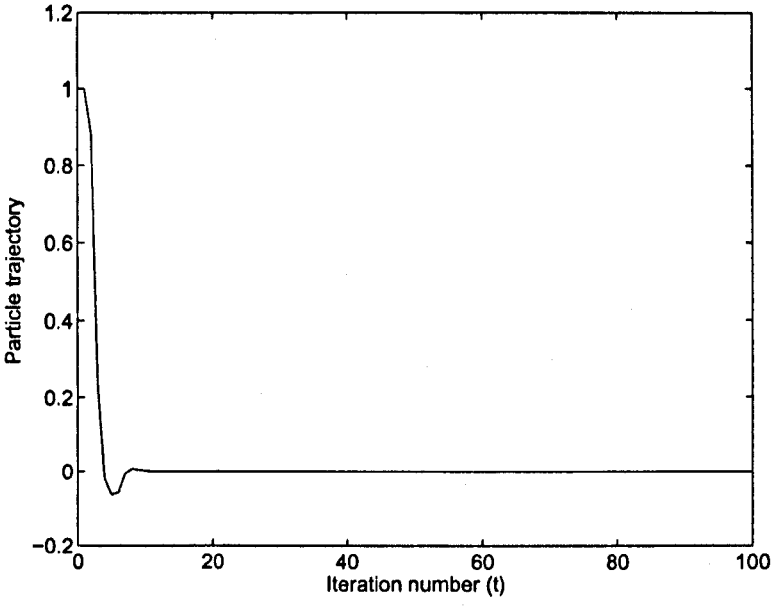

Figure 4.8: Particle trajectories with $K = 1$ and $w = 0.2$

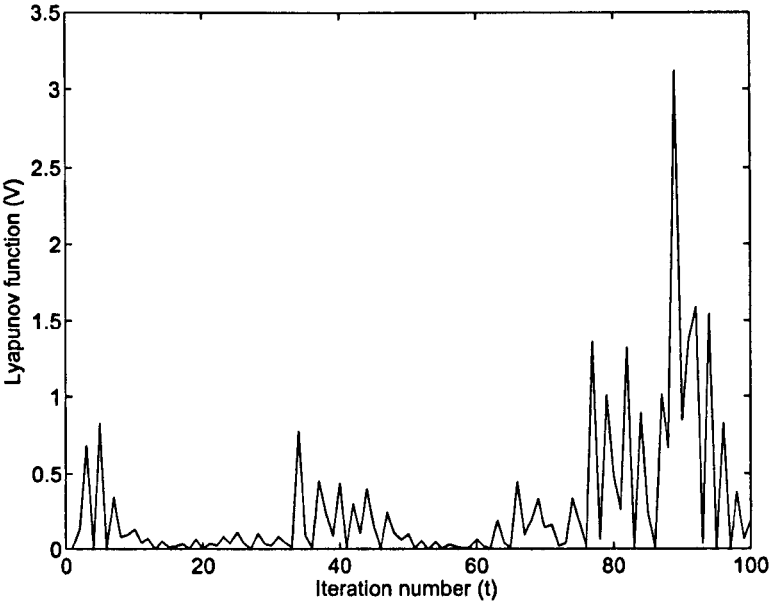

Figure 4.9: Lyapunov function with $K = 2.5$ and $w = 0.8$

observed. As seen in Figure 4.9 for a single realization, the energy decreases, but not monotonically, showing an increase at various times. In fact, the results were consistently similar. The associated particle trajectory is given in Figure 4.10, and shows asymptotic stability despite the stability conditions not being satisfied. A similar analysis was carried out with the design choices of $w = 0.2$ and $K = 2$ which also violate the required stability conditions. Figures 4.11 and 4.12 show the evolution of the Lyapunov function (4.53) and the corresponding particle trajectory. Again, the figures demonstrate the conservativeness of the stability result by showing asymptotic stability for the particle trajectory even when the design parameters do not meet the required conditions.

However, instability does occur even at reasonable design parameter values when the stability conditions are violated as shown Figures 4.13, 4.14 and 4.15. It may appear at first sight that the conservative stability conditions derived here is not useful for design. However, the utility of such analysis is in providing insights into particular features of the algorithm and thereby guide design choices. In particular,

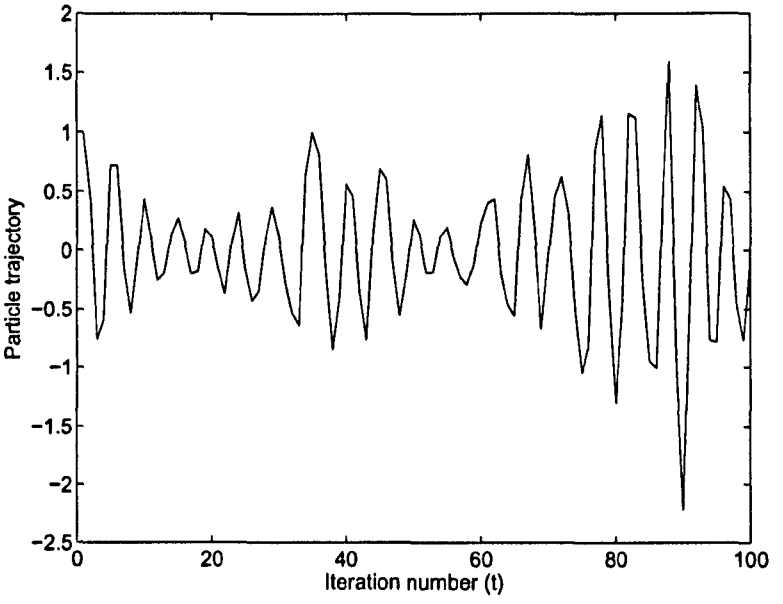

Figure 4.10: Particle trajectories with $K = 2.5$ and $w = 0.8$

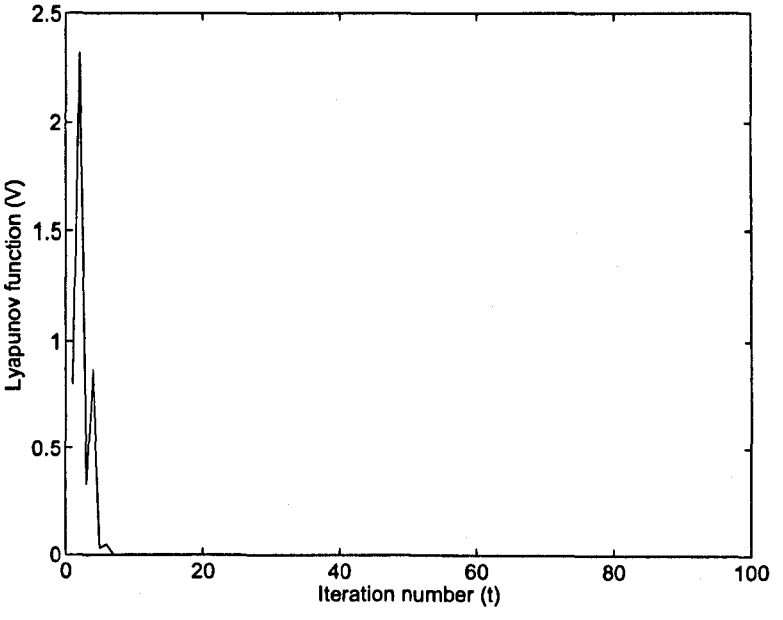

Figure 4.11: Lyapunov function with $K = 2$ and $w = 0.2$

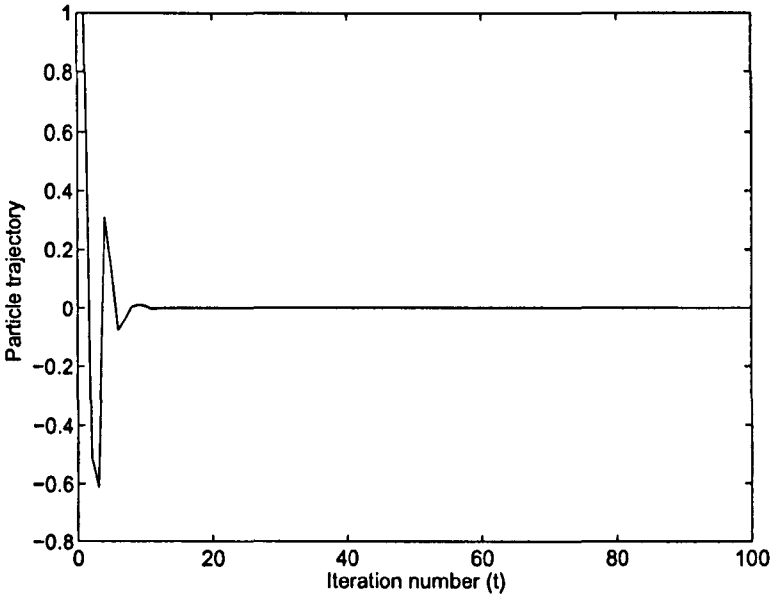

Figure 4.12: Particle trajectories with $K = 2$ and $w = 0.2$

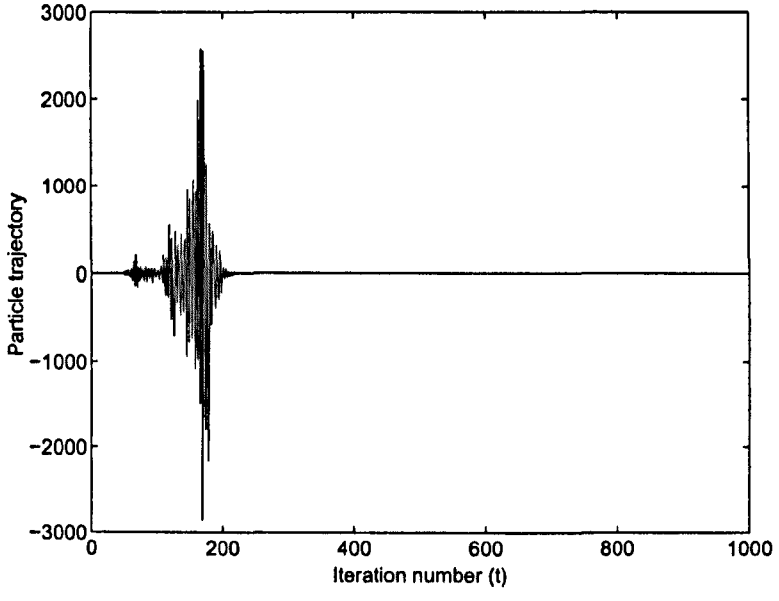

Figure 4.13: Particle trajectories with $K = 3.5$ and $w = 0.8$

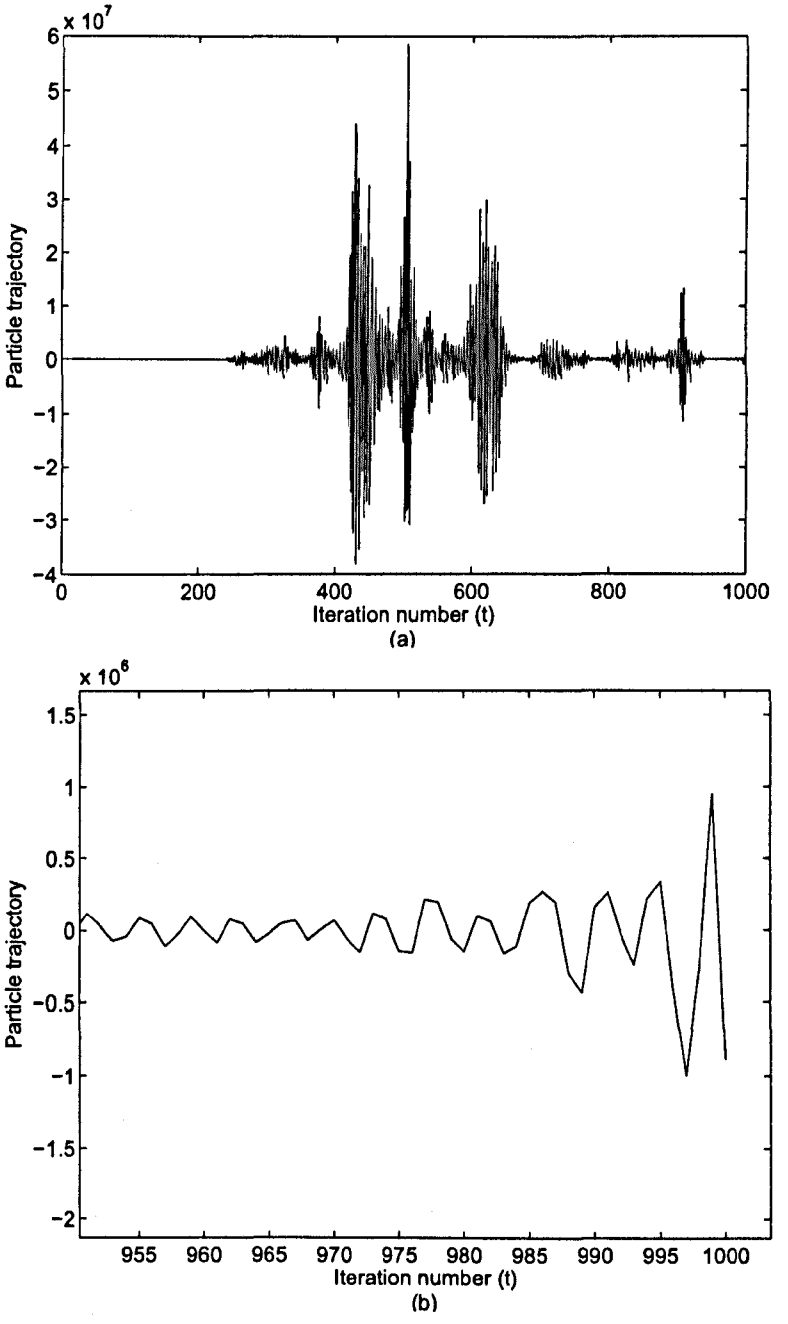

Figure 4.14: Particle trajectories with $K = 3.5$ and $w = 0.9$. (a) from initial time to t=1000. (b) Zoomed trajectory in time interval [950, 1000]

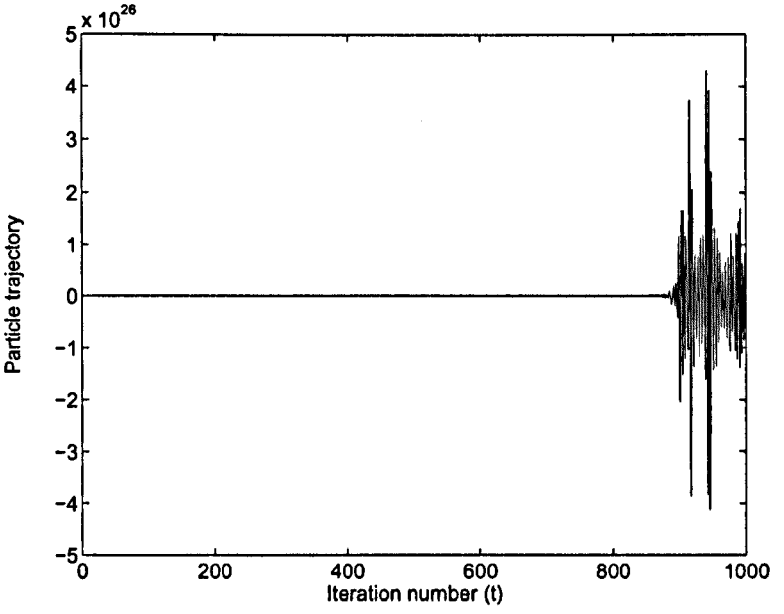

Figure 4.15: Particle trajectories with $K = 3.8$ and $w = 0.95$

we have shown that for $0 < w < 1$, decreasing w should be associated with increased K if we want to maintain the same level of exploration/convergence. It is also possible to arrive at adaptive designs in which parameters such as w and K are changed over time while stability is maintained, within the analytical framework such as those in control systems literature [86, 88, 90].

When $-1 < w < 0$ show stability and the particle trajectories have alternating signs which leads to large jumps in the particle motion, which is undesirable for local exploration. Ideally the choice for w is for it to lie in the region $0 < w < 1$ as identified by [38, 84].

It is interesting to note that under instability conditions, the particle trajectories reach very high values suggesting that particles escape from the search region, not monotonically, but at various times. This effect has been observed in the literature and solutions such as imposing a limit on the particle velocity have been proposed albeit with further problems [37] to mitigate this effect. It is possible that the stability analysis provided here can not only be used to analyse such schemes but also can

provide a guide to deriving new stabilising particle dynamics algorithms.

A characteristic feature of some of the selected particle trajectories with design choices in the region outside stability guarantees is that the particle position magnitudes were very large albeit temporarily. Such movement of particles outside relevant search region is undesirable which is one of the aims of addressing stability. In order to investigate the relationship of the number of times in a simulation the particle exceed some search region defined by a threshold for specific w values and varying K, 1000 Monte Carlo simulations for each design choice were carried out. The relevant search region was defined as

$$\mathcal{S} = \{x : |x| < \delta\}, \tag{4.54}$$

where δ is a threshold. Simulations were carried out for $\delta = 10, 100, 1000, 10000$ for three design choices that are outside the stability region identified in this paper but inside the stability region identified in [40, 38, 84]. The results are given in Table 4.1 where the number of simulations in which the particle escaped the region S at some time during the particle motion referred to as instability count. A further set of experiments were carried out with with $\delta = 100$ and $w = 0.8, 0.9, 0.95, 0.99$, while varying K in the region (0,5). Figure 4.16 shows the count of the simulations in which the particle escaped region $\mathcal{S}$ for these parameter choices.

The results clearly show that the on-set of instability as defined by the count of simulations escaping some search region and how instability increases with increasing K. The results also show the conservative nature of the theoretical bounds derived here. However, it is also noteworthy that going from $w = 0.8$ to $w = 0.9$, to achieve the same level of stability, the choice for K has to be decreased. This trend is predicted by the theoretical results shown in Figure 4.2. The critical values of K for the on-set of instability as defined here is also in between the values predicted theoretically in this paper and that advocated in [40, 38, 84].

Table 4.1: Threshold and Instability count for 1000 Monte Carlo runs

Threshold	w=0.8 and K=3.5	w=0.9 and K=2.5	w=0.95 and K=2
10	93	240	817
100	14	75	609
1000	1	18	374
10000	0	2	215

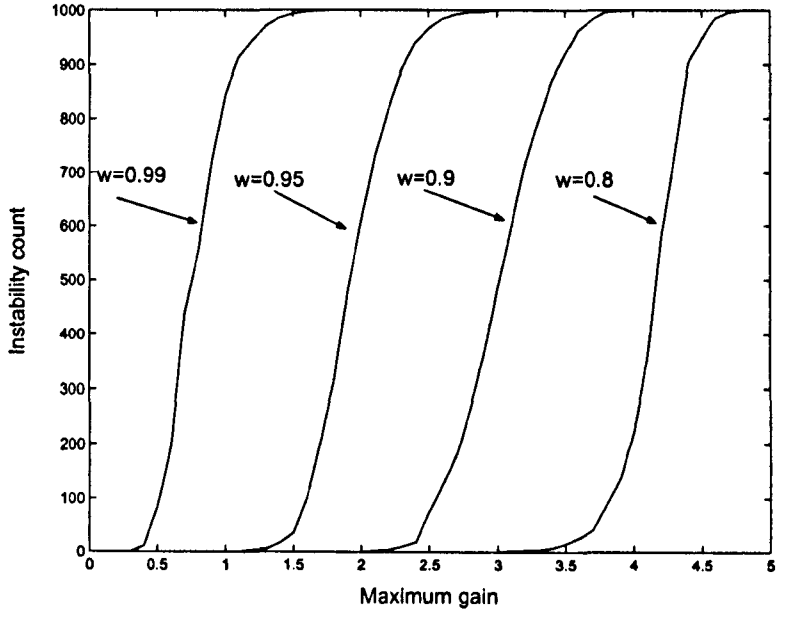

Figure 4.16: Monte Carlo trials for different w values with threshold 100

4.6 Conclusions

We have provided a different approach to the stability analysis of PSO with stochastic parameters. The passivity theorem [86] and Lyapunov stability [89] methods were applied to the particle dynamics in determining sufficient conditions for asymptotic stability and hence convergence to the equilibrium point. Since the results are based on the Lyapunov function approach, they are conservative and hence violation of these conditions do not imply instability. Nevertheless, the results can be used to infer qualitative design guidelines. Illustrative examples were given to demonstrate the application of the technique.

The analysis provided in this chapter has addressed only the issue of absolute stability. The primary aim of PSO however is optimisation while maintaining stability. For instance, adaptation rules on K and/or w design parameters such that exploration is facilitated while maintaining stability is needed.

Chapter 5

Energy Efficient Sink Node Placement in Sensor Networks

5.1 Introduction

The evolution of sensor technology and wireless communication have led to the development of wireless sensor networks. Wireless Sensor Networks (WSN) consist of small nodes with sensing, computation and wireless communication capabilities. These small nodes are inexpensive, portable wireless nodes with limited power, memory and computational capabilities. The energy supply of the sensor node is one of the main constraints in the design of sensor networks. Many routing protocols, power management techniques and data dissemination protocols have been designed for WSN where energy awareness is an essential design issue in wireless sensor networks. Wireless sensor networks are capable of observing the environment, processing data, and making decisions based on these observations. These networks are important for a number of applications in health monitoring, surveillance, target detection and environment monitoring [10].

In the past few years, intensive research has been carried out on the collaboration among the sensors in data gathering and processing and in the coordination management of the sensing activity. However sensor nodes are constrained by energy supply where it is not feasible to replace the batteries after deployment, by bandwidth lim-

itation and by limited computational and processing capabilities. Such constraints combined with typical deployment of a few hundreds of sensor nodes pose many challenges to the design and management of the WSN. It is highly desirable to find methods for energy efficient deployment, route discovery and relaying of data from sensor nodes to the sink node so that the lifetime of the network is maximised.

WSN nodes and the sink node can be deployed in different ways such as random placement and controlled placement, depending on the application. Generally, fewer sensors are required to perform the same task in the deterministic deployment than a random deployment. Previous research in sensor networking has largely focused on routing problems [11, 12, 13] and has ignored other problems such as the sensor placement and sink node placement issues.

In this chapter, we formulate a nonlinear optimisation problem to find the optimal sink node position for a given WSN where sensor nodes generate different amount of data to send to the sink node. The problem is NP-hard in general and so we use the particle swarm optimisation technique to solve the optimisation problem.

5.2 Related Work

Previous research in sensor networking has largely ignored the optimal node placement issues. Recently, works have emerged which address the node placement problem in the WSN [16, 17, 65]. Most closely related to our work discussed here are papers relating to the optimisation models of multi hop wireless sensor networks [16, 17]. The ideas in this chapter build on the optimisation formulations of these literature. We briefly discuss the modelling work given in the literature in the following sections.

5.2.1 Optimal information extraction in energy limited wireless sensor networks

The current practice in wireless sensor networks is to develop functional system designs and protocols for information extraction using intuition and heuristics, and validate them through simulations and implementations. Optimal information ex-

traction in energy limited wireless sensor networks was proposed in [16]. Here, the authors addressed the need for a symmetric methodology by developing formal non-linear optimisation models of static WSN that yield fundamental performance bounds and optimal designs with respect to the energy constraints. They have addressed two problems, that of maximising the total information gathered subject to the energy constraints and minimising the energy usage subject to information gathering.

It is assumed that n sensors are placed in fixed locations, each with limited energy supply E_i and d_{ij} denotes the physical distance between node i and j. The purpose is to extract as much as information as possible to the sink node (node $n+1$ with unlimited energy resources –a reasonable assumption if the sink node is plugged in). Each node consumes C units of energy per-bit received and units of energy per-bit sensed. They also assume that the sensor node can adjust both the information flow rate and the transmission power, which are denoted by f_{ij} and P_{ij} for link between nodes i and j. The relation between the flow rate and transmission power on a link is given by Shannon's capacity equation for a white Gaussian noise channel, assuming a square-law signal decay d_{ij}^{-2} and noise of η on the communication channel. The fraction of the total information that reaches the sink node from node i is denoted by α_i. The objective is to find the coordinated operation of all nodes by setting transmission powers and flow rates in order to maximise the amount of information that reaches

the link. The problem is expressed by the following nonlinear programming problem.

$$\max \sum_{j=1}^{n} f_{jn+1} \tag{5.1}$$

Subject to

$$\sum_{j=1}^{n+1} f_{ij} - \sum_{j=1}^{n} f_{ji} \geq 0 \tag{5.2}$$

$$\sum_{j=1}^{n+1} f_{ij} - \sum_{j=1}^{n} f_{ji} \leq \alpha_i \sum_{j=1}^{n} f_{jn+1} \tag{5.3}$$

$$\beta(\sum_{j=1}^{n+1} f_{ij} - \sum_{j=1}^{n} f_{ji}) + \sum_{j=1}^{n+1} P_{ij} + \sum_{j=1}^{n} C f_{ji} \leq E_i \tag{5.4}$$

$$f_{ij} \leq \log(1 + \frac{P_{ij} d_{ij}^{-2}}{\eta}) \tag{5.5}$$

$$f_{ij} \geq 0, P_{ij} \geq 0. \tag{5.6}$$

The constraint in 5.2 represents rate of data sensed by node i. The next constraint in 5.3 guarantees that each node sends a fraction of the total information to the sink node. The constraint in 5.4 limits the available energy E_i for each sensor node and equation (5.5) comes from the Shannon's capacity equation [47].

Adding the consumption of energy of every node i, the following expression can be obtained for the total energy consumed by the sensor nodes of the WSN:

$$\sum_{i=1}^{n} \epsilon_i = \sum_{i=1}^{n} (\beta(\sum_{j=1}^{n+1} f_{ij} - \sum_{j=1}^{n} f_{ji}) + \sum_{j=1}^{n+1} P_{ij} + \sum_{j=1}^{n+1} P_{ij} + \sum_{j=1}^{n} C f_{ji}) \tag{5.7}$$

$$= \sum_{i=1}^{n} (\beta f_{in+1} + P_{in+1} + \sum_{i=1}^{n} \sum_{j=1}^{n} (C f_{ij} + P_{ij}). \tag{5.8}$$

The formulation is a very interesting and incorporates the multi hop nature of the problem. In the simulation experiments, consideration is given only to line of topology and square topology of the networks which are easily configured for optimal routing in the network. The problem when the optimal multi-path information is not easily known, or extracted was not addressed.

5.2.2 Energy aware node placement in wireless sensor networks

Energy aware node placement in wireless sensor networks was developed in [17], where the formulation of a constrained multi-variable nonlinear programming problem to determine both the locations of the sensor nodes and data transmission patterns. The sensor networks model considered there has a single static sink node located randomly in the sensor networks region. Optimal placement strategies are numerically calculated for the linear network where the multi-path connection patterns are easily known. The following section explain the modelling framework as in [17].

Let us first consider a linear network, which consists of a set of sensor nodes placed a long and narrow area with the sink node at the end. Each node collects the data within its sensing range, sending information to the sink node for control. Each sensor has a certain amount of initial energy E_0 and a sensing range D. Let d_i be the distance between node i and $i+1$, $i = 1, ...n-1$, and d_0 the area covered by node 1. A general scenario where each sensor node continuously or periodically collects constant bit data rate is considered. A further assumption is made so that the amount of data generated in a unit area per unit time is a constant denoted by c.

If the sensor nodes are deployed using uniform placement in which sensor nodes are placed with equal distance in between, then the power consumption of sensor networks can be modelled as follows:

$$P_i = i\frac{L}{n}(\frac{L}{n})^m c \tag{5.9}$$

$$T = \min_i(\frac{E_0}{P_i}) \qquad i = 1, 2, ..., n-1 \tag{5.10}$$

where P_i denotes the power consumption of the ith node to relay all the collected data, L the length of the linear network, n the number of nodes, E_0 the energy allocated to each node and T the lifetime of the network. L/n is the sensing area of each node and m is the communication path loss index. In this type of placement technique, nodes closer to the sink node carry more loads, consume more power and lose all its energy quickly thus the total life time of the network is reduced.

It is infeasible in practice to allocate energy arbitrarily among different nodes.

Hence, the assumptions are made that the sensor nodes have the same initial energy E_0 and f_{ij} is the amount of data rate to be sent directly from node i to node j for a unit time period. The corresponding power dissipation can be expressed by,

$$f_{ij}(\sum_{k=i}^{j-1} d_k)^m \tag{5.11}$$

The placement problem can be formulated as follows [17]:

$$\max T \tag{5.12}$$

Subject to:

$$\sum_{j=i+1}^{n} f_{ij} = \sum_{k=1}^{i-1} f_{ki} + d_{i-1}c, \quad i = 2, ..., n-1 \tag{5.13}$$

$$\sum_{j=2}^{n} f_{1j} = d_0 c \tag{5.14}$$

$$\sum_{j=i+1}^{n} f_{ij}(\sum_{k=i}^{j-1} d_k)^m \leq \frac{E_0}{T} \quad i = 1, ..., n-1 \tag{5.15}$$

$$\sum_{i=0}^{n-1} d_i = L \tag{5.16}$$

$$0 \leq d_i \leq D, \quad i = 0, 1, ..., n-1. \tag{5.17}$$

Equation (5.13) represents the flow constraint of the network and Equation (5.14) is the flow constraint at node 1. Equation (5.15) and Equation (5.16) present the energy constraints at each node and the length of the network respectively. The objective of this work is to place sensor nodes in an optimal way to maximise the lifetime of the sensor networks consisting of n sensor nodes with the same initial energy deployed in a certain area. As this type of problem has no analytical solution, the authors use a numerical algorithm to maximise the network lifetime. The simulation results show that using the above optimal node placement and data transmission pattern leads to a significant benefit over the other placements techniques.

5.2.3 Sink node placement methods

Nodes closer to a sink node will experience heavier traffic load since they not only collect data within their sensing range but also forward data to the sink node or to the next node. Such an unbalanced traffic load introduces an asymmetric power consumption among the sensor nodes. Hence, sink node placement methods will have considerable impact on the life time of the WSN.

The idea of exploiting the mobility of the sink node for the purpose of increasing the lifetime of WSN was developed in [93]. Here the authors formulated a linear optimisation model to determine which nodes should be visited by the sink in order to maximise the lifetime of the WSN. They consider WSN nodes that are arranged in a two dimensional grid and the one sink node travels along the grid line. These assumptions lead easily to a formulation of a linear programming problem that can be solved using existing linear programming methods.

Deploying multiple, mobile sink nodes idea was proposed in [94]. The experimental results demonstrate that nodes which are one hop away from a base station drain energy faster than other nodes in the network. This is attributed to the fact that nodes which are one hop away from the sink node need to forward messages originating from many other nodes, in addition to delivering their own message. In doing so, these nodes deplete their energy quicker and become in-operational. A solution that determines the new location based on the residual energy of nodes is proposed based on the integer linear program [94]. It does not minimise the energy usage of the network but maximises the network lifetime by sharing the energy resources within the network.

In [66], the energy provisioning for wireless sensor networks was investigated for a two-tier wireless network. Deployment of a relay node into the network was proposed to mitigate network geometric deficiencies and prolong the network lifetime. Optimal sink node locations in two-tiered wireless sensor networks can be found in [95]. The main contribution of this work is to algorithmically obtain optimal sink node position for given cluster head nodes. It includes an analytically derived upper and lower bound bounds of maximal topological lifetime by exploring some intrinsic properties

of WSNs.

5.3 System Models

In this chapter, we define a system model based on the modelling work given in section 4.2 and we assume a sensor network model similar to those used in [12, 13] with the following properties:

- The sensor nodes are energy constrained with a uniform energy allocation.
- The nodes are equipped with power control capabilities to vary their transmit power
- All sensor nodes are immobile and their locations are known.
- Data sending rates of sensors are not the same.
- The sink node has no energy constraints and it can be placed anywhere in the given sensor network region.

5.3.1 Energy Model

Energy efficiency is the vital design parameter for sensor networks. Power is defined by the rate of change in the energy. Therefore the amount of energy which is necessary to operate for time t consuming power $P(t)$ can then be defined as,

$$E = \int P(t)dt \tag{5.18}$$

Power consumption in the sensor node in the sensor networks can be divided into the following components depending on the operations performed within the node:

- Transmitter energy: The data gathered from the environment need to be transmitted to the sink node. Therefore, the transmitter circuitry needs to be operated. For this process, transmitter energy is consumed which depends on the transmitter power, size of the data packet and data transfer rate.

- Receiver energy: Sensors not only collect and transmit their data within their sensing range but also receive and forward data to the sink node or to the next node. The receiver energy will be consumed during this process which is independent of the distance between the two sensor nodes. Receiver energy depends on the size of the data packet received and data transfer rate.

From communication theory, the radio propagation model in a single path channel can be modelled as follows:

$$P_r = G_t G_r \left(\frac{c}{4\pi df}\right)^m P_t \tag{5.19}$$

where P_t and P_r are the transmitted power and received power respectively. G_r and G_t are the receiver and transmitter antenna gain respectively. c is the velocity of the radio wave propagation in free space and f is the frequency of its waves. We can write the received power in terms of the transmitted power as [96],

$$P_t = C d^m P_r \tag{5.20}$$

where C is a constant, d is the distance between the transmitter and the receiver and m is known as path loss index. In many sensor networks application scenarios, path loss index m can be assumed to lie between 2 and 4 [96].

Our optimisation model considers the problem of finding optimal location for a sink node for a given WSN in the most efficient manner. Let us assume that N sensor nodes are placed in a region to collect data from a specific location. Each sensor has limited energy supply E_i, with x_i and y_i denoting the Cartesian coordinates of the sensor locations and d_{ik} denoting the distance between nodes i and k. The purpose of this network is to extract required information from sensor nodes (which have limited power supply) to the sink node, which is possibly connected to a main power supply (no energy constraint, some road monitoring applications may have this type of scenarios). We also assume that sensor nodes can adjust their transmission power which is denoted $P_i(i,k)$ for the link between nodes i and k [12].

We assume that the rate of data generated at sensor node i (after data aggregation) is g_i constant bit rate. The power consumption in the data communication (by

receiving and transmitting) is the important criteria for consideration. The power dissipation at the transmitter can be modelled as

$$P_t(i,k) = C_{ik} f_{ik} \tag{5.21}$$

where f_{ik} is the bit rate transmitted from node i to k and C_{ik} is the power consumption cost of radio link from node i to k, written as,

$$C_{i,k} = \alpha + \beta((x_i - x_k)^2 + (y_i - y_k)^2)^{m/2} \tag{5.22}$$

where α is a distance independent constant term, β is a coefficient term associated with the distance-dependant term, and m is the path loss index, with $2 \leq m \leq 4$. Typical values for these parameters are $\alpha = 50nJ/b$ and $\beta = 0.0013pJ/b$, and $m = 3$ [96].

The power dissipation at a receiver can be modelled as,

$$P_r(i) = \rho \sum_{k \neq i} f_{ki} \tag{5.23}$$

where $\sum_{k \neq i} f_{ki}$ (in b/s) is the rate of the received data stream at node i and ρ is a receiver constant with a typical value is 50nJ/b [96].

5.4 Energy Efficient Sink Node Placement

For a network with N sensor nodes, where each node i senses and generates data with the rate of g_i. The data rates from node i to node k and to the sink node are f_{ik} and f_{iS} respectively. (x_i, y_i), $1 \leq i \leq N$, are fixed coordinates for the placement of the sensor nodes and (x, y) are the coordinates of the sink node which is to be placed efficiently in the sensor network region $(-L, L) \times (-L, L)$. For each node in the WSN, the following flow balance equation and location constraint must be met [16]:

$$f_{iS} + \sum_{1 \leq k \leq N}^{k \neq i} f_{ik} = \sum_{1 \leq m \leq N}^{m \neq i} f_{mi} + g_i \tag{5.24}$$

$$-L \leq x \leq L \tag{5.25}$$

$$-L \leq y \leq L \tag{5.26}$$

The goal here is to place the sink node in an optimal way so as to maximise the lifetime of a sensor network consisting of N sensors with the same initial energy deployed in a certain area. According to the problem setup, maximising the life time is achieved by minimising total power consumption of N sensor nodes. The power consumption of node i, P_i, can be represented as follows:

$$P_i = \sum_{1\leq k\leq N}^{k\neq i} c_{ik} f_{ik} + \sum_{1\leq m\leq N}^{m\neq i} \rho f_{mi} + c_{iS} f_{iS} \tag{5.27}$$

The total power consumption of the WSN can be calculated as follows:

$$P = \sum_{i=1}^{N} P_i \tag{5.28}$$

which can be expanded by substituting (7)

$$P = \sum_{i=1}^{N} \Big(\sum_{1\leq k\leq N}^{k\neq i} C_{ik} f_{ik} + \sum_{1\leq m\leq N}^{m\neq i} \rho f_{mi} + C_{iS} f_{iS} \Big) \tag{5.29}$$

$$= \sum_{i=1}^{N} \Big(\sum_{1\leq k\leq N}^{k\neq i} C_{ik} f_{ik} + \sum_{1\leq m\leq N}^{m\neq i} \rho f_{mi} \Big) + \sum_{i=1}^{N} C_{iS} f_{iS} \tag{5.30}$$

The optimisation function can be rewritten as follows:

$$\min \sum_{i=1}^{N} \Big(\sum_{1\leq k\leq N}^{k\neq i} C_{ik} f_{ik} + \sum_{1\leq m\leq N}^{m\neq i} \rho f_{mi} \Big) + \sum_{i=1}^{N} C_{iS} f_{iS} \tag{5.31}$$

subject to the following constraints,

$$-L \leq \ x \ \leq L \tag{5.32}$$

$$-L \leq \ y \ \leq L \tag{5.33}$$

$$f_{iB} + \sum_{1\leq k\leq N}^{k\neq i} f_{ik} \ = \ \sum_{1\leq m\leq N}^{m\neq i} f_{mi} + g_i \tag{5.34}$$

There is high level of complexity in the equation (5.31) to determine the optimal location for sink node. In the following sections, we propose three different suboptimal strategies to place the sink node in the given sensor network region.

5.4.1 Strategy 1

This strategy minimises the total radio link power consumption cost between every node and sink node and does not take data rate into account. This strategy does not consider optimal multi hop connection pattern to the sink node from each sensor node. The optimisation function in 5.31 can be simplified to,

$$P = \sum_{i=1}^{N} C_{iS} \tag{5.35}$$

The optimisation function for strategy 1 can be written

$$\min \sum_{i=1}^{N} (\alpha + \beta((x_i - x)^2 + (y_i - y)^2)^{m/2}) \tag{5.36}$$

Here, the sub optimality comes in the way that multihop communication between each node and sink node and data rate of individual sensor node are not considered. This simplifies the optimisation task but becomes suboptimal. In most cases suboptimality gives better solution where optimal solution is not easily obtained.

5.4.2 Strategy 2

This strategy minimises the total radio link power consumption cost between every sensor node and the sink node. This strategy takes the data rate into account where each node needs to generate different data rates but does not consider the optimal multi hop connection pattern to the sink node from each sensor node. The optimisation function in 5.31 can be simplified to,

$$P = \sum_{i=1}^{N} C_{iS} f_{iS} \tag{5.37}$$

The optimisation function for strategy 2 can be written

$$\min \sum_{i=1}^{N} (\alpha + \beta((x_i - x)^2 + (y_i - y)^2)^{m/2}) f_{iS} \tag{5.38}$$

Here, the sub optimality comes in the way multihop communication between each node and sink node is not considered. This simplifies the optimisation task but becomes suboptimal.

5.4.3 Strategy 3

This strategy minimises the multi hop communication transmission problem. Here, we consider the multi hop nature of the sensor node to the sink node where direct communication is not simply possible in sensor networks. Here, the sub optimality comes in the way the region is partitioned for sink node placement and as long as the sink node stays in the cluster region, no re-routing is needed. This simplifies the optimisation task but becomes suboptimal. However, this strategy is more accurate than strategy 1 and strategy 2.

The optimisation function for strategy 3 can be written as,

$$\min \sum_{i=1}^{N} \Big(\sum_{1 \le k \le N}^{k \neq i} C_{ik} f_{ik} + \sum_{1 \le m \le N}^{m \neq i} \rho f_{mi} \Big) + \sum_{i=1}^{N} C_{iS} f_{iS} \tag{5.39}$$

To find the optimal location of the sink node in the proposed three strategies we have to perform search algorithms such as genetic algorithms or particle swarm optimisation as the problems are NP-hard in general. Here we choose particle swarm optimisation because its implementation is simple and gives better results in most cases than genetic algorithms for these types of optimisation problems [37, 22].

5.5 Optimisation Strategies

Particle swarm optimisation is a recently proposed optimisation techniques that poses several highly desirable attributes, including the fact that the algorithm is very easy to understand and implement. It is similar in some ways to genetic algorithm and evolutionary algorithms, but requires less computational cost. The following sections explain the particle swarm optimisation and genetic algorithm.

5.5.1 Particle swarm optimisation

Particle swarm optimisation (PSO) is a swarm intelligence based technique developed by Kennedy and Eberhart [20], inspired by social behaviour of bird flocking or fish schooling. The PSO formulation defines each particle as a potential solution to

a problem in d-dimensional space with memory of its previous best position and the best position amongst all particles, in addition to a velocity component. At each iteration, the particles are combined to adjust the velocity along each dimension which in turn is used to compute the new particle position. The position vector and the velocity vector of the i^{th} particle can be represented as $x_i = (x_{i1}, x_{i2}, x_{i3}.....x_{id})$ and $v_i = (v_{i1}, v_{i2}, v_{i3}.....v_{id})$ respectively. According to the optimisation function to be optimised, let say the particle best position at iteration t is $P^{(l)}(t) = (p_{i1}, p_{i2}, p_{i3}.....p_{id})$ and the best global position or the best solution amongst all particles at iteration (t) is $P^{(g)}(t) = (p_{g1}, p_{g2}, p_{g3}.....p_{gd})$. Then the new velocities and positions of j^{th} dimension of the particle i for the next fitness evaluation are calculated:

$$v_{ij}(t+1) = w * v_{ij}(t) + c_1 * r_1(p_{ij}^{(l)}(t) - x_{ij}(t)) + c_2 * r_2(p_{gj}^{(g)}(t) - x_{ij}(t)) \quad (5.40)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t) \quad (5.41)$$

where c_1 and c_2 are constants known as acceleration coefficients, w is known as inertia factor and r_1 and r_2 are two different uniformly distributed random number in the range [0,1].

PSO is initialised with a population of particles (initial solutions) with random positions and velocities. The fitness of each particle is then evaluated according to the optimisation function. At each iteration the velocity and position of each particle is calculated according to equations (5.40-5.41). Each time a particle finds a better position than the previously found best position, its location is stored in a memory. Generally, a maximum velocity v_{max} for each dimension of the velocity vector of the particles is defined in order to control excessive roaming of the particles outside the search region.Whenever v_{ij} exceeds the defined limit, its velocity is set to v_{max}.

For the best performance of PSO algorithm, the condition for design parameters were derived in the literature [84, 40, 38],

$$w < 1 \quad \text{and} \quad (5.42)$$

$$K < 2(w+1). \quad (5.43)$$

where $K = c_1 + c_2$.

The suitable selection of inertia factor w provides a balance between global and

local explorations. Shi and Eberhart [37] have found a significant improvement in the performance of the PSO method with a linearly varying inertia factor over the iteration. In general, the inertia factor w is set according to the following equation.

$$w = w_{max} - \frac{w_{max} - w_{min}}{I_{max}} \times I \tag{5.44}$$

where I_{max} is the maximum number of iterations, I is the current number of iteration, w_{max} is the initial value of w and w_{min} is the final value of w.

5.5.2 Genetic algorithm

The genetic algorithm (GA) [97] also begins its search from randomly generated population of designs that evolve over successive generations. To perform its optimisation process, the GA employs three operators to propagate its population from one generation to other. The first operator is the *Selection* operator that mimics the principal of *Survival of the fitness.* The second operator is the *Crossover* operator which mimics the mating in biological populations. The crossover operator propagates features good surviving designs from the current population into the future population. The last operator is *Mutation*, which promotes diversity in population characteristics. The mutation operator allows global search of the design space and prevents the algorithm from getting trapped in local minima.

The use of genetic algorithm (GA) requires the proper selection of a set of genetic operations between many possibilities. The number of generations and the population size, crossover and mutation probabilities are values that must be given to initialise the optimisation process. All these parameters have great influence on the GA performance. Although, there is no clear indication about the population size in the GA, larger population size may increase the computational cost. However, for small population size the cross over and mutation operations can not be implemented properly. The probability of crossover is always greater than probability of mutation. Generally, the probabilities of crossover and mutation are taken as 0.75 to 0.9 and 0.05 to 0.2 respectively. A real coded genetic algorithm and its parameter selection guidelines are discussed in [98].

5.6 Sink Node Shortest Path Problem

We use the centralised method for sink node shortest path placement problem. The static sensor networks need not to rely on the sensor network routing protocols which do not always find the optimal multi path to sink node from each sensor node. Although, many centralised algorithms have been devised for finding the shortest path problem, Dijkstra's algorithm is perhaps the earliest and also one of the most efficient algorithm for the shortest path problem [99].

Dijkstra's algorithm solves the single-source shortest path problem for a directed graph. To find the shortest path between the sink node and a sensor node, length of a path is calculated as the sum of the weights of the edges in the path. A path is the shortest path if there is no path from sink node to source node with lower weight. Dijkstra's algorithm finds the shortest path from sink node to sensor node in order of increasing distance from sensor node. That is, it chooses the first minimum edge, stores this value and adds the next minimum value from the next edge it selects. It starts out at one vertex and branches out by selecting certain edges that lead to new vertices.

Let $D(v)$ be the distance from the source s to a node v. Let $l(v, w)$ be the given cost between nodes v and w. There are two main steps in the algorithms which are an initialisation step and a step to be repeated until the algorithm terminates [99].

- Initialisation: Set $N = \{s\}$. For each node v not in N, set $D(v) = l(s, v)$. We use ∞ for nodes not connected to s.

- Iteration Step: Find a node w not in N for which $D(w)$ is a minimum and add w to N. Then update $D(v)$ for all nodes remaining that are not in N by computing

$$D(v) = min[D(v), D(w) + l(w, v)] \tag{5.45}$$

  This step repeated until all nodes are in N.

The Figure 5.1 shows the optimal multi path routing connection pattern between 10 sensor nodes and sink node using Dijkstra's algorithm.

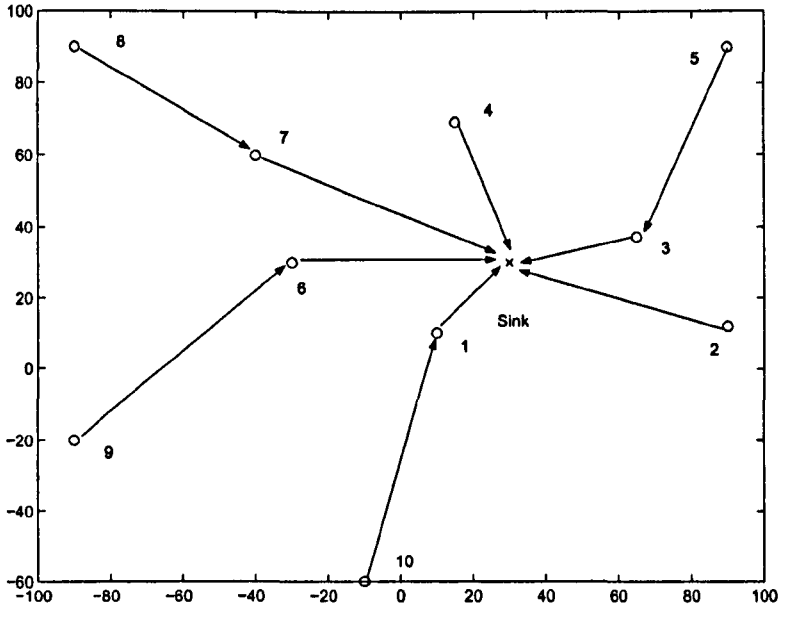

Figure 5.1: Optimal multi path routing connection pattern by Dijkstra's algorithm.

5.7 Sink Node Placement in Sensor Networks

We only consider the optimal placement of the sink node, so that the vector of design variables DV is of size 2.

$$DV = [x\ y] \tag{5.46}$$

The design variables are x and y coordinates of the sink node. For strategy 1 and strategy 2, we do not need multi-hop routing path connection information to calculate the optimal sink node position and perform the particle swarm optimisation algorithm as normal with the constraints of x and y coordinates of the sensor networks region. For strategy 3, we need to know the optimal connection pattern for every single search point in the sensor networks region as the optimisation function in strategy 3 depends on the multi-hop optimal path connection pattern of every single node to the sink node. If we perform routing algorithms after every iteration (every possible point for the sink node in the region) online, it can be a computationally expensive process and adds more complexity to the optimisation algorithm. To alleviate this problem

Table 5.1: PSO simulation parameters

Parameter	Value
w_{min}	0.1
w_{max}	1
c_1	1.2
c_2	1.2
V_{max}	10
Population	20
Iteration Number	1000

we consider the sensor networks region as being partitioned into several small clusters to reduce the computational complexity. We perform the optimal multi-hop routing algorithms off-line by assuming the sink node is placed in centre of each cluster. Then we calculate the cost function using the optimal routing connection pattern for each cluster. We assume that if the sink node is placed anywhere in the given cluster it has the same optimal routing path (a realistic assumption to calculate near optimal position). After each iteration, our optimisation algorithms identifies the appropriate cluster and the cost function for its cluster which depends on the multi-hop optimal routing connection from each sensor node to the sink node.

We have also performed the optimisation process using genetic algorithm to compare with the particle swarm optimisation. The results are compared after 1000 iterations for strategy 1 using particle swarm optimisation and genetic algorithms. Table 5.1 and Table 5.2 show PSO and GA parameter values which we use for the optimisation process respectively. We have performed the PSO and GA algorithms with the range of different parameter values and parameters given in Table 5.1 and Table 5.2 are the best parameter of this particular optimisation problem.

5.8 Simulation Results

A square network topology is considered for computational experiments that are easily scalable. In the square topology, a WSN is chosen in which all sensors lie in

Table 5.2: GA simulation parameters

Parameter	Value
Size of the population	20
Probability of crossover	0.8
Probability of mutation	0.02
Tournament probability	0.7
Scale for mutations	0.1
Number of runs	1000

the region of $([-100m, 100m] \times [-100m, 100m])$, with the liberty to place the sink node anywhere in the region. We performed our computational experiment with the particle swarm optimiser for the three strategies proposed in section 5.4. Experiments with 5, 10, 15, 20, 25 and 30 sensor nodes were carried out and the positions of the sensor nodes are given in Table 5.3.

The sensor network region is divided into 16 different clusters ($50m \times 50m$ blocks). Without loss of generality we assume that the sink node placed in (25m,25m) for the random placement method. The total power consumption of WSN with 5, 10, 15, 20, 25 and 30 sensor nodes for the three proposed strategies and the random placement method were calculated. The path loss index value $m(2 \leq m \leq 4)$ was considered as 3 for the calculations. We assume that each node has the same initial energy $100KJ$[95]. Figure 5.2 shows that our proposed three strategies give better result than random placement. The strategy 3 gives better result than strategy 1 and strategy 2. The accuracy of the strategy 3 depends on the size of the cluster we have chosen. If we choose large number of clusters it becomes computationally more costly and more complex to implement.

We have also performed a simulation study with different parameter choices for strategy 1 using PSO. The PSO algorithm gave better performance for $c_1 = c_2 = 1.2$ than $c_1 = c_2 = 1.8$. Figure 5.3 show the performance differences clearly and support the claim that the deterministic version of the convergence analysis does not always give good design choices [100].

In this work, an attempt was also made to examine the claim that PSO has same

Table 5.3: Locations, data generating rate, and initial energy for each sensor nodes

Sensor No	$(x_i, y_i)(m)$	$g_i(kb/s)$	$e_i(KJ)$	Sensor No	$(x_i, y_i)(m)$	$g_i(kb/s)$	$e_i(KJ)$
1	(10,10)	1	100	16	(-60,-20)	5	100
2	(90,12)	2	100	17	(-90,-90)	2	100
3	(65,37)	4	100	18	(90,-90)	3	100
4	(15,69)	5	100	19	(40,-40)	6	100
5	(90,90)	3	100	20	(-80,40)	2	100
6	(-30,30)	7	100	21	(65,85)	3	100
7	(-40,60)	8	100	22	(35,48)	9	100
8	(-90,90)	1	100	23	(55,5)	10	100
9	(-90,-20)	1	100	24	(0,-100)	2	100
10	(-10,-60)	5	100	25	(35,-85)	3	100
11	(60,-60)	8	100	26	(-40,-40)	7	100
12	(60,-90)	2	100	27	(30,-85)	1	100
13	(-80,30)	6	100	28	(-12,18)	6	100
14	(-20,20)	6	100	29	(17,-18)	5	100
15	(-10,90)	1	100	30	(-40,10)	3	100

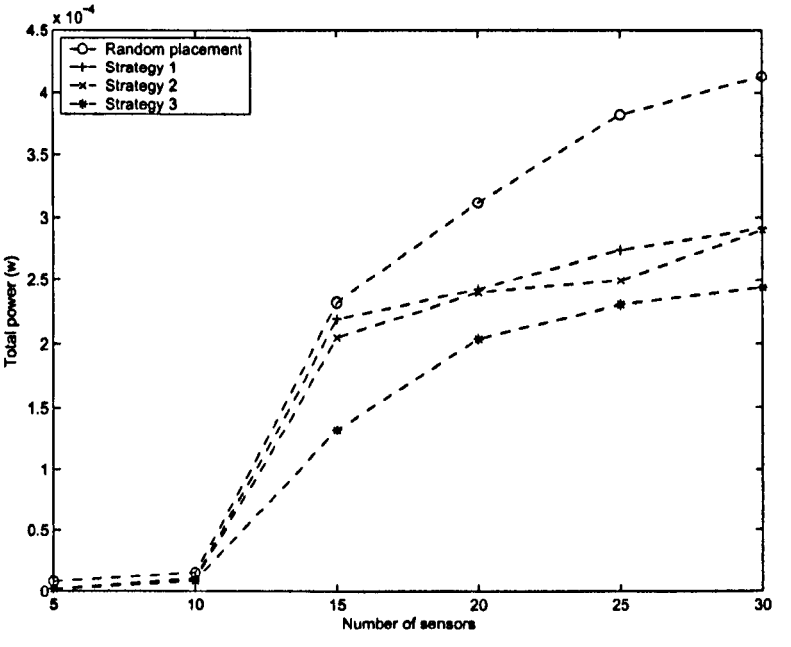

Figure 5.2: Comparison of the network total power for strategy 1, strategy 2, strategy 3 and random placement

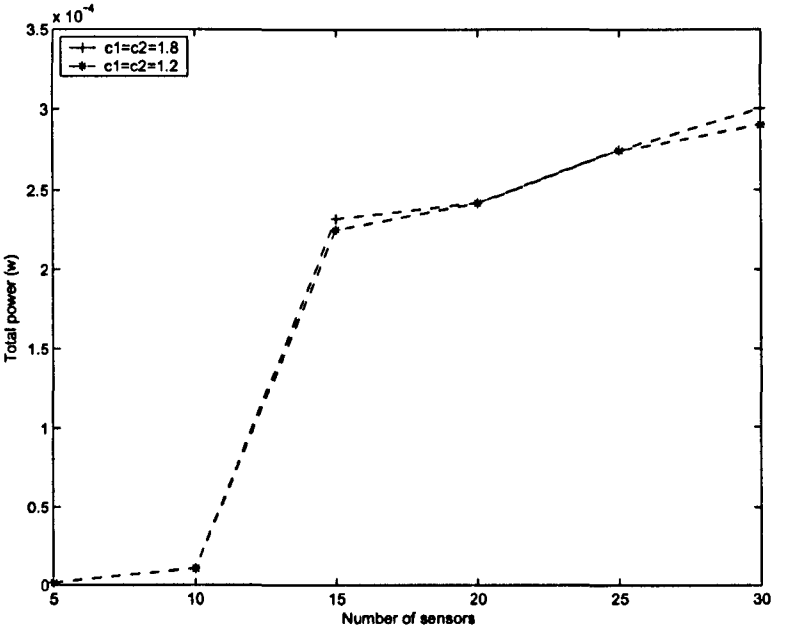

Figure 5.3: Comparison of the strategy 1 with different PSO parameter

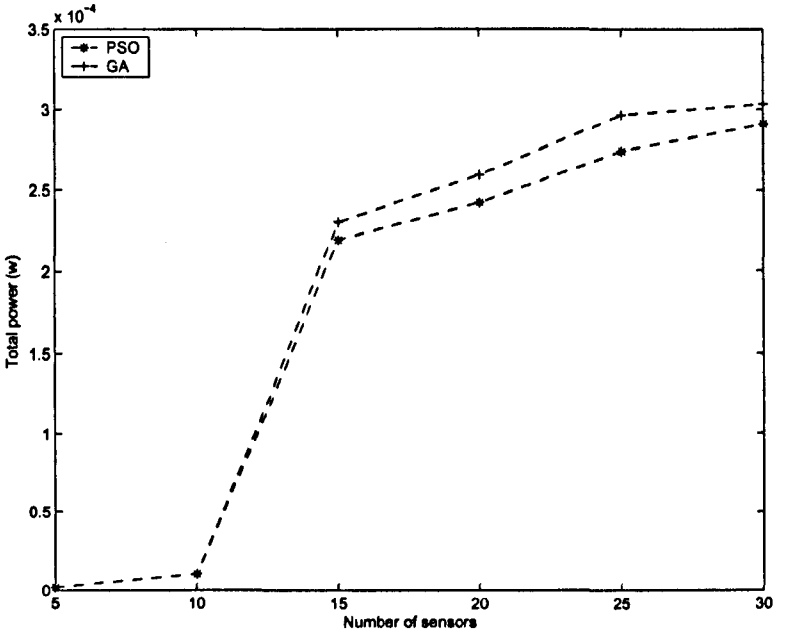

Figure 5.4: Comparison of the strategy 1 with PSO and GA

effectiveness as GA but with significantly better computational efficiency using less functions evaluations. The problem in the strategy 1 was solved using PSO and GA over 1000 iterations. The best values are used to find the optimal location of the sink node. The total power of the sensor networks are calculated based on those results. Figure 5.4 shows that PSO always finds a better solution than the GA for 1000 iterations, with PSO taking significantly less time than GA over 1000 runs.

In this work, PSO algorithm gives better result in most cases with comparatively less computational time. This results are interesting for online networking optimisation algorithms where computational cost and time may have significant performance differences in network operations.

5.9 Conclusions

In this chapter we investigated the sink node placement problem in wireless sensor networks. The novel idea in this work is the placement of sink node in a given wireless sensor networks region rather than the placement of sensor nodes as commonly investigated. We formulated a nonlinear programming problem to determine the location of the sink node inside the given sensor network region. Our simulation results show that the three proposed optimal strategies are of significant benefit over random placement scenarios where energy consumption is vital in wireless sensor networks. In this work we have also utilised the particle swarm optimiser which is effective in solving NP-hard nonlinear optimisation problems than GA. GA is a well established algorithm with many versions and many applications. In the other hand, PSO is a recently proposed algorithm and has become popular in many applications. Our results also shows that PSO is relatively better at finding solutions quickly than GA, as already established in the literature [20, 67].

Chapter 6

Swarm Intelligence Based Routing for Mobile Ad Hoc Networks

6.1 Introduction

Without relying on any existing, pre-configured network infrastructure or centralised control, ad hoc networks are useful in many situations where impromptu communication facilities are required. Lack of a fixed network and the nature of the nodes give rise to the challenges such as reliable data routing, dynamic network topologies, changing environments, selfish nodes and scarce radio resources [2, 1]. Routing in ad hoc networks faces extreme challenges from node mobility, potentially very large number of nodes and limited communication resources (bandwidth and energy). Therefore routing protocols for ad hoc networks have to adapt to frequent and unpredictable topology changes and optimise the limited resources. Ad hoc networking protocols are typically subdivided in two main categories based on how they perform routing function: proactive routing protocols and reactive on demand routing protocols.

Proactive routing protocols are derived from Internet distance-vector and link-state protocols. The main characteristic of these protocols is the constant maintaining of route by each node to all other network nodes. The route creation and maintenance are performed through both periodic and event driven messages. Destination

sequenced distance vector (DSDV) [9] and optimised link state routing (OLSR) are proactive routing protocols.

Reactive on demand routing protocols establish the route to a destination only when there is a demand for it. This technique reduced the overhead packets which is vital in the limited resources communication networks. Ad hoc on demand distance vector (AODV) routing [5] and dynamic source routing (DSR) protocol [7] are examples of reactive on demand routing protocols.

In this chapter, we propose a Swarm intelligence based routing protocol for mobile Ad hoc Networks (SwAN) to mitigate the problems in mobile ad hoc networking. Mapping the pheromone laying and following behaviour of biological ants, our algorithm allows each node to choose the next node for packets to be forwarded on the basis of mobility influenced pheromone table.

6.2 Related Work

Swarm intelligence techniques are distributed, adaptive, robust and scalable solutions to the complex systems management and control problems. It boasts a number of advantages due to the use of mobile agents and stigmergy for the communication network management problem. A new class of algorithms, inspired by swarm intelligence, is currently being developed [101, 102, 32, 103] that can potentially solve problems of modern ad hoc mobile networks. In this section, we give an introduction to existing ant based routing protocols for mobile ad hoc networks.

6.2.1 Ant-colony based Routing Algorithm (ARA)

Ant-colony based routing algorithm (ARA) was proposed for mobile ad hoc networks in [101]. The main goal in the design of the algorithm was to reduce the overhead for routing. The routing algorithm is similar to many other routing approaches and consists of three phases such as route discovery, route maintenance and route failure handling. The creation of new routes requires the use of a forward ant (FANT) and a backward ant (BANT). Forward ant is an agent which establishes

the pheromone track to the destination and backward ant establishes the pheromone track back to its origin.

Route discovery phase

Route discovery is the mechanism by which a node wishing to send a packet to a destination node obtains a source route to destination and it allows any node in the ad hoc network to dynamically discover a route to any other host in the ad hoc network. A FANT is an agent which is initiated at the source node to establishes route to the destination node and a BANT is initiated at the destination node to source node when any source node receives a FANT. A node which receives a FANT for the first time assign a value in its routing table. Every routing table has three values namely, destination address, next hop and pheromone value. The node interprets the source address of the FANT as destination address, the address of the previous node as the next hop, and calculates the pheromone value depending on the number of hops it took the FANT to reach the node. The node then forwards the FANT to its neighbour. The destination node extracts the information of the FANT and creates a BANT, and returns to its source node. The BANT follow the same path as the FANT but in the opposite direction as found in the FANT. When the source node receives a BANT from the destination node, the route discovery phase is completed, the source node can send the data packet to the specific destination.

Route maintenance phase

Route maintenance is another important phase in the routing algorithm and this phase is responsible for the maintenance of the routes during the communication. ARA does not use any control packet for route maintenance. Once the routes established between a source and destination nodes regular data packets are used to maintain the path. In the self-organising systems, established paths do not keep their initial values forever. For example in the ant systems, the pheromone evaporate with time. When a node n_i send a data packet to destination d to a neighbour node n_j, it increases the pheromone value of the entry by $\triangle\rho$. In other words, this path to the

destination is strengthened by data packet. The next hop n_i increases the pheromone value of the entry by $\Delta\rho$, i.e., the backward path to the source node is also strengthened. The evaporation process of the real pheromone is modelled by decreasing the pheromone values according to the following equation.

$$\rho_{i,j}(t+\tau) = (1-q).\rho_{i,j}(t) \quad q \in (0,1] \tag{6.1}$$

Route failure handling phase

Route failure handling is another important phase especially in mobile ad hoc networks because node mobility causes frequent route failures. ARA recognise a route failure through a missing acknowledgement on the MAC layer. The ARA algorithm was implemented in IEEE 802.11 on the MAC layer. If node receives a ROUTE_ERROR message from a broken link, the node searches for an alternative link in its routing table. If there is an alternative route to the destination then it will send the data packet through that path. Otherwise, the node will send this information through the neighbour. Then, the source node has to initiate a new route discovery.

The main goals of the ARA was to reduce the necessary routing overhead while maintaining the other network performances. ARA uses a multipath and purely reactive scheme and this results in a reduction of in the number of control packets. However, ARA does not use any neighbour management techniques. Therefore other network performance metrics suffer. In ARA [101], a pheromone evaporation factor is used, i.e., decrease pheromone value with time, as found in biological systems, which does not utilise any information from the neighbouring nodes.

6.2.2 Probabilistic emergent routing for mobile ad hoc networks

Probabilistic emergent routing mobile ad hoc networks (PERA) was described in [102] and it uses an entirely proactive approach. This algorithm uses three kinds of agents: regular forward ants, uniform forward ants and backward ants. Uniform and

regular forward ants are agents that are of unicast type. These agents pro-actively explore and reinforce available paths in the network. They create a probability distribution at each node for its neighbours. The probability or goodness value at a node for its neighbour reflects the likelihood of a data packet reaching its destination by taking the neighbour as a next hop. Backward ants are utilised for propagating the information collected by forward ants through the network and to adjust the routing table entries according to the perceived network status. Nodes pro-actively and periodically send out forward regular uniform ants to randomly chosen destinations. Each routing table has the values of destination, next hop and probability. It contains the goodness values for a particular neighbour to be selected for particular destination. In addition to that each node also maintains a table of statistics for each destination d to which a forward ant has been previously sent; the mean and the variance (μ_{sd},σ^2_{sd}) for the routes between source node s and destination node d. Thus, regardless of whether a packet needs to be sent from a node to any destination in the network or not, each node creates and periodically updates the routing tables.

PERA technique uses positive and negative reinforcement to manage the network multipath routes for the destinations. PERA uses regular forward ants to find new routes to the destinations even after a route has been established. This seems an inefficient way of maintaining the routes considering the fact that forward ants increase the number of control packets.

6.2.3 Mobile agent based routing protocol for mobile ad hoc networks

A routing algorithm for mobile ad hoc networks, which combines the on-demand routing algorithm capability of AODV routing protocol with a distributed topology mechanism using ant mobile agent is proposed in [104]. This method forms a hybrid of both ant based routing and AODV routing protocol to overcome some of their inherent drawbacks. In Ant-AODV ant agents work independently and provide routes to the source nodes. The nodes use on demand route discovery to find routes to destinations for which they do not have a fresh enough route entry. The use of ants

with AODV increases the node connectivity which in turn reduces the amount of route discoveries. As a direct result of providing topology information to the nodes using ants, the foundations for designing a distributed network control and management get automatically laid. Higher connectivity and reduced end-to-end-delay are achieved at the cost of extra processing of the ant message and the slightly higher overhead occupying some network capacity.

6.2.4 An adaptive swarm-based distributed routing algorithm

An adaptive swarm-based distributed routing algorithm, Adaptive-SDR, was proposed for wireless networks in [103]. This algorithm clusters the network nodes into colonies. Then, it uses two types of ants like mobile agents called local ants (within the colony) and colony ants (between the colonies). The task of the colony ants is to find routes one cluster to other cluster and local ants are responsible for finding routes within their colonies. In the colony routing table, all the neighbours of the nodes are included in the next hop list. The local routing table only includes the neighbour nodes belong to the colony. The routing table are updated similar to AntNet algorithm [18, 3], while adding some modifications.

An ant colony system algorithm solving the minimum broadcast problem can be found in [105]. In [32, 103], authors summarised that many distributed time varying network communications problems are thus well suited to swarm-based optimisation.

As the algorithms proposed in this sections are based on ant foraging behaviour and using distributed control to forward data to the destination. These types of algorithms are normally scalable and robust but these properties are not yet studied well.

6.3 The SwAN Protocol

6.3.1 Motivation

Most of the proposed ad hoc networking routing protocols use proactive or hybrid (reactive and proactive methods) routing techniques. Normally, proactive routing causes high routing control packet overhead and bandwidth problems. The proposed swarm intelligence based routing protocols described in section 6.2 do not use any neighbour node update to maintain route connections. When established route connections are broken, normally error messages are sent to the source node and route discovery phase initiated again – this process increases the number of control packets and affects the network performances (end-to-end delay and delivery ratio). In ARA [101], a pheromone evaporation factor is used, i.e., decrease pheromone value with time, as found in biological systems, which does not utilise any information from its neighbour node. This process may result in the loss of optimal path for data forwarding. In the route discovery process, ARA uses the number of hops to reach specific node for computing its initial pheromone value which is not efficient in mobile ad hoc networks (frequent topology changes due to node mobility). SwAN uses an entirely reactive approach and also maintains possible neighbour nodes information to forward data packets to the given destination in an efficient way. Each node which is participating in the transmission process, maintains next possible neighbour nodes information and its pheromone values. The pheromone values are assigned to be inversely proportional to the excess received packets power level. We also use pheromone evaporation factor, i.e., decrease pheromone values with time, as a function of node mobility. It is clear that mobility is one of the most important factors for route changes in mobile ad hoc networks. In SwAN, the rate of change of mobility information in the networks is predicted using received HELLO packets(HELLO packets are used for neighbour management in dynamic networks) power level which is then translated into a pheromone decay. By this mobility prediction techniques, we may increase the interval between HELLO packets broadcast and thereby reduce the control packets overhead, while still maintaining sufficient connectivity between

forwarding nodes.

6.3.2 Packet design

SwAN protocol needs to maintain four types of control packets. These are forward ant (FANT) packet, backward ant (BANT) packet, route error (RERROR) packet and hello (HELLO) packet.

FANT: Forward ant packet is sent when a node wants to send data packets to an unknown destination. This packet contains information of source address, destination address, message type and time to live (TTL).

BANT: Backward ant is launched from destination node if route FANT packet is received by a destination node. This contains source address, destination address, message type and time to live (TTL).

RERROR: Route error packets are sent to source node if any intermediate node can not forward a data packet to the next node.

HELLO: Hello packets are sent to neighbour nodes by active nodes which are participating in data forwarding with the default value of a packet in every 5s.

6.3.3 Protocol description

The algorithm for the SwAN protocol can be defined as a procedure involving the following steps:

- Any source node (s) sends forward ant (FANT) as a broadcast in a control manner (with time to live (TTL)) if it wants to send packets to the destination node (d) and does not have any previous record in its routing table.
- When a node receives a FANT, it forwards to neighbour nodes. If such a FANT has been received, the node silently discards the newly received FANT.
- When the destination node receives a FANT from any source node, then it launches a backward ant (BANT) in the same way as the FANT broadcast. Nodes also discard a BANT if it has already received a BANT, but it records the information of where it is coming from.

- When nodes receive a BANT from any other node, it assigns pheromone values inversely proportional to the excess received signal power level associated with the neighbour node and destination node (all packets have the power level information).
- When the source node (s) receives the first BANT it starts to forward data packets to the destination node (d).
- Each node's routing table contains neighbour node information and its pheromone values related to the specific destination.
- Nodes which are participating in the forwarding process also send a HELLO packet to the neighbours. Any node hearing a HELLO packet sends back a HELLO packet to the neighbour node.
- Mobility information of the node can be predicted from the sequential received HELLO packets power level differences. The pheromone table update is then made utilising this information.
- When a node receives a data packet to be forwarded, it chooses the highest pheromone value from its routing table.
- If a node does not receive acknowledgement (ACK) from next node, it selects the next highest pheromone value node to forward the data packet.
- If the node routing table does not have an alternative neighbour node to select from, then it sends a route error (RERROR) packet to the source node. Then the source re-initiates path finding with FANT.

6.3.4 Pheromone table initialisation

Each node which is receiving the BANT maintains a pheromone table with respect to the destination (d) node. When node i receives a BANT packet from its sequential neighbour node j, it assigns initial pheromone value represented by $\tau^0(i, j, d)$, using

the following equation:

$$\tau^0(i,j,d) = \frac{1}{P_r^0(i,j,d) - P_{thres}} \tag{6.2}$$

where $P_r^0(i,j,d)$ is the received BANT packet power level at node i from node j with respect to the destination (d) and P_{thres} is the minimum power level for which a packet can be correctly received by node i. Note that $P_r^0(i,j,d) > P_{thres}$. We assume that P_{thres} is equal for all nodes. We have modelled our pheromone values inversely proportional to the excess received power level to perform shortest path routing i.e., possibly minimum hops intended to reduce end to end delay and re-transmission problems.

6.3.5 Mobility information

The SwAN protocol does not need any additional control packet to estimate mobility information and it uses HELLO packets. Nodes check received power level differences in certain time period and predict the mobility factor of neighbour nodes. If a high difference in power level is found then node pheromone values decay are very fast. Pheromone values for each active nodes will be updated related to the estimated mobility factor as follows:

$$\tau^n(i,j,d) = \frac{1}{P_r^n(i,j,d) - P_{thres}} \tag{6.3}$$

$$\tau_m^n(i,j,d) = \tau^n(i,j,d)(1 - m^n(i,j,d)) \tag{6.4}$$

where $\tau^n(i,j,d)$ is the current pheromone value without the effect of mobility factor and $\tau_m^n(i,j,d)$ is the current pheromone value with the effect of mobility factor. m^n is defined as the current mobility factor computed at the last HELLO packet received and maintained until the next HELLO packet is received, given by,

$$m^n(i,j,d) = \begin{cases} \frac{|P_r^n(i,j,d) - P_r^{n-1}(i,j,d)|}{P_t - P_{thres}} & \text{if } i \text{ received HELLO packets} \\ 1 \quad \text{otherwise} \end{cases} \tag{6.5}$$

where $P_r^n(i,j,d)$ and $P_r^{n-1}(i,j,d)$ are received power level of two consequent HELLO packets by node i from j and P_t is the packet transmission power of each node. Note

that $0 \leq m^n(i,j,d) \leq 1$. The idea of introducing a mobility factor is to avoid high mobility node to participate in the forwarding process which can cause frequent link failure. HELLO packets broadcasting interval can be increased by predicting the neighbour node mobility, and this will significantly reduce control packet overhead which is crucial for energy constrained nodes.

6.4 Simulation Model

Simulations were carried out to compare AODV, DSR, DSDV, ARA and SwAN performances such as end-to-end delay, packet delivery ratio and routing overhead. We have implemented our protocol in NS 2.27 which is a common discrete event simulator used by the mobile ad hoc networks research community. MAC layer is implemented using IEEE 802.11 Distributed Coordination function (DCF). The transmission range of each of the mobile nodes is set to 250m. Our protocol evaluations are based on the simulation of 50 wireless nodes forming an ad hoc network, moving about over a rectangular 1500m × 300m flat space for 600 seconds simulation time (This is a general simulation model used by the mobile ad hoc networking research community to compare the protocols). We pre-generated 60 different scenario files with varying movement patterns and traffic loads, and then ran all four routing protocols for each of these scenario files. The mobility model uses the random way point model in a rectangular area. The pre-generated movement scenario files we used for each simulation is characterised by a pause time. Each node station begins the simulation by remaining stationary over the pause time seconds. It then selects a random destination in the 1500m ×300m space and moves to that destination at a speed distributed uniformly between 0 and a maximum speed. We ran our simulations with movement patterns generated for 6 different pause times: 0, 30, 60, 120, 300 and 600 seconds and with maximum speeds of $20ms^{-1}$ (high mobility) and $1ms^{-1}$ (low mobility). Constant bit rate (CBR) traffic sources were chosen, as the aim is to compare the performance of each routing protocol. The traffic used was 512 bytes sent from 10 sources with a rate of 4 packets per second. The mean of the performance values were computed for comparing the five protocols.

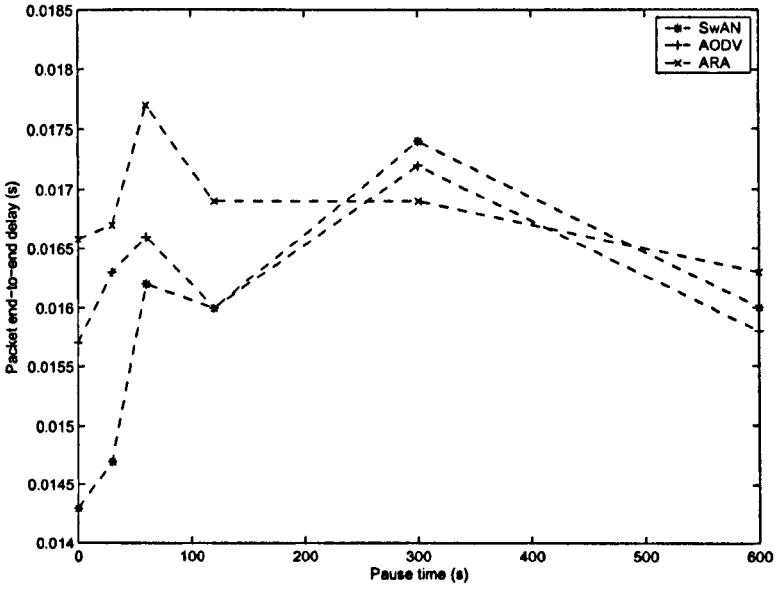

Figure 6.1: Comparison of the delay as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$

6.5 Simulation Results

We have compared the proposed SwAN protocol end-to-end delay, packet delivery ratio and routing packet overhead with the de facto ad hoc routing protocol AODV, DSR and DSDV. We have also compared SwAN with AODV and ARA [101]. The results for each of the performance parameter is given below.

6.5.1 Average end-to-end delay

The average end to end delay includes buffering delay during route discovery, queueing delay at interface queue, re-transmission delays and transfer times. The Figures 6.1 and 6.2 show that end to end delay of SwAN protocol is better than AODV and ARA in most cases. But Figure6.1 shows a peak at pause time 300s where the SwAN end to end delay is higher than ARA and AODV. This due to the random components in mobility and traffic models. In the high mobility scenarios

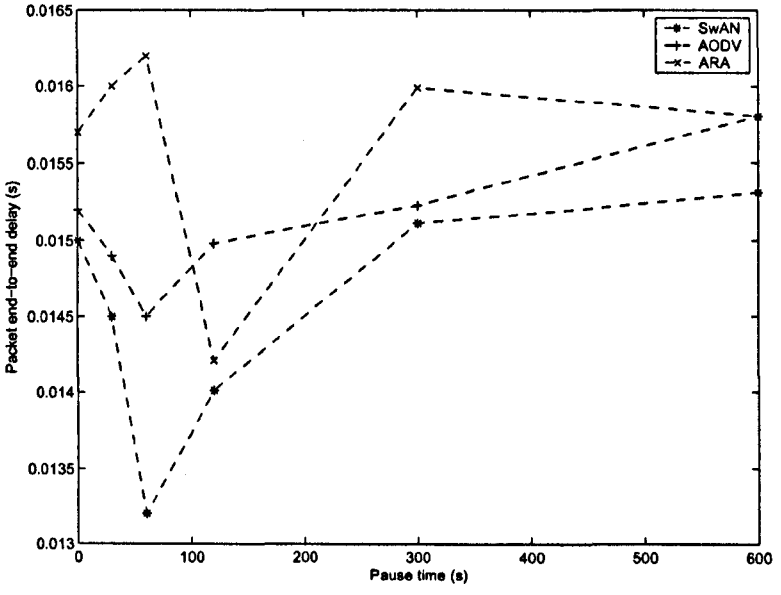

Figure 6.2: Comparison of the delay as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$

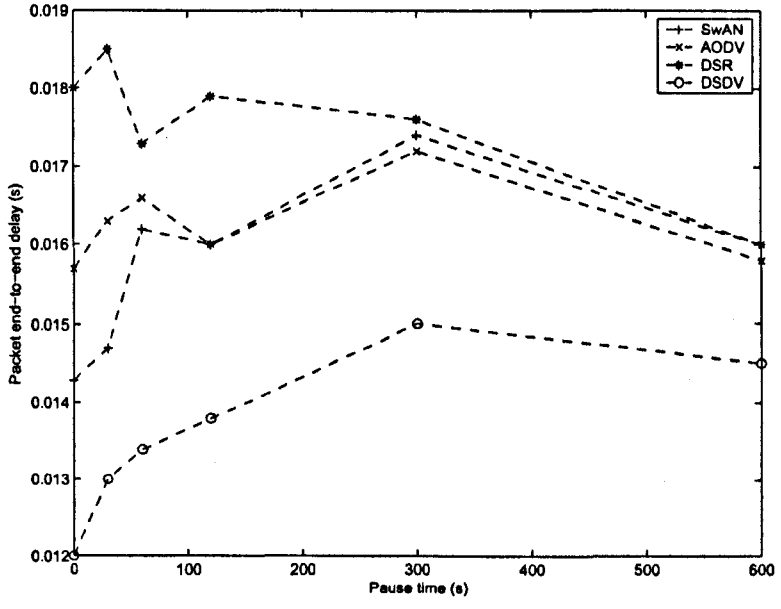

Figure 6.3: Comparison of the delay as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$

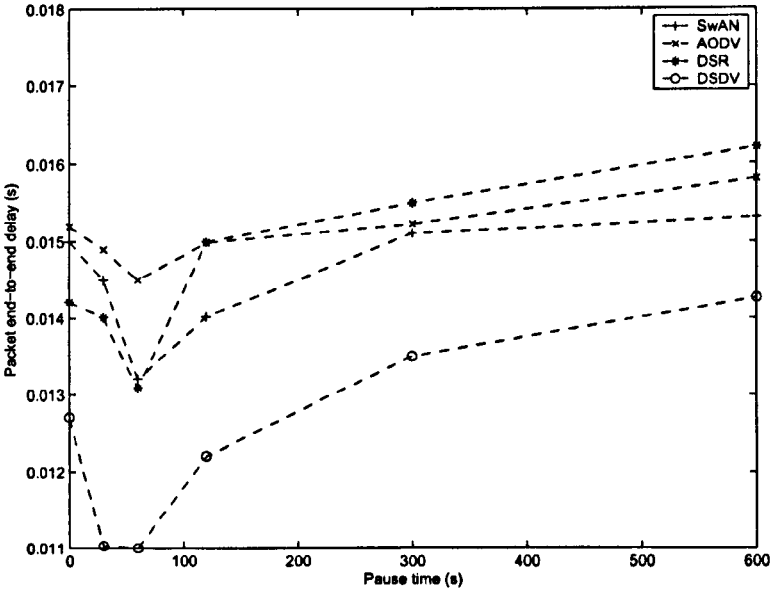

Figure 6.4: Comparison of the delay as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$

in Figure 6.1 with 20 m/s maximum velocity, SwAN gives better performance than AODV. The SwAN performs shortest path routing and gives each node an alternative route to packet forwarding which results in a better end to end delay. Use of mobility prediction in SwAN which chooses low mobility nodes to forward data, reduces frequent route failure. SwAN always gives better performance than ARA because ARA does not efficiently maintain routing tables which in turn causes frequent route failure and hence generates high end to end delay in packet forwarding. Figures 6.3 and 6.4 show that SwAN always gives better performance than DSR because DSR uses source routing mechanisms that cause frequent route failure and hence generates high end to end delay in packet forwarding. Being proactive, DSDV on the other hand gives the better end to end delay. It can be seen in all Figures that there is a high differences in performance at 0 pause time(high mobile) and small differences in (low mobility).

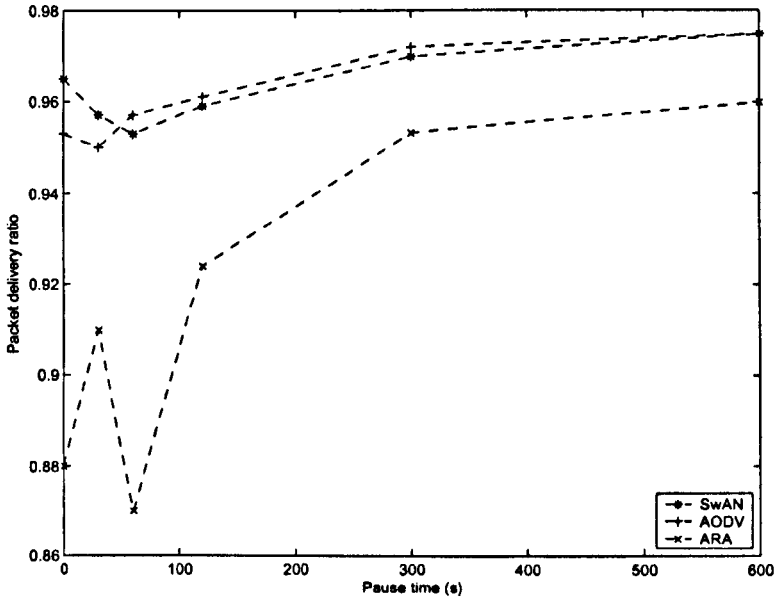

Figure 6.5: Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$

6.5.2 Packet delivery ratio

Packet delivery ratio is the ratio between the number useful packets received at all destination nodes and number of data packets sent by the CBR source. Figures 6.5, 6.6, 6.8 and 6.7 show the fraction of data packets received and data packets sent with maximum velocity 20 m/s and 1 m/s. Comparing the five protocols (SwAN, AODV, ARA, DSDV and DSR) AODV and SwAN deliver between 94% and 100% of the packets in all cases. In high mobility scenario, SwAN gives better delivery ratio and is very close to the AODV. However in the low mobility scenario SwANs' performance is slightly lower than the AODV packet delivery ratio. ARA packet delivery ratio is lower than AODV and SwAN in all scenarios because ARA does not use any neighbour HELLO packets to update its next possible forwarding neighbour nodes. Figures 6.5, 6.6, 6.8 and 6.7 show unpredictable peak in some pause times due to the random components are used in networking simulations. To reduce this effect, we have performed the simulations number of times and have taken the average

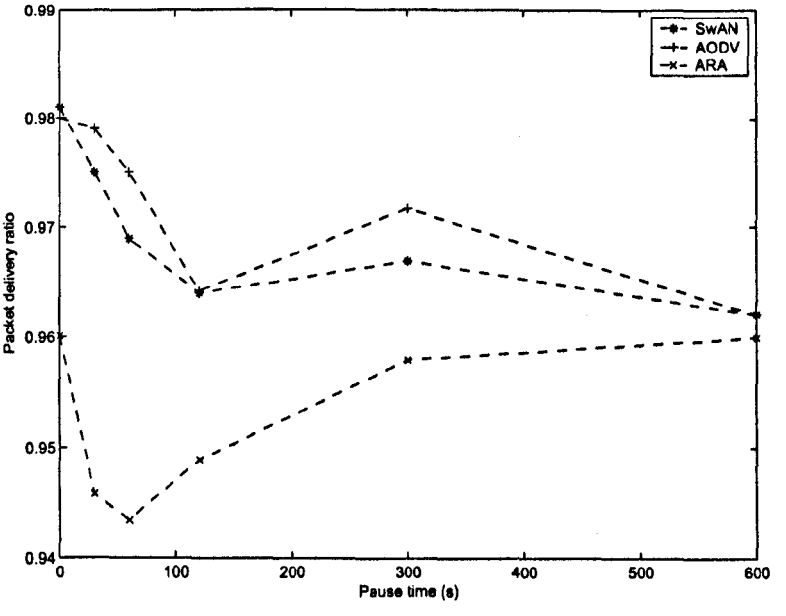

Figure 6.6: Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$

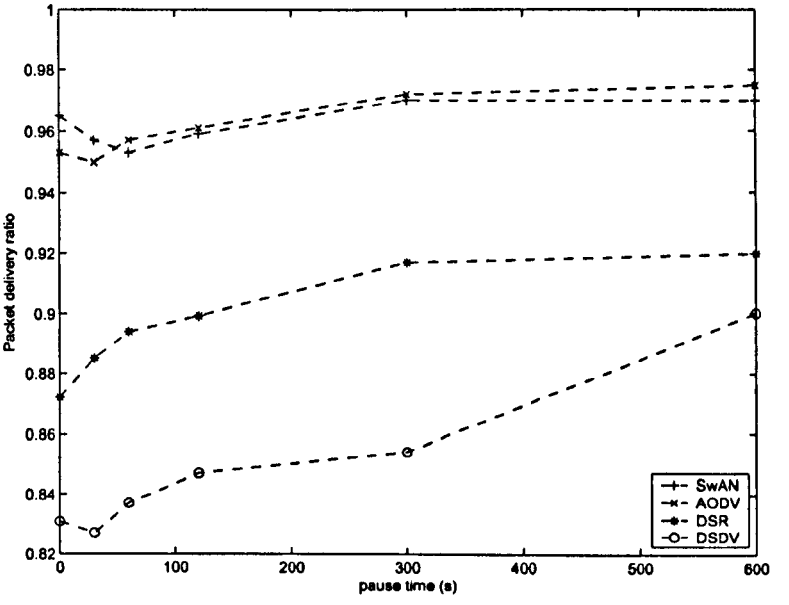

Figure 6.7: Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $20ms^{-1}$

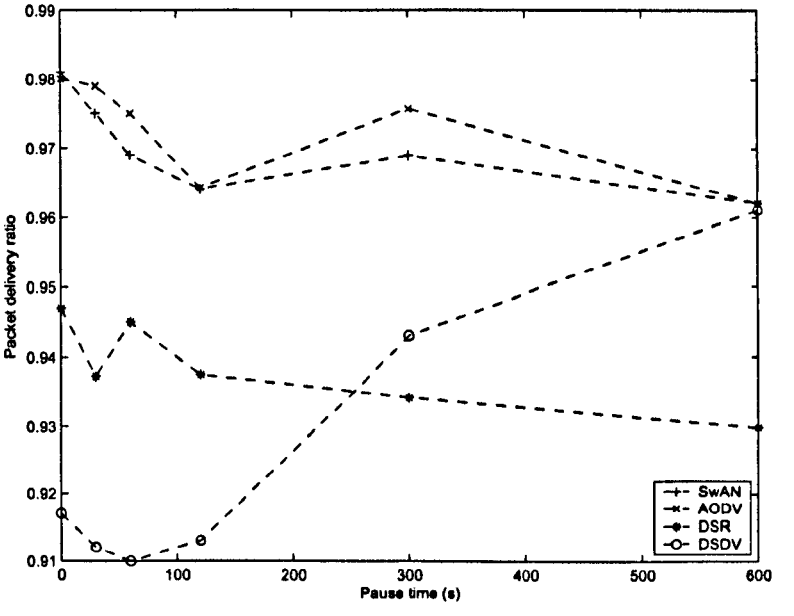

Figure 6.8: Comparison of the throughput as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$

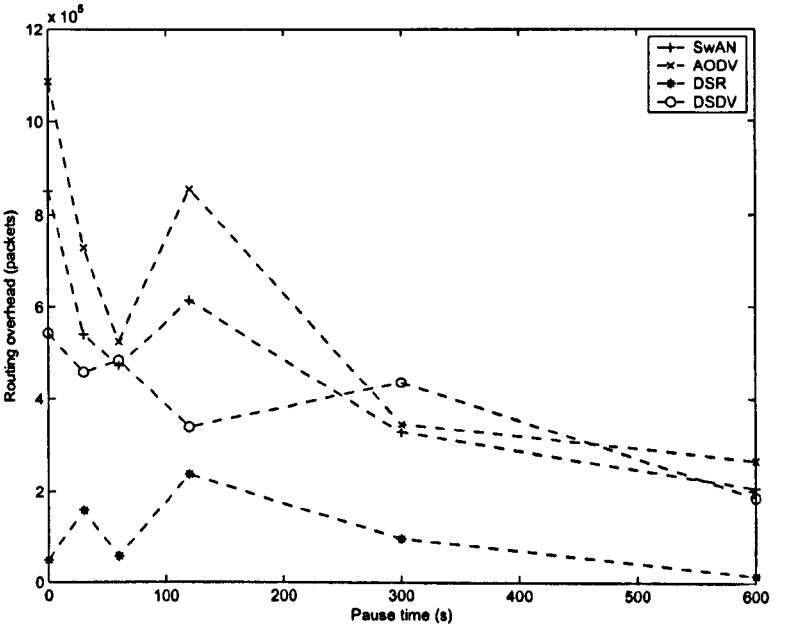

Figure 6.9: Comparison of overhead packets as a function of pause time for node number 50 and maximum velocity of $20ms^{-1}$

values to plot the performance metrics.

6.5.3 Routing overhead

Routing overhead is the number of routing packets transmitted per data packet received at the destination. As seen in figures 6.11 and 6.12, ARA uses low number of control packets to establish a route due to ARA not broadcasting any HELLO packets. However, ARA performances like packet delivery ratio and end to end delay are very low. The SwAN makes lower use of overhead packets than AODV in all cases as demonstrated in figures 6.11 and 6.12. It is clear that the use of mobility prediction and multi-path for data packets could reduce the number of control packets. AODV normally requires that each node periodically transmits a HELLO message, with a default rate of once per second. SwAN on the other hand requires only 5s intervals because of the mobility prediction and it could be increased for the low mobility scenarios while maintaining similar performance levels.

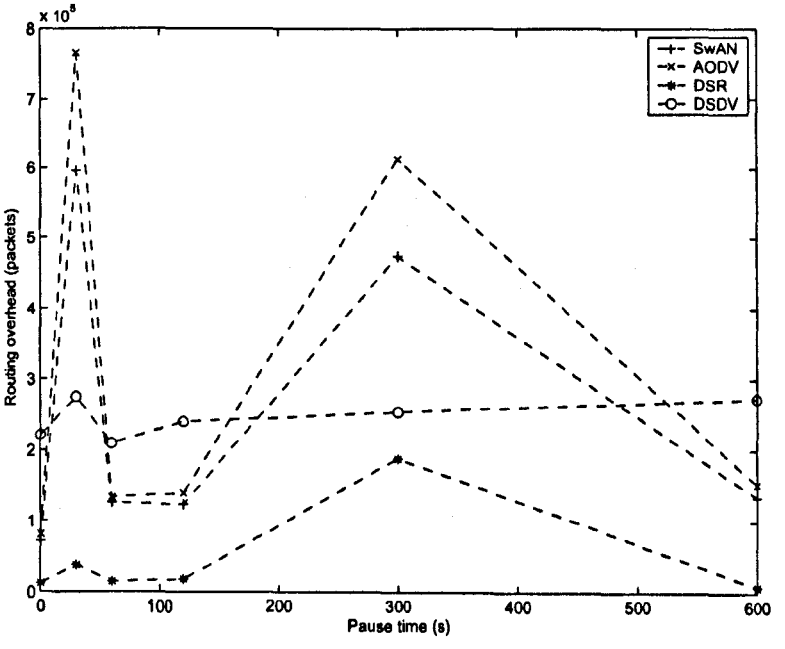

Figure 6.10: Comparison of the overhead packets as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$

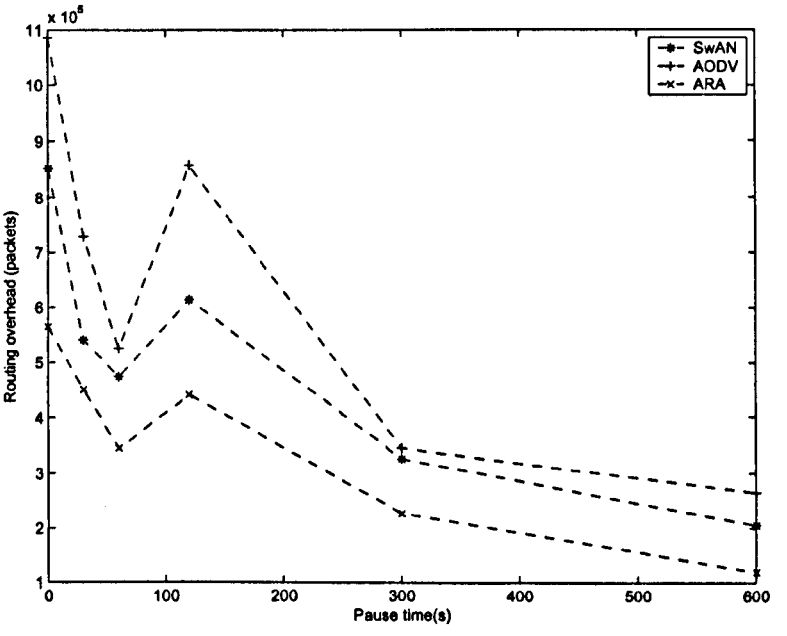

Figure 6.11: Comparison of overhead packets as a function of pause time for node number 50 and maximum velocity of $20ms^{-1}$

As seen in figures 6.9 and 6.10, the total number of routing overhead packets in DSDV is independent of traffic patterns and is always sending control packets. However DSR uses very low number of control packets to establish a route. Its other performances like packet delivery ratio and end to end delay are very low. Figures 6.13 and 6.14 show the performance measures packet overhead, end-to-end delay and throughput for maximum velocity $20ms^{-1}$ and $1ms^{-1}$ respectively. The normalised performance values in the parallel coordinate representation clearly show that SwAN gives better performance in all scenarios.

The simulations results are reported in this section for SwAN HELLO packet interval of 5s. We have also carried out the simulations for other HELLO packet intervals (1s, 2s and 10s). However the best performances are obtained for 5s intervals. Some figures show unexpected peak values for particular pause time due to the many random parameters involved in the simulation model and the transience of the network links.

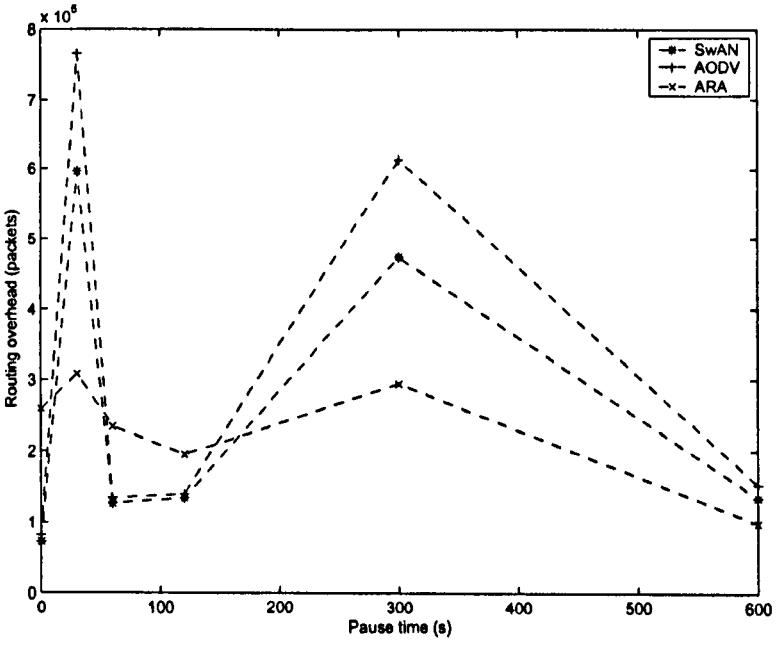

Figure 6.12: Comparison of the overhead packets as a function of pause time for node number 50 and maximum velocity $1ms^{-1}$

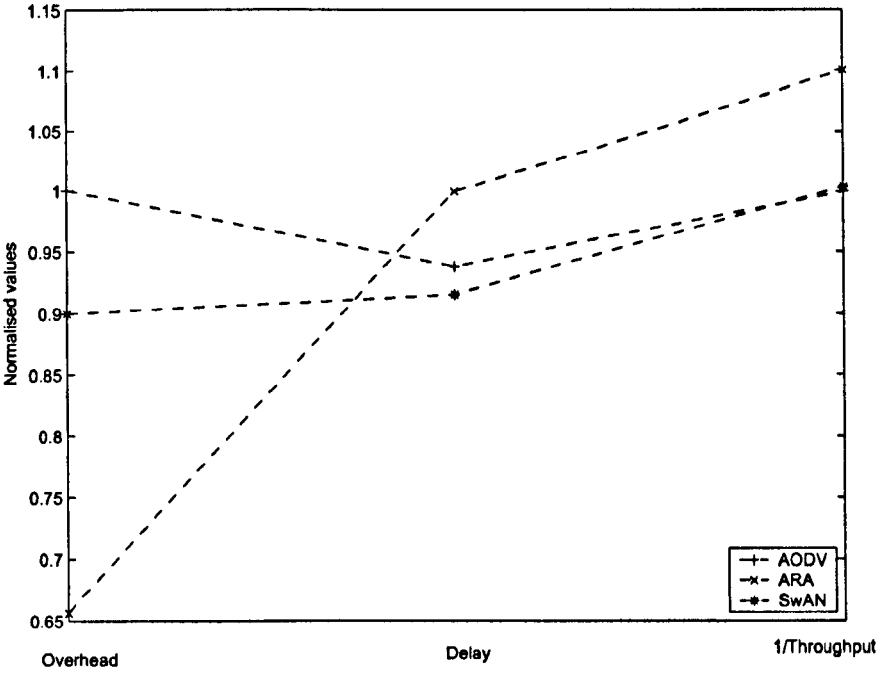

Figure 6.13: Parallel coordinate presentation for pause time 300s and maximum velocity of $20ms^{-1}$

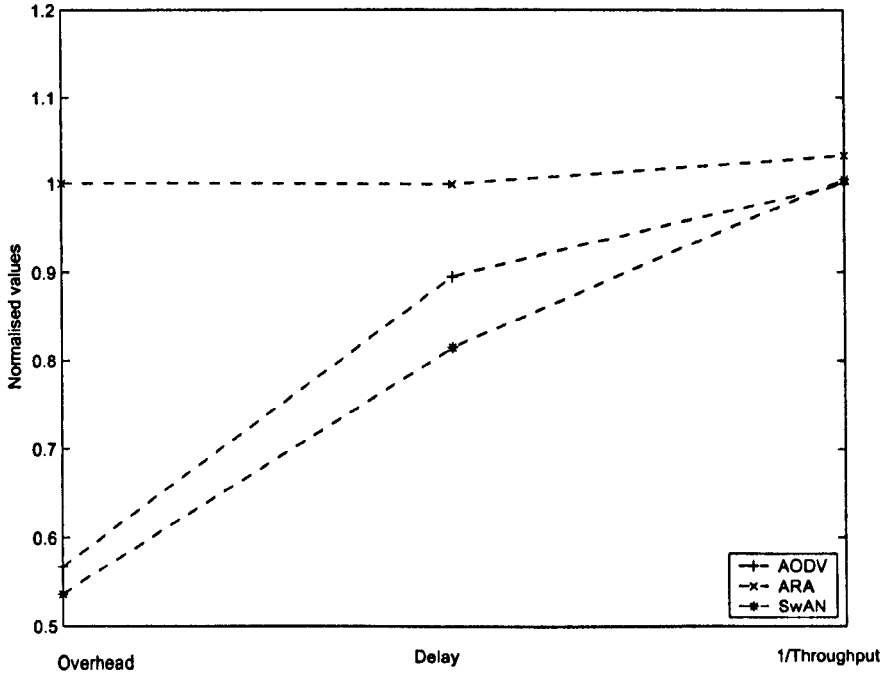

Figure 6.14: Parallel coordinate presentation for pause time 300 and maximum velocity of $1ms^{-1}$

6.6 Conclusions

In this chapter, a swarm intelligence based reactive routing protocol for mobile ad hoc networks (SwAN) was proposed. Mapping the pheromone laying and following behaviour of biological ants, our algorithm allows nodes to choose the next node for packets to be forwarded on the basis of mobility influenced pheromone table. The effectiveness of the proposed approach is demonstrated through an extensive simulation study. Simulation results show that SwAN gives better end to end delay than AODV. Interestingly, SwAN always uses a lower number of overhead packets to perform routing in all cases than AODV. On the other hand packet delivery ratio in SwAN is slightly lower than AODV in the low mobility case, but in the high mobility case SwANs' performance is significantly closer to AODV.

Scalability in ad hoc mobile networks is an inherently difficult goal due to the mobility of the nodes and the transience of the network links. Existing ad hoc network routing protocols, which experience some performance degradation when used in increasingly large networks, is a challenge and a significant amount of work is needed to reach this goal.

Chapter 7

Swarm Intelligence Based Energy Aware Routing Algorithm

7.1 Introduction

Ad hoc mobile networks have received widespread attention in recent years. The node in an ad hoc mobile network is typically powered by batteries with a limited energy supply. One of the main important and challenging issues in ad hoc wireless networks is how to conserve energy for maximising the lifetime of its nodes which increases the lifetime of the whole network.

As we discussed in chapter 6, a routing protocol is an essential and challenging function to facilitate communication within the networks. The dynamic nature (mobility of nodes) of the ad hoc mobile network presents many challenges to routing function. When a node runs out of its available energy, it affects the network functioning. Therefore, an important issue in ad hoc mobile networks is how to conserve energy. Limited energy resources in the ad hoc networks adds more challenges to efficient routing in ad hoc mobile networks [106].

The power required by each node can be classified into two main categories such as communication related power consumption and computation related power consumption [107]. Communication related power consumption involves usage of the transceiver at the source, intermediate node and destination nodes. The transmitter

is used for sending control, route request and response, as well as data packets originating at or routed through the transmitting node. The receiver is used to receive data and control packets. A typical mobile radio may use modes in their operation which are transmit, receive and standby. Notably, maximum power is consumed in the transmit mode and the least in the standby mode. The computation power consumption related with protocol processing aspects. It involves usage of the CPU and main memory and other components of the node [108].

Physical layer, data link layer and network layer in ad hoc networks are closely coupled to power consumption. We therefore briefly summarise the power conservation schemes for physical layer, data link layer and network layer below.

At the physical layer, transmission power can be adjusted [109]. The use of excessive transmission power can increase the interference to other node and will cause an increase in transmission power by other nodes. Therefore, physical layer functions should include transmitting data at the minimum power level to maintain links and adapt to changes in transmission environment due to node mobility. Power control can maintain a link at the minimum power level, but can also prolong an existing link against interference by increasing the transmission power.

At the data link layer, energy conservation can be achieved by using effective retransmission request schemes and sleep node operation. The data link layer is thus responsible for wireless link error control, mapping network layer packets into frames and packets retransmission. A node transmitting packets to its destination nodes will be overhead all neighbouring nodes. Hence, all neighbouring nodes will consume power even though the packets transmission was not directed to them. Therefore, to reduce power consumption a node transceiver should be powered off when not in use [59].

Finally, the network layer is responsible routing packets, establishing the network service type, and transferring between the transport and data link layers. In general, paths are computed based on minimising hop count or delay. Nonetheless, to maximise the life of the mobile nodes, routing algorithms must select the best path from the view point of power constraints as a part of route stability. Ad hoc networks will require a routing algorithm where power efficiency is considered that can evenly

distribute packet relaying loads to each node to prevent nodes from being over-used.

In this chapter, we propose a Swarm intelligence based Energy Aware Routing (SEAR) algorithm for mobile ad hoc networks to mitigate the problems in mobile ad hoc networking. The neighbour node energy level and drain rate information are predicted and related to a pheromone decay as found in the natural foraging ant systems in the node routing table.

7.2 Related Work

In this section we present a brief description of the four main existing energy aware routing algorithms.

7.2.1 The minimum total transmission power routing (MTPR)

The Minimum Total Transmission Power Routing (MTPR) was proposed in [62]. MTPR makes use of a simple energy metric representing the total energy consumed along the route. If we consider a generic route $r_d = n_0, n_1 ..... n_d$, where n_0 is the source node, and n_d is the destination node and a function $T(n_i, n_j)$ denoting the energy consumed in transmitting over hop (n_i, n_j), then the total transmission energy for a route is calculated as:

$$P(r_d) = \sum_{i=0}^{d-1} T(n_i, n_{i+1}) \tag{7.1}$$

The optimal route r_0 satisfies the following condition:

$$P(r_0) = \min_{r_j \in r_*} P(r_j) \tag{7.2}$$

where r_* is the set of all possible routes. This algorithm significantly reduce power requirements and interferences. The power saving translates directly into increased node lifetime and increased network capacity through reductions in interference.

7.2.2 The min-max battery cost routing (MMBCR)

The Min-Max Battery Cost Routing (MMBCR) was proposed in [59]. The MMBCR considers a route with the best condition amongst paths impacted by each crucial node over each path is selected, whereas MTPR can only reduce the total transmission energy consumed per packets and MTPR does not reflect directly on the lifetime of each node which is crucial for determining the life time of the entire network. Let $c_i(t)$ be the battery capacity of node n_i at time t and $f_i(t)$ be a battery capacity function of each node n_i. The less capacity a node has, the more reluctant it should be to forward packets, so that the proposed value is $f_i(t) = 1/c_i(t)$. If only the summation of battery cost is considered, a route containing nodes with little remaining battery capacity may still be selected. The Min-Max Battery Cost Routing (MMBCR) defines the route cost as:

$$R(r_j) = \min_{r_i \in r_j} f_i(t), \tag{7.3}$$

The desired route r_o is obtained so that $R(r_o) = \min_{r_j \in r_*} R(r_j)$, where r_* is the set of all possible routes.

7.2.3 The conditional max-min battery capacity routing (CMMBCR)

The goal of the routing protocols is to maximise the lifetime of each node and use battery fairly. However, these two goals can not be achieved simultaneously by applying MTPR or MMBCR schemes. The MMBCR mechanism, for example, does not guarantee that the total transmission energy consumed per packet over a chosen path is minimised. The Conditional Max-Min Battery Capacity Routing (CMMBCR) [63] is a hybrid approach combining the MTPR and MMBCR mechanisms. The basic idea behind CMMBCR is that when all nodes in some possible routes between the source and destination have sufficient power remaining, a route with minimum total transmission power among these routes is chosen. The relaying packets for the low energy level nodes must be reduced because less total energy is required to forward packets for each connection and their lifetime is extended. However, if all have routes

with low battery capacity (i.e., below a threshold γ), a route including nodes with the lowest battery capacity must be avoided in order to extend lifetime of these nodes. The battery capacity for route r_j can be defined at time t as $R_j(t) = \min_{n_i \in r_j} c_i(t)$. For two nodes n_a and n_b, this mechanism considers two sets of routes, namely Q and A. Q is the set of all possible routes between n_a and n_b at time t, and A is the set of routes between any two nodes at time t for which the condition $R_j(t) \geq \gamma$ holds. If all nodes in a given path have remaining battery capacity higher than γ, select a path applying MTPR scheme, otherwise choose a route r_i with the maximum battery capacity.

7.2.4 The minimum drain rate mechanism (MDR)

The Minimum Drain rate (MDR) and the Conditional Minimum Drain Rate (CMDR) was proposed in [64]. MDR uses the drain rate as the metric that measures the energy dissipation rate in a given node. Each node n_i monitors its energy consumption caused by transmission, reception and overhearing activities and computes the energy drain rate, denoted by DR_i, for every T sampling interval by averaging the amount of energy dissipation per second during the past T seconds. The DR_i value can be calculated using the drain rate values DR_{old} and DR_{sample} which represent the previous and the newly calculated values.

$$DR_i = \alpha \times DR_{old} + (1 - \alpha) \times DR_{sample} \tag{7.4}$$

The value of α is set to 0.3 to give higher priority to the current sample . The corresponding cost function can be defined as:

$$C_i = \frac{RBP_i}{DR_i} \tag{7.5}$$

where RBP_i denotes the residual battery energy at node n_i. The minimum value of C_i for the maximum lifetime of a given path r_p can be defined as:

$$L_p = \min_{\forall n_i \in r_p} C_i \tag{7.6}$$

The minimum Drain Rate (MDR) mechanism is based on choosing the route r_M, contained in the set of all possible routes r_* between the source and the destination

nodes can be calculated as:

$$r_M = r_p = \max_{\forall r_i \in r_*} L_i \tag{7.7}$$

As seen above, MDR does not guarantee that the total transmission energy is minimised over a chosen route, as in MMBCR. A modified version of MDR called CMDR (Conditional Minimum Drain Rate) was then proposed. The CMDR technique is based on choosing a path with minimum total transmission energy among all possible routes with a lifetime higher than a given threshold, i.e., $\frac{RBP_i}{DR_i} \geq \delta$ as in the MTPR mechanism. If there are no routes in this condition CMDR switches to the basic MDR mechanism.

7.3 Swarm intelligence based Energy Aware Routing algorithm (SEAR)

Since most of the mobile nodes of an ad hoc network operate on battery power. It is very important to minimize the power consumption of each node in the network to maximise the total lifetime of the network. Most of the existing energy aware routing algorithms in the literature were based on the source routing methods which are commonly impractical where the node mobility is high. Each source node has to decide all nodes in the path with the availability of the energy level or drain rate information, which is inefficient for nodes with high dynamic topologies. All existing energy aware algorithms do not perform any shortest path routing; rather, they make decisions based only on the energy level information of the nodes which does not guarantee any other important performance measures like number of control packets, end-to-end delay or throughput. Those algorithms do not use any neighbour nodes information when established route connections are broken, normally send error message to the source node and initiate route discovery phase again – this process increases the number of control packets.

In this chapter, we propose a Swarm intelligence based Energy aware Routing (SEAR) algorithm for mobile ad hoc networks. Here, we use an entirely distributed

approach and maintain possible neighbour nodes information to forward data packets to the given destination in an optimal way. SEAR algorithm performs a shortest path approach to select the next node to forward to the destination. Each node which is participating in the transmission process maintains next possible neighbour node energy levels and drain rates information in a table called pheromone table. The pheromone values are assigned inversely proportional to the excess received packets power level to perform shortest path routing. It is clear that current node energy information is one of the most important actions for route changes in mobile ad hoc networks. In SEAR, the node energy level and drain rate information in the networks are predicted using neighbour HELLO packets and relate it to pheromone decay in the node pheromone table. By this energy prediction techniques, we may increase the network life time and interval between HELLO packets transmission and thereby reduce the control packets overhead, while still maintaining sufficient connectivity information between forwarding nodes.

7.3.1 Protocol Description

SEAR protocol needs to maintain four types of control packets. Those are forward ant packet (FANT), backward ant packet (BANT), route error packet (RERROR) and hello packet (HELLO). The algorithm for the SEAR protocol can be defined as a procedure involving the following steps:

- Any source node sends forward ant (FANT) as a broadcast in a control manner (with time to live (TTL)) if it wants to send packets to the destination node and does not have any previous record in its routing table.
- When a node receives a FANT, it forwards it to neighbour nodes. If such a FANT has been received, the node silently discards the newly received FANT.
- When the destination node receives a FANT from any source node, it launches a backward ant (BANT) in the same way as the FANT broadcast. Nodes also discard a BANT if it has already received a BANT, but it records the information of where it is coming from.

- When nodes receive a BANT from any other node, it assigns pheromone values inversely proportional to the excess received signal power level associated with the neighbour node and destination node. Each node's pheromone table contains neighbour node information and its pheromone values related to the specific destination.

- When the source node receives the first BANT it starts to forward data packets to the destination node.

- Nodes which are participating in the forwarding process also send a HELLO packet to the neighbours. Any node hearing a HELLO packet sends back a HELLO packet to the neighbour with the information of current energy level. HELLO packets are broadcasted in a regular time interval of Ts.

- Energy level information and the drain rate of the node can be predicted from the sequential received HELLO packets energy level information. The pheromone table update is then made utilising this information.

- When a node receives a data packet to be forwarded, it chooses the highest pheromone value from its pheromone table.

- If a node does not receive acknowledgement (ACK) from next node, it selects the next highest pheromone value node to forward the data packet.

- If the node pheromone table does not have an alternative neighbour node to select from, then it sends route error packet (RERROR) the to source node. Then the source re-initiates path finding with forward ant (FANT).

7.3.2 Pheromone table initialisation

Each node (i) which is receiving the BANT maintains a pheromone table with respect to neighbour node (j) and destination (d) node. When nodes receive BANT packets from its neighbour node, it assigns a pheromone value represented by $\tau^0(i, j, d)$,

using following equation:

$$\tau^0(i,j,d) = \frac{1}{P_r^0(i,j,d) - P_{thres}} \tag{7.8}$$

where $P_r^0(i,j,d)$ is the received BANT packet power level at node i from node j with respect to the destination (d), P_{thres} is the minimum power level for which a packet can be correctly received by node i, so that $P_r^0(i,j,d) > P_{thres}$. We assume that P_{thres} is equal for all nodes. We have modelled our pheromone values inversely proportional to the excess received power level to perform shortest path routing i.e, possibly minimum hops intended to reduce end-to-end delay and re-transmission problems.

7.3.3 Energy level information

The SEAR protocol does not need any additional control packets to estimate energy level information and it uses neighbour HELLO packets. When any node receives a HELLO packet from its neighbour, it sends back a HELLO packet with current energy level information. Nodes get the current energy level and predict the energy drain rate (over Ts intervals). If a high energy drain rate is found then the node pheromone table values decay are very fast. Pheromone values for each active node will be updated relative to the estimated energy drain rate and its current energy level information as follows:

Let $E^n(i,j,d)$ be the current energy level information received from HELLO packet and $E^{n-1}(i,j,d)$ is the energy level information of node j from previous HELLO packet information. The drain rate of node j can be calculated as,

$$D^n(i,j,d) = \frac{E^n(i,j,d) - E^{n-1}(i,j,d)}{T} \tag{7.9}$$

Energy factor of node j can be defined by considering the current energy level $E^n(i,j,d)$ as,

$$F^n(i,j,d) = \begin{cases} \frac{E^n(i,j,d)}{D^n(i,j,d)} \\ 0 \quad \text{if } i \text{ has not received HELLO} \end{cases} \tag{7.10}$$

where $F^n(i,j,d)$ is the current energy factor computed at the last HELLO packet received.

The pheromone values for each active node will be updated relative to the estimated energy factor as,

$$\tau^n(i,j,d) = \frac{1}{P_r^n(i,j,d) - P_{thres}} \tag{7.11}$$

$$\tau_F^n(i,j,d) = \tau^n(i,j,d) \times F^n(i,j,d) \tag{7.12}$$

where $\tau^n(i,j,d)$ is the current pheromone value without the effect of energy factor and $\tau_F^n(i,j,d)$ is the current pheromone value with the effect of the energy factor.

The idea of introducing the energy factor is to avoid high drain rate and low remaining energy level nodes to participate in the forwarding process which can cause certain nodes to run out of energy faster than other nodes in the networks.

7.4 Simulation Model

We implemented the SEAR protocol in NS 2.27 which is a common discrete event simulator used by the mobile ad hoc networks research community. MAC layer is implemented using IEEE 802.11 distributed coordination function (DCF). The transmission range of each of the mobile nodes is set to 250m. Our protocol evaluations are based on the simulation of 50 wireless nodes forming an ad hoc network, moving about over a rectangular 1500m × 300m flat space for 900 seconds simulation time(Here we have choosen the same network model used by mobile ad hoc networking research community). The different scenario files were pre-generated with varying movement patterns and traffic loads, and then each routing protocol were run against each of these scenario files. The average of the performance values(node with empty battery, data packets delivered and number of control packets) were computed for comparison of the protocols.

7.4.1 Mobility and traffic model

The mobility model in our simulation uses the random way point model in a rectangular working area of 1500 × 300 meters. The pre-generated movement scenario files used for each simulation are characterised by a pause time. Each node begins

the simulation by remaining stationary over the pause time. It then selects a random destination in the 1500m ×300m space and moves to that destination at a velocity distributed uniformly between 0 and a maximum velocity. We ran our simulations with 5 different movement patterns generated for pause time of 100s and with maximum speeds of $10ms^{-1}$ (high mobility) and $1ms^{-1}$ (low mobility, approximately equivalent to walking speed).

Constant bit rate (CBR) traffic sources were chosen, as the aim is to compare the performance of each routing protocol. The traffic used was 512 bytes sent from 10 sources with a rate of 4 packets per second.

7.4.2 Energy model

We have implemented a specific energy expenditure model in NS 2.27 based on the model and values predicted by Feeney and Nilsson [110]. The energy consumed by the network interface when a node sends, receives and discards a packet can be modelled using the linear equation $E = m * p + n$, where p is the packet size in bytes and m and n are constants. The experimental results confirmed the accuracy of the linear model and were used to determine values for the coefficient m and n for different mode of operation such as send, receive and discard. In our simulation model, 50 nodes were assigned full energy level at the beginning of the simulation experiment.

7.5 Simulation Results

We have compared the performance, using the number of nodes with empty battery, number of packet delivered to destination and routing overhead of proposed SEAR protocol with the de facto ad hoc routing protocol AODV. We have set HELLO packets broadcasting interval to $5s$ in SEAR. Figures 7.1 and 7.2 show the average number of nodes alive as a function of simulation time. It can be seen in the figures that AODV drains the battery of a large number of nodes faster than SEAR. This is because, AODV selects routes without considering the availability of the energy

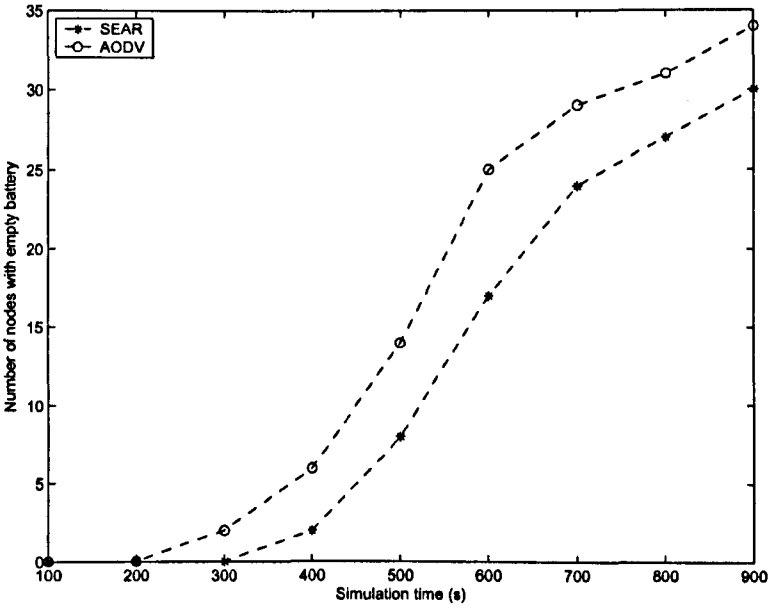

Figure 7.1: Average number of nodes with empty battery for 50 total nodes with maximum velocity of $1ms^{-1}$

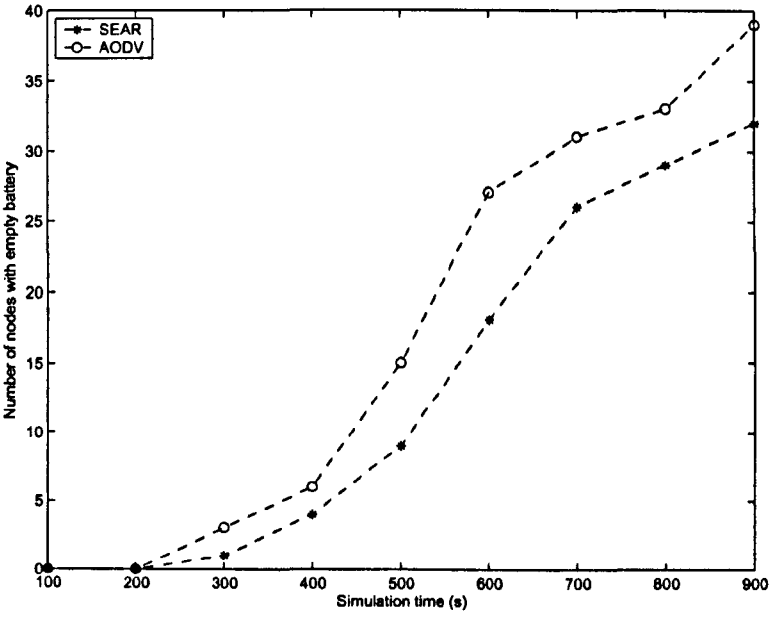

Figure 7.2: Average number of nodes with empty battery for 50 total nodes with maximum velocity of $10ms^{-1}$

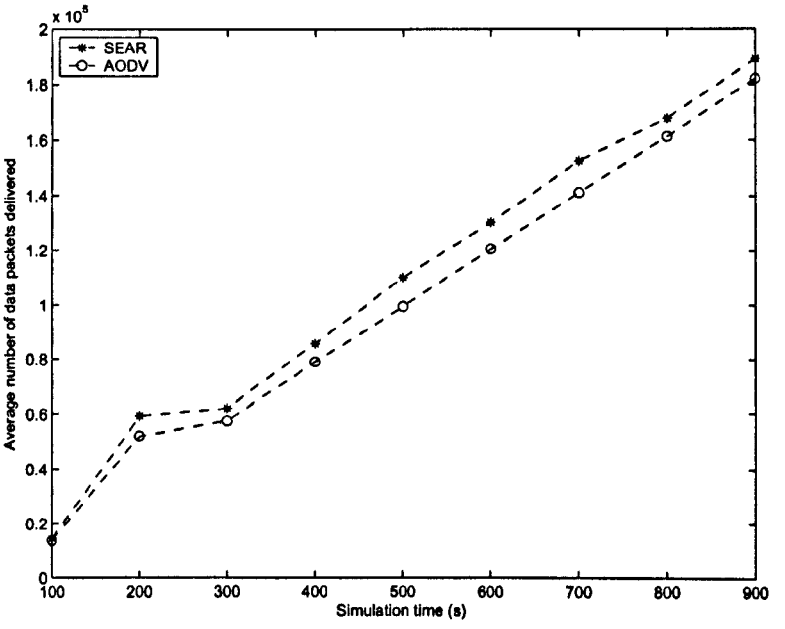

Figure 7.3: Average number of data packets delivered to destination for 50 total nodes with maximum velocity of $1ms^{-1}$

level information of the nodes while SEAR considers the remaining energy and its drain rate. This will result in a low number of nodes with battery power completely depleted. When the node mobility increases to maximum velocity of $10ms^{-1}$, the number of nodes with empty battery is higher than when the maximum velocity is $1ms^{-1}$, however SEAR still performs better than AODV.

Figures 7.3 and 7.4 show the average number of data packets delivered to the destination with maximum velocity of $1ms^{-1}$ and $10ms^{-1}$ respectively. Comparing the two protocols, SEAR always delivers more data packets than AODV. In high mobility scenario ($10ms^{-1}$), the difference between SEAR and AODV is higher than in the low mobility scenario. The energy prediction and neighbour update present in the SEAR gives better performance than AODV.

The SEAR makes lower use of overhead packets than AODV in all cases as demonstrated in figures 7.5 and 7.6. It is clear that the use of energy prediction and multipath for data packets reduces the number of control packets.

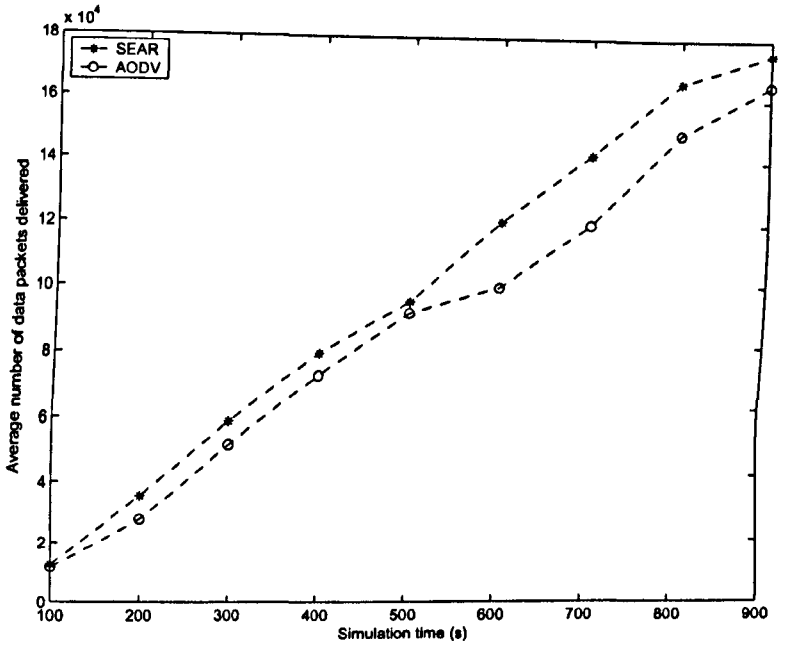

Figure 7.4: Average number of data packets delivered to destination for 50 total nodes with maximum velocity of $10ms^{-1}$

7.6 Conclusions

In chapter, we proposed a Swarm intelligence based Energy Aware Routing (SEAR) algorithm for mobile ad hoc networks. SEAR chooses the next for packets to be forwarded on the basis of the pheromone table which is influenced by the energy level and drain rate. The pheromone table is mimicked by the foraging behaviour of the natural biological ants. The effectiveness of the proposed approach is demonstrated through an extensive simulation study. Simulation results show that SEAR gives better network life time than AODV. Interestingly, SEAR always uses a lower number of overhead packets to perform routing in all cases than AODV. It is clear that the use of energy prediction and multi-path for data packets could reduce the number of control packets. AODV normally requires that each node periodically transmits a HELLO message, with a default rate of once per second. SwAN on the other hand requires only 5s intervals because of the energy prediction and it could be increased for the low mobility scenarios while maintaining similar performance levels. SEAR

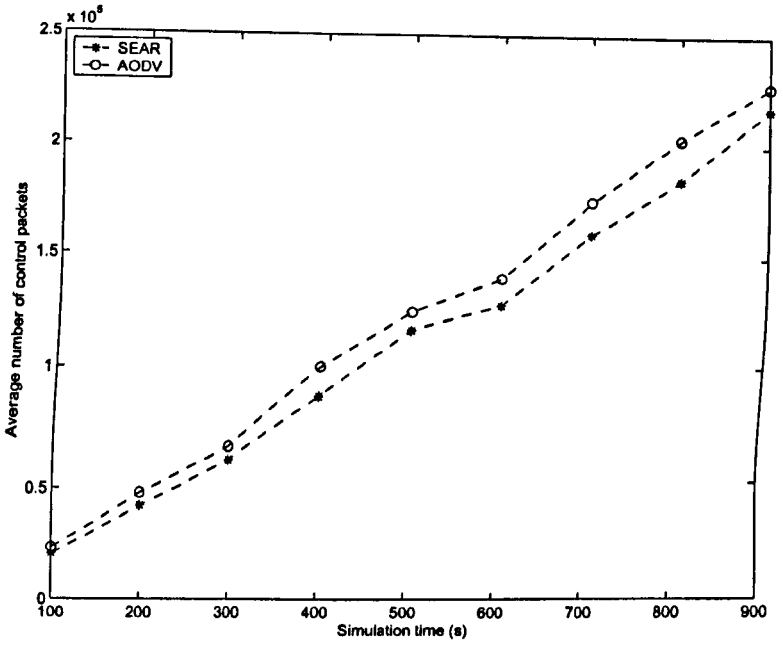

Figure 7.5: Average number of control packets used for 50 total nodes with maximum velocity of $1ms^{-1}$

delivers more data packets than AODV because It has multi paths and reliable routes for each data packets.

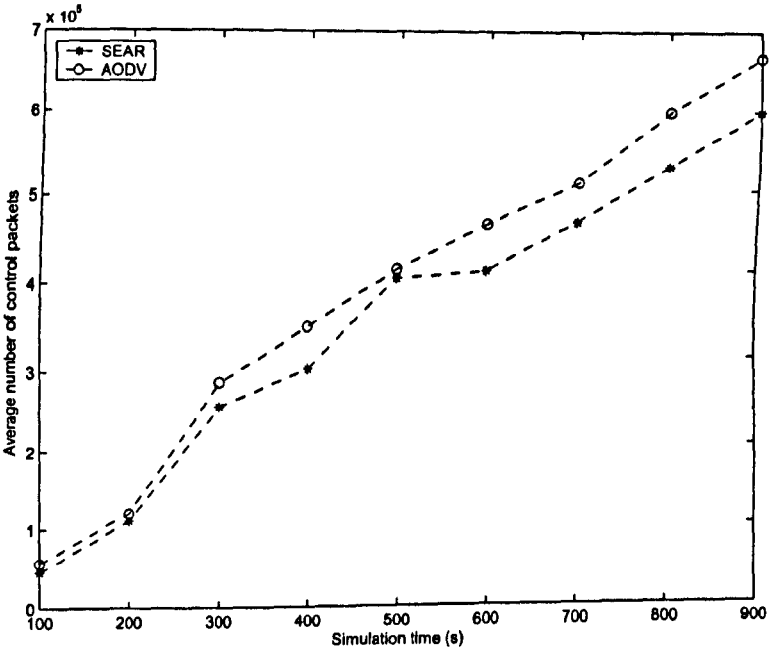

Figure 7.6: Average number of data packets delivered to the destination for 50 total nodes with maximum velocity of $10ms^{-1}$

Chapter 8

Conclusions and Future Work

8.1 Conclusions

Modern communication networks management and control problems become more complex and require solving in a distributed manner. The distributed problem solving techniques in swarm intelligence have numerous properties to solve communication network optimisation and control problems. This thesis investigated swarm intelligence applications to wireless ad hoc and sensor networks.

This thesis has provided a different approach to the stability analysis of PSO with stochastic parameters. The passivity theorem [86] and Lyapunov stability [89] methods were applied to the particle dynamics in determining sufficient conditions for asymptotic stability and hence convergence to the equilibrium point. Since the results are based on the Lyapunov function approach which are conservative and hence violation of these conditions do not imply instability. Nevertheless, the results can be used to infer qualitative design guidelines. Illustrative examples were provided to demonstrate the application of the technique. The analysis has only addressed the issue of absolute stability. The primary aim of PSO, however, is optimisation while maintaining stability. For instance, adaptation rules on K and/or w design parameters is such that exploration is facilitated while maintaining stability is required.

Investigation into the sink node placement problem in wireless sensor networks was also addressed. The novel idea of this research was the placement of sink node

in a given wireless sensor networks region, rather than the placement of sensor nodes for a given sink node position. A nonlinear programming problem was formulated to determine the location of the sink node inside the given sensor network region. The simulation results have shown that the three proposed optimal strategies are of significant benefit over random placement scenarios where energy is a vital design parameter in wireless sensor networks. This research also utilised the particle swarm optimiser, which is effective in solving NP-hard nonlinear optimisation problems and improves upon genetic algorithms.

Next, a swarm intelligence based routing protocol for mobile ad hoc networks (SwAN) was developed. Mapping the pheromone laying and following behaviour of biological ants, SwAN allows nodes to choose the next node for packets to be forwarded on the basis of the mobility influenced pheromone table. The effectiveness of the proposed approach was demonstrated through an extensive simulation study. Simulation results have shown that SwAN gives better end to end delay than AODV. Interestingly, SwAN always uses a lower number of overhead packets to perform routing in all cases than AODV. On the other hand, packet delivery ratio in SwAN is slightly lower than AODV in the low mobility case, but in the high mobility case SwANs' performance is significantly closer to AODV. SwAN always looks for an alternative path to send the packet to destination rather than re-initiates the route discovery phase as in AODV. This may cause some loss in data packets.

Finally, a swarm intelligence based energy aware routing (SEAR) algorithm for mobile ad hoc networks was proposed. The pheromone laying and following behaviour of biological ants was mimicked in the protocol design. SEAR allows nodes to choose the next node for packets to be forwarded on the basis of the pheromone table which is influenced by the energy level and drain rate. The effectiveness of the proposed approach was demonstrated through an extensive simulation study. Simulation results show that SEAR gives better network lifetime than AODV. Notably, SEAR always uses a lower number of overhead packets to perform routing in all cases than AODV.

It is clear form this thesis that swarm intelligence methods have a very useful role to play in the optimisation problem associated with wireless ad hoc and sensor networks. This thesis has given a number of example applications and has demonstrated

its usefulness in improving performance over pseudo standards.

8.2 Future Work

The work reported in this thesis raises a number of questions that need to be addressed in future work. There are issues related to the analysis of swarm intelligence method as well as for performance improvement in wireless networks.

The advent of particle swarm optimisation based on swarm intelligence techniques is a new resource for optimisation problem solving, which provides an efficient approach for complex real world problems. PSO has been successfully applied in various real world problems. An analysis for the PSO best particle dynamics was presented based on the Lyapunav and passivity theorem which does, however, prescribe highly conservative design requirements. An approach to mitigate this is the investigation of the condition for decreasing the Lyapunov energy function over a time interval rather than at every time instant, which is likely to lead to a less conservative condition for stability. The best particle dynamics in the PSO have been considered and it would be desirable for a complete PSO systems analysis, which could result in design guidelines for a PSO algorithm that is linked to performance improvements.

Sensor networks applications have the potential of significantly impacting the lives of people and their work environment. Three strategies were developed to place the sink node in an energy efficient way which results in saving significant amounts of energy, thus improving the lifetime of the entire network. The future direction of the research is in finding the optimal position for the sink node for large scale sensor networks based on the residual energy of the sensor nodes and their positional information. Therefore, the point of issue would become an online process and would require a faster real-time solution. This is linked to the need to develop an online PSO that is capable of adaptive sink node placement.

The proposed SwAN protocol outperforms the de-facto mobile ad hoc networking protocol AODV in overhead control packets and end-to-end delay and lacks in packet delivery ratio. AODV was proposed in [5] and was later adapted for several modifications and improvements. The simulation studies were undertaken with the latest

version of the AODV. It is advised that a more rigorous investigation is required, thus enabling further enhancements to the SwAN protocol, such as on the improvement of packet delivery ratio. For example, mechanisms that limit the use of control packets while maintaining other performance measures require development. The scalability of the SwAN protocol also necessitates further investigation.

The proposed SEAR protocol significantly improves the lifetime of the network while performing better than AODV. Hence, the potential of SEAR method is high because the energy is the vital design issue in the mobile ad hoc networks. The scalability of the SEAR protocol should be investigated thoroughly. SEAR opens new avenues for the QoS based routing and mechanisms need to be identified as how QoS parameters are incorporated into the pheromone table.

Appendix A

Ns2 Simulation Scripts

This section gives the NS2 simulation script which is used for performance analysis in chapter 6 and chapter 7.

```
set val(chan) Channel/WirelessChannel
set val(prop) Propagation/TwoRayGround
set val(netif) Phy/WirelessPhy
set val(mac) Mac/802_11
#set val(ifq) CMUPriQueue ;#for dsr
set val(ifq) Queue/DropTail/PriQueue
set val(ll) LL
set val(ant) Antenna/OmniAntenna
set val(x) 1500 ;# X dimension of the topography
set val(y) 300 ;# Y dimension of the topography
set val(ifqlen) 50 ;# max packet in ifq
set val(seed) 1.0
set val(ragent) SWARM
#routing protocol set val(nn) 50 ;# how many nodes are simulated
set val(cp) "cbr-50-10"
#traffic pattern set val(sc) "sear10"
# mobility pattern set val(stop) 200 ;# simulation time
```

```
set val(energymodel) EnergyModel
set val(initialenergy) 0.5 ;# Initial energy in Joules
# unity gain, omni-directional antennas
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0
# Initialisee the tranceiver parameters
Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.2818
#Phy/WirelessPhy set Pt_ 0.1000
#Phy/WirelessPhy set Pt_ 7.214e-3
#Phy/WirelessPhy set Pt_ 8.5872e-4
Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0

# Main Program
# Initialize Global Variables
set ns_ [new Simulator] ;# create simulator instance
set topo [new Topography]; # setup topography object
set val(nn) 50 ;# number of numbers
set tracefd [open swarm1.tr w] ;# create trace object for ns and nam
set namtrace [open swarm1.nam w]
$ns_ use-newtrace; # use new-trace format
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
$topo load_flatgrid $val(x) $val(y); # define topology
```

```
set god_ [create-god $val(nn)]; # Create God
#global node setting ; # define how node should be created
$ns_ node-config -adhocRouting $val(ragent)
-llType $val(ll)
-macType $val(mac)
-ifqType $val(ifq)
-ifqLen $val(ifqlen)
-antType $val(ant)
-propType $val(prop)
-phyType $val(netif)
-channelType $val(chan)
-topoInstance $topo
-agentTrace ON
-routerTrace ON
-macTrace ON
-movementTrace ON
-energyModel $val(energymodel)
-initialEnergy $val(initialenergy)

# Create the specified number of nodes [$val(nn)] and "attach" them to the channel.

#[[lindex [$node_(0) array get netif_] 1] set initialEnergy_ 0.05]
for {set i 0} $i ¡ $val(nn)  incr i {
set node_($i) [$ns_ node]
$node_($i) random-motion 0 ;#disable random motion
}

puts "Loading connection pattern..."
source $val(cp)

# Define traffic model
```

```
puts "Loading scenario file..."
source $val(sc)

# Define node initial position in nam
for {set i 0} {$i ¡ $val(nn)} {incr i}
# 50 defines the node size in nam, must adjust it according to yourscenario
# The function must be called after mobility modelis defined
$ns_ initial_node_pos $node_($i) 20
}

# Tell nodes when the simulation ends
for {set i 0} {$i ¡ $val(nn) } {incr i}
$ns_ at $val(stop).0 "$node_($i) reset";
}
$ns_ at $val(stop).0001 "stop"
$ns_ at $val(stop).0002 "puts N̈S EXITING...;̈
$ns_ halt"
proc stop {} {
global ns_ tracefd namtrace
$ns_ flush-trace
close $tracefd
close $namtrace
}

puts $tracefd "M 0.0 nn $val(nn) x $val(x) y $val(y) rp $val(ragent)"
puts $tracefd "M 0.0 sc $val(sc) cp $val(cp) seed $val(seed)"
puts $tracefd "M 0.0 prop $val(prop) ant $val(ant)"

puts "Starting Simulation..."
$ns_ run
```

Appendix B

Sample Scenario File

This section gives extract of node movement pattern for the NS2 simulator. We create a node-movement scenario consisting of 50 nodes moving with maximum speed of 20.0m/s with 300s pause time and the topology boundary is defined as 1500 X 300.

$node_(0) set X_ 799.657336047088
$node_(0) set Y_ 135.455153527360
$node_(0) set Z_ 0.000000000000
$node_(1) set X_ 390.085389287621
$node_(1) set Y_ 194.243182754745
$node_(1) set Z_ 0.000000000000
$node_(2) set X_ 1428.986456880622
$node_(2) set Y_ 165.457012368147
$node_(2) set Z_ 0.000000000000
$node_(3) set X_ 1259.848559985054
$node_(3) set Y_ 19.517756593342
$node_(3) set Z_ 0.000000000000
$node_(4) set X_ 71.236765784502
$node_(4) set Y_ 2.714858615000
$node_(4) set Z_ 0.000000000000
$node_(5) set X_ 543.918916870638

$node_(5) set Y_ 46.356298365190
$node_(5) set Z_ 0.000000000000
$node_(6) set X_ 889.221664159118
$node_(6) set Y_ 249.445444109070
$node_(6) set Z_ 0.000000000000
$node_(7) set X_ 317.271808277009
$node_(7) set Y_ 262.459155333113
$node_(7) set Z_ 0.000000000000
node_(8) set X_ 881.277909573588
$node_(8) set Y_ 236.994077031214
$node_(8) set Z_ 0.000000000000
$node_(9) set X_ 533.835494550360
$node_(9) set Y_ 171.531909163724
$node_(9) set Z_ 0.000000000000
$node_(10) set X_ 430.434439105532
$node_(10) set Y_ 220.956168570296
$node_(10) set Z_ 0.000000000000
$node_(11) set X_ 681.693016503041
$node_(11) set Y_ 297.436216815080
$node_(11) set Z_ 0.000000000000
$node_(12) set X_ 1463.412884015097
$node_(12) set Y_ 123.201256554076
$node_(12) set Z_ 0.000000000000
$node_(13) set X_ 1371.776557371375
$node_(13) set Y_ 93.848444339930
$node_(13) set Z_ 0.000000000000
$node_(14) set X_ 894.093436415176
$node_(14) set Y_ 298.113139487694
$node_(14) set Z_ 0.000000000000
$node_(15) set X_ 1005.744923184380
$node_(15) set Y_ 7.800207082751

$node_(15) set Z_ 0.000000000000
$node_(16) set X_ 1364.456800233343
$node_(16) set Y_ 167.252251549733
$node_(16) set Z_ 0.000000000000
$node_(17) set X_ 1089.633238884554
$node_(17) set Y_ 156.497818313433
$node_(17) set Z_ 0.000000000000
$node_(18) set X_ 1219.839811261436
$node_(18) set Y_ 128.401805764897
$node_(18) set Z_ 0.000000000000
$node_(19) set X_ 1035.049745298537
$node_(19) set Y_ 261.406315740317
$node_(19) set Z_ 0.000000000000
$node_(20) set X_ 675.243439956251
$node_(20) set Y_ 85.053898252363
$node_(20) set Z_ 0.000000000000
$node_(21) set X_ 1340.836036809230
$node_(21) set Y_ 130.988983331646
$node_(21) set Z_ 0.000000000000
$node_(22) set X_ 994.109882687883
$node_(22) set Y_ 95.679627848175
$node_(22) set Z_ 0.000000000000
$node_(23) set X_ 406.589047999708
$node_(23) set Y_ 207.558301014855
$node_(23) set Z_ 0.000000000000
$node_(24) set X_ 1461.871272784012
$node_(24) set Y_ 151.294652342165
$node_(24) set Z_ 0.000000000000
$node_(25) set X_ 689.892076537374
$node_(25) set Y_ 107.747922863711
$node_(25) set Z_ 0.000000000000

$node_(26) set X_ 614.799591451491
$node_(26) set Y_ 201.998907424457
$node_(26) set Z_ 0.000000000000
$node_(27) set X_ 1215.101271458218
$node_(27) set Y_ 135.232436849816
$node_(27) set Z_ 0.000000000000
$node_(28) set X_ 61.520854662251
$node_(28) set Y_ 41.170834136087
$node_(28) set Z_ 0.000000000000
$node_(29) set X_ 1.046435608915
$node_(29) set Y_ 25.126403715995
$node_(29) set Z_ 0.000000000000
$node_(30) set X_ 1135.784161333718
$node_(30) set Y_ 175.997474046302
$node_(30) set Z_ 0.000000000000
$node_(31) set X_ 1325.058467479460
$node_(31) set Y_ 123.590456276856
$node_(31) set Z_ 0.000000000000
$node_(32) set X_ 862.063971070877
$node_(32) set Y_ 227.628113526536
$node_(32) set Z_ 0.000000000000
$node_(33) set X_ 297.552463689566
$node_(33) set Y_ 152.464232992511
$node_(33) set Z_ 0.000000000000
$node_(34) set X_ 721.468381948169
$node_(34) set Y_ 60.155041774793
$node_(34) set Z_ 0.000000000000
$node_(35) set X_ 614.914094890033
$node_(35) set Y_ 156.806367499690
$node_(35) set Z_ 0.000000000000
$node_(36) set X_ 688.822053204055

$node_(36) set Y_ 60.670347167047
$node_(36) set Z_ 0.000000000000
$node_(37) set X_ 1474.305566902170
$node_(37) set Y_ 245.368363763350
$node_(37) set Z_ 0.000000000000
$node_(38) set X_ 126.987960518686
$node_(38) set Y_ 74.875615135331
$node_(38) set Z_ 0.000000000000
$node_(39) set X_ 291.958693118628
$node_(39) set Y_ 108.356699333106
$node_(39) set Z_ 0.000000000000
$node_(40) set X_ 614.850166111448
$node_(40) set Y_ 138.537673027216
$node_(40) set Z_ 0.000000000000
$node_(41) set X_ 260.377163686429
$node_(41) set Y_ 100.772417350827
$node_(41) set Z_ 0.000000000000
$node_(42) set X_ 732.353791787659
$node_(42) set Y_ 54.904015972287
$node_(42) set Z_ 0.000000000000
$node_(43) set X_ 903.030079167862
$node_(43) set Y_ 229.755005913098
$node_(43) set Z_ 0.000000000000
$node_(44) set X_ 1471.847532816298
$node_(44) set Y_ 172.553146325763
$node_(44) set Z_ 0.000000000000
$node_(45) set X_ 380.209193817226
$node_(45) set Y_ 244.563395778925
$node_(45) set Z_ 0.000000000000
$node_(46) set X_ 783.417730347926
$node_(46) set Y_ 29.095895763474

$node_(46) set Z_ 0.000000000000
$node_(47) set X_ 909.186948815101
$node_(47) set Y_ 95.000044479968
$node_(47) set Z_ 0.000000000000
$node_(48) set X_ 719.281252033659
$node_(48) set Y_ 68.662992069009
$node_(48) set Z_ 0.000000000000
$node_(49) set X_ 1104.597197439572
$node_(49) set Y_ 276.670265278643
$node_(49) set Z_ 0.000000000000
$ns_ at 300.000000000000 "$node_(0) setdest 1346.091719108419 65.207689363323 12.349623574112"
$ns_ at 300.000000000000 "$node_(1) setdest 1255.388488671930 74.220808813797 13.088875348327"
$ns_ at 300.000000000000 "$node_(2) setdest 565.457769812834 32.416526005286 10.156625009276"
$ns_ at 300.000000000000 "$node_(3) setdest 653.575681020446 76.046574888213 4.481282297547"
$ns_ at 300.000000000000 "$node_(4) setdest 1361.115842268825 268.092036355087 8.454328509630"
$ns_ at 300.000000000000 "$node_(5) setdest 498.263997058131 264.346803325260 18.302900010488"
$ns_ at 300.000000000000 "$node_(6) setdest 126.243104054259 55.346440992332 2.804014846504"
$ns_ at 300.000000000000 "$node_(7) setdest 192.972184423873 108.317456238445 13.876785348722"
$ns_ at 300.000000000000 "$node_(8) setdest 1328.569386001311 68.946179570818 15.286684413354"
$ns_ at 300.000000000000 "$node_(9) setdest 275.760558284399 96.433042026607 7.326701116737"
$ns_ at 300.000000000000 "$node_(10) setdest 214.092789457948 28.034900648347

17.939423283422"
$ns_ at 300.000000000000 "$node_(11) setdest 361.998223069969 279.779753972355 10.333207913047"
$ns_ at 300.000000000000 "$node_(12) setdest 587.866233726073 201.139774391678 5.395688335948"
$ns_ at 300.000000000000 "$node_(13) setdest 524.159201892352 15.415055562354 8.977936121498"
$ns_ at 300.000000000000 "$node_(14) setdest 1138.959505805647 8.460045829343 9.318060059602"
$ns_ at 300.000000000000 "$node_(15) setdest 711.209456187572 116.446417155875 18.879015386765"
$ns_ at 300.000000000000 "$node_(16) setdest 287.461093578466 150.406869124768 4.192334062421"
$ns_ at 300.000000000000 "$node_(17) setdest 1411.454219506699 223.527606319110 17.371409580403"
$ns_ at 300.000000000000 "$node_(18) setdest 761.271230607391 48.879667922615 1.584001399919"
$ns_ at 300.000000000000 "$node_(19) setdest 1004.348658352280 186.722528431166 9.999779057678"
$ns_ at 300.000000000000 "$node_(20) setdest 707.590469782897 227.326777108007 10.182360400895"

Appendix C

CBR Connection Pattern

This section gives node connection pattern for the NS2 simulator. We create a CBR connection file between 50 nodes, having maximum of 10 connections, with a seed value of 1.0 and a rate of 4.0.

```
# 1 connecting to 2 at time 2.5568388786897245
# set udp_(0) [new Agent/UDP]
$ns_ attach-agent$node_(1) $udp_(0)
set null_(0) [new Agent/Null]
$ns_ attach-agent $node_(2) $null_(0)
set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 0.25
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ connect $udp_(0) $null_(0)
$ns_ at 2.5568388786897245 "$cbr_(0) start"
#
# 4 connecting to 5 at time 56.333118917575632
#
```

```
set udp_(1) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(1)
set null_(1) [new Agent/Null]
$ns_ attach-agent $node_(5) $null_(1)
set cbr_(1) [new Application/Traffic/CBR]
$cbr_(1) set packetSize_ 512
$cbr_(1) set interval_ 0.25
$cbr_(1) set random_ 1
$cbr_(1) set maxpkts_ 10000
$cbr_(1) attach-agent $udp_(1)
$ns_ connect $udp_(1) $null_(1)
$ns_ at 56.333118917575632 "$cbr_(1) start"
#
# 4 connecting to 6 at time 146.96568928983328
#
set udp_(2) [new Agent/UDP]
$ns_ attach-agent $node_(4) $udp_(2)
set null_(2) [new Agent/Null]
$ns_ attach-agent $node_(6) $null_(2)
set cbr_(2) [new Application/Traffic/CBR]
$cbr_(2) set packetSize_ 512
$cbr_(2) set interval_ 0.25
$cbr_(2) set random_ 1
$cbr_(2) set maxpkts_ 10000
$cbr_(2) attach-agent $udp_(2)
$ns_ connect $udp_(2) $null_(2)
$ns_ at 146.96568928983328 "$cbr_(2) start"
#
# 6 connecting to 7 at time 55.634230382570173
#
set udp_(3) [new Agent/UDP]
```

```
$ns_ attach-agent$node_(6)$udp_(3)
set null_(3) [new Agent/Null]
$ns_ attach-agent $node_(7) $null_(3)
set cbr_(3) [new Application/Traffic/CBR]
$cbr_(3) set packetSize_ 512
$cbr_(3) set interval_ 0.25
$cbr_(3) set random_ 1
$cbr_(3) set maxpkts_ 10000
$cbr_(3) attach-agent $udp_(3)
$ns_ connect $udp_(3) $null_(3)
$ns_ at 55.634230382570173 "$cbr_(3) start"
#
# 7 connecting to 8 at time 29.546173154165118
#
set udp_(4) [new Agent/UDP]
$ns_ attach-agent $node_(7) $udp_(4)
set null_(4) [new Agent/Null]
$ns_ attach-agent $node_(8) $null_(4)
set cbr_(4) [new Application/Traffic/CBR]
$cbr_(4) set packetSize_ 512
$cbr_(4) set interval_ 0.25
$cbr_(4) set random_ 1
$cbr_(4) set maxpkts_ 10000
$cbr_(4) attach-agent $udp_(4)
$ns_ connect $udp_(4) $null_(4)
$ns_ at 29.546173154165118 "$cbr_(4) start"
#
# 7 connecting to 9 at time 7.7030203154790309
#
set udp_(5) [new Agent/UDP]
$ns_ attach-agent $node_(7) $udp_(5)
```

```
set null_(5) [new Agent/Null]
$ns_ attach-agent $node_(9) $null_(5)
set cbr_(5) [new Application/Traffic/CBR]
$cbr_(5) set packetSize_ 512
$cbr_(5) set interval_ 0.25
$cbr_(5) set random_ 1
$cbr_(5) set maxpkts_ 10000
$cbr_(5) attach-agent $udp_(5)
$ns_ connect $udp_(5) $null_(5)
$ns_ at 7.7030203154790309 "$cbr_(5) start"
#
# 8 connecting to 9 at time 20.48548468411224
#
set udp_(6) [new Agent/UDP]
$ns_ attach-agent $node_(8) $udp_(6)
set null_(6) [new Agent/Null]
$ns_ attach-agent $node_(9) $null_(6)
set cbr_(6) [new Application/Traffic/CBR]
$cbr_(6) set packetSize_ 512
$cbr_(6) set interval_ 0.25
$cbr_(6) set random_ 1
$cbr_(6) set maxpkts_ 10000
$cbr_(6) attach-agent $udp_(6)
$ns_ connect $udp_(6) $null_(6)
$ns_ at 20.48548468411224 "$cbr_(6) start"
#
# 9 connecting to 10 at time 76.258212521792487
#
set udp_(7) [new Agent/UDP]
$ns_ attach-agent $node_(9) $udp_(7)
set null_(7) [new Agent/Null]
```

```
$ns_ attach-agent $node_(10) $null_(7)
set cbr_(7) [new Application/Traffic/CBR]
$cbr_(7) set packetSize_ 512
$cbr_(7) set interval_ 0.25
$cbr_(7) set random_ 1
$cbr_(7) set maxpkts_ 10000
$cbr_(7) attach-agent $udp_(7)
$ns_ connect $udp_(7) $null_(7) $ns_ at 76.258212521792487 "$cbr_(7) start"
#
# 9 connecting to 11 at time 31.464945688594575
#
set udp_(8) [new Agent/UDP]
$ns_ attach-agent $node_(9) $udp_(8)
set null_(8) [new Agent/Null]
$ns_ attach-agent $node_(11) $null_(8)
set cbr_(8) [new Application/Traffic/CBR]
$cbr_(8) set packetSize_ 512
$cbr_(8) set interval_ 0.25
$cbr_(8) set random_ 1
$cbr_(8) set maxpkts_ 10000
$cbr_(8) attach-agent $udp_(8)
$ns_ connect $udp_(8) $null_(8)
$ns_ at 31.464945688594575 "$cbr_(8) start"
#
# 11 connecting to 12 at time 62.77338456491632
#
set udp_(9) [new Agent/UDP]
$ns_ attach-agent $node_(11) $udp_(9)
set null_(9) [new Agent/Null]
$ns_ attach-agent $node_(12) $null_(9)
set cbr_(9) [new Application/Traffic/CBR]
```

```
$cbr_(9) set packetSize_ 512
$cbr_(9) set interval_ 0.25
$cbr_(9) set random_ 1
$cbr_(9) set maxpkts_ 10000
$cbr_(9) attach-agent $udp_(9)
$ns_ connect $udp_(9) $null_(9)
$ns_ at 62.77338456491632 "$cbr_(9) start"
#
#Total sources/connections: 7/10
#
```

Appendix D

Ns2 Trace Files for SwAN Protocol

NS2 produces text-based output files that contain detailed simulation data when a simulation is finished. The data files is used for simulation analysis. This section gives extract of the NS2 simulation trace files.

M 0.0 nn 50 x 1500 y 300 rp SWARM
M 0.0 sc scenpause60 cp cbr-50-10 seed 1.0
M 0.0 prop Propagation/TwoRayGround ant Antenna/OmniAntenna
Request and Reply Packets
s -t 2.573647828 -Hs 31 -Hd -2 -Ni 31 -Nx 1450.99 -Ny 271.03 -Nz 0.00 -Ne 0.488091 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 100 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
s -t 2.573753774 -Hs 32 -Hd -2 -Ni 32 -Nx 454.70 -Ny 218.48 -Nz 0.00 -Ne 0.490633 -Nl RTR -Nw — -Ma 0 -Md ffffffff -Ms 2e -Mt 800 -Is 32.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 27 -P swarm -Pt 0x2 -Ph 4 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.573800165 -Hs 9 -Hd -2 -Ni 9 -Nx 766.69 -Ny 217.42 -Nz 0.00 -Ne 0.486226 -Nl MAC -Nw — -Ma 3c4 -Md 9 -Ms 0 -Mt 0
s -t 2.573810165 -Hs 9 -Hd -2 -Ni 9 -Nx 766.69 -Ny 217.42 -Nz 0.00 -Ne 0.486226 -Nl MAC -Nw — -Ma 13a -Md 2 -Ms 9 -Mt 806 -P arp -Po REPLY -Pms 9 -Ps 9 -Pmd

2 -Pd 2
r -t 2.574448161 -Hs 44 -Hd -2 -Ni 44 -Nx 1427.46 -Ny 173.92 -Nz 0.00 -Ne 0.487260 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.574448213 -Hs 41 -Hd -2 -Ni 41 -Nx 1396.52 -Ny 169.13 -Nz 0.00 -Ne 0.486641 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.574448325 -Hs 42 -Hd -2 -Ni 42 -Nx 1320.49 -Ny 198.46 -Nz 0.00 -Ne 0.485975 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.574448377 -Hs 43 -Hd -2 -Ni 43 -Nx 1429.93 -Ny 107.55 -Nz 0.00 -Ne 0.487260 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.574448380 -Hs 26 -Hd -2 -Ni 26 -Nx 1475.81 -Ny 107.27 -Nz 0.00 -Ne 0.487563 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.574448405 -Hs 20 -Hd -2 -Ni 20 -Nx 1330.45 -Ny 146.64 -Nz 0.00 -Ne 0.486187 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.574448498 -Hs 27 -Hd -2 -Ni 27 -Nx 1261.22 -Ny 204.59 -Nz 0.00 -Ne 0.485967 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST
r -t 2.574448645 -Hs 47 -Hd -2 -Ni 47 -Nx 1334.80 -Ny 55.16 -Nz 0.00 -Ne 0.486278 -Nl MAC -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM

-Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST

r -t 2.574450737 -Hs 2 -Hd -2 -Ni 2 -Nx 604.13 -Ny 162.23 -Nz 0.00 -Ne 0.488281 -Nl MAC -Nw — -Ma 13a -Md 2 -Ms 9 -Mt 806 -P arp -Po REPLY -Pms 9 -Ps 9 -Pmd 2 -Pd 2

s -t 2.574460737 -Hs 2 -Hd -2 -Ni 2 -Nx 604.13 -Ny 162.23 -Nz 0.00 -Ne 0.488281 -Nl MAC -Nw — -Ma 0 -Md 9 -Ms 0 -Mt 0

r -t 2.574473161 -Hs 44 -Hd -2 -Ni 44 -Nx 1427.46 -Ny 173.92 -Nz 0.00 -Ne 0.487260 -Nl RTR -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST

r -t 2.574473213 -Hs 41 -Hd -2 -Ni 41 -Nx 1396.52 -Ny 169.13 -Nz 0.00 -Ne 0.486641 -Nl RTR -Nw — -Ma 0 -Md ffffffff -Ms 1f -Mt 800 -Is 31.255 -Id -1.255 -It SWARM -Il 48 -If 0 -Ii 0 -Iv 26 -P swarm -Pt 0x2 -Ph 5 -Pb 1 -Pd 2 -Pds 0 -Ps 1 -Pss 22 -Pc REQUEST

Data Packets

s -t 8.909172744 -Hs 1 -Hd 9 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.283630 -Nl MAC -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 584 -If 0 -Ii 32 -Iv 30 -Pn cbr -Pi 27 -Pf 0 -Po 2

r -t 8.913845305 -Hs 9 -Hd 9 -Ni 9 -Nx 807.19 -Ny 173.15 -Nz 0.00 -Ne 0.265294 -Nl MAC -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 32 -Iv 30 -Pn cbr -Pi 27 -Pf 1 -Po 2

s -t 8.913855305 -Hs 9 -Hd -2 -Ni 9 -Nx 807.19 -Ny 173.15 -Nz 0.00 -Ne 0.265294 -Nl MAC -Nw — -Ma 0 -Md 1 -Ms 0 -Mt 0

r -t 8.913870305 -Hs 9 -Hd 9 -Ni 9 -Nx 807.19 -Ny 173.15 -Nz 0.00 -Ne 0.265093 -Nl RTR -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 32 -Iv 30 -Pn cbr -Pi 27 -Pf 1 -Po 2

f -t 8.913870305 -Hs 9 -Hd 2 -Ni 9 -Nx 807.19 -Ny 173.15 -Nz 0.00 -Ne 0.265093 -Nl RTR -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 32 -Iv 29 -Pn cbr -Pi 27 -Pf 1 -Po 2

r -t 8.914159865 -Hs 1 -Hd -2 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.280426 -Nl

MAC -Nw — -Ma 0 -Md 1 -Ms 0 -Mt 0
s -t 8.914489305 -Hs 9 -Hd -2 -Ni 9 -Nx 807.20 -Ny 173.15 -Nz 0.00 -Ne 0.265093 -Nl
MAC -Nw — -Ma 14be -Md 2 -Ms 9 -Mt 0
r -t 8.914842055 -Hs 2 -Hd -2 -Ni 2 -Nx 582.72 -Ny 156.99 -Nz 0.00 -Ne 0.307050 -Nl
MAC -Nw — -Ma 14be -Md 2 -Ms 9 -Mt 0
s -t 8.914852055 -Hs 2 -Hd -2 -Ni 2 -Nx 582.72 -Ny 156.99 -Nz 0.00 -Ne 0.307050 -Nl
MAC -Nw — -Ma 1384 -Md 9 -Ms 0 -Mt 0
r -t 8.915156805 -Hs 9 -Hd -2 -Ni 9 -Nx 807.20 -Ny 173.14 -Nz 0.00 -Ne 0.264741 -Nl
MAC -Nw — -Ma 1384 -Md 9 -Ms 0 -Mt 0
s -t 8.915166805 -Hs 9 -Hd 2 -Ni 9 -Nx 807.20 -Ny 173.14 -Nz 0.00 -Ne 0.264741 -Nl
MAC -Nw — -Ma 13a -Md 2 -Ms 9 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 584 -If 0 -Ii 32
-Iv 29 -Pn cbr -Pi 27 -Pf 1 -Po 2
r -t 8.919839555 -Hs 2 -Hd 2 -Ni 2 -Nx 582.70 -Ny 156.99 -Nz 0.00 -Ne 0.305004 -Nl
MAC -Nw — -Ma 13a -Md 2 -Ms 9 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 32
-Iv 29 -Pn cbr -Pi 27 -Pf 2 -Po 2
s -t 8.919849555 -Hs 2 -Hd -2 -Ni 2 -Nx 582.70 -Ny 156.99 -Nz 0.00 -Ne 0.305004 -Nl
MAC -Nw — -Ma 0 -Md 9 -Ms 0 -Mt 0
r -t 8.919864555 -Hs 2 -Hd 2 -Ni 2 -Nx 582.70 -Ny 156.99 -Nz 0.00 -Ne 0.304803 -Nl
AGT -Nw — -Ma 13a -Md 2 -Ms 9 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 32
-Iv 29 -Pn cbr -Pi 27 -Pf 2 -Po 2
r -t 8.920154306 -Hs 9 -Hd -2 -Ni 9 -Nx 807.23 -Ny 173.11 -Nz 0.00 -Ne 0.261537 -Nl
MAC -Nw — -Ma 0 -Md 9 -Ms 0 -Mt 0
s -t 9.059362578 -Hs 7 -Hd -2 -Ni 7 -Nx 292.11 -Ny 99.52 -Nz 0.00 -Ne 0.366999 -Nl
AGT -Nw — -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 7.2 -Id 9.0 -It cbr -Il 512 -If 0 -Ii 33 -Iv 32
-Pn cbr -Pi 5 -Pf 0 -Po 3
r -t 9.059362578 -Hs 7 -Hd -2 -Ni 7 -Nx 292.11 -Ny 99.52 -Nz 0.00 -Ne 0.366999 -Nl
RTR -Nw — -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 7.2 -Id 9.0 -It cbr -Il 512 -If 0 -Ii 33 -Iv 32
-Pn cbr -Pi 5 -Pf 0 -Po 3
s -t 9.059362578 -Hs 7 -Hd 23 -Ni 7 -Nx 292.11 -Ny 99.52 -Nz 0.00 -Ne 0.366999 -Nl
RTR -Nw — -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 30
-Pn cbr -Pi 5 -Pf 0 -Po 3

s -t 9.059517578 -Hs 7 -Hd -2 -Ni 7 -Nx 292.11 -Ny 99.52 -Nz 0.00 -Ne 0.366999 -Nl MAC -Nw — -Ma 14be -Md 17 -Ms 7 -Mt 0

r -t 9.059870293 -Hs 23 -Hd -2 -Ni 23 -Nx 448.23 -Ny 246.54 -Nz 0.00 -Ne 0.306465 -Nl MAC -Nw — -Ma 14be -Md 17 -Ms 7 -Mt 0

s -t 9.059880293 -Hs 23 -Hd -2 -Ni 23 -Nx 448.23 -Ny 246.54 -Nz 0.00 -Ne 0.306465 -Nl MAC -Nw — -Ma 1384 -Md 7 -Ms 0 -Mt 0

r -t 9.060185007 -Hs 7 -Hd -2 -Ni 7 -Nx 292.11 -Ny 99.52 -Nz 0.00 -Ne 0.366646 -Nl MAC -Nw — -Ma 1384 -Md 7 -Ms 0 -Mt 0

s -t 9.060195007 -Hs 7 -Hd 23 -Ni 7 -Nx 292.11 -Ny 99.52 -Nz 0.00 -Ne 0.366646 -Nl MAC -Nw — -Ma 13a -Md 17 -Ms 7 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 584 -If 0 -Ii 33 -Iv 30 -Pn cbr -Pi 5 -Pf 0 -Po 3

r -t 9.064867722 -Hs 23 -Hd 23 -Ni 23 -Nx 448.26 -Ny 246.54 -Nz 0.00 -Ne 0.304419 -Nl MAC -Nw — -Ma 13a -Md 17 -Ms 7 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 30 -Pn cbr -Pi 5 -Pf 1 -Po 3

s -t 9.064877722 -Hs 23 -Hd -2 -Ni 23 -Nx 448.26 -Ny 246.54 -Nz 0.00 -Ne 0.304419 -Nl MAC -Nw — -Ma 0 -Md 7 -Ms 0 -Mt 0

r -t 9.064892722 -Hs 23 -Hd 23 -Ni 23 -Nx 448.26 -Ny 246.54 -Nz 0.00 -Ne 0.304218 -Nl RTR -Nw — -Ma 13a -Md 17 -Ms 7 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 30 -Pn cbr -Pi 5 -Pf 1 -Po 3

f -t 9.064892722 -Hs 23 -Hd 46 -Ni 23 -Nx 448.26 -Ny 246.54 -Nz 0.00 -Ne 0.304218 -Nl RTR -Nw — -Ma 13a -Md 17 -Ms 7 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 29 -Pn cbr -Pi 5 -Pf 1 -Po 3

r -t 9.065182437 -Hs 7 -Hd -2 -Ni 7 -Nx 292.11 -Ny 99.52 -Nz 0.00 -Ne 0.363443 -Nl MAC -Nw — -Ma 0 -Md 7 -Ms 0 -Mt 0

s -t 9.065551722 -Hs 23 -Hd -2 -Ni 23 -Nx 448.26 -Ny 246.54 -Nz 0.00 -Ne 0.304218 -Nl MAC -Nw — -Ma 14be -Md 2e -Ms 17 -Mt 0

r -t 9.065904390 -Hs 46 -Hd -2 -Ni 46 -Nx 645.28 -Ny 210.89 -Nz 0.00 -Ne 0.297069 -Nl MAC -Nw — -Ma 14be -Md 2e -Ms 17 -Mt 0

s -t 9.065914390 -Hs 46 -Hd -2 -Ni 46 -Nx 645.28 -Ny 210.89 -Nz 0.00 -Ne 0.297069 -Nl MAC -Nw — -Ma 1384 -Md 17 -Ms 0 -Mt 0

r -t 9.066219057 -Hs 23 -Hd -2 -Ni 23 -Nx 448.27 -Ny 246.54 -Nz 0.00 -Ne 0.303866

-Nl MAC -Nw — -Ma 1384 -Md 17 -Ms 0 -Mt 0

s -t 9.066229057 -Hs 23 -Hd 46 -Ni 23 -Nx 448.27 -Ny 246.54 -Nz 0.00 -Ne 0.303866 -Nl MAC -Nw — -Ma 13a -Md 2e -Ms 17 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 584 -If 0 -Ii 33 -Iv 29 -Pn cbr -Pi 5 -Pf 1 -Po 3

r -t 9.070901724 -Hs 46 -Hd 46 -Ni 46 -Nx 645.27 -Ny 210.88 -Nz 0.00 -Ne 0.295022 -Nl MAC -Nw — -Ma 13a -Md 2e -Ms 17 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 29 -Pn cbr -Pi 5 -Pf 2 -Po 3

s -t 9.070911724 -Hs 46 -Hd -2 -Ni 46 -Nx 645.27 -Ny 210.88 -Nz 0.00 -Ne 0.295022 -Nl MAC -Nw — -Ma 0 -Md 17 -Ms 0 -Mt 0

r -t 9.070926724 -Hs 46 -Hd 46 -Ni 46 -Nx 645.27 -Ny 210.88 -Nz 0.00 -Ne 0.294822 -Nl RTR -Nw — -Ma 13a -Md 2e -Ms 17 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 29 -Pn cbr -Pi 5 -Pf 2 -Po 3

f -t 9.070926724 -Hs 46 -Hd 9 -Ni 46 -Nx 645.27 -Ny 210.88 -Nz 0.00 -Ne 0.294822 -Nl RTR -Nw — -Ma 13a -Md 2e -Ms 17 -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 28 -Pn cbr -Pi 5 -Pf 2 -Po 3

r -t 9.071216392 -Hs 23 -Hd -2 -Ni 23 -Nx 448.30 -Ny 246.54 -Nz 0.00 -Ne 0.300662 -Nl MAC -Nw — -Ma 0 -Md 17 -Ms 0 -Mt 0

s -t 9.071645724 -Hs 46 -Hd -2 -Ni 46 -Nx 645.26 -Ny 210.87 -Nz 0.00 -Ne 0.294822 -Nl MAC -Nw — -Ma 14be -Md 9 -Ms 2e -Mt 0

r -t 9.071998283 -Hs 9 -Hd -2 -Ni 9 -Nx 808.20 -Ny 172.05 -Nz 0.00 -Ne 0.256949 -Nl MAC -Nw — -Ma 14be -Md 9 -Ms 2e -Mt 0

s -t 9.072008283 -Hs 9 -Hd -2 -Ni 9 -Nx 808.20 -Ny 172.05 -Nz 0.00 -Ne 0.256949 -Nl MAC -Nw — -Ma 1384 -Md 2e -Ms 0 -Mt 0

r -t 9.072312841 -Hs 46 -Hd -2 -Ni 46 -Nx 645.26 -Ny 210.87 -Nz 0.00 -Ne 0.294469 -Nl MAC -Nw — -Ma 1384 -Md 2e -Ms 0 -Mt 0

s -t 9.072322841 -Hs 46 -Hd 9 -Ni 46 -Nx 645.26 -Ny 210.87 -Nz 0.00 -Ne 0.294469 -Nl MAC -Nw — -Ma 13a -Md 9 -Ms 2e -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 584 -If 0 -Ii 33 -Iv 28 -Pn cbr -Pi 5 -Pf 2 -Po 3

r -t 9.076995399 -Hs 9 -Hd 9 -Ni 9 -Nx 808.23 -Ny 172.01 -Nz 0.00 -Ne 0.254903 -Nl MAC -Nw — -Ma 13a -Md 9 -Ms 2e -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 28 -Pn cbr -Pi 5 -Pf 3 -Po 3

s -t 9.077005399 -Hs 9 -Hd -2 -Ni 9 -Nx 808.23 -Ny 172.01 -Nz 0.00 -Ne 0.254903 -Nl MAC -Nw — -Ma 0 -Md 2e -Ms 0 -Mt 0

r -t 9.077020399 -Hs 9 -Hd 9 -Ni 9 -Nx 808.23 -Ny 172.01 -Nz 0.00 -Ne 0.254702 -Nl AGT -Nw — -Ma 13a -Md 9 -Ms 2e -Mt 800 -Is 7.2 -Id 9.0 -It cbr -Il 532 -If 0 -Ii 33 -Iv 28 -Pn cbr -Pi 5 -Pf 3 -Po 3

r -t 9.077309958 -Hs 46 -Hd -2 -Ni 46 -Nx 645.25 -Ny 210.86 -Nz 0.00 -Ne 0.291266 -Nl MAC -Nw — -Ma 0 -Md 2e -Ms 0 -Mt 0

s -t 9.203030043 -Hs 1 -Hd -2 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.273512 -Nl AGT -Nw — -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 1.0 -Id 2.0 -It cbr -Il 512 -If 0 -Ii 34 -Iv 32 -Pn cbr -Pi 28 -Pf 0 -Po 2

r -t 9.203030043 -Hs 1 -Hd -2 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.273512 -Nl RTR -Nw — -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 1.0 -Id 2.0 -It cbr -Il 512 -If 0 -Ii 34 -Iv 32 -Pn cbr -Pi 28 -Pf 0 -Po 2

s -t 9.203030043 -Hs 1 -Hd 9 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.273512 -Nl RTR -Nw — -Ma 0 -Md 0 -Ms 0 -Mt 0 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 34 -Iv 30 -Pn cbr -Pi 28 -Pf 0 -Po 2

s -t 9.203385043 -Hs 1 -Hd -2 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.273512 -Nl MAC -Nw — -Ma 14be -Md 9 -Ms 1 -Mt 0

r -t 9.203737595 -Hs 9 -Hd -2 -Ni 9 -Nx 809.04 -Ny 171.13 -Nz 0.00 -Ne 0.254563 -Nl MAC -Nw — -Ma 14be -Md 9 -Ms 1 -Mt 0

s -t 9.203747595 -Hs 9 -Hd -2 -Ni 9 -Nx 809.04 -Ny 171.13 -Nz 0.00 -Ne 0.254563 -Nl MAC -Nw — -Ma 1384 -Md 1 -Ms 0 -Mt 0

r -t 9.204052147 -Hs 1 -Hd -2 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.273160 -Nl MAC -Nw — -Ma 1384 -Md 1 -Ms 0 -Mt 0

s -t 9.204062147 -Hs 1 -Hd 9 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.273160 -Nl MAC -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 584 -If 0 -Ii 34 -Iv 30 -Pn cbr -Pi 28 -Pf 0 -Po 2

r -t 9.208734699 -Hs 9 -Hd 9 -Ni 9 -Nx 809.07 -Ny 171.09 -Nz 0.00 -Ne 0.252517 -Nl MAC -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 34 -Iv 30 -Pn cbr -Pi 28 -Pf 1 -Po 2

s -t 9.208744699 -Hs 9 -Hd -2 -Ni 9 -Nx 809.07 -Ny 171.09 -Nz 0.00 -Ne 0.252517 -Nl

MAC -Nw — -Ma 0 -Md 1 -Ms 0 -Mt 0
r -t 9.208759699 -Hs 9 -Hd 9 -Ni 9 -Nx 809.07 -Ny 171.09 -Nz 0.00 -Ne 0.252317 -Nl
RTR -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 34
-Iv 30 -Pn cbr -Pi 28 -Pf 1 -Po 2
f -t 9.208759699 -Hs 9 -Hd 2 -Ni 9 -Nx 809.07 -Ny 171.09 -Nz 0.00 -Ne 0.252317 -Nl
RTR -Nw — -Ma 13a -Md 9 -Ms 1 -Mt 800 -Is 1.0 -Id 2.0 -It cbr -Il 532 -If 0 -Ii 34
-Iv 29 -Pn cbr -Pi 28 -Pf 1 -Po 2
r -t 9.209049250 -Hs 1 -Hd -2 -Ni 1 -Nx 885.74 -Ny 24.43 -Nz 0.00 -Ne 0.269956 -Nl
MAC -Nw — -Ma 0 -Md 1 -Ms 0 -Mt 0
s -t 9.209218699 -Hs 9 -Hd -2 -Ni 9 -Nx 809.08 -Ny 171.09 -Nz 0.00 -Ne 0.252317 -Nl
MAC -Nw — -Ma 14be -Md 2 -Ms 9 -Mt 0
r -t 9.209571458 -Hs 2 -Hd -2 -Ni 2 -Nx 581.73 -Ny 156.75 -Nz 0.00 -Ne 0.295766 -Nl
MAC -Nw — -Ma 14be -Md 2 -Ms 9 -Mt 0
s -t 9.209581458 -Hs 2 -Hd -2 -Ni 2 -Nx 581.73 -Ny 156.75 -Nz 0.00 -Ne 0.295766 -Nl
MAC -Nw — -Ma 1384 -Md 9 -Ms 0 -Mt 0

Bibliography

[1] M. Frodigh, S. Parkvall, C. Roobol, and P. Larsson, "Future generation wireless networks," *IEEE Personal Communication Magazine*, vol. 8, no. 5, pp. 10–17, 2001.

[2] I. Chlamtac, M. Conti, and N. Liu, "Mobile ad hoc networking: imperatives and challenges," *Ad Hoc Networks*, vol. 1, no. 1, pp. 13–64, 2003.

[3] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems.* Oxford University Press, 1999.

[4] J. Weatherall and A. Jones, "Ubiquitous networks and their applications," *IEEE Wireless Communications*, vol. 9, no. 1, pp. 18–29, 2002.

[5] C. Perkins and M. Royer, "Ad hoc on demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, 1999, pp. 90–100.

[6] M. R. Pearlman and Z. J. Hass, "Determining the optimal configuration of for the zone routing protocol," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1395–1414, 1999.

[7] B. Johnson and D. A. Maltz, *Dynamic source routing in ad hoc wireless networks.* Kluwer Academic Publishers, 1996, ch. 3.

[8] M. Jiang, J. Li, and Y. C. Tay, "Cluster based routing protocol (cbrp) functional specification," Internet Engineering Task Force, Tech. Rep., 1998.

[9] C. E. Perkins and P. Bhagvat, "Highly dynamic destination-sequenced distance vector routing for mobile computers," *Computer Communication Review*, pp. 234–244, 1994.

[10] I. Fakyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[11] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *Proc. 5th Aunual ACM/IEEE International Conference on Mobile Computing and Networking*, Seattle, WA, August 1999.

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

[13] S. Lindsey, C. Ragavendra, and K. M. Sivalingam, "Data gathering algorithms in sensor networks using energy metrics," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 9, pp. 924–935, 2002.

[14] V. Rodoplu and T. H. Ming, "Minimum energy mobile wireless networks," *IEEE Journal of Selected Areas in Communications*, vol. 17, no. 8, pp. 1333–1344, 1999.

[15] K. Akkaya and M. Younis, "An energy aware qos routing protocol for wireless sensor networks," in *Proc. International Conference on Distributed Computing Systems Workshops (ICDCSW'03)*, 2003.

[16] F. Ordonez and B. Krishnamachari, "Optimal information extraction in energy-limited wireless sensor networks," *IEEE Transactions on Selected Areas in Communications*, vol. 22, no. 16, pp. 1121–1129, August 2004.

[17] P. Cheng, C. N. Chuah, and X. Liu, "Energy aware node placement in wireless sensor networks," in *Proc. IEEE Globcom*, 2004.

[18] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 26, no. 1, pp. 29–41, 1996.

[19] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz, "Ant-based load balancing in telecommunications networks," *Adaptive Behaviour*, vol. 5, pp. 169–207, 1996.

[20] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. Sixth International Symposium on Micromachine and Human Science*, vol. 1, 1995, pp. 39–43.

[21] R. Kube and E. Bonabeau, "Cooperative transport by ants and robots," *Robotics and Autonomous Systems*, vol. 30, pp. 85–101, 2000.

[22] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proc. IEEE Congress on Evolutionary Computation*, 1999, pp. 1945–1950.

[23] C. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," in *SIGGRAPH '87 Conference proceedings*, vol. 21, no. 4, 1987, pp. 25–34.

[24] S. Camazine, J. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-organization in biological systems.* Princeton University Press, 2001.

[25] M. Dorigo, E. Bonabeau, and G. Theraulaz, "Ant algorithms and stigmergy," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 851–871, 2000.

[26] C. A. Dosoer and M. Vidyasagar, *Feedback Systems: Input – Output Properties.* Academic Press, 1975.

[27] M. Dorigo and M. Gambardella, "Ants colony systems: A cooperative learning approach to the travelling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, April 1997.

[28] M. Dorigo, G. D. Caro, and L. M. Gambardella, "Ant algorithms for discrete optimisation," *Artificial life*, vol. 5, no. 2, pp. 137–172, 1999.

[29] M. Sim and H. Sun, "Ant colony cptimization for routing and load balancing:survey and new directions," *IEEE Transactions on Systems, Man, and Cybernetics-Part A:Systems and Humans*, vol. 33, no. 5, pp. 560–572, 2003.

[30] V. Maniezzo and A. Colorni, "The ant systems applied to the quadratic assignment problem," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 5, pp. 769–778, 1999.

[31] T. Stutzle and M. Dorigo, "A short convergence proof for a class of ant colony optimization algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 358–365, 2002.

[32] P. Arabshahi, A. K. Das, I. Kassabalidis, and A. El-Sharkawi, "Adaptive routing in wireless communication networs using swarm intelligence," in *Proc. 19th AIAA Int. Communication Satellite Systems Conference*, 2001.

[33] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels, "The self-organizing exploratory pattern of the argentine ant," *Insect Behaviour*, vol. 3, pp. 159–168, 1990.

[34] D. Come, M. Dorigo, and F. Glover, Eds., *The ant colony optimization metaheuristic.* McGraw-Hill, London, 1999.

[35] D. Subramanian and J. Chen, "Ants and reinforcement learning: A case study in routing in dynamic networks," in *Proc. IJCAI*, 1998.

[36] J. Sum, H. Shen, C. sing Leung, and G. Young, "Analysis on a mobile agent based algorithm for network routing and management," *IEEE Transactions on parallel and Distributed Systems*, vol. 14, no. 3, 2003.

[37] R. C. Eberhart and Y. Shi, "Parameter selection in particle swarm optimization," in *Proc. 7th Conference on Evolutionary Programming*, Mar 1998, pp. 591–600.

[38] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 58–73, Feb 2002.

[39] F. D. Bergh and A.P.Engelbrecht, "A new locally convergent particle swarm optimiser," in *Proc. Systems, Man and Cybernetics, IEEE International Conference*, vol. 3, 2002, pp. 6–9.

[40] I. C. Trelea, "The pariticle swarm optimization algorithm: convergence analysis and parameter selection," *Information Processing Letters*, vol. 85, pp. 317–325, Mar 2003.

[41] Y. F. Liu and K. M. Passino, "Boimimcry of social foraging of distributed optimization and control," *IEEE Control Systems Magazine*, vol. 115, pp. 603–628, Dec 2002.

[42] V. Gavi and K. M. Passino, "Stability analysis of social foraging swarms," *IEEE Transactions on Systems Man and Cybernetics*, vol. 34, no. 1, pp. 539–557, 2004.

[43] N. Davies, K. Cheverst, A. Friday, and K. Mitchell, "Future wireless applications for a networked city: services for visitors and residents," *IEEE Wireless Communications*, vol. 9, no. 1, pp. 8–16, 2002 2002.

[44] J. Zou and V. K. Bhargava, "Design issues in a cdma cellular system with heterogeneous traffic types," *IEEE Transactions on Vehicular Technology*, vol. 47, no. 3, pp. 871–884, 1998.

[45] R. Ramanathan and J. Redi, "A brief overview of ad hoc networks: challenges and directions," *IEEE Communication Magazine*, vol. 40, no. 5, pp. 20–22, 2002.

[46] K. Pahlavan, T. H. Probert, and M. E. Chase, "Trends in local wireless networks," *IEEE Communications Magazine*, vol. 33, no. 3, pp. 88–95, 1995.

[47] D. Tse and P. Viswanath, *Fundamentals of wireless communication.* Cambridge University Press, 2005.

[48] S. Kapp, "802.11:leaving the wire behind," *IEEE Internet Computing*, pp. 82–85, 2002.

[49] W. S. Jeon, D. G. Jeong, and C. Choi, "An integrated services mac protocol for local wireless communications," *IEEE Transactions on Vehicular Technology*, vol. 47, no. 1, pp. 352–364, 1998.

[50] S. S. Sairam, N. Gunasekaran, and S. R. Reddy, "Bluetooth in wireless communication," *IEEE Communications Magazine*, pp. 90–96, June 2002.

[51] C. E. Perkins, *Ad hoc Networking.* Addison Wesley, 2001.

[52] D. Cordeiro, C. Gossain, and P. Agrawal, "Multicast over wireless mobile ad hoc networks: present and future directions," *IEEE Network*, vol. 17, no. 1, pp. 52–59, 2003.

[53] D. P. Bertsekas and R. G. Gallager, "Distributed asynchronous bellman-ford algorithm," in *Data Networks.* Prentice Hall, Englewood Cliffs, 1987, ch. 5.2.4.

[54] M. Royer and C. K. Toh, "A review of current routing protocols for ad hoc networks," *IEEE Personal Communications*, vol. 6, no. 2, pp. 46–55, 1999.

[55] G. H. Forman and J. Zahorjan, "The challenges of compile computing," *IEEE Computer*, vol. 27, no. 4, pp. 38–47, 1994.

[56] J. R. Lorch and A. J. Smith, "Software strategies for portable computer energy management," *IEEE Personal Communications*, vol. 5, no. 3, pp. 60–73, 1998.

[57] ——, "Scheduling techniques for reducing processor energy use in macos," *ACM/Baltzer Wireless Networks*, vol. 3, no. 5, pp. 311–324, 1997.

[58] D. P. Helmbold, D. E. Long, and B. Sherrod, "A dynamic disk spin-down technique for mobile computing," in *Proc. Second Annual ACM international conference on Mobile Computing and Networking*, 1996, pp. 130–142.

[59] S. Singh and C. S. Raghavendra, "Power aware multi-acess protocol with signalling for ad hoc networks," in *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking*, 1998.

[60] J. C. Chen, K. M. Sivalingam, P. Agrawal, and S. Kishore, "A comparision of mac protocols for wireless local networks based on battery power consumption," in *Proc. IEEE INFORCOM*, 1998.

[61] R. Ramanathan and R. Rosales-Hain, "Topology control of multi-hop wireless networking using transmit power adjustment," in *Proceeding of IEEE INFOCOM*, Israel, 2000.

[62] K. Scott and N. Bambos, "Routing channel assignment for low power transmission in pcs," in *Proc. IEEE Int'l Conf. Universal Personal Communications*, 1996.

[63] C. K. Toh, "Maximum battery life routing to support ubiquitous mobile computering in wireless ad hoc networks," *IEEE Communication Magazine*, vol. 39, no. 6, pp. 138–147, June 2001.

[64] D. Kim, J. J. Garcia-Luna-Aceves, and K. Obraczka, "Routing mechanisms for mobile ad hoc networks based on the energy drain rate," *IEEE Transactions on Mobile computing*, vol. 2, no. 2, pp. 161–173, April-June 2003.

[65] K. Chakrabarty and S. S. Iyengar, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Transactions on Computers*, vol. 51, no. 12, pp. 1448–1453, December 2002.

[66] Y. H. Hou, Y. Shi, H. D. Sherali, and S. F. Midkiff, "On energy provisioning and relay node placement for wireless sensor networks," *IEEE Transaction on Wireless Communications*, vol. 4, no. 5, pp. 317–327, Semptember 2004.

[67] R. C. Eberhart and Y. Shi, "Comparision between genetic algorithms and particle swarm optimization," in *Proc. 7th Conference on Evolutionary Programming*, vol. 1447, 1998, pp. 611–616.

[68] R. C. Eberhart and X. Hu, "Human tremor analysis using particle swarm optimization," in *Proc. IEEE Congress on Evolutionary Computation*, Washington, DC, March 1999, pp. 1927–1930.

[69] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Naknishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *IEEE Transactions on Power Systems*, vol. 15, no. 4, pp. 1232–1239, Nov 2000.

[70] G. Ciuprina, D. Ioan, and I. Munteanu, "Use of intellignet particle swarm optimization in electromagnetics," *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 1037–1040, Mar 2002.

[71] M. P. Wachowiak, R. Smolikova, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby, "An approach to multimodal biomedical image registration utilizing particle swarm optimization," *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 3, pp. 289– 301, 2004.

[72] L. Messerschmidt and A. P. Engelbrecht, "Learning to play games using a pso-based competitive learning approach," *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 3, pp. 280– 288, 2004.

[73] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 3, pp. 240– 255, 2004.

[74] R. C. Eberhart and X. Hu, "Adaptive particle swarm optimization: detection and response to dynamic systems," in *Proc. IEEE Congress on Evolutionary Computation*, Hawaii, USA, May 2002, pp. 1666–1670.

[75] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 3, pp. 211–224, 2004.

[76] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Trans. on Evolutionary Computation*, vol. 8, no. 3, pp. 256– 279, 2004.

[77] R. C. Eberhart and X. Hu, "Multiobjective optimization using dyamic neighborhood particle swarm optimization," in *Proc. IEEE Congress on Evolutionary Computation*, Hawaii, USA, May 2002, pp. 1677–1681.

[78] K. E. Parsopouls and M. N. Vrahatis, "Particle swarm optimization method in multiobjective problems," in *Proc. ACM Symposium on Applied Computing (SAC) on Evolutionary Computation*, Madrid, Spain, May 2002.

[79] X. Hu, R. C. Eberhart, and Y. Shi, "Particle swarm with extended memory for multiobjective optimization," in *Proc. IEEE Swarm Intelligence Symposium 2003 (SIS 2003)*, Indianapolis, Indiana, 2003, pp. 193–197.

[80] P. J. Angeline, "Using selection to improve particle swarm optimization," in *Proc. IEEE Congress on Evolutionary Computation*, Anchorage, Alaska, USA, 1998.

[81] P. N. Suganthan, "Particle swarm optimizer with neighbourhood operator," in *Proc. IEEE Congress on Evolutionary Computation*, Piscataway, NJ, USA, 1999, pp. 1958–1962.

[82] J. Kennedy, "Particle behaviours," in *Proc. 7th Annual Conference on Evolutionary Programming*, 1998, pp. 581–589.

[83] E. Ozcan and C. K. Mohan, "Surfing the waves," in *Proc. IEEE Congress on Evolutionary Computation*, 1999.

[84] F. V. Bergh, "An analysis of particle swarm optimizers," Ph.D. dissertation, University of Pretoria, Dapartment of Computer Science, 2002.

[85] H. M. Emara and H. A. A. Fattah, "Continuous swarm optimization technique with stability analysis," in *Proc. American Control Conference*, Boston, Massachusetts, USA, 2004, pp. 2811–2817.

[86] M. Vidyasagar, *Nonlinear Systems Analysis.* Englewood Cliffs, New Jersey: Prentice-Hall, 1993.

[87] H. K. Khalil, *Nonlinear Systems.* Macmillan Publishing Company, 1992.

[88] P. A. Cook, *Nonlinear Dynamical Systems.* Prentice Hall International (UK) Limited, 1994.

[89] W. J. Rugh, *Linear System Theory.* Prentice-Hall Information and System Sciences Series, 1996.

[90] B. S. R. T. Stefani, C. J. Savant and G. H. Hostetter, *Design of feedback control systems.* Saunders College Publishing, 1994.

[91] C. I. Byrnes and W. Lin, "Discrete-time lossless systems,feedback equivalence and passivity," in *32nd IEEE Conference on Decision and Control*, Dec 1993, pp. 1775–1781.

[92] C. Xiao and D. J. Hill, "Generalizations and new proofs of the discrete-time positive real lemma and bounded real lemma," *IEEE Transactions on Circuits and Systems - I: Fundamental Theory and Applications*, vol. 46, no. 6, pp. 740–743, 1999.

[93] Z. M. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting sink mobility for maximizing sensor networks life time," in *Proc. 38th Hawai International Conference on Systems Sciences*, 2005.

[94] S. R. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, "Energy efficient schemes for wireless sensor networks with multiple mobile base stations," in *Proc. IEEE Globecom*, vol. 22, 2003.

[95] J. Pan, L. Cai, T. Hou, Y. Shi, and S. X. shen, "Optimal base-station locations in two-tiered wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 5, pp. 458–473, Semptember 2005.

[96] T. S. Rappaport, *Wireless Communications: Principle and Practice.* Prentice Hall, New Jersey, 1996.

[97] M. Mitchell, *An Introduction to genetic algorithms.* MIT Press Cambridge, Massachusetts, 1996.

[98] F. Herrera and M. Lozana, "Gradual distributed real-coded genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 1, pp. 43–63, 2000.

[99] T. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms.* MIT Press and McGraw-Hill, 2001.

[100] V. Kadirkamanathan, K. Selvarajah, and P. Fleming, "Stability analysis of the particle dynamics in particle swarm optimizer," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 245 – 255, June 2006.

[101] M. Gunes and O. Spaniol, "Ant routing algorithm for mobile multi-hop ad hoc networks," in *Proc. Network control and engineering for Qos, security and mobility II.* Norwell, MA, USA: Kluwer Academic Publishers, 2003, pp. 120–138.

[102] S. Baras and H. Mehta, "A probabilistic emergent routing algorithms for mobile ad hoc networks," in *Proc. WiOpt03: Modelling and Optimization in Mobile, Ad Hoc and wireless Networks*, 2003.

[103] I. Kassabalidis, M. A. El-Sharkawi, P. Arabshahi, and A. Gray, "Adaptive-sdr: adaptive swarm-based distributed routing," in *Proc.IEEE World Congress on computational Intelligence*, Hawaii, May 2002.

[104] S. Marwaha, C. K. Tham, and D. Srinivasan, "Mobile agents based routing protocol for mobile ad hoc networks," in *Proc. IEEE GLOBECOM*, 2002.

[105] A. K. Das, R. J. Marks, M. A. El-Sharkawi, P. Arabshahi, and A. Gray, "The minimum power broadcast problem in wireless networks:an ant colony system approach," in *Proc. IEEE world congress on computational Intelligence*, Hawaii, May 2002.

[106] J. Li, D. Cordes, and J. Zhang, "Power aware routing protocols in ad hoc wireless networks," *IEEE Wireless Communications*, pp. 69–81, December 2005.

[107] C. E. Jones, K. M. Sivalingam, P. Agrawal, and J. C. Chen, "A survey of energy efficient network protocols for wireless networks," *Wireless Networks*, vol. 7, pp. 343–358, 2001.

[108] K. M. Sivalingam, "Design and analysis of low-power access protocols for wireless and mobile atm networks," *Wireless networks*, vol. 6, no. 1, pp. 73–87, 2000.

[109] A. Alwan, R. Bagrodia, N. Bambos, M. Gerla, L. Kleinrock, J. Short, and J. Villasenor, "Adaptive mobile multimedia networks," *IEEE, Personal Communications*, vol. 3, no. 2, pp. 34–51, 1996.

[110] L. Feeney and M. Nilsson, "Investigating the energy consumption of wireless network interface in an ad hoc networking environment," in *Proc. IEEE INFOCOM*, 2001.