INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI[®]

Data Gathering and Communication for Wireless Sensor Networks Using Ant Colony Optimization

by

Nian Nian Ding

A Thesis submitted to the Faculty of Graduate Studies and Research in
partial fulfilment of the requirements of the degree of

Master of Applied Science

Ottawa-Carleton Institute of Electrical and Computer Engineering

Department of System and Computer Engineering

Carleton University

May, 2005

Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

0-494-08370-0

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file *Votre référence*

ISBN:

Our file *Notre référence*

ISBN:

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Acknowledgments

First of all, I would like to express my deep gratitude to my supervisor Dr. Peter Xiaoping Liu. I am deeply indebted to Dr. Liu for my intellectual formation. His encouragement, guidance and tolerance made my study in Carleton University enjoyable.

I would like to thank my friend Mrs. Tamara Bicic. She used her expertise help me correct my English in this thesis. I learn a lot from her.

I would like to thank my husband Jian and my lovely daughter Annie. Without their support, it would be impossible for me to finish this thesis. I would also like to thank my parents; they always believe in me, encourage me and support me even when they are at the other side of Pacific Ocean.

Abstract

This thesis introduces a centralized approach to data gathering and communication for wireless sensor networks. Inspired by the social behaviors of ants, we clearly divide the work for the base station and sensor nodes according to their different functions and capabilities. We use the Ant Colony Optimization (ACO) method in the base station to form a near-optimal chain. The sensor nodes in the network can form a bi-direction chain structure, which is self-adaptive to minor changes. The simulation results show that the AntChain algorithm performs much better than the LEACH and PEGASIS methods, in terms of network structure, energy-efficiency, data quality and lifetime, when the base station is near the area where sensor nodes are deployed.

Table of Content

1	Introduction.....	1
1.1	Sensor network applications and characteristics.....	2
1.2	Challenge: Reliable performance at lower cost.....	4
1.2.1	Lower redundancy architecture.....	4
1.2.2	Communication scheme providing overall reliable performance.....	5
1.3	Solution: Efficient communication scheme using ACO Optimization.....	6
1.3.1	Natural inspired optimization approach---AntChian Algorithm.....	7
1.3.2	Centralized optimization.....	7
1.3.3	Adaptive bi-direction chain algorithm.....	9
1.4	Contribution of this thesis.....	11
1.5	Thesis structure.....	13
2	Background.....	15
2.1	Wireless sensor network architecture.....	15
2.1.1	Component of sensor network system.....	16
2.1.2	Sensor network structure.....	16
2.2	Sensor network related technologies.....	20
2.2.1	Localization in wireless sensor network.....	20
2.2.2	Time synchronization for wireless sensor network.....	24
2.3	Data-gathering in sensor networks.....	26

2.3.1 Research works of sensor network data-gathering communications.....	27
2.3.2 The reactive behaviour of sensor networks.....	29
2.4 MAC protocols for wireless sensor networks.....	32
2.4.1 MAC protocols for wireless voice communication networks.....	32
2.4.2 MAC protocols for wireless sensor networks.....	33
2.4.2.1 IEEE 802.11 distributed coordination function.....	34
2.4.2.2 Energy-efficient MAC for wireless sensor network.....	35
2.4.2.3 TDMA approach MAC protocols for wireless sensor network.....	36
3 The Ant Colony Optimization.....	37
3.1 Biological inspiration of natural ants.....	38
3.2 ACO algorithm overview.....	40
3.2.1 Construction algorithms and local search algorithms.....	41
3.2.2 ACO algorithms.....	43
3.2.3 TSP problem and ACO algorithms.....	46
3.2.3.1 AS and TSP.....	47
3.2.3.2 MMAS and TSP.....	48
3.3 MMAS extending with local search.....	51
3.4 MMAS performance comparison.....	53
3.5 ACO in dynamic applications.....	54
3.6 ACO approaches for dynamic TSP.....	56

4 ACO approach in Sensor Network Data-gathering.....	59
4.1 The constraints and design objectives.....	59
4.2 Ant behaviour and centralized approach.....	61
4.3 WSN chain structure: a constrained TSP problem	63
4.3.1 Energy consumption analyse for data-gathering.....	64
4.3.2 Constrained TSP for chain based WSN.....	66
4.4 A local weaving ACO Strategy or network changing during data-gathering.....	68
4.4.1 The cutting strategy.....	69
4.4.2 ACO optimization for the sub-graph.....	71
5 The AntChain data-gathering communication protocol.....	73
5.1 Chain architecture in AntChain protocol	74
5.1.1 Uni-direction AntChain	74
5.1.2 Bi-direction AntChain.....	75
5.1.3 Chain schemes built based on query type.....	76
5.2 AntChain communication protocol.....	78
5.2.1 AntChain communication phases.....	78
5.2.2 Periodical listening and wakeup message at Pre-setup phase.....	79
5.3 Chain setup phase.....	82
5.3.1 Chain ID and sensor node routing table.....	82
5.4 Data-gathering process in AntChain.....	84
5.4.1 Data-gathering in Uni-directional AntChain.....	85

5.4.2 Data-gathering in Bi-directional AntChain.....	86
5.5 Reconfiguration phase.....	87
5.6 Sensor node operation upon receiving the messages.....	88
6. Discussion and Simulation.....	90
6.1 The parameters in the MMAS algorithm.....	90
6.2 The network and energy model.....	91
6.3 Other clarification of the Simulation and Experiments.....	94
6.3.1 Head Rotation Issue for PEGASIS.....	94
6.3.2 About Uni-AntChain Algorithm	97
6.4 Experiment and Simulation results	97
6.4.1 Chain structure comparison	98
6.4.2 Data amount and energy-efficiency comparison	103
6.4.3 Energy consumed by each sensor nodes in detail	106
6.4.4 Sensor Node Death Distribution and Unexpected Node Death.....	109
6.4.5 Data Integrity comparison	112
7 Conclusion and Discussion	115
7.1 Summary of contribution.....	115
7.2 Future work.....	117

List of Figures and Tables

Figure 2-1	A general view of sensor network without illustrating how sensor nodes connected to each other.....	16
Figure 2-2	An illustration of the multi-layer sensor network structure.....	18
Figure 2-3	An illustration of single-layer sensor network structure. Left: Some of the sensor nodes setup their communication route to reach the base station; Right: A chain structure of WSN.....	19
Figure 2-4	Three master approach localization [30]: Triangulation in the three masters	23
Figure 3-1	Ants' food path between their nest and the food source without obstacles	38
Figure 3-2	Ants explore new paths after encountering obstacle	39
Figure 3-3	Ants find the new shortest path between the food and their nest	39
Figure 4-1	Visualization of cutting strategy, (a) the Path strategy; (b) Centre strategy; after one node dead. The background is the optimal solution of a TSP instance	70
Figure 4-2	Prepare the sub-graph/sub-chain for ACO optimization (a) for the path cutting strategy; between the nodes go in and out the chain, one arbitrary edge added; (b). for the centre cutting strategy, following the chain, start from the change untill back to the start point, arbitrary edges are added between the leaving node and the conjuncted entering node	71
Figure 5-1	The structure of unidirectional AntChain and its TDMA scheduling	75
Figure 5-2	The structure of bi-directional AntChain and its TDMA scheduling	75
Figure 5-3	Sensor network architectures in different chain schemes. a. Main chain scheme; b. Partial chain scheme; C. Temporary chain scheme	76
Figure 5-4	The format of wakeup message sent from base station.....	80

Figure 5-5	The time-line of pre-setup phase: The sensor nodes operate periodic listening while waiting for the base station to notify them for wakeup time	81
Figure 5-6	Format of chain-info message for main chain.....	83
Figure 5-7	Format of chain-info message for partial chain	83
Figure 5-8	Format of chain-info message for temporary chain.....	83
Figure 5-9	Sensor node routing table.....	84
Figure 5-10	Format of query message	84
Figure 5-11	Format of data message	85
Figure 5-12	Data gathering in Uni-direction AntChain.....	85
Figure 5-13	Data gathering in Bi-direction AntChain.....	86
Figure 5-14	Format of chain reconfiguration message.....	88
Figure 5-15	Sensor node operation upon receiving messages sent from the base station.....	89
Figure 6-1	Time slot number for each sensor node in PEGASIS data-gathering round i, round i+1 and round i+2	95
Figure 6-2	Actual time slot number for each sensor node in PEGASIS simulation for data gathering round i, round i+1 and round i+2	96
Figure 6-3	The network structures produced by different algorithms. In the figure, the X,Y axis present the geography position(m). Base Station is at (0,0). (a) Network structure produced by Bi-AntChain ; (b) Network structure produced by Uni-AntChain; c)Network structure produced by PEGASIS ; (d) Network structure produced by LEACH	98
Figure 6-4	Total cost comparison for the four structures produced by uni-antchain, bi-antchain, LEACH and PEGASIS algorithms. Left: The sum of square of the length of each link in each structure; Right: The sum of the length of the all links in the each structure	101

Figure 6-5	Data Amount(unit)/Energy(J)/Node.....	106
Figure 6-6	Network lifetime (round) when dead node percentage threshold is 20% and 50% . The network is 100 node in 100*100 area, initial enery is 0.5J. The base station is at (0,0)	106
Figure 6-7	Energy utilization of each sensor node when first sensor node dies. Base station located at (0,0). (a) sensor area is 100*100m;(b) sensor area is 200*200m;(c) sensor area=300*300m	108
Figure 6-8	Sensor node death distribution: base station at (0,0), each sensor nodes has 0.5j at beginning. (a), 100*100 sensor field ; (b) 200*200 sensor field	111
Figure 6-9	Data Integrity compariion at different sanity threshold: when 20%, 50% and 80% nodes died; Sensor area=100*100; base station at (0,0)	113
Table 6-1	Parameters for MMAS during the simulation	91
Table 6-2	Parameters related to the radio model and hardware	93
Table 6-3	Data amount (in unit) received by base station when first node, 20% , 50% , 80% and 100% nodes dies. The base station is at (0,0),(50,50) and (0,-87), nodes are in 100*100 area. Initial energy for each node is 0.5J	104
Table 6-4	Data ammount (in unit) received by base station when the size of sensor area varies. The base station is at (0,0). nodes are in 100*100, 200*200, 300*300 area. Initial energy for each node is 0.5J.The result is collected when first node dies, 20% nodes die and all the nodes die	105

Chapter 1

Introduction

“Smart dust”, known by many people through Neal Stephenson's science-fiction novel ‘*The Diamond Age*’, is a cloud of tiny machines that float around, sensing, gathering data and transmitting information. In recent years, there has been more and more interest in and attention to the realistic version of those tiny machines: large-scale, low-cost sensor networks that could be used to get information from dangerous zones and remote areas for military goals; or for civilian purposes where, for example, monitoring is needed.

With continuing advances in the technologies of microelectronics, digital signal processing and wireless communication, large scale wireless sensor networks (WSNs) are becoming a rapidly developing area in both research and application [51]. Although they might have been driven by military applications at the earliest stage [70], WSNs are being investigated and applied to many different civilian applications. For example, sensor networks have been studied for vehicle tracking [68, 69], habitat monitoring [58], forest surveillance [65], earthquake observation [80, 83], biomedical or health-care applications [7] and building condition monitoring [81, 82].

A WSN usually consists of a large number of sensor nodes. These tiny sensor nodes have functions for sensing, communicating, computing and power supplying. Although the technology used in a particular sensor network might be very much application oriented, these networks share some unique attributes: limited power supply, limited

transmission distance, few or no point-to-point communications and data-centric transmission.

1.1 Sensor network applications and characteristics

Sensor networks are very much application oriented and the requirements may vary quite widely from one application to another. The nodes can be mobile when used in habitat monitoring. In the environment monitoring or in healthcare applications, however, the location of sensor nodes may be mostly fixed. Sensor networks may need to gather data in a real-time manner when used in the military application. In contrast, when used in environment monitoring applications such as temperature or moisture sensing, real-time is not an issue at all. Sensor networks can be used indoors, in a remote harsh area, or even inside the human body. They may use the wireless radio or laser as the transmission medium as in the smart dust project [66]. Sensor networks could be deployed with a fixed infrastructure in a human body or it could be thrown in mass in a battlefield. The diversity and unique features of sensor network drive a lot of research interest in this new field.

Nevertheless, sensor networks have some common features and challenges that researchers are trying to face and address. A large number of sensor nodes are the fundamentals of sensor networks. Unlike traditional sensors, the tiny sensor node itself consists of elements for sensing, computing/processing and communicating. The power supply is mostly by battery. The ability of computation makes sensor node an intelligent role in the distributed sensing system.

Recent advances in the integrated circuits have made a promising picture for large-scale sensor networks. The cost of each sensor is approximately \$10 in [57]. Sensor

researchers are trying to develop sensor nodes that cost just \$1 each. As sensor nodes get cheaper, smaller and more powerful, large-scale sensor network applications become more realistic.

Unlike other networks such as cellular systems, or wireless LAN, or normal ad hoc networks, sensor networks have unique features. No single existing technique can satisfy all the requirements of these sensor network applications. For example:

- Since sensor networks are often deployed in remote areas or in harsh environments, the node as a single entity can easily die out or fail, This often changes the network topology and thus requires self-adaptive communication protocols and a robust architecture
- Sensor nodes are used to accept query from base stations and gather/send information back. This makes them much different from ad hoc networks which mainly use point-to-point communications.
- With a limited power supply, and often no recharge for the nodes, sensor networks thus may have a limited lifetime. Therefore the techniques for power efficiency and overall longer lifetime for the sensor networks are the most important goals.
- Limited computation capability. Complicated computation not only wastes energy but is also limited by the capability of the processor and memory.
- Limited transmission range. As a small sensor node with a lot of power constraint, a sensor node often cannot transmit data to a distant location in a single hop, which makes most sensor networks to adopt a multi-hop communication scheme.

- The number of nodes are large and the node density are often very high; contrast to its limited power capability, the communication overhead is one of the critical issues for a power-efficient sensor network.

These unique features raise many challenges for the researchers in this area. One important challenge is the constraint of energy supply. For wireless sensor networks, communication consumes a significant amount of energy. Consequently, much attention has been given to energy-efficient communications schemes for the system.

1.2 Challenges: more reliable performance and lower cost

As more and more sensor networks are used for or potentially aimed at civilian applications, there are some critical challenges: long lifetime, reliable performance and low cost.

For example, in the medical application where an array of sensor nodes is employed within the human body, the required lifetime for WSNs could be as long as ten years. In the application of building monitoring, sensor nodes might be installed within the building structure and need to work reliably for many years. In the meantime, the cost of the system must be low enough to make the WSNs practical for civilian applications. These mean less redundancy, and a more optimized and reliable system requiring minimal maintenance.

1.2.1 Lower redundancy architecture

In some military applications, such as in a battlefield, sensor nodes may be placed in either a remote area or a harsh environment. A high rate of node death is expected and thus it is reasonable that sensor nodes are densely deployed with high redundancy in

those applications. The high redundancy can easily cause more collisions, complicated communication, more traffic load, and more data fusion task and management issues. Therefore, it decreases the efficiency of the whole sensor network system. Given these issues and the cost of the sensor nodes themselves, high redundancy increases the cost of sensor network significantly. Low redundancy thus becomes one important requirement for civilian applications.

1.2.2 Communication scheme providing overall reliable performance

It is understandable that low redundancy can be achieved only when sensor nodes are highly reliable. The reliability of sensor nodes can be explained in many different aspects. The most important one is the lifetime of each sensor node. In the civilian application where the environment is more stable and consistent, for the same type of sensor node, the lifetime is expected to be within a certain similar range. Although unexpected node death is not avoidable in sensor network applications, from the system point of view, we can decrease the “unexpected death” by providing better technologies and more optimum algorithms. For example, in the communication scheme, one thing we can do is to prevent some sensor nodes use up all the energy much more quickly than others just because their position or their specific role in the particular communication algorithm.

Low unexpected node death rate is important because it not only reduces the cost of WSNs, but also it is critical for the performance/reliability of the whole network. Even in the case that sensor nodes are recharged or reinstalled, lower unexpected node death rate still means lower maintenance costs, because maintenance is performed less frequently.

1.3 Solution: Communication scheme using ACO method

In this thesis, we developed a novel data gathering and communication algorithm called the AntChain algorithm for wireless sensor networks. It has been shown that it provides energy-efficient and low-cost communication compared to some other existing protocols.

1.3.1 Nature inspired optimization approach—AntChain algorithm

Ant Colony optimization (ACO) [18, 17] is inspired by natural ants. It has been successfully applied in many different applications such as scheduling [16] and vehicle routing [15]. Tiny insects, such as ants, are neither physically strong nor highly intelligent, but they are able to successfully make their colony a highly organized society. By effectively communicating with each other through a medium, they can perform very complicated tasks, such as finding the shortest path from the colony to a food source, or building a structured colony. The similarity between the ant colony and a network of energy-constrained sensor nodes inspired us to look at this swarm intelligence.

One thing we have learned from ant colonies is that ants clearly divide the work into different roles according to each ant's different abilities. In this thesis, we employ a similar rule to the data-gathering for sensor networks. The WSN base station, which usually has unlimited energy recourse, stronger communication and computation ability, will perform all the complicated computation, optimization and set-up over-head. In our approach, unlike many other WSN routing algorithms where sensor nodes have to communicate with each other in order to set up a transmission route, sensor nodes in AntChain WSN only receive useful information from the base station's broadcast. In

addition, sensor nodes only need to report their sensing data when being asked (by the base station). The non-involvement of routing operation can preserve a significant amount of the routing set-up energy for sensor nodes.

1.3.2 Centralized optimization

A large scale WSN often has hundreds or thousands of sensor nodes. To optimize its communication scheme to a high quality level can be a time and energy consuming task. A centralized approach allows the base station, which has unlimited energy resources and powerful computation capability, to perform complicated jobs such as optimization, analysing, coordinating and large-scale computation.

In many civilian applications, the location of sensor nodes is often priori known or can be got by the base station after being deployed. This information is critical for centralized optimization. Other research and simulations also show that the centralized approach can improve performance significantly. LEACH and LEACH-C [61] is one of the examples. LEACH is a clustering communication protocol for WSN, in which the sensor nodes promote themselves as cluster heads by random chance; the clustering results are often unbalanced. In LEACH-C, the clusters are formed by the base station using stimulated annealing optimization. The simulation result shows that LEACH-C, which is a centralized version of LEACH, performs 40% better than LEACH in terms of energy efficiency.

Using a centralized approach, AntChain algorithm can take the full advantage of the priori known global information. The base station employs the ACO optimization method to compute a chain based structure, called “AntChain”, as routing information. By efficiently using ACO, AntChain can produce a very high quality (near-

optimal/optimal) chain for which the cost is computed based on the energy consumed during the data-gathering process. The base station can then provide the optimized routing information to the sensor nodes. The simulation results show that the AntChain algorithm can provide a much better solution with higher level energy-efficiency and data quality. It also outperforms LEACH and LEACH-C significantly.

With the known global knowledge and the real time information sent by the sensor nodes, the base station can dynamically responds to the changes occurring in the network such as node death. According to different applications, the base station using ACO optimization can also respond to the end user's request or can be designed to respond to the events that happening in the target area. For example it can request information from part of the network regarding a specific query. In this case, what the base station needs to do is sending out the routing (chain) information which just includes this part of sensor nodes. Using the centralized approach, these operations are very convenient and can achieve high energy-efficiency by the optimized routing information. On the other hand, some application operations/queries which respond to or are produced by the event analysing results can be also conveniently controlled by the applications at the base station side. For example, data gathering can be set more frequently if a particular area is suspected or is having higher event rate. High performance and flexibility is thus both achievable in AntChain algorithm.

The optimization algorithm used in AntChain is one from the ACO algorithm family: Max-Min Ant System (MMAS) [25]. MMAS has been successfully used in many different applications and provides world-class level performance in many problems. Base station performs MMAS extending with local search which also decreases the run-time of the optimization. Our simulation results and experiments in

ACO society (for larger instance) show, when the communication between the base station and sensor nodes is not hard real-time, the MMAS optimization performed by the base station is a very practical and promising approach to achieve very high quality network structure within an acceptable run time. The high quality structure builds a solid foundation for other high-performance output such as energy-efficiency.

1.3.3 Adaptive bi-direction chain structure

Chain structure is proven to be an efficient way for data-gathering process. For example in PEGASIS [55]; the chain structure is more efficient than the cluster structure of LEACH and LEACH-C [61]. But one of the disadvantages for PEGASIS is that it provides little/no self-adaptive ability, which is critical for the ever changing sensor network. In order to provide a self-adaptive, robust and highly reliable network, we introduce the bi-direction AntChain structure.

In the bi-direction AntChain, the chain head is the one nearest to the base station, and the chain tail is the second closest one. The chain head and tail take turns sending data to the base station. During the data-gathering process, starting from the chain head/tail, each node within the chain receives data from its left/right neighbour (except the first node) and performs simple data aggregation using the data it receives and the data of its own. Next the sensor node sends its final data to its right/left neighbour. The process repeats and finally reaches the tail/head. The head or tail then sends the data to the base station. In case of a node dying, the node's neighbours can detect it because they have not received data from the dead node and the chain is broken into two for the next round. The neighbours activate themselves as new head/tail and send data directly to base

station. The base station thus detects the topology change in the network and can decide whether or not to perform another reconfigure process for the whole network.

In the bi-direction chain structure, each node just transmits to/receives from its both neighbours in the chain. The optimized number of the total energy utilized by all the sensor nodes in the process is produced by performing the MMAS algorithm through selecting and optimizing the chain orders. The optimization can be seen as a variant of the typical Travel Salesman Problem (TSP), in which by given a finite number of cities along with the cost of travel between each pair of them, one try to find the cheapest way of visiting all the cities and returning to the starting point [87]. In a round of data-gathering, each node in the sensor network has to send a package of data to another node or to the base station once and only once (while the last step is the base station). The objective is to use as little energy as possible. With the global knowledge, we can analyse the cost of communication between senders and receivers, then apply a proven MMAS method for TSP problem into the WSN setting.

The bi-direction chain also makes the network more robust. It avoids large amounts of data loss in case of the death of one single or a small number of sensor nodes, which often happens in the clustering algorithm. It also provides self-adaptive ability which most other chain based algorithms cannot provide. More importantly, it uses much less energy for the overhead (for example, computing the next round chain/cluster head). Contrast to other algorithms, in the bi-direction AntChain algorithm, the chain head and tail can save more energy by not receiving data from other nodes in every other round (round $0, 2, 4, \dots n..$). Therefore, it has more energy for the transmission task, which means it can transmit to the further base station without visibly shorten its lifetime. This

We performed the simulation to compare our algorithms to other data-gathering and communication algorithms. The results show that AntChain algorithm can significantly improve the performance of the WSNs in terms of reliability, data quality, energy-efficiency and longer lifetime. Our algorithm can also provide flexibility in handling different requests/queries for the sensor network, which other self-adaptive data-gathering algorithms cannot do.

1.4 Contribution of this thesis

The developed algorithms and simulation results show that for a highly constrained sensor network, the developed centralized ACO optimization can achieve very high performance which is required by most civilian WSN applications such as high energy-efficiency, longer network lifetime, reliability, high data quality and robustness. The other important benefit is that by providing a fair and reliable scheme for the sensor nodes; the AntChain algorithm can also lower the cost of the application. Because the constraints are so obvious in the sensor network applications, a good optimization algorithm can significantly improve performance and service quality.

The centralized approach is the basic thinking, while research interest is mainly focused on the distribution aspect of sensor network; this thesis provides a different point of view which is also compliant with its own unique feature. The centralized approach is supported by both the resource-constrained characteristics and the communication feature of sensor networks. A sensor network system has two main components: the sensor nodes and the base station. Unlike Internet-like networks, which mostly each node is very much independent and the communication is mainly peer to peer; sensor nodes in sensor networks can be dependent on the base station for many issues and the communication goal is often to exchange data from sensor nodes and the

base station. Evidence that supports the centralized approach is related to the application purpose, most sensor networks are data-centralized with no/little demand of peer-to-peer communication, meaning that the sensor nodes do not need to communicate with each other except for routing purposes, which makes a centralized scheme possible in practice.

In a summary, the main contributions of this thesis are:

- Development of a nature inspired optimization method for sensor network applications. The similarity between a network of sensor nodes and a colony of ants provides the foundation of the potential benefit of applying the Ant Colony Optimization. The simulation results show that the AntChain algorithm can provide the resource-constrained sensor nodes simple but efficient communication scheme with lower overhead. It also provides high network performance results in terms of energy-efficiency, lowered cost, reliability and data quality.
- Applying a highly recognized ACO optimization method, AntChain algorithm can build an optimum/near optimum structure for the WSNs. The chain formed by AntChain algorithm is not only optimized the total energy cost for the network, but it also provides each individual sensor node a fair and better position. The energy-constrained sensor nodes thus can meet the high requirement especially on energy-efficiency, similar or expectable lifetime and less unexpected node death.
- Development of the bi-direction AntChain architecture and the data-gathering scheme with its robustness and flexibility. This architecture can be self-adaptive at two levels, the sensor nodes level and the network level (performed by base

station). The simulation shows it overcomes other data-gathering algorithms in terms of energy-efficiency, network life time and data quality.

- Provides the high flexibility for different data-gathering and communication operations. The base station can complete different data-gathering tasks according to different queries or requirements. It could be for the whole network or could be for some its nodes. It can be set operation at different paces when necessary.

1.5 Thesis Structure

First, we introduce the background of sensor networks in chapter 2. We describe the sensor network characteristics and data-gather algorithms. We also describe other advances in wireless sensor network communication such as routing algorithms and localization algorithms.

In chapter 3, we introduce the Ant Colony Optimization which we apply in our AntChain algorithm. While we cannot prove that it is the best optimization method which can be used in the centralized approach, we can provide the existing evidence to support that our selection is not only naturally inspired by a colony of the small but efficient-working ants; we also show that, the ACO optimization especially the MMAS algorithm also has proven achievement in many computational problems and real-world applications.

In Chapter 4, we describe how the MMAS is applied into the WSN setting. We also design new strategies that can be used to deal with the dynamic issue such as node death in the chain (as a typical dynamic TSP problem).

In chapter 5, the AntChain algorithm and communication protocols are introduced in detail. The simulation and related discussion is presented in chapter 6. In chapter 7, we make the conclusion and discuss the future work.

Chapter 2

Background

Wireless sensor networks have been applied in many different areas and are very much application oriented. The application examples are in [58, 68, 69, 70, 65, 66, 7]. The sensor nodes are mobile when used in habitat monitoring [58], but in the environment monitoring or in the healthcare application sensor nodes are mostly fixed [80, 83]. The sensor networks may need to gather data in a real-time manner when used in the military applications [68, 69] but by contrast, when used in some environment monitoring applications for example building temperature or moisture sensing, real-time is not a sensitive issue. Sensor networks can be used in the indoor environment, in a remote harsh area or even inside the human bodies. It often uses wireless radio signal as its transmission medium. It also may use laser as its medium as in the smart dust project [66]. Although sensor networks applications are very diverse, they also have common characters which attract a lot of efforts of research.

2.1 Wireless sensor network architecture

Wireless sensor networks are composed of a large number of sensor nodes. The sensor nodes are unique tiny machines which have computation and communication power but are limited by a number of factors. The architectures of a sensor network are mainly determined by constrained sensor nodes and the application goals.

2.1.1 Components of a sensor network system

A sensor network may have hundreds to hundreds of thousands of nodes within the target area. The node density can reach up to 20 nodes per m^3 [56]. The architecture is affected by many application facts and requirements such as environment and cost. It is also affected by the hardware or other constraints. Basically, sensor networks are used to detect events in the target area via sensing, processing and efficiently transmitting data that are required outside the sensor networks. The main components are illustrated in Figure 2-1. The component which collects all the data and connects the nodes to the outside could be a sink, or a base station. This role often does not have constraints such as power limitation. Thus, the challenge here is how to organize the sensor nodes and make sure every sensor have a route to the base station or related gateway whenever it is needed.

Figure2-1. A general view of sensor network without illustrate how sensor nodes connected with each other

2.1.2 Network structure

Since wireless sensor networks have a large number of nodes within a large area but with limited transceiver range, most common sensor networks are designed to be

multi-hop ad hoc networks. Unlike other ad hoc networks, nodes in sensor networks do not have to be mobile. Often they are fixed.

There are different structures of sensor networks that researchers have addressed and applied in different applications:

Multi-layer Structure: It also can be called as Hierarchical Structure [52, 53, 54, 59]. LEACH is a good example of this structure.

Shown in Figure 2-2, in a multi-layer WSN, sensor nodes are assigned to different layers. A sensor node in the higher layer collects data from multiple child sensor nodes and thus the network is often formed into a pyramid shape. Most sensor nodes are often assigned to the lowest layer to perform the basic function, which is to sense and send data. In higher layer, other than the basic function, a sensor node also acts as a collect agent and a gateway node.

Sensor nodes are often grouped into different clusters according to their location, neighbourhood or even data attributes. In each cluster, sensor nodes are assigned in different layers. The higher layer node which connects to its child nodes and collects information is called cluster head. Only a cluster head can communicate with its higher level node (if there are). In multi-layer WSN, sensor nodes in the same layer may or may not be able to communicate with each other. The base station is the highest layer of the network.

Figure 2-2 An illustration of the multi-layer sensor network structure

Single-layer structure: In a single-layer WSN, the relationship between all the sensor nodes is equivalent and straightforward. As shown in Figure 2-3, sensor nodes can logically directly connect with all the other nodes and cooperate with each other in this single layer. They perform the same functions of sensing and sending data, as well as can act as a relay node for other nodes. In a single-layer WSN, no single node acts as a 'centre nodes' to directly collect data from multiple sensor nodes.

This structure was first proposed by R. Wesson et al in [64], in which all the sensor nodes send data directly to the base station. In terms of large-scale WSNs that are composed of thousands of sensor nodes, this solution does not seem to be a feasible or practical one. Instead, more examples of this structure let sensor nodes connect to their neighbours and each sensor node can serve as router to another node with no different layers or clusters in architecture. Most reactive sensor networks can be put in this category.

In the reactive sensor networks [38, 39], the sensor node sends information when certain events occur or the sensing data passes certain thresholds. The sensor node

builds its route to the base station by contacting its available neighbours. The selected neighbour relays the data by continuing the routing process until the data is received by the base station.

In [55], S. Lindsey proposed a single-layer structure, in which all the sensor nodes are formed into a chain. Every sensor node communicates with their direct neighbours in the Chain. H.R. Qi also proposed a mobile agent algorithm which is based on chain structure [63]. In this algorithm, the mobile agents carry the required functions (algorithms) which are sent from the base station, and travel through the network to collect and aggregate data. Single-layer structure is straightforward and often has less overhead for route building and self-adaptive.

Figure 2-3. An illustration of single-layer sensor network structure. Left: Some sensor nodes setup their communication route until it reach the base station; Right: a chain structure of WSN

As mentioned before, the topology of a sensor network often changes dynamically due to the node movement or failure. As the deployment approaches of sensor nodes vary for different applications, some sensor networks have non-predictable topologies. Therefore the sensor nodes need to be self-organized and self-adaptive at all the stage of

the network lifetime. The overhead of self-organization and self-adaptation can consume a considerable amount of sensor nodes' energy. This is a big challenge for the WSN communication protocols. Localization algorithm is one of the solutions of this problem, by which the base station can get the location of every sensor nodes in the network after they have been deployed. The literature survey shows a promising picture that a knowable topology is very possible and feasible for sensor network application.

2.2 Sensor network related technologies

There are many technologies that are related to the wireless sensor networks including wireless technology, semiconductor and sensing, etc... In this thesis, we mainly introduce technologies that are related to the wireless communication aspect.

2.2.1 Localization in wireless sensor networks

A predicable topology is important for an energy constrained sensor network. It relates sensing data to the particular sensing spot. It also helps to find the optimal routing for the data transmission. One of the most distinguishing advantages of wireless sensor networks when compared to wired networks is that the former one can be deployed with much more flexibility. The topology of a wireless sensor network is often unpredictable before deployment, and it often changes even after deployment. Thus the localization for wireless sensor networks has attracted a lot of research attention.

There are some methods which can be used for the localization in sensor networks. One of them is global positioning system (GPS) which can provide the absolute global position of the sensor nodes [85, 86]. GPS uses a constellation of 24 satellites in precise

orbits approximately 11,000 miles above the earth. The satellites transmit data via high frequency radio waves back to Earth. A GPS receiver can lock onto these signals and process the data to triangulate its precise location on the globe. In simple words, the GPS receiver simply measures the travel time of the signals transmitted from the satellites then multiplies them by the speed of light to determine its distance to every satellites from which it is receiving signals. By locking onto the signals from a minimum of three different satellites, a GPS receiver can calculate a 2D (two-dimensional) position, consisting of the latitude and longitude. By locking onto a fourth satellite, the GPS can compute a 3D (three-dimensional) location on the globe, calculating the altitude as well as the latitude/longitude position.

As the CMOS technology develops, low cost and low power consumption GPS receivers are becoming available. The development makes it possible that the GPS equipment sensors are ready and suitable for sensor network application. [31, 32]

Other approaches which can be more practical for sensor networks have also been studied by many researchers recently. These algorithms are designed to look for the relative location of the sensor nodes. One of self-localization solutions uses triangulation to compute the location of the sensor nodes by measuring the received signal strength (RSS), the angle of arrival of a signal (AOA) and the time of arrival of a signal (TOA)[30, 36]. Depending on the different algorithms, different number of reference nodes (also called beacons or anchor nodes, which the location are known) can be used to achieving different level of performance in terms of accuracy or robustness. The number of reference nodes range from one or many.

In [30], the authors give two architectures for self-localizing sensor networks. One uses one master node as reference. Through four operations, the algorithm can locate

and active all the sensor nodes in the network. First it identifies each sensor by performing data transmission between the master and a sensor node. Through a long process sensor nodes identify themselves as Son, Parent, Neighbour of other sensor nodes or of the master. Through the Son identification process, sensor nodes get its identification number (ID). After the base station collects all the Parent information and uses triangulation method to calculate each sensor node position, it can construct a rough sensor location map. The algorithm then uses Neighbour information to refine the map.

The other architecture shown in Figure 2-4 uses three masters. This approach significantly simplifies the map construction process. Once the three masters detect the signal, they calculate its strength and send result to the base station. The base station collects results from three masters then performs the triangulation. In the simulation it also shows that by using this architecture, sensor nodes consume more energy for the whole constellation than the one master approach. This architecture is also faster and more precise.

Figure2-4. Three master approach localization: Triangulation in the three masters [30]

Another approach is to use distance vector exchange to find the distance from the non-beacon nodes to beacon nodes. Each node performs tri-lateration or multi-lateration to estimate its location [40].

In [37], the method uses the distances between the nodes to build a relative coordinate system in which the node positions are computed. First, sensor nodes calculate local maps of adjacent nodes. Then, the results can be compiled to compute the relative or physical location of each sensor node.

In a civilian application, to deploy a few more powerful reference nodes at a designed location is feasible and practical. Therefore, based on these researches we can see that for the civilian application, self localization is an achievable goal with relatively lower cost.

There is also another possibility for civilian application, that the position (the relative position) of the sensor nodes is possibly known before the deployment as in the building surveillance or health care applications.

2.2.2 Time synchronization for wireless sensor networks

Unlike other traditional networks, time synchronization for wireless sensor networks has many constraints but also draws a lot of attention. There are many related researches that have been put in literature [72, 73, 74, 75, 76, 77, 78, 9, 8].

Basically, there are three different methods for sensor network synchronization, they are Reference Broadcast, TINY/MINI-SYNC, and Level Synchronization.

In [72], Elson and Estrin proposed *post-facto synchronization* for wireless sensor networks. This scheme supposes that the nodes are normally unsynchronized with each other. When a stimulus is detected, the receiving nodes record the time stamp according to their own clock. Next a “third party” node sends a beacon to all the related nodes in this area as a synchronization pulse. Those nodes thus can reference with this beacon node and synchronized with each other. This approach can be categorized as Reference Broadcast.

In [9], a scalable Timing-sync Protocol was proposed based on a created hierarchical topology. This method is also called Level Synchronization. It uses two steps to synchronize a WSN; first the algorithm creates the hierarchical topology and assigns a level to each node. One node is assigned as root node and all the other nodes are eventually synchronized to this node. The root node can be equipped with a GPS receiver. It is the gateway node between the sensor nodes and outside world (or the base station). The second step starts from the root node. The algorithm uses a classic

approach of sender-receiver handshake synchronization to pair these two nodes. The node that is already synchronized can initial the synchronization process with the nodes that are under its level. Similar process is carried on from the root to all the nodes. Eventually all the sensor nodes are synchronized with the root node which is synchronized with the outside world.

Greunen and Rabaey proposed a lightweight tree-based synchronization method [78]. It is another example of Level Synchronization. Based on the single-hop pair-wise synchronization, the authors propose two approaches for multi-hop synchronization. One approach is the simple linear extension of the single-hop pair-wise synchronization, the centralized multi-hop synchronization. Another approach uses the distributed scheme, in which the leaf nodes initialize the synchronization process instead of the root node. This scheme is designed for applications in which most of the nodes do not need frequent synchronization. The simulation results also show that the centralized approach is more energy-efficient.

In [75], Sichitiu and Veerarittiphan proposed a TINY/MINI-SYNC approach for WSN synchronization. In this scheme, it is assumed that the clock drifts of the nodes are of a linear form. Thus the offset between two nodes is also linear. When performing pair-wise synchronization, TINY/MINI-SYNC nodes exchange time-stamped packets to estimate the best-fit offset line between the two nodes. As more packets are exchanged, the computation complexity required for calculating the best-fit line increases. The linear constraint is used to identify redundant packets that are discarded to lower computation complexity. Each node performs this pair-wise synchronization scheme with each of its neighbors until all the nodes get synchronized.

2.3 Data-gathering in sensor network

WSNs are often designed to be data-centric which means that the communication in WSNs is more concentrated on data or data attributes instead of the information-changing between particular sensor nodes themselves. This is not a surprise if we consider the purpose of the sensor networks. But for different applications, WSNs can have very different structures, topologies and requirements. The particular communication and routing scheme could be very much application dependant. But no matter what the application is, they have common constraints and face similar challenges. The communication schemes of sensor networks have some general requirements.

- **Keep efficiency.** Efficiency can be shown in many aspects during communication in WSNs, such as data efficiency. Not every row data needs to be sent to the outsider, therefore, data fusion distributed in each sensor can increase efficiency by sending less amount but meaningful information. This function is also supported by the fact that the computation capability of sensor node increases significantly even within a node smaller than a penny. Energy efficiency is critical to sensor networks because the power is mostly supplied by battery. This should be considered in finding the optimal communication route, schemes or protocols.
- **Be self-adaptive.** WSNs' dynamic topology requires that communication protocols have a strong self-adaptive capability. At different stages, the sensor nodes need to self-organize for network configuration or reconfiguration. The network needs to be self-adaptive when some or any of its nodes die-out or malfunction. It needs to self-recover after some part of a network is shut down or even the whole network is down.

- **Be fault-tolerant.** A single sensor node is small and can easily be moved or affected by the environment. It may send false data or no data when malfunctioned. It could not be seen as a reliable entity by itself. The communication and data gathering scheme therefore need to insure that the whole network can provide the correct overall information. Redundancy and coding technique are also essential.
- **Handle Resource constraints.** Resources are often limited with sensor networks. Some constrained resources include energy, lifetime for individual sensor node, the limited and changing power-level, the limited computation capability and limited transmission range. A communication scheme has to provide a satisfactory function and must reasonably deal with all those constraints.
- **Provide reasonable response time.** Most sensor networks are data-centric, which means that all the interests of the network is focus on the data itself, sensor nodes may be identified by the data attributes. Real-time communication may still be a key requirement for some applications such as “fire-detection in forest” applications and security surveillance applications. For some other applications such as temperature or building moisture surveillance applications, real-time may not be a critical issue. But a timely response manner is still preferred and certain requirements have to be met.

2.3.1 Research works of sensor network data-gathering and communications

In recent years, many researches have been carried on for the communication protocols for WSNs. Beside some traditional techniques such as flooding algorithm and gossip algorithm, many other methods have been proposed.

LEACH. Heinzelman proposes an energy-efficient communication protocol called LEACH (Low-Energy Adaptive Clustering Hierarchy) [61]. In LEACH, it is assumed that the sensor nodes can adjust their power level according to the distance to the receiver. Every sensor node has the ability to send data directly to the base station which far away from the sensor area. Sensor nodes have limited energy. LEACH protocol clusters nearby nodes into the same cluster. In each cluster, one node, called the cluster head, collects, aggregate and send data to the base station. In order to perform tasks of the cluster head, sensor node consume much more energy than other nodes in the cluster. In order to prolong the life of the network and avoid quick die out of certain sensor nodes (the cluster heads), the cluster head is randomly rotated in every data collection round. A sensor node n promotes itself as cluster head when randomly select it a number between 0 and 1, which is bigger than a priori decided threshold $T(n)=P/(1-P*(r \bmod 1/P))$. P is the desired percentage of the cluster heads in the network. In order to provide high level performance results, it is necessary to carefully select the parameters according to the power consumption formula and the size and density of sensor network.

One of the drawbacks is the random selection of cluster heads could not insure that the cluster heads are ideally and evenly scatter in the whole network. Thus some nodes may still need long-distance transmission.

Another drawback is that since the parameter is priori decided, when the sensor nodes gradually die out and the network changes, the existing best parameters often are not the best choice later.

SPIN (Sensor Protocol for Information via Negotiation) [62]. This approach uses meta-data which is a very short message and thus not energy-consuming. The protocol

has three different messages: ADV, REQ and DATA. The first two are meta-data message to negotiate with the sender and receiver before large amounts of sensing data are transmitted in the network. By using a short and quick handshake, sending large amounts data to unnecessary nodes can be avoid. In this way, the protocol achieves its goal of energy-efficiency.

PEGASIS [55], a chain-based protocol was proposed to improve LEACH by shortening the transmission range and lessen the number of nodes that communicate with a distant base station. Starting with the node that is the nearest to the base station, every node seeks out its nearest neighbour that has not joined the chain. Each sensor node performs the same greedy chain formation algorithm. With the assumption that every sensor nodes have the identical and current global information, the output chain order of each node is the same. After the chain is setup, each node just communicates with its direct neighbour in the chain. Every sensor node in the chain also performs the data fusing for data of themselves and of their neighbours. In each data gathering round, one node is used to transmit fused data to the base station. Every node in the chain takes turns along the chain to perform this task. Compared with LEACH in terms of energy efficiency, it performs much better in the simulations.

2.3.2 The reactive behaviour of sensor networks

In a traditional sensor system, sensors have to send all their data to the data fusion center. As more powerful computation ability is available to sensor nodes, more and more decisions and data fusion are carried out locally by individual sensor nodes instead of the data fusion center (normally the base station). Recent years, many researches have focused on how the smart sensor nodes can be organized and can

perform various data fusion and analyzing tasks and send meaningful output on their own.

Because of the computation ability of sensor nodes, they can be programmed to “react” to the environment, events and commands in accordingly. Using local data fusion, less data needs to be transmitted between sensor nodes or to the base station. Local data fusion also yields better response time to rapid changes in target area. Data fusion and decisions that are carried out by the sensor nodes are classified as reactive behavior of sensor networks. The reactive behavior of sensor nodes can be also applied to data-gathering communication.

In [38] TEEN protocol is proposed for reactive sensor networks. This paper first classified the sensor network as a proactive network where sensor nodes periodically switch on their sensors and transmitters to sense the environment and transmit the data. This type of sensor network provides a snapshot of the relevant parameters at regular intervals. Another type of sensor network is the reactive network, in which the nodes react immediately to sudden and drastic changes in the value of a sensed attribute. They are well suited for time critical applications. In the TEEN protocol, two different thresholds, the “hard threshold” and “Soft threshold”, are sent from the cluster head to the nodes when the cluster changes. When the “hard threshold” is reached, the sensor node must switch and send the data immediately. If the “soft threshold” is reached, the sensor nodes only send their data when they are in between the current cluster period and the priori period, and the difference is greater than the “soft threshold”.

In the so-called reactive networks, sensor nodes transmit their data only when they have to do so. This conserves energy by limiting transmission which is the most energy-consuming operation.

Another similar and adaptive approach is presented in [39]. The APTEEN protocol combines the proactive and reactive approaches by sending data periodically and reacting to the sudden or dramatic changes.

When considering the computation ability of sensor nodes, local reactive behavior is important and has shown many advantages. On the other hand, when considering the resource constraints and other limitation of individual sensor nodes, the centralized data fusion, analyzing and organizing still cannot be ignored when attempting to increase the overall response speed, performance and flexibility of the whole network.

During our research, we found that the overall reactive behavior of sensor networks should include the nodes level and system level. At the node level, sensor nodes react to local data or events that are sensed by the individual sensor nodes. They also respond to the particular commands or queries sent to them. At the system level, the sensor network needs to react to events or phenomena in certain areas up to the whole target zone, thus makes the system level decision. These decisions include sensor network reconfiguration, connection route reset, controlling and changing the frequency of data gathering operation and certain parameters changing. In order to provide the above flexibility, the sensor network must have a suitable architecture and supportive communication protocols.

2.4 MAC protocols of wireless sensor networks

In wireless networks, before any station or node transmits a frame, it must first gain access to the medium, which is a radio channel that stations and nodes share. The medium access control (MAC) Layer manages and maintains communications by

coordinating access to a shared radio channel and utilizing protocols that enhance communications over a wireless medium. There are many MAC protocols for wireless voice communication. In recent years, many research works have been done in the low power wireless sensor network area too. In this section we will first introduce some traditional Medium Access Control (MAC) protocols. Next we will introduce some researches of MAC protocol in the wireless sensor networks domain.

2.4.1 MAC protocols from wireless voice communication networks

- **TDMA: Time Division Multiple Access**

TDMA is digital transmission technology that allows a number of users to access a single radio frequency (RF) channel without interference by allocating unique time slots to each user within each channel. It is an available, well-proven technique used in commercial operation in many systems as well as in Ad-hoc wireless networks. It is also an economic approach for wireless equipments and systems.

TDMA is based on digitized audio signal. The “time” is divided into a number of milliseconds-long packets/channels. Each user of the channel takes turns transmitting and receiving in a round-robin fashion. TDMA allocates a single frequency channel for a short time to a user and then moves to another channel. At any given moment only one user is actually using the medium.

- **CDMA: Code Division Multiple Access**

CDMA is a form of *spread-spectrum*, a family of digital communication techniques that have been used in military applications for many years. It allows many users to occupy the same time and frequency allocations in a given band/space. It uses unique spreading codes to spread the base band data before transmission. The signal is

transmitted in a channel, which is below noise level. CDMA receivers do not eliminate analogue processing entirely, but they separate communication channels by means of a pseudo-random modulation that is applied and removed in the digital domain, not on the basis of frequency. By demodulation and decoding, the receiver extracts the wanted signal and information data, which is passed through a narrow band pass filter. Unwanted signals are not decoded and do not pass through the filter. The codes used for spreading have low cross-correlation values and are unique to every user. This is the reason why a receiver which has knowledge about the code of the intended transmitter is capable of selecting the desired signal.

- **FDMA: Frequency Division Multiple Access**

FDMA allocates a single channel to one user at a time. If the transmission path deteriorates, the controller switches the system to another channel. Although technically simple to implement, FDMA wastes much more bandwidth compared with other MAC protocols: the channel is assigned to a single conversation whether or not somebody is speaking. Moreover, it cannot handle alternate forms of data, only voice transmissions.

2.4.2 MAC protocols for wireless sensor networks

MAC for wireless sensor networks can be divided into contention-based and TDMA protocols. One example of a contention-based protocol is the standardized IEEE 802.11 distributed coordination function (DCF) [43]. Because of its simplicity and robustness when considering the hidden terminal problem, it is widely used in ad hoc wireless networks.

2.4.2.1 IEEE 802.11 distributed coordination function (DCF)

The 802.11 standard defines two forms of medium access, DCF and PCF (point coordination function). DCF is mandatory and is based on the CSMA/CA (carrier sense multiple access with collision avoidance) protocol. With DCF, 802.11 nodes contend for access and attempt to send messages when there is no other node transmitting. If another node is sending a message, nodes politely wait until the channel is free.

The MAC layer contains a network allocation vector (NAV), which is a counter resident at each node that represents the amount of time that it needs to send the previous frame. When the node tries to access the medium, the MAC layer checks the value of the NAV, which must be zero before a node can attempt to send a frame. Prior to transmitting a frame, a node calculates the amount of time necessary to send the frame based on the frame's length and data rate. The node places a value representing this time in the duration field in the header of the frame. When nodes receive the frame, they examine this duration field value and use it as the basis for setting their own related NAVs. This process reserves the medium for the sending nodes. The receiving node needs to send an acknowledgement (ACK) if it detects no errors in the received frame. If the sending station does not receive an ACK after a specified period of time, the sending node will assume that there was a collision (or RF interference) and retransmit the frame.

An important aspect of the DCF is a random back off timer that a node uses if it detects a busy medium. If the channel is in use, the node must wait a *random* period of time before attempting to access the medium again. This ensures that multiple stations wanting to send data do not transmit at the same time. The back off timer significantly

reduces the number of collisions and corresponding retransmissions, especially when the number of active users increases.

One important function of DCF is the optional Power Save Mode that a user can switch on or off the radio circuits in order to conserve battery power when there is no need of sending data. In order to continue to receive data frames, the sleeping nodes must wake up periodically (at the right time) to receive regular beacon transmissions coming from the access point. These beacons identify whether or not sleeping nodes have frames been waiting and buffered at the access point for delivery to their respective destinations. The node with awaiting frames will request them from the access point. After receiving the frames, the node can go back to sleep.

2.4.2.2 Energy-efficient MAC protocols for wireless sensor network

One of the most important requirements for the MAC protocol of sensor networks is energy-efficiency. The work in [42] shows that one of the shortcomings of the IEEE 802.11 DCF is the high energy consumption of sensor nodes in idle mode. This is mainly due to idle listening. PAMAS [44] made an improvement by trying to eliminate the overhearing among neighboring nodes.

S-MAC [41] exploits a similar method as PAMAS but with greater energy savings. It is explicitly designed for wireless sensor networks. S-MAC consists of three major components: periodic listen and sleep, collision and overhearing avoidance, and message passing. For *Periodic Listen and Sleep*, each node chooses a sleep schedule and exchanges it with its neighbors. It goes to sleep for a time, and then wakes up and listens to see if any other node wants to talk to it. For collision and overhearing avoidance, S-MAC tries to avoid overhearing by letting interfering nodes go to sleep.

During the message passing, the S-MAC fragments the long message into many smaller fragments, and transmits them in burst. Thus it can avoid the high cost of re-transmitting the long packet.

2.4.2.3 TDMA approach MAC protocols for wireless sensor network

TDMA protocols have the natural advantage of having no collision or control-packet overhead which often plagues contention-based MAC protocols. In literature, many communication protocols adapt the TDMA approach, as with LEACH, PEGASIS and Bluetooth [47].

Sohrabi and Pottie propose a TDMA protocol for wireless networks [46], in which each node schedules different time slots to communicate with its known neighbors. The protocol uses FDMA or CDMA to avoid interference between adjacent links.

Another example of the TDMA approach is DE-MAC proposed in [45]. DE-MAC introduces the concept of periodic listen and sleep. A sensor node switches off its radio and goes into sleep mode only when it is in its own time slot and does not have anything to transmit. The node has to keep the radio awake in the slots assigned to its neighbors in order to receive packets from them. It also has a leader election process to allow low energy nodes to sleep more than higher energy nodes. This balances and conserves energy among the sensor nodes and consequently extends the network's lifetime.

Chapter 3

The Ant Colony Optimization

The sensor networks are mainly composed of two different components, the base station and a network of sensor nodes. One unique feature of sensor networks is that the sensor nodes have limited energy and bandwidth, and have limited computation and transmission capabilities. In order for sensor networks to meet application objectives, all the sensor nodes need to be well organized and energy efficient. These features remind us of emerging swarm intelligence Ant Colony Algorithms [17, 18], which is inspired by the natural ants. This biological analogy to ants' behavior provides us with a unique point of view when we consider the communication protocol of WSNs.

In this chapter, we will first describe the Ant Colony Optimization (ACO) Algorithms, which we applied in our data gathering protocol. We will start with the original experiments of real ants, and then we will specify what ACO is. We will also introduce a family of ACO algorithms and show their performance comparison, which provides basic data to explain why we choose the MMAS algorithm as our optimization algorithm for the base station.

3.1 Biological inspiration of natural ants

Ant Colony Optimization was originally inspired by the behaviour of natural ants. It has been proven as one efficient approach to solve hard combinatorial computation

problems by multi agent cooperation using indirect communication through modifications in the environment. In the real world, ants release a certain amount of pheromone while walking, and each ant probabilistically prefers to follow a direction which is rich in pheromone. The famous bridge experiment [11] shows that ants always can find the shortest path between the colony and food source. In the following illustration, we explain why ants are able to adjust to changes in the environment, such as new obstacles interrupting the shortest path.

Figure 3-1. Ants' food path between their nest and the food source without obstacles

At first, as shown in figure 3-1, ants follow the shortest path to carry food back to their colony. If there is an obstacle obstructs the path the ants will randomly choose another way around the obstruction (right, left, over or under). If we assume that the ants cannot go under or over the obstruction, we can safely assume that approximately half of the ants will go right and the other half left when ants reach points A and B, as illustrated in figure 3-2.

The ants that happen to pick the shorter path will obviously create a strong trail of pheromone a lot faster than the ants choosing the longer path. This will cause more and more ants to choose the shorter path until eventually all ants have found the shortest path as shown in figure 3-3.

Figure 3-2 Ants explore new paths after encountering obstacle

Figure 3-3 Ants find the new shortest path between the food and their nest

Ant Colony Optimization attempts to apply similar techniques in order to solve much more complex problems in real life. The main idea is to repeatedly use simulations of artificial ants as agents to generate new solutions to the problem at hand. The ant agents use information collected during past simulations to direct their next step

in current run and this information is available from the modifiable medium (pheromone intensity on the trail). Many different algorithms have been implemented and many have been proven to have world class performance on the problems such as Traveling salesman problem [25], vehicle routing [15], scheduling [16] and routing in telecommunication networks [19, 20, 21].

3.2 ACO algorithm overview

There are many complex problems such as the *NP*-hard problems [88], are often difficult to solve. When solving a *NP*-hard problem, the time required grows exponentially with the instance size in the worst case. For these problems, often approximate algorithms are the only feasible way to obtain near optimal solutions at relatively low computational cost. ACO algorithms are proposed as approximate algorithms targeting hard combinatorial optimization problems. Recently ACO algorithm researchers have been more interested in applying ACO algorithms in the constrained, dynamic optimization problems.

The first ACO algorithm is Ant System (AS), proposed in 1991 [17]. It was first applied to solve the TSP problem, and gave initially an encouraging result. It is also stimulated further research on algorithmic variants and new applications. After AS, many other ACO algorithms have been proposed including Ant Colony System [23, 24], the ranked-based version of Ant System [22], Max-Min Ant System [24, 25, 26] etc. These algorithms improve performance in terms of the solution quality and run time requirements in different applications. At present, ACO algorithms have successfully applied to many different applications, such as the quadratic assignment problem [13, 14], which is a standard problem in location theory. If there is a set of n locations and n

facilities, each facility need to be assigned to a location, the aim is to find the assignment that minimizes the cost by: (1), measuring the cost of each possible assignment and, (2), multiplying the prescribed flow between each pair of facilities by the distance between their assigned locations, and then adding together all the pairs. Other applications include sequential ordering [5], vehicle routing [15], scheduling [16] or packet-switched network routing [19, 20, 21]. In these cases, ACO algorithms obtain world-class performance.

3.2.1 Construction algorithms and local search algorithms

Two main traditional approximate algorithms that are normally used to solve these hard combinatorial optimization problems are *construction* algorithms and *local search* algorithms [84]. These are two different types of methods. Construction algorithms work on partial solutions and try to extend these in the best possible way to complete problem solutions. One example of this method is known as the *nearest neighbor* tour construction heuristic, which is used to solve the TSP problem (the PEGASIS protocol uses this approach to construct its chain). Although the solutions given by this method are of a better quality than the randomly generated solutions, the main disadvantages of the construction method are:

- Only a very limited number of solutions can be generated, which are often of a low quality.
- Greedy decisions in the early stages of the construction process strongly constrain the available possibilities later on, often causing very poor moves in the final phases of the solution construction.
- The solutions are not guaranteed to be optimal in cases of small changes

The results produced by the construction methods can often be improved by local search algorithms. Local search methods move in the search space of complete solutions. Starting from a complete initial solution, local search methods try to find a better solution in an appropriately defined neighbourhood of the current solution (known as iterative improvement). If such a solution is found, it replaces the current solution and the local search continues. The steps are repeated until no improving of the solution is found in the neighbourhood of the current solution and the algorithm ends in a local optimum. An example neighbourhood for the TSP is the *k-opt* neighbourhood in which neighbour solutions differ at most arcs. Figure 3-4 shows the example of a 2-opt neighbourhood. A common disadvantage with local search algorithms is that they may easily get trapped in local minima and that the result strongly depends on the initial solutions.

Figure 3-4: 2-opt algorithm for TSP illustration. Starting from the left, after changes of 2 arcs, the cost (distance in this case) of the tour is reduced

3.2.2 ACO algorithms

Although they are inspired by natural ants, artificial ants in ACO have more capabilities and can solve more complicated problems. Artificial ants have memory to remember their path and can trace all the way back (in order to update the pheromone trail). They can compute the quality of the path and update the trail with a proportional

pheromone amount while real ants can deposit more pheromone if more food is found at the food source.

Ant Colony Optimization is proposed as an approach for solving hard combinatorial optimization problems. It can be seen as an extension of the construction method, using a special pheromone trail. In the implementation of many recently proposed ACO algorithms, local search is often added-in to improve the performance.

Starting from AS, new ACO algorithms have been proposed for problems which can not easily be solved with classical techniques, such as shortest path problems where costs change dynamically (like routing problem in Internet); *NP*-hard problems such as the constrained shortest path problem, etc. Although they retain some of the original biological inspiration, new algorithms of the ACO family are less biologically inspired and more motivated by the need to make ACO algorithms competitive or an improvement over state-of-the-art algorithms. In other words, researchers are making efforts to develop new ACO algorithms with higher level performance.

The basic framework of the ACO method remains the same. The ACO meta-heuristic described in [18] is as the pseudo code as below:

Ant Colony Optimization meta-heuristic

WHILE termination conditions not met *DO*

ScheduleActivities

AntBasedSolutionConstruction()

PheromoneUpdate()

DaemonActions() {optional}

END ScheduleActivities

ENDWHILE

The *ScheduleActivities* construction consists of three parts.

AntBasedSolutionConstruction(): An ant constructively builds a solution to a problem by moving through the nodes of the construction graph G (in which ants can construct solution by moving from component to component). Ants move by applying a stochastic local decision policy that makes use of the pheromone values and the heuristic values on components and/or connections of the construction graph. While moving, the ant remembers the partial solution it has built based on the path it was following on the construction graph.

PheromoneUpdate(): When adding a component to the current partial solution, an ant can update the values of the pheromone trails that were used for this construction step. This kind of pheromone update is called an online step-by-step pheromone update. Once an ant has built a solution, it can retrace the same path backwards by using its memory and update the pheromone trails of the used components and/or connections according to the quality of the solution it has built. This is called online delayed pheromone update. Another important concept in Ant Colony Optimization is pheromone evaporation. Pheromone evaporation is the process by which the pheromone trail intensity on the components decreases over time. From a practical point of view, pheromone evaporation is needed to avoid a too rapid convergence of the algorithm toward a sub-optimal region. It implements a useful form of forgetting, favoring the exploration of new areas in the search space.

DaemonActions(): Daemon actions can be used to implement centralized actions which cannot be performed by single ants. Examples are the use of a local search procedure applied to the solutions built by the ants, or the collection of global information that can be used to decide whether it is useful or not to deposit additional pheromone to bias the search process from a non-local perspective. As a practical

example, the daemon can observe the path found by each ant in the colony and choose to deposit extra pheromone on the components used by the ant that built the best solution. Pheromone updates performed by the daemon are called offline pheromone updates.

Informally, the construction procedure can be simply described as following:

In the search space, the artificial ants move concurrently and asynchronously while applying a stochastic local decision policy that makes use of pheromone trails and heuristic information. By continuing to move, ants incrementally build solutions to the optimization problem. Once an ant has built a solution, or while the solution is being built, the ant evaluates the solution and deposits pheromone trails on the components or paths it passed. This pheromone information will direct the search of the future ants. The moving activity is repeated until the end condition is reached. Besides the ants' activity, an ACO algorithm includes two more procedures that make ACO different to other construction method: *pheromone trail evaporation* and *daemon actions* (this is optional). By introducing the pheromone evaporation, the ACO algorithms avoid a too rapid convergence towards sub-optimal solutions and maintain a strong capability of exploring new solutions. Daemon actions are used to implement centralized actions which cannot be performed by single ants. For example, global information can be collected and the ACO algorithm can decide whether to or not to deposit additional pheromone to bias the search process from a non-local perspective.

3.2.3 TSP problem and ACO algorithms:

The Travelling Salesman Problem (TSP) is a test-bed for many combinatorial optimization algorithms. The goal is to find shortest tour for a given set of cities. Each

city can be visited exactly just once. The inter-city distances are known. The TSP problem has attracted much research efforts.

In this section we introduce the ACO algorithms from AS and describe how MMAS solves the TSP problem. The reason why we begin with AS is because it is the first ACO algorithm and it is more conceptual than newer ACO algorithms. It is also easy to understand. For the more recently proposed MMAS, which is one of the best ACO algorithms, we will specify what makes it different from and better than AS, and why it can get better performance results.

3.2.3.1 AS and TSP

When using AS to solve the TSP problem [17], a group of m ants are initially put on randomly chosen m cities within n cities in the instance. Each ant has a memory of the partial solution it has constructed so far (initially the memory contains only the start city). Beginning with its start city, an ant iteratively moves from city to city. While at city i , the ant chooses to go to the yet unvisited city j with a probability given by

$$P_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta} \quad (3.1)$$

Where $\eta_{ij} = \frac{1}{d_{ij}}$, d_{ij} is a priori known heuristic information: the distance between city i to city j (it also could be the cost of any form). $\tau_{ij}(t)$ is the pheromone strength number of the arc between city i and city j at time t . Parameters α and β determine how much the pheromone trail and heuristic information can influence the ants' behaviours. N_{ij}^k represents the feasible neighbourhood of ant k , that is, the set of cities

which ant k has not yet visited. If $\alpha=0$, the selection probabilities are inverse proportional to the cost the arc and the closest cities are more likely to be selected. In this case the AS corresponds to a classical stochastic greedy algorithm (with multiple starting points since ants are initially randomly distributed on the cities). If $\beta=0$, only the pheromone strength number is at work. This will lead to the rapid emergence of a situation where all the ants are building the same result and it could be very possibly a strong suboptimal solution.

The pheromone trails are updated after each ant completes its tour. At first, all trails are assigned a constant of pheromone evaporation. Next then each ant deposits pheromone on the arcs that belong to its tour:

$$\Delta \tau_{ij}^k(t) = \begin{cases} 1/L^k(t) & \text{if ant } k \text{ passes the arc}(i,j) \\ 0 & \text{otherwise} \end{cases} \quad (3.2)$$

$L^k(t)$ is the tour length (or a function of length/cost) of ant k . This formula means that arcs within the shorter tours are receiving more pheromone. For the next run, if there are m ants building solutions then the pheromone amount for arc (i,j) at time $t+1$ is,:

$$\tau_{ij}^k(t+1) = (1-\rho) \cdot \tau_{ij}^k(t) + \sum_{k=1}^m \Delta \tau_{ij}^k(t) \quad \forall (i,j) \quad (3.3)$$

In this formula, $0 < \rho \leq 1$ is the pheromone trail evaporation rate. This parameter is to avoid unlimited accumulation of pheromone trails and to make ants forget the previous bad decisions.

The building operation is repeated. The arcs which are used by the most ants and which are contained in shorter tours will receive more pheromone. These same arcs therefore are also more likely to be chosen in future iterations of the algorithm.

3.2.3.2 MMAS and TSP

The performance of AS is not compatible with other well-known approximation algorithm. MMAS (MAX-MIN Ant System) was proposed by Stutzle and Hoos in 1996 for achieving better performance [25]. MMAS has been proven one of the best performing ACO algorithms. MMAS improved upon the AS by allowing only the best ant updates the pheromone trail after each run. It also extends the algorithm with local search algorithm. MMAS can guide the local search process to a much bigger search space and thus increases the possibility of a global optimal solution.

One shortage of AS is that the pheromone intensity on the arcs often becomes too high and thus reduces the possibility of ants exploring other efficient search spaces. The ants tend to select arcs with high intensities of pheromone causing the algorithm to reach a situation called stagnation of search. Because there are few arcs with a high intensity of pheromone, many other promising trails are ignored by the ant this leads to premature convergence to sub-optimal solutions. This shortcoming is a fact in many other natural inspired algorithms, too. To overcome this issue MMAS provides *dynamically evolving bounds* on the pheromone trail intensities. The pheromone intensity on all paths is always within a specified limit of the path with the greatest pheromone intensity, which is between the maximum and the minimal numbers. As a result, all paths always have a non-trivial probability of being selected and thus wider exploration of the search space is encouraged.

MMAS uses upper and lower bounds to ensure pheromone intensities lie within a given range, that is $\tau_{\min}(t) \leq \tau_{ij}(t) \leq \tau_{\max}(t)$. The upper and lower bounds $\tau_{\max}(t)$, $\tau_{\min}(t)$ are given by Stutzle & Hoos:

$$\tau_{\max}(t) = \frac{1}{1-\rho} \frac{1}{f(S^{gb}(t-1))} \quad (3.4)$$

In this equation S^{gb} is the global best solution of this iteration. $f(S(t))$ is a function of the solution. In the application, it can be seen as the cost of the solution. A global best solution is found during each run. The $\tau_{\max}(t)$ is updated as well, thus leading the $\tau_{\max}(t)$ to a dynamically changing value.

In order to determine $\tau_{\min}(t)$, MMAS assumes that

- 1) The best solutions are found shortly before search stagnation occurs. Therefore the probability of re-constructing the global-best solution in one algorithm iteration is significantly higher than zero.
- 2) The solution construction is determined by the relative difference between upper and lower pheromone trail limits, rather than by the relative differences of the heuristic information.

Given these assumptions, good values for $\tau_{\min}(t)$ can be found by relating the convergence of the algorithm to the minimum trail limit. P_{best} , which is the probability of finding the best solution, is significantly higher than 0 when AS has converged. If at each choice point, an ant makes the “right” decision then it can construct the best solution. MMAS assumes that the probability P_{dec} of choosing the corresponding

solution component at a choice point is constant at all decision points. Then, an ant has to make n times of the “right” decision and hence will construct the best solution with a probability of P_{dec} . By setting

$$P_{dec}^n = P_{best} \quad \text{thus} \quad P_{dec} = \sqrt[n]{P_{best}}$$

Finally MMAS gives

$$\tau_{min}(t) = \frac{\tau_{max}(t)(1 - \sqrt[n]{P_{best}})}{(NO_{avg} - 1)\sqrt[n]{P_{best}}} \quad (3.5)$$

where P_{best} is the probability that the current global-best path, $S_{gb}(t)$, will be selected given that all non-global best edges have a pheromone level of $\tau_{min}(t)$ and all global-best edges have a pheromone level of $\tau_{max}(t)$. n is the number of decision points and NO_{avg} is the average number of edges at each decision point.

In summary, what makes MMAS different with the basic AS are:

- 1) After each run (iteration), only the best ants get to update the pheromone trail.
- 2) The intensity of the pheromone trail always has an up bound and lower bound limit, which is dynamically changed when the best-global solution is found in each run.
- 3) MMAS initially assigns high pheromone intensity to each arc. The initialization increases the exploration solutions during the first iterations of the algorithm.

3.3 MMAS extended with local search

It is known that the performance of Genetic Algorithms can often be greatly improved when coupled with local search algorithms. This is the case when MMAS combines local search with it in application of the TSP. The experiment results in [25] show that local search can improve the solutions built by MMAS algorithm. At the same time, local search can improve MMAS performance significantly by decreasing the time needed to find optimal solutions.

In [25], detailed experimental investigation has been conducted to find how MMAS should be coupled with local search. Basically, the algorithm has to decide which ant or ants can perform local search after the initial MMAS algorithm and how many ants should be used in the MMAS search. Three options have been investigated. The first option (10+all-LC) is given a constant number 10 of ants to perform MMAS, and all these ants are allowed to do 2-opt local search for symmetric TSPs (3-opt local search for asymmetric TSPs). The second option (10+best-LC), is same as the first one except only the iteration-best ant is allowed to perform local search. In the third option (MMAS+best-LC), the number of ants that perform MMAS is proportional to problem size and only the iteration-best ant is allowed to do local search. The results clearly show that given a less limited CPU-time ($k \cdot n \cdot 100$ steps) the method with the best performance is the second method (10 ants perform MMAS and only the iteration-best ant performs the local search). The instance size ranges from 100 to larger than 700. In the experiment in which the longest CPU-time is given ($k \cdot n \cdot 2500$ steps), all three options have produced the best solution for different applications/instances. When the given time is medium ($k \cdot n \cdot 500$ steps) the third option has the overall best performance.

This experiment is encouraging for the dynamic application in which a limited run time is important and an easy parameter setting is also needed. It shows that in the applications which require quick optimization and easy setting parameter, MMAS+best-Local search is potentially a good choice.

One explanation of the good performance of a combination of MMAS with local search is that these two search methods are complementary. The MMAS algorithm initially performs a rather coarsely-grained search at the beginning of each iteration. By the time the initial solution is built, a local search algorithm is performed to search in the surroundings of it.

Finding good initial solutions is critical for quality results and the run-time limit. Thus, in MMAS, the iteration-best ant generates new promising solutions based on previously found optima. The local search then is given good starting points. The experiments proved that this combination gives outstanding performance output. In one of the experiments, for the instance “lin318”, in 500 seconds (about 10 runs), MMAS without Local search has almost no chance to find a good solution while, MMAS plus local search (option 2) produces a very high quality solution and the difference to the optimal solution is only 2%.

Other greedy algorithms (for example, the nearest neighbor construction heuristic) also build the initial solutions and then followed with local search. But the experiments show that the MMAS+local-search clearly produces higher quality results. The quality difference becomes more significant in ATSP instances. Since ATSP represents a more general aspect of most real applications, this observation shows that MMAS is more suitable for complicated application and it is more robust. This robustness feature is a strong asset in dynamic setting applications.

3.4 MMAS performance comparison

When talking about the TSP problem, it is well known that the sophisticated Lin-Kernighan local search algorithm [16] often gives the best performance with respect to solution quality. But it is also known that its run time is very sensitive to particular instance and the algorithm is very complex and requires a lot of fine tuning to run quickly and produce high-quality solutions. The obvious disadvantage is that it is very difficult to apply this algorithm to the dynamic application with run-time constraints.

Except for the traditional algorithms that are used to solve the NP-hard problems, many natural-inspired-algorithms have been considered as good approaches for these problems too. Some of the well-known natural inspired algorithms are Ant Colony Optimization, Genetic Algorithm (GA) and Stimulated Annealing algorithm. In [2], ACO algorithm (MMAS is used in particular) is compared with two other GA algorithms as well as other algorithms and the experiment results show that ACO is compatible with those well-known best algorithms in the TSP problem.

Many ACO algorithms have been proposed after Ant System. Except MMAS, Ant Colony System (ACS), Best-worst Ant System, and Rank-Based version of Ant System are well known. Among those algorithms, MMAS is one of the best algorithms when considering the performance on TSP problem, [25, 26, 29]. Similar performance also can be obtained by ACS. One interesting observation in the Stutzle's experiments is that for different instances, the average results of MMAS are all better than those of ACS. On the other hand the best solutions produced by ACS are often better than those produced by MMAS. This observation gives us another indication that MMAS is good at finding high quality solutions, and the risk of producing worse (unwanted) solution is

comparatively lower than other algorithms. The stable performance of MMAS is a desired asset for real application that required high quality solutions but do not necessarily need to be the exact best solution.

3.5 ACO in dynamic applications

Dynamic applications are applications in which the data and/or the structure of the problem can change while a problem is being solved. Many industrial problems fall into this category as operational conditions constantly change. Compared with the amount of research undertaken on static optimization problems, relatively little work has been devoted to dynamic problems, despite the potential economic advantage in doing so. ACO is one of few techniques that have really been applied to dynamic problems, such as telecommunication routing problems.

ACO algorithms show a strong potential that they can be applied in highly dynamic applications where local search algorithms might not be easy to apply. One successful example is the routing algorithm in telecommunication networks [3,6, 4].

In 1996, Schoonderwoerd, Holland, Bruten and Rothkrantz applied an ACO algorithm to a routing problem. This is the so-called ant-based control (ABC) algorithm [4]. It was applied to a model of the British Telecom (BT) telephone network. The model they studied was not capable of connecting all calls at a given time so the objective was to minimize the number of lost calls. In their ant-based solution, the ants operate independently of the calls. Each ant travels between an origin and a random destination node based on a function of pheromone, distance and node congestion. Calls are routed according to the pheromone level on each neighbouring node. The link with

the most pheromone is always chosen. This approach compares favourably with the existing mobile agent methods of British Telecom.

Several ACO algorithms have been developed for routing on connectionless networks taking inspiration both from AS in general and from ABC in particular.

Di Caro and Dorigo developed AntNet [3,6], an ACO algorithm for distributed adaptive routing in Internet-like data networks. It built forwarding tables for packet routing problems (such as those that occur on the Internet). The authors have shown that their system compares favorably with existing routing algorithms.

In AntNet, ants move using the same link queues as data while building the path to the destination. In this way ants experience the same delays as data packets and the time elapsed while moving from the source node to the destination node can be used as a measure of the path quality. Ants evaluate path quality by their experience and local statistical models. In AntNet, pheromone is deposited after a complete path is built other than deposit while passing each arc. This makes AntNet becomes applicable to more general applications in which the cost of the path might be asymmetric. A complete trip for the moving ants is a round trip between the source and destination. After reaching their destination nodes, ants move backwards to their source nodes via the same route. In order to obtain a fast propagation of the collected information, high priority queues are used for “backward ants”.(in AntNet the term “forward ant ”is used for ants moving from source to destination nodes, and the term “backward ant ”for ants going back to their source nodes). During the backward path, the pheromone value of each visited link is updated.

AntNet is similar to ABC in which ants launch the tours from each node to destination and the ant path is updated with respect to all the successor nodes of the (forward) path.

In a comparison to other routing algorithms, AntNet shows a much better performance in terms of throughput, packet delays and robustness.

Similar research in telecommunication has also been performed by White, Pagurek and Oppacher [34] except that separate ant colonies are used to determine the routes, allocate traffic to the routes and deallocate routes. In addition, Varela, G. and Sinclair worked on applying ACO to routing and wavelength allocation with optical networks [33].

3.6 Recent proposed ACO strategies for dynamic TSP

In recent years, the ACO community has shown more interest in the ACO strategies or algorithms that can be used in dynamic applications, especially the dynamic travel salesman problem. In the dynamic TSP (DTSP), the distance of between the cities may change over time. The cities can be inserted or deleted. The challenge is to find how ACO optimization can cope with the changes and meet the high requirement of time and solution quality.

Although the simplest way to react to a change is to restart the ACO algorithm, if the change is small enough or relatively small, it is reasonable to assume that the new optimal solution is somehow similar or highly related to the old one. In literature, there are different ACO approaches that have been proposed for DTSP.

In 2001, Guntsch and Middendorf propose three different strategies for Ant Algorithm applied to DTSP [28, 35]. Their study investigate the pheromone modification methods that can be used in the circumstance that one city is deleted/inserted. The main point is to reserve and use the information of the original good solution to achieve the goal which can reduce the standard ant algorithm run time without sacrificing the solution quality. The pheromone information which is stored in the old pheromone matrix is the foundation of the original solution. Guntsch and Middendorf investigate the methods about how the matrix can be transferred and used according to the change. By using and transforming the old pheromone matrix, the algorithm can be benefited from the knowledge. At the same time, the algorithm also should be able to avoid keeping too much knowledge, which can make the algorithm be easily trapped into a local optimal solution without exploring better choices.

The first strategy they used is called restart strategy. In restart strategy, each city is assigned the same degree to reset the pheromone value of the edges that incident to it. In this strategy, all the pheromone information is transferred at the same degree without considering where the change happened.

The second method, η -strategy modifies the pheromone value of the edges for each city according to its distance from inserted/deleted city. The degree that is transferred is proportionate to the Euclidian distance.

The third method, τ -strategy modifies pheromone value by using the distance that is measured based on the pheromone information of all the edges that on the path from the city to the deleted/inserted city.

The experiment results show that the restart strategy and η -strategy perform best. τ -strategy also provides a very close result. The τ -strategy provides the best performance immediately after the change, and the other two strategies provide better quality solution later.

Other approaches include population based ACO [49, 50], which keeps a pool of K solutions. The solutions in the pool are updated according to the age or quality as the time goes on. For DTSP which has no change of instance size, it gives promising experiment results.

In [67], a “shaking” approach is proposed for DTSP when traffic jams frequently occurred between cities. This approach shakes the level of the pheromone value after a change occurs. “Shaking” smoothes the difference between the edges while preserving the relative order. To prevent losing too much global information during the shaking process, the study defines the “local shaking” strategy, the shaking occurs only within the area close to the change.

Chapter 4

ACO Approach in Sensor Network

Ant Colony Optimization has been successfully applied in many difficult discrete optimization problems including the travelling salesman problem, scheduling, vehicle routing problems, etc. It also has successfully applied in the telecommunication networks routing problem, which is a typical dynamic problem, where the traffic at every node is keeping changing as time varies.

In this chapter, we introduce the strategies of applying the ACO optimization in WSN data-gathering and communication. The main goals are to achieve high energy-efficiency and longer lifetime for WSN.

4.1 The constraints and design objectives

Many sensor networks in civilian applications such as in health care, building management or manufacture are mostly used to provide reliable monitoring of the target site. Often the base station is not too far away from the sensor area. The challenge for these WSNs in civilian application is how to obtain better and more predictable performance based on the lower cost. In these applications, some objectives are different with the original requirements of the WSNs in military applications. For example, civilian applications may normally have less redundancy of sensor nodes in order to lower the cost. Another point is that the civilian applications may not be expected to have their sensor nodes die/malfunctioned as frequently as they do when

deployed in a harsh environment such as a battlefield. Furthermore, when we deploy the same type of sensor nodes in an application they are often expected to have similar lifetime span.

Our objectives for the energy-efficient data-gathering and communication protocol for WSN can be summarized as:

- Energy efficiency
- Balanced workload for sensor nodes thus lessen systematically caused earlier node death rate
- Predicable and similar lifetime for all the sensor nodes
- High data quality and reliable performance
- Self adaptive

Although civilian WSN applications often have different goals compared with military applications, they share similar constraints when we design the communication protocols. In a simple summary, the main constraints that need to be considered include energy constraint, limited computation ability, limited transmission range and the cost requirements.

In the design of the protocol, we also use some basic assumptions which are the ones used by many other WSN protocols, such as LEACH-C and PEGASIS:

- Sensor nodes can adjust their transmission power according the distance between the sender and the receiver.
- During most their life time, sensor nodes always have enough neighbours existing within its transmission range
- Sensor nodes always change to sleeping status while they are not busy
- Base station can get global localization information

4.2 Ant behaviours and centralized approach

Nature ants are not strong or intelligent as individuals, but they successfully make the colony a highly organized and efficient society. By effectively communication between each other through the medium, ants can perform very complicated tasks such as finding the shortest path between the colony and food resources, or building the complicate colony nests. The similarity between the ant colony and a network of energy-constrained tiny sensor nodes inspired us to look at this swarm intelligence more closely.

One thing we have learned from the natural ants is that they clearly divide the work loads to different roles according to their different abilities. We apply this rule to our WSN data-gathering and communications scheme too. Under this principle, all the complicated computation, optimization and set-up operation should be performed by the base station which we has unlimited energy recourse, much stronger communication and computation abilities. On the other hand, sensor nodes keep all the basic works such as sensing, data gathering, simple data fusion and data sending/receiving.

Unlike other self-organized sensor network protocols, in which sensor nodes are can organize and communicate between themselves without any interference of the base station; our approach is that the base station plays the central role for organizing the whole network communication.

Centralized approach can bring many advantages for WSN.

- **Complicated optimization is possible and can bring significant differences**

Optimization often brings significant benefits to applications. The effects become more significant when the applications, such as WSNs, which have more constraints. One of the good examples is LEACH-C, which is a centralized version of LEACH algorithm. In LEACH-C, the base station has the global knowledge, and the base station performs stimulated annealing optimization to obtain better clustering results. It provides evidently better performance results than LEACH.

The original simulation results show that the LEACH-C outperforms LEACH by 40% in terms of energy efficiency. Another thing that needs to be mentioned is that LEACH-C performed set-up process, rotated the cluster heads and re-clustered sensor nodes in every round during the simulation; while the LEACH algorithm finely selected a re-clustering time that allowed one node to perform the cluster head just one time during its whole life time. In another words, LEACH-C used less amount of energy but can gather much more amount of data.

In a summary, centralized optimization may bring significant effect for sensor network in terms of energy efficiency and lifetime.

- **Lower set-up overhead for sensor nodes**

In protocols such as LEACH or other reactive protocols, sensor nodes need to communicate with each other in order to set up a data gathering route. In a centralized approach, base station performs the optimization to produce better routing solution, and broadcasts the information to the sensor nodes. This approach can preserve a significant amount of set-up energy for sensor nodes as well as simplify the routing set-up process.

- **High flexible communication style controlled by application**

One of the disadvantages of LEACH and PEGASIS is that the sensor nodes work on their own pace with no pause or stop, which may cause very short network lifetime and is not desirable for civilian application. Thus many reactive protocols are proposed in which sensor nodes send data when it is necessary.

However, for the self-organized reactive protocols, when many nodes in the same area have to transmit their data at the same time, it may easily produce traffic congestion and introduce data loss and long time delay. It could also affect the energy efficiency because nodes have to keep in waking-up state and receiving unnecessary signal or sending the same message for more than one time. In a centralized approach, sensor nodes' behaviour can be very much controlled by base station. In the scenario that a certain number of sensor nodes which are in the same area detect similar event, the base station can select these particular sensor nodes and just gather data from those particular nodes and let other nodes keep at sleeping status. The application can also easily adjust its data gathering pace according to the situation at the particular time.

4.3 WSN chain structure: a constrained TSP problem

In each round of the WSN data-gathering process, one node needs at least once to send its sensing data directly or indirectly to the base station. In most cases, direct transmission is not feasible because the sensor nodes have limited transmission ability and energy constraints. The distances between each sensor nodes and the base station varies and many of the sensor nodes may be too far from the base station and it implies that a large amount of energy is needed for the data transmission. This situation means

that many sensor nodes will have a very short lifetime if they send the data directly to the base station. Therefore the communication in most WSNs is using multi-hop approaches to suit the ability of sensor nodes and for energy efficiency consideration.

4.3.1 Energy consumption analyse for data gathering

According to the radio propagation model [89], the received power of signal decreases as the distance between the transmitter and the receiver increases. Generally, the propagation of radio can be modelled as a power law function of the distance between the transmitter and the receiver. For example, in the direct line-of-sight model, according to the Friss free space equation the transmit power is attenuated to proportion with the square of the distance between the sender and the receiver [89].

$$\text{The Friss free space equation is: } P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^2 L} \quad (4.1)$$

Where $P_r(d)$ is the receive power; P_t is the transmission power; G_t and G_r are the gain of the transmitting and receiving antenna; λ is the wavelength of the carrier signal; d is the distance between transmitter and receiver; L is a system loss factor which is not related to propagation.

While in the two-way ground propagation model [89], the equation of transmitter power is: $P_r(d) = \frac{P_t G_t G_r h_t^2 h_r^2}{d^4}$ (4.2); where h_r and h_t is the height of the receiving antenna and transmitting antenna; other parameters mean the same as in the equation 1.

No matter which propagation model, the energy that is needed for transmitting is increased with the distance increases, and it can roughly modeled as proportion to the

power law of the distance (So that we can ensure that the receiver can receive the signal at or above certain required power threshold).

During the data-gathering round, every node has to send its data message once directly or indirectly to the base station. How can the sensor network accomplish this task with the least energy? This is a typical NP-hard problem. But if we consider that during a data-gathering process each sensor node must and send data only once, the problem is very similar to the famous TSP problem. The difference is that often the base station is included and is the last node to be visited in a data-gathering round. Also this special node can send /receive more than once because it does not have those constraints which sensor nodes have. This difference decides that the lowest cost solution based on our requirements is the one that among a group solutions that meet the criteria as following:

- 1) The solution constructs the whole WSN as one or more Hamilton loop/loops in which the base station is included.
- 2) Each sensor node has to be in only one of the loop/loops.

The range of the number of loops is from 1 to the number of sensor nodes. At the lowest bound 1, the whole networks is constructed into one chain, while at the highest bound, every sensor node sends its data directly to the base station. It is well known that the direct approach is not feasible in most applications. In our strategy, instead of looking for the best solution in the whole solution pool, we choose to simplify the optimization process by looking for an optimal/near-optimal solution on the lowest bound, which is putting the whole network into one chain.

When considering the energy cost during the data-gathering process, the main part for each sensor node is energy used in transmission and receiving. In our approach, we

are trying to find the optimal solution for the equation: $\sum_{i=0}^N E_r(i) + E_t(i)$ (4.3) ;

where $E_r(i)$, the energy needed for receiving a certain size package, is related to the sensor nodes hardware, but not to the particular communication method. As we introduced above, the transmission energy can be roughly noted as $E_t(i) = a * D^n(i)$ where $n=2,3,4$ etc; D is the transmission distance; a is a parameter that independent with the routing protocol, it is reasonable to assume that as the same type of sensor node in the same application, the parameter can be treated as the same. To simplify the equation, in

our approach, we look for an optimal solution for the equation: $\sum_{i=0}^n D^{\alpha(i,j)}(i)$ (4.4) ;

Where $\alpha(i, j)$ is an integer that depends on different application and also could depend on the distance between the pairs of transmitter node i and receiver node j .

4.3.2 Constrained TSP for chain based wireless sensor networks

Similar to many other industry applications, when we model the chain based WSN communication into a TSP problem, there are some constraints that need to be considered.

1) Transmission range of the sensor nodes is limited. Unlike in a typical TSP, all the cities can be directly connected. In wireless sensor network, there may be only part of the nodes are the eligible neighbors for each node. This constraint can be easily well programmed in the MMAS algorithm that operated by base station. Specifically in the MMAS algorithm, when being at a city i , an ant chooses to go to a still unvisited city j with a probability given by

$$P_{ij}(t)=0 \quad \text{if } d_{ij}(t) > D \text{ otherwise,}$$

$$P_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{i \in N_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}(t)]^\beta} \quad (4.5)$$

Where $\eta_{ij}(t) = \frac{1}{d_{ij}(t)}$, $d_{ij}(t)$ is a priori known heuristic information: the distance or

the cost at any form between city i to city j . D is the sensor node transmission range limit.

2) As wireless ad-hoc network, the “cost” between two nodes may not be always symmetric, that makes our TSP problem become an unsymmetrical TSP problem. From chapter 3 we can see, MMAS has very good performance at unsymmetrical TSP.

3) The “cost” of between two nodes can change over time while the environment changing.

4) The sensor nodes can be dead and new nodes can be added. The networks topology is changing over time. As for the TSP, the instance size can changes over time.

Constraints 3 & 4 are typical for a dynamic TSP problem (DTSP). Although the straightest way to deal with the dynamics in TSP is to restart the optimization algorithm (MMAS as we choose here) again when changes occur, there are other approaches to consider as we discussed in the chapter 3. As we assume no mobility in our sensor network, the events described in constraint 4 happen more often. We have designed a new strategy approach for this specific constraint.

4.4 A local weaving ACO strategy for dynamics in WSN

For a WSN communication protocol, it is required to be adaptive to the changing of the topology of the network. When the changes happen, the communication schemes need to 1) keep routing information updated/optimal. 2) change accordingly without lose too much data 3) change accordingly before any bigger damage, such as causing a area of sensor nodes run out of energy, may happen.

As changes happen during the run-time, one of the important objectives for a communication strategy is to cope with the changes in the timely manner.

In order to provide a good run-time performance, to ensure that the WSNs response any minor change within certain time, we design a new “local weave” strategy specifically for the AntChain protocols. As we can see from chapter 3, all the existing ACO approaches for DTSP is to use and modify the information in the pheromone matrix to give the ACO a very good start point to begin its solution construction process. The approaches give a reasonable assumption that the solution after a minor change is somehow similar and related with the old one. Based on this assumption, our local weaving strategy is not only use and modify the pheromone matrix but also we assume if a city is really far away from the change, the original solution of this part is still part of a good quality solution after the change happened.

In another word, our approach is a “cut, weave and patch’ solution, and we believe that, with an optimal patch with a proper size for the change, the overall solution will keep a high quality and can provide high efficiency for the sensor networks.

Another motivation for this approach is the run-time requirements (for example, the response time) for the adaptive action. The run-time of ACO algorithms is largely

decided by the size of the TSP instance, although those existing ACO strategies give a good start (which is to keep using the pheromone information in certain way/degree), as we can find from experiments, the good new solutions still come after hundreds of iterations, run time still goes high if the instance/network size is too big.

In our “local weaving” strategy, we first decide how big the patch will be. For example, if one node dies, according to the time requirement we can estimate a certain amount of nodes will be “weaved again locally” in the chain. This decision is mainly depended on the application requirements. For example, if the base station can have more than a couple of minutes after it detects the change in a hundred node sensor network, we can simply restart our MMAS algorithm or use any strategy that already existing to construct a new optimal/near optimal solution.

If the sensor network has more than 500 nodes, a couple of minutes may not be enough to produce a high quality solution. Our strategy is, we cut out some nodes that surrounded the dead node, for example about 100 nodes, then we do the MMAS optimization for this area, after we have gotten the solution for this part, we weave the patch back to the old solution at the same position as before. The new solution is then sent to the sensor nodes for reconfiguration.

4.4.1 The “cutting” strategies

The “cutting” can be along the chain/path, or surrounding the area of the change.

Figure 4-1: Visualization of cutting strategy, (a) the Path strategy; (b) Centre strategy; after one node dies. The background is the optimal solution of a TSP instance

As we see in the Fig 4-1, the path strategy is to cut part of the chain beside the dead node. After cutting, base station can run MMAS optimization to form an optimal/near optimal local chain as a new process. It can also apply the related pheromone modification strategy to ensure a good start. To simplify the process, we can use the restart strategy that is described in last section.

Centre strategy is to cut a certain area that surrounds the changes. This method is also suitable when the change is within a small area and more than one node died/added. Because of the pheromone modification involves several nodes and changes, which is a complicated process; we can simply start an ACO optimization from the very beginning without using the old pheromone information.

4.4.2 ACO optimization for the sub-graph

Figure 4-2: Prepare the sub-graph/sub-chain for ACO optimization (a) for the Path cutting strategy; between the nodes go in and out the chain, one arbitrary edge is added; (b) for the Centre cutting strategy: following the chain, start from the change until back to the start point, arbitrary edges are added between the leaving node and the conjunct entering node.

As shown in the Figure 4-2, in order to patch the solution back to the remaining part, we must keep the same structure at the interface between the remaining part and the sub-graph/sub-chain. By this way we also make sure it becomes a small size and complete TSP problem, thus we can use the same ACO/MMAS algorithm as well.

In our strategy, the remaining part is represented as a cost zero arc. Those edges are marked with the highest pheromone level and zero cost in the pheromone matrix, which means that in the ACO algorithm, if the ants reach any of the cities that adjoined to these edges, it will certainly select them as their next paths. This can be also programmed as an arbitrary edge which is the only choice when ants approaching the any of the adjoin cities.

Thus the ACO constructs an optimal solution within the patch and then put back to the remaining part to construct the whole solution. In this way the run time can be significant decreased compared to build the whole solution all over again.

Finally two parts of solution can be merged together with the arbitrary edge/edges are replaced accordingly.

Chapter 5

The AntChain Data-gathering and Communication Protocol

The requirements on sensor networks in civilian applications may be different from those in military application. To meet the requirements of sensor networks in those civilian applications, inspired by ant social behaviour, we developed the AntChain data-gathering communication scheme. It is a centralized approach with self-adaptive capability. It introduces the Ant Colony optimization algorithm to produce optimal/near optimal routing for the sensor nodes to achieve high performance in term of energy-efficiency, high data-quality and long life time.

The developed AntChain algorithm is a base station controlled network communication scheme, for which sensor nodes are behaved by following the commands from the base station. Compared to other data-gathering schemes, which are self-configuring during the entire lifetime, the base station plays a critical role in the AntChain algorithm. However, using and maintaining the base station are normal routine in sensor network applications, and the unlimited resource availability and capacity make the base station capable to play such a role.

The main features that the AntChain protocol can provide are:

- Bi-direction Chain architecture with adaptive capability
- Ant Colony Optimization used by the base station to produce very high quality near shortest chain

- The cross-layer protocol enables the interaction between base station and sensor nodes, giving the flexibility to the end user/application for dynamic control over the data-gathering process.
- Simple data fusion operation is performed by each node. No node need to act as a control centre which requires complicated computation and operation.

The simulation results show that the AntChain algorithm performs better than other data gathering communication algorithms such as LEACH, LEACH-C and PEGASIS.

5.1 Chain architecture in AntChain protocol

During data-gathering communication, in the AntChain algorithm, all the participating nodes are formed into a chain structure. AntChain can provide two different types of chains to fit different data queries.

5.1.1 Unidirectional AntChain

Unidirectional AntChain is a simple chain structure, in which each node transmits data to its neighbour of the same direction during the data gathering process. Starting from the first node (node 0), the data is sent to the next node (right-side neighbour) in the chain until reach the end of the chain (node n). The last node (node n) acts as the chain head and send data directly to the base station. The data gathering process repeats in the same TDMA schedule in each data gathering round.

When a temporary chain is formed for a specific query, the application can choose the simple unidirectional AntChain for the data-gathering process. The structure is shown in the Figure 5-1.

Figure 5-1. The structure of Unidirectional AntChain and its TDMA scheduling

5.1.2 Bi-direction AntChain

Bi-direction Antchain is an adaptive chain structure in which sensor nodes can transmit to both its neighbors in the chain. As shown in the Figure 5-2, in the first round of data gathering process, the operation is the same as in the uni-direction Ant-chain algorithm. In the following round, data gathering starts from the end of the chain (node n), transmitting the data to its left-side node until reaching the chain head (node 0). Then the node 0 transmits the data to the base station directly.

a. Round 0, 2, 4,.... TDMA schedule

b. Round 1, 3, 5, 7.....TDMA schedule

Figure 5-2. The structure of Bi-directional AntChain and its TDMA scheduling

The bi-direction AntChain is used to regularly gathering of the data from network, the operation often needs to be performed repeatedly without any interruption from the base station. The bi-direction AntChain can provide adaptive ability to deal with the changes during the data-gathering operation.

5.1.3 Chain schemes built based on query type

We propose three chain schemes in AntChain communication to fit with different requirements of applications or according to different data queries. These schemes are shown in Figure 5-3.

Figure 5-3. Sensor network architectures when it uses different chain schemes.
a. Main chain scheme; b. Partial chain scheme; C. Temporary chain scheme

a. Main chain scheme:

All the sensor nodes in a sensor network are formed into one bi-direction chain. The scheme can be used when sensor network working at a steady and light load working

stage. Often the data gathering process is operated at the low frequency over the whole networks but needs continuing work for a long time. The bi-direction AntChain algorithm thus fits this requirement well since it is adaptive and can provide high performance as well as robustness.

b. Partial chain scheme:

The sensing area is evenly divided into several groups and forms several independent bi-direction AntChains. When use this scheme in a simultaneous manner in which every chain can perform data-gathering at the same time (if the hardware supports), the application can achieve higher data throughput and decrease the response time. It is also able to achieve higher data accuracy since the data aggregation in each chain involves less sensor nodes.

The grouping of sensor nodes can also be made based on application needs. For example, the applications can randomly select sensor nodes into a chain without considering their physical position if a coarser grid monitoring fits the current need.

The chains in a sensor network can perform data-gathering sequentially or simultaneously according to their hardware and application needs.

c. Temporary chain scheme:

Sensor nodes are selected by the base station to form a chain in order to gather data from certain specified sensor node/nodes. This scheme uses a simple Uni-direction AntChain.

It is a special chain that AntChain can make. The sensor nodes in these chains can be of two types: data node and relay node. The data nodes are the specific nodes in which the base station is interested at the query time. The relay node is used to relay the data to the base station. Therefore, for the relay node, its own data do not need to be aggregated in

the package. The number of relay nodes may be zero if the base station decides that one of the data nodes can send the information directly to the base station.

The temporary chain is used when certain specified sensor nodes are queried. For example, in a building moisture monitoring sensor network, when the base station detects that certain area in the building needs to be investigated more closely, the application can form one or more temporary chains to find out the exact “suspected” location.

The data-gathering process in this chain scheme is often performed just once or for a limited times, thus no adaptive ability is needed during the process.

5.2 AntChain communication protocol

Data-gathering in AntChain begins with the command of base station. The sensor nodes then uses the TDMA approach to collect data within the chain and finally sends the data to the base station.

5.2.1 AntChain communication phases

For the AntChain data-gathering algorithm, the base station has a priori global knowledge of the sensor nodes. It can also get the global knowledge by collecting the information through the nodes’ special device such as GPS. The base station can detect if the nodes die. The global knowledge also can be updated when the new nodes that added in. When we start with the information collection, the AntChain data-gathering process can be described with following phases:

- **Pre-setup phase:** The base station collects information from all the sensor nodes in the sensor area, then broadcasts wakeup time to all the sensor nodes.

- **Chain-setup phase:** According to the application or the end user queries, the base station uses MMAS optimization to form a main chain or several partial chains. It can also build a temporary chain by selecting related target nodes and relay nodes. The base station notifies all the sensor nodes of the data-gathering route (The TDMA schedule) by sending a broadcast message to all the sensor nodes at their wakeup time. The sensor nodes receive the message and then compute their own time slot and store the route information.
- **Independent data-gathering phase:** In this period, sensor nodes use the time slot and route information receiving, aggregating and sending their data within the AntChain and finally sending data to the base station. The data-gathering process can be performed multiple rounds without interference by the base station, but the base station can detect node death via the normal data message it receives from the sensor nodes.
- **Re-configuration phase:** when base station detects a node death, or it receives any request of an end user, it goes through the set-up process and re-sends the route information.

5.2.2 Periodical listening and wakeup message at pre-setup phase

In our algorithms, sensor nodes always perform the periodic listening function after been deployed. The length of the period depends on different application. The periodical listening is initialized when the sensor nodes are deployed and keeps functioning after that. It also can be changed by base station afterwards.

If the period of the listening is setup as P_L after the nodes have been deployed, for every P_L seconds, the node will listen for a short period of time t . If the base station is sending a short wakeup message during this time t , the sensor nodes setup themselves into receiving status. During other time within the period, sensor nodes are set at sleeping status, thus the energy cost is minimal.

After all the sensor nodes in the network are synchronized, the sensor nodes can be set on periodical listening status at the same pace and phase (The methodologies of sensor network synchronization are introduced in Chapter 2). Thus, the base station can begin to send its message at the time when all the sensor nodes are at the listening status.

After all the nodes have been deployed, before any long messages including chain information (TDMA schedule) message is sent out, base station will send the short wakeup message for several times until every sensor node gets the message. The wakeup message is a very short message which only includes a message head, two time stamps and a number to indicate the listening period. The format of setup-enable message is in Figure 5-4:

Head	Current Time(T1)	Wakeup Time(T2)	Listening period
------	------------------	-----------------	------------------

Figure 5-4. The format of wakeup message which is sent from the base station.

In the message, the current time (T1) is the base station clock time when it is sending the message. The wakeup time (T2) is the base station clock time when it will send the message of TDMA schedule; it is also the time when all the sensor nodes wake up and receive it. As we assumed that all the sensor nodes are fixed, we thus assume that the transmission delay between the sender and the receivers will keep the same during this short period of time. (Between the current time and the wakeup time)

As shown in the Figure 5-5, this wakeup message periodically sent from the base station at the period time less than the sensor nodes' listening time t . To ensure that the message is received by every node in the network, it is sent frequently by the base station before the defined wakeup time. In the wakeup message, the difference between

the two time stamps should be bigger than P_L . For each sensor node, after receives the wakeup message, it sets up its wakeup time then turns to sleep status again to avoid overhearing of other wakeup messages. It will turn its clock to T1, and set wakeup time as T2. This wakeup message can also be used to change the period of periodical listening (P_L) and the WSN's operation pace.

Figure 5-5. The time-line of pre-setup phase: The sensor nodes operate periodical listening while waiting for the base station to notify them for wakeup time.

5.3 Chain setup phase

For the AntChain algorithm, the location of sensor nodes is priori known and the computation objective is to find the data-gathering route which costs a near least amount of energy. The base station uses the MMAS algorithm to form the chain.

The operation of chain set-up is performed before the data-gathering process. At the beginning of the sensor network lifetime, the base station collects the global information and forms a main chain or several partial chains for the sensor network. When there is a query for a particular set of sensor nodes, the base station will form a temporary chain for the query. Sensor nodes receive the chain-setup message and calculate their own TDMA schedule, store their transmission route and the ID's of their neighbouring nodes.

In AntChain Algorithm, the chain can be formed into 2 different structures: Uni-direction AntChain and Bi-direction AntChain. These two structures can fit for different chain schemes. For the main chain, it will use a bi-direction structure; while for a temporary chain it will use a uni-direction structure. For the partial chain, it can use either of these structures depending on the query or the application. When the base station sends the Chain information, the message specifies the structure and the schemes.

5.3.1 Chain ID and sensor node routing table

The base station applies the ACO optimization method to form an optimal /near-optimal chain and sends the information in the setup message to the sensor nodes. The chain set-up messages are in the format as in Figure 5-6, Figure 5-7 and Figure 5-8.

In the message, the Chain type is to define the chain as main chain, partial chain or temporary chain. The partial chain could be bi-direction (type 01) or uni-directional (type 10). The sensor nodes which receive the message then can build the TDMA schedule according to the chain type. For the main chain, it will form a bi-direction chain. For the temporary chain, it will be a simple uni-directional chain.

Head	Chain type:00	Chain ID	Num-data node	Node 0		Node n
------	---------------	----------	---------------	--------	-------	--------

Figure 5-6. Format of Chain-Info message for main chain.

Head	Chain type:01 or 10	Num of Chain	Chain ID	Num of node			Node n		Chain IDm	Num of node	Node 0		Node nm
------	---------------------	--------------	----------	-------------	--	----------------	--------	----------------	-----------	-------------	--------	----------------	---------

Figure 5-7. Format of Chain-Info message for partial chain.

Head	Chain type:11	Chain ID	Num of node	Num of data node	Data Node 0		Data Node m	Relay Node 0		Relay Node n-m-1
------	---------------	----------	-------------	------------------	-------------	----------------	-------------	--------------	-------------	------------------

Figure 5-8. Format of Chain-Info message for temporary chain.

In the AntChain sensor network, each chain has a unique chain-ID. The later on query process is based on the ID. For the main chain and regular partial chains, the information can be always stored in the sensor nodes. For the temporary chain, it follows the FIFO rule and the most recent chains are always stored in the sensor node, because it is more likely that theses chains will be queried again in the nearest future time period.

The sensor nodes store all their chain information in their routing table as shown in figure 5-9. The size of the table depends on the application.

After the chain has been setup and there is no known changes happening, the base station can operate the query using the Chain ID. It can also setup a new chain according to any query or application requests.

Chain ID	Chain type	Right-S neighbor	R-T slot	Left-side neighbor	L-T slot	Node type	Node number
0	00	R-N-ID	..	L-N-ID	..	data	..
1	01	R-N-ID	..	L-N-ID	..	data	..
i	01	..	..	..	..	data	..

Figure 5-9. Sensor node routing table

5.4 Data-gathering process in AntChain

AntChain uses the TDMA approach to data-gathering. The base station forms an optimal/near optimal chain and sends the chain information to the sensor nodes. Sensor nodes calculate its TDMA time slot according to the chain order and the chain type, storing the information in their routing table.

Head	Chain ID	Round needed
------	----------	--------------

Figure 5-10. Format of Query message

The data-gathering process begins with the query/command from the base station. As shown in Figure 5-10, a query message is a short message that just indicates the chain ID and how many rounds are needed. The sensor nodes receive the query message and start the data-gathering process. After the required number of runs is done, the

sensor nodes are set back to the periodical listening status, waiting for next further commands from the base station.

5.4.1 Data-gathering in unidirectional AntChain

In the uni-directional AntChain, each node receives data from its left-side node and transmits data to its right-side node. In a round of data-gathering (base station gets information from every sensor node in the network once), except the first node in the chain, each sensor node performs data-aggregation for the receiving message and its own data. The message size and format keep the same during the data gathering process.

The data message is shown in Figure 5-11:

Head	Data	Number of participate nodes
------	------	-----------------------------

Figure 5-11. Format of Data message

If a sensor node dies, its neighboring sensor node will send its data and reset the number of participating nodes as 1. The process and message format is illustrated in Figure 5-12. The base station can detect node death by analyzing the number of participating node in the data message it receives.

Figure 5-12. Data gathering in Uni-direction AntChain

5.4.2 Data-gathering in Bi-direction AntChain

In Bi-direction AntChain, data-gathering is like a round-trip tour. We call that the data-gathering starts from the chain head the forward-round, and the return trip from the chain last node (tail) of the chain is called back-round.

The data-gathering in bi-direction AntChain is shown in Figure 5-13:

Figure 5-13. Data gathering in Bi-direction AntChain

A sensor node (node i for example) in the middle of the chain, during the forward-round, receives messages from its left-hand neighbour node $i-1$, performs data fusion with its own data, and sends its data message to its right-hand neighbour node $i+1$. When the process reaches the chain tail, after receiving and performing data fusion, the

chain tail sends the data message directly to the base station. During the back-round, it receives messages from its right-hand neighbour node $i+1$, performs data fusion with its own data, and sends its data message to its left-hand neighbour node $i-1$. As long as the chain head receives data packets, it sends the data out to the base station.

When node i dies, its both neighbours will detect it because no message is received. Each of them will then active itself as a chain head (or tail) when they receive data from it only remained neighbour. They will send data directly to the base station. By this mean, there are no data losses due the dead node. And the base station can observe the information of death.

5.5 Reconfiguration phase

The partial chains and main chain, which perform most the data gathering task, use the bi-direction chain and they can be self-adaptive to minor changes of the network. But after the changing, the chain may no longer be a near optimal chain in terms of energy-efficiency. If we keep the broken chain for a long time, the node next to the dead nodes, which is forced to behave as a chain head but may be actually far away from the base station, is at a disadvantage position and may use up all its energy very quickly.

Therefore in our algorithm, once the chain is broken, the data-gathering process will be just performed another round thus to inform the base station without losing any sensing data. That means the neighbors of the dead nodes will just perform one time chain head role and send the data directly to the base station. During this last run, every living node can detect the node death. Then all the living sensor nodes in this chain can set their status to sleeping and periodical listening status.

After the base station receives the information, it can quickly know which node/nodes are dead and decide to rebuild a chain directly according to the information.

As we described in the previous chapter, the base station can form the chain using our DTSP strategy or simply re-start the MMAS algorithm. At the same time it can use other chain, which does not include the dead node for the data-gathering task. When the chain is reconfigured, only the sensor nodes in the old chain need to receive the chain reconfigure message, other part of the sensor nodes can return to the sleeping status. During the chain reconfiguration, only the order of the nodes and the number of the nodes are changed. The chain type and chain ID will be kept the same as the original one. The format is shown in Figure 5-14.

Head	Chain ID	Num-data node: n	Node 0		Node n-1
------	----------	------------------	--------	-------	----------

Figure 5-14. Format of Chain reconfiguration message

5.6 Sensor node operations upon receiving the messages

Sensor node operation is controlled and coordinated by the base station. It always is set back to the periodical listening status after finishing a task. The operations are summarized in the Figure 5-15.

Figure 5-15. Sensor node operations upon receiving messages sent from the base station

Chapter 6

Simulation and Discussion

We used NS2 [79] on UNIX to simulate our AntChain protocols and other protocols include PEGASIS. The original simulation of LEACH can be added in as an extension of NS2. NS2 is an event-driven network simulator with extensive support of wireless network protocols. It can give detailed event information and data of each sensor nodes during their whole lifetime. For the Ant optimization algorithm especially MMAS, we made the use of the program that was originally developed by Professor Stutzle on Linux platform in C/C++. We made the platform change and changed interface and transferred it to our system according to our algorithm requirements and application constraints. Through these detailed simulation and experiments, the performances of these protocols are clearly presented in all the aspects that are need to be compared with each other.

6.1 The parameters in the MMAS algorithm

In the simulation, we define a sensor network with 100 sensor nodes. As many optimization algorithms need fine-tuning for parameter setting, we have tried different parameter settings to compare their computing speed. In all those parameter settings, the MMAS algorithm found the same optimal chain result. The run time of the algorithm had no significant difference. (For 15 times of execution with different parameter settings, total run time of each run was close to and less than 30 second on DELL PC, CPU speed 2.0G, 512 RAM).

Parameter	Description	Value
n_ants	number of ant agents	100
nn_ants	nearest neighbors in tour construction	20
max_tries	Max number of independent trials	10
alpha	influence of pheromone trails	1
beta	influence of heuristic information	1
rho	pheromone trail evaporation	0.9
max_time	maximum time for each trial	2.0s

Table 6-1. Parameters we used for MMAS during the simulation

The experiment results are consistent with lots of literature in which state that MMAS has strong easy-tuning asset compared with other optimization algorithms. The parameters we use in MMAS are summarized in Table 6-1:

6.2 The network and energy model

In our simulation, we used the sensor network in which its base station was within or away from the sensor area at the reasonable distance. The sensor nodes were deployed randomly, also considerably evenly in the sensing area, thus there would be no significant disadvantage for particular node/nodes. In the network, each sensor node can communicate with other sensor nodes; it can also directly send data to base station. All the sensor nodes can be on one of the state of sleeping, idle, data receiving or data transmission. The amount the energy consumed in these state is increased in order. A sensor node always changes back to the sleeping status when possible. In order to compare different schemes fairly, all the algorithms performed data-gathering from the all sensor nodes in the network, thus the network worked in the full workload.

As we used network simulator NS2 with the LEACH extension to conduct our network experiments, the sensor node hardware and radio energy model kept the same parameters as the parameters used by the original LEACH simulation. In addition to the Uni-AntChain algorithm and Bi-AntChain algorithm, we also implemented PEGASIS algorithm in NS2.

The main energy model and radio model we used in the simulation are summarized as following:

- For the wireless channel propagation, we adapt model used in LEACH which used free space model and two-way ground propagation model according to the sensor node transmission distance. The crossover distance is decided by the height of receiving/transmitting antennas, we use the definition as: $d_{crossover} = \frac{4\pi\sqrt{L}h_r h_t}{\lambda}$

Where $L \geq 1$ is the system loss and λ is the carrier signal wavelength

- Sensor nodes adjust their transmit power according to the sensitivity of the receiver and the distance between the sender and receiver. Described in a simple way:

$$P_t = \begin{cases} \alpha P_{r-t} d^2 & d < D_{cross-over} \\ \beta P_{r-t} d^4 & d \geq D_{cross-over} \end{cases} \quad (6.1)$$

where P_{r-t} is the sensitivity of the receiver, α , β is the constant that is related to the propagation parameters.

- The energy cost for transmitting is defined in two parts; one part is the energy that is used by the electric circuits other than the transmission radio amplifier, such as digital coding and signal-filtering. The main part that we need to consider is the

part that is cost by the transmission amplifier. The amount of this part of energy is proportional to the power of the distance between the sender and receiver. For transmitting k bit data for distance d :

$$P_k = \begin{cases} kE_{elec} + k\eta_1 d^2 & d < D_{cross-over} \\ kE_{elec} + k\eta_2 d^4 & d \geq D_{cross-over} \end{cases} \quad (6.2)$$

Where η_1, η_2 are decided by the sensor node wireless communication hardware and: the height of antenna, ..., and the receiver sensitivity. E_{elec} is decided by the sensor node hardware.

The main parameters we use in the simulation are list in Table 6-2.

Description	Parameter	Value
Crossover distance for Friss and two-ray ground model	$d_{crossover}$	87m
Radio electrics energy	E_{elec}	50nJ/bit
Radio amplifier energy	η_1 for $d < d_{crossover}$	10 pJ/bit/m ²
	η_2 for $d \geq d_{crossover}$	0.0013pJ/bit/m ⁴
Signal wavelength	λ	0.325 m
Antenna height	h_r, h_t	1.5m
Sensitivity of receiver	P_{r-t}	6.3 nW

Table 6-2. Parameters we used which related to the radio model and hardware

6.3 Other clarification of the simulation and experiments

6.3.1 Head rotation issue for PEGASIS

When we implement the PEGASIS in NS2, instead of letting each sensor node carry the chain formation computation, we let the base station do the computation and then send the information to sensor nodes. We also found that the sensor nodes could not get themselves synchronized with each other if we just used a simple TDMA approach (Same as LEACH and AntChain). Because the chain head rotated along the chain for every data-gathering round, extra synchronization efforts/energy was needed to fully complete the data-gathering process in PEGASIS.

In PEGASIS, when the chain head (*node i*) is not the first/last node of the chain, the chain is divided by the chain head into two parts beside it: node 0 to node $i-1$ and node $i+1$ to node $n-1$. The data-gathering process starts with the first node (*node 0*) of the chain. For the sensor nodes in the first part, whose index are in front of the chain head, they are assigned the time slot according to their index in the chain. All the first part nodes take their turns (time slots) to send data to their direct neighbour until the chain head receives data.

The sensor nodes in the second part, whose indexes are behind the chain head (and have larger index number than the head), start the data-gathering from the last node of the chain. Since the PEGASIS rotates the chain head in every round, the sensor nodes need computation to decide which time slots they will take in the next round. Thus the actual position (to send and receive data) of node k on the chain is somehow a random number. When we used the same strategy as the simulation of AntChain, in which each sensor nodes compute their next waking-up time (for sending or receiving data)

independently based on its own current clock, the PEGASIS chain could never get to be synchronized. As a result the base station could just receive data from part of the sensor nodes. In figure 6-1, we illustrate the time slot (transmission order) results in round i , round $i+1$ and round $i+2$ according the PEGASIS algorithm.

Figure 6-1: Time slot number for each sensor node in PEGASIS data-gathering round i , round $i+1$ and round $i+2$

In the paper [55], the author proposed a token passing approach, by which the nodes have to listen to a short token message sent by the base station (or the chain head), and then the second part of the nodes start their data-gathering process. When use this method the nodes need first to setup a wakeup time close to the “token message sending time”, and may receive data messages sent to other nodes, which also could cost too much energy. Using this approach also implies that the PEGASIS needs consume extra energy for synchronization in every round of data gathering.

To simplify the implementation and fully represent energy consuming aspects of PEGASIS, we simply ensured that every nodes' sending, receiving and head rotating behaviours be fully following the description of the algorithm in each data-gathering round; but we did not use its proposed sending and receiving order. The actual number and size of sending/receiving data packages were designed consistently with the PEGASIS algorithm; but data content or the pairing of sender and receiver might have been changed. As we showed in Figure 6-2, by following this actual order of sending and receiving operations, all the sensor nodes have gotten synchronized during the data-gathering process in our simulation. No extra energy was consumed for the synchronization proposes (as proposed by the author).

Figure 6-2: Actual time slot number for each sensor node in PEGASIS simulation for data gathering round i , round $i+1$ and round $i+2$

Therefore in the energy-consuming point of view our simulation fully represented sensor nodes behaviours in the PEGASIS and its performance results represented the same as or better than of which the actually algorithm could provide. At the base station side we assumed that every time when base station receives a data package, it then gets

the data of all the living nodes in the network, which is the idea performance for any protocols.

Since PEGASIS gives no mention of how to deal with node death, in our implementation, we simply kept all the behaviour and transmission distance the same as before (as if no node die); we assume that at base station it can get data from all the living nodes.

6.3.2 About Uni-AntChain algorithm

The main purpose of Uni-AntChain structure is to provide a simple and energy-efficient algorithm for short term or temporary data-gathering process. It gives the base station the capability to construct an energy-efficient chain with certain interested sensor nodes. In the uni-AntChain, some of the sensor nodes might just act as routers which do not fuse their own data into the output messages. Uni-AntChain does not provide self-adaptive capability since the network does not need to maintain this structure for a long time.

In our simulation, we mainly focused on the study of energy-efficiency property of this algorithm. The results showed that the uni-direction AntChain performs much better than PEGASIS and LEACH, but is slightly outperformed by the bi-direction AntChain.

6.4 Experiment and simulation results

In our simulation, the sensor network had 100 randomly deployed sensor nodes within a 100*100m, 200*200m or 300*300m area. The base station was within or at a

reasonable distance away from the sensor area. It had been set at centre, edge or away but near the sensor nodes area. No mobility was involved.

6.4.1 Chain structure comparison

The network structure formed by different algorithms make the fundamental difference for the overall performance of the algorithm. It is worth to compare output results with its shape, structure and total cost. The networks that were produced by Bi-AntChain, Uni-AntChain, PEGASIS and LEACH are shown in Figure 6-3 (a, b, c, d).

(a) Network structure produced by Bi-AntChain

(b) Network structure produced by Uni-AntChain

(c) Network structure produced by PEGASIS

(d) Network structure produced by LEACH

Figure 6-3. The network structures produced by different algorithms. In the figure, the X,Y axis present the geography position(m). Base Station is at (0,0). (a) Network structure produced by Bi-AntChain ; (b) Network structure produced by Uni-AntChain; c) Network structure produced by PEGASIS ; (d) Network structure produced by LEACH.

The sensor network structures which were produced by AntChain algorithms show visible advantages over other structures. It is optimized on several aspects regarding the needs of sensor networks.

First, as shown in the Figure 6-4, the total length of the route in each structure can be ordered as following, starts with the least length: Uni-AntChain, Bi-AntChain, PEGASIS and LEACH. Since the energy cost in WSN is proportionable to the power of the distances between the senders and receivers, the less length implies better energy-efficiency performances.

Figure 6-4 Total cost comparison for the four structures produced by uni-antchain, bi-antchain, LEACH and PEGASIS algorithms. Left: The sum of square of the length of each link in each structure; Right: The sum of the length of the all links in the each structure. The structures are shown in Figure 6-3.

The Bi-AntChain and Uni-AntChain produced very close results. The length of the chains produced by AntChain algorithms is about 75% of the chain produced by PEGASIS. While in LEACH, the clustering changes every round, we chose a better-than-average instance for the comparison. Compared with the total length of all the links in this instance, the chain formed by AntChain only had 25% of its length. The differences got more significant when we used the squared length of each link. The sum of the squared length of each link in AntChain was less than 30% of that in PEGASIS, and about 6% of that sum of structure in the selected instance of LEACH.

Second, there was no sensor node being set at any apparent disadvantage position in the AntChain structures. But in the PEGASIS algorithm as shown in the figure, some nodes were set on very disadvantaged positions, which both sides of their neighbours were far away from it. Some nodes had one of its neighbours being set at a distant position. This setting would cause rapid death for those disadvantaged nodes; gradually it caused some disadvantaged area in the network. One direct consequence of these

disadvantage nodes was that PEGASIS provided a much less robust performance than the AntChain algorithm and the LEACH.

The cause of this disadvantage is that the PEGASIS used a simple local optimization algorithm to form the chain. There is no global control over the “neighbour selection”. If a node gets into the chain later than most of its close neighbour, it has to select from the distant nodes, which have not been put in the chain as its next neighbour. Thus, the later a node gets into the chain the higher the possibility that its get “farther” neighbours.

Third, the Chain direction can be well controlled in AntChain algorithm. For the bi-direction AntChain, we can produce a TSP instance including the base station, than the neighbours of the base station can be assigned as chain head/tail.

For Uni-direction AntChain, which we could use to gather data in a short range of time, we simply produced a chain which started at the farthest node from the base station and ended at the nearest sensor nodes. The nearest sensor node was naturally the chain head. In order to get this output, in the MMAS optimization we assigned the link between the farthest node and the nearest node a lowest cost value and highest pheromone value. Therefore, every “ant agent” would naturally choose this link when they reached either of these two nodes. After the best result had been produced, we simply broke this link to form a chain, and the nearest node would set itself as chain head.

While PEGASIS algorithm, which also started from the farthest nodes and intended to go towards the nearest nodes, because of the limitation of its local search algorithm,

it could not control the chain direction at all. The final chain direction were opposite, as shown in the figure.

6.4.2 Data Amount and energy-efficiency comparison

The main objective of energy-efficiency is to send biggest amount of data with lowest energy cost. In Table 2, we give the total data amount when we set each node same amount energy as 0.5J. We present all the data that is related to the different dead node percentage. As we can see in the table, no matter the base station was within the sensor area (50, 50), or at the edge of the area (0, 0), or away from the area at (0,-87), Bi-AntChain algorithm always had the best performance in terms of the data amount received by base station. When the BS was close or within the area, no matter in which stage of the network, AntChain algorithm always performed the best.

In the Table 6-3 we can see, Bi-AntChain outperformed PEGASIS most when the first node dies, for example when the base station at the edge (0,0) or at the centre (50,50), Bi-AntChain algorithm can receive more 2 times of data the PEGASIS can receive and more that 8 time than LEACH algorithm. While the base station is away at (0,-87), the advantage became less distinguishable, but still Bi-AntChain receives more than 160% data compare to PEGASIS algorithm and about 6 times of LEACH algorithm. Comparing the total data amount received by base station with LEACH algorithm until all the sensor nodes die, when the base station is within the area, the total amount of data received is 7 times data of LEACH algorithm, and when the base station is away at (0,-87), the data received using Bi-AntChain is 5 times data of LEACH algorithm.

Base Station	Protocol	When First node	20%	50%	100%
(0,0)	LEACH	13209	14550	15646	15980
	PEGASIS	48100	70273*	79106*	81474*
	Bi-AntCHAIN	116500	116870	11700	117527
(50,50)	LEACH	13257	14580	15960	16207
	PEGASIS	50299	70312*	79423*	81889*
	Bi-AntCHAIN	117000	117400	117841	118014
(0,-87)	LEACH	13257	14580	15960	16207
	PEGASIS	49999	67943*	76606*	77861*
	Bi-AntCHAIN	83799	87888	88442	88713

TABLE 6-3: Data ammount (in unit) received by base station when first node, 20% , 50% , 80% and 100% nodes die. The base station is at (0,0),(50,50) and (0,-87), nodes are in 100x100 area. Initial energy for each node is 0.5J. *: The number represent the data received in the experiments, which is bigger than the possible actual number.

In Table 6-4, we compared the results for different algorithms when the same 100 sensor node were set to the sensor areas of different size, which are 100m*100m, 200m*200m and 300m*300m area. The topolonogy keeps the same, the distance between the sensor nodes increased proportionately as the sized increased. The differencs between AntChain algorithm and other algorithms increased significiantly when the sensor area size got bigger. Comparing the data recieved when the first node dies, in 100*100 setting, Bi-AntChain received more that 2 times more of data compared with PEGASIS, recieved more than 8 times amount od data compared with LEACH. When the sensor area incresed to 200*200, the differences were 16 times and 25 times more accordingly. Compared with LEACH, the total data received when

all the sensor nodes died, Bi-AntChain outperformed it by about 7 times, 11 times and 13 times more accordingly.

Sensor area	Check point	LEACH	PEGASIS	Bi-AntChain
100X100 (mXm)	When first node dies	13209	48180	116500
	20% nodes die	14450	70273*	116870
	100% nodes die	15980	81474*	117527
200X200 (mXm)	When first node die	4358	6841	109501
	20% nodes die	8286	12230*	110702
	100% nodes die	9899	31199*	112340
300X300 (mXm)	When first node dies	711	1276	84583
	20% nodes die	5229	8956*	87350
	100% nodes die	6969	25724*	90935

Table 6-4: Data ammount (in unit) received by base station when first node, 20% , 50% , 80% and 100% nodes die. The base station is at (0,0),(50,50) and (0,-87), nodes are in 100x100 area. Initial energy for each node is 0.5J. *: The number represents the data receive in the experiement, which is bigger that the possible actual number.

In Figure 6-5, we compared the data received per energy unit for each node. In order to compare the three algorithms fairly, because PEGASIS does not provide a solution to deal with node death, we used the results of which was before the first node died. The results show that when Base station is near or within the sensor area, Bi-AntChain performs more that 2 times better that PEGASIS, more that 8 times better than LEACH in terms of energy-efficiency.

Figure 6-5. Data Amount(unit)/Energy(J)/Node

Figure 6-6. Network lifetime (round) when dead node percentage threshold is 20% and 50% . The network is 100 node in 100x100 area, initial enery is 0.5J. The base station is at (0,0)

Network lifetime is another measurement that is used to compare the energy efficiency performance. We use “round” to measure the lifetime, in each round of data gathering, base station receives data from every sensor nodes in the network once. Of the three algorithms that were compared in the simulation, in the Figure 6-6, we can see that Bi-AntChain performance is much better that those of PEGASIS and LEACH. This is consistent with the data amount and energy-efficiency results.

6.4.3 Energy utilization of each sensor nodes

NS2 can provide detailed event information during the simulation. Through the detailed information, we can analyse how an algorithm performs in each stage of their

life time. In this section we will compare the energy consumed by each sensor nodes in different experiments until the first node died. The results show that Bi-AntChain can provide a very fair solution for each sensor nodes in the network, which means that when there is one node uses up all its energy, most other nodes are about to use up their energy too. In another word, when start with the same initial energy amount, sensor nodes in the Bi-AntChain can provide similar life time. Uni-AntChain is a little less than Bi-AntChain in term of fairness, which is rational because the bi-direction chain levels the cost by transmitting to both directions. The other advantage they have is they perform steadily when the sensor area got bigger. That means the AntChain increased its advantage when the distance between gets longer or in another word, when the density of sensor nodes gets less.

As we can see from the Figure 6-7, both LEACH and PEGASIS algorithms provided much less robustness than AntChain algorithm. The environmental changing made big differences on the algorithms performances.

(a) Sensor area is 100X100m

(b) Sensor area is 200X200m

Figure 6-7. Energy Consumed by each sensor node when first sensor node dies. Base station located at (0,0)

Another thing that also needs to note is that the performance results of LEACH were often very different even with the same setting (at different runs). The reason is that the cluster heads were chosen fully by chance. When we increased the sensor area, the differences between two runs with the same setting often increased too.

In the 100mX100m setting, when first node dies, most other sensor nodes in two AntChains also used up most of their energy (about 98%). For LEACH the average energy consumed for the rest of sensor nodes was around 60%, while for PEGASIS, the number was about 53%.

In a 200mX200m area, the difference between Bi-AntChain and Uni-AntChain becomes obviously. When the first node dies, the average energy consumed for the rest of sensor nodes is about 92% in Bi-AntChain, while in Uni-AntChain the number is about 87%. PEGASIS has been largely affected by this setting; the average energy consumed for the rest of sensor nodes is just around 36%. For LEACH, this number is nearly 40%.

6.4.4 Sensor node death distribution and unexpected node death

In most sensor network applications, sensor nodes are expected to have similar expectable lifetime in order to lower the cost of maintenance and keep the integrity of the network during most of the lifetime. If many sensor nodes die at the early stage of the sensor network lifetime, it also affects the overall performance of the network.

Keeping a higher redundancy is one of the solutions to deal with the early node death rate problem while provide an overall accepted integrity of the networks. But high redundancy not only complicates the network and communication, it also increases the cost of the hardware. Another approach for civilian applications is to frequently re-

change sensor nodes or frequently add new sensor nodes, but this also increases the cost of maintenance and management.

If sensor nodes can have a similar lifetime then the maintenance can be performed after a longer period. This attribute is very important in the civilian application, in which the cost must be low.

In Figure 6-8 (a), we can see that Bi-AntChain has a very good shape in the node death distribution. Nodes are dead in a very short time period, which means the maintenance can be done just before this point. LEACH also offers a good shape but much less Lifetime. When using PEGASIS, nodes begin to die at very early stage and nearly 50% of its lifetime, the maintenance is needed.

In the Figure 6-8 (b), we also can see that as the distances between sensor nodes increase, both LEACH and Bi-AntChain still keep the similar distribution shape as before. But for PEGASIS, the increase of sensor area and distance between nodes caused obvious increase of unpredictability of sensor nodes' lifetime.

(a)

(b)

Figure 6-8. Sensor Node Death Distribution: base station at (0,0), all sensor nodes has 0.5j at beginning. (a), 100X100 sensor field ;(b) 200X200 sensor field

When the sensor nodes are put in a larger area (200mX200m), the range of node death time increases largely and sensor nodes start to die at the very earliest time. Sensor nodes in PEGASIS had its first node died at the similar time as LEACH.

From the comparison, we can see that Bi-AntChain can provide more predictable behaviour such as similar lifetime for all the sensor nodes in the application. In another word, Bi-AntChain algorithm lessens the systematically unexpected death of node, which is because of the near optimal communication scheme. At the same time, sensor nodes in Bi-AntChain have the longest lifetime.

6.4.5 Data integrity comparison

Sensor networks are data-centric networks. In applications such as health care, each sensor node may be related to a particular patient or a particular equipment spot. In building monitoring application, each node may be related a specific room or a specific building part. The quality and reliability of data is thus an important requirement. Moreover, it is rational to say that at some point before all sensor nodes die out; the network already lost sanity according to certain requirement of the applications. In our simulation, we assumed that a sensor network kept its sanity if the living sensor nodes number was above certain thresholds.

We use the metrics I to measure the overall data quality during its lifetime.

$$Q = \sum_{i=1}^n (Data_Amount / Num_node) / n \quad (6.3)$$

In this formula, n is the total round of data-gathering at the current measuring point, Num_Node is the total nodes number of the original sensor network. $Data_Amount$ is the number of nodes that report its information to base station in each

round. Thus starts from the first round, if all the deployed sensor nodes are alive and can perform data-gathering tasks successfully, the data-quality Q keeps at its maximum value, which is 1.

In the figure 6-9, we compared the value of Q at three points, when 20% nodes died, when 50% nodes died and when 80% nodes died. The data quality Q was close to 1 at all three points for Bi-AntChain algorithm. Our experiment also showed that for Bi-AntChain algorithm, more that 99% lifetime, the network provides highest data quality which is 1. Therefore we can say that Bi-AntChain not only provides longer lifetime and higher energy-efficiency of the network, it also provides very high quality of data during its lifetime.

The LEACH algorithm performed better than PEGASIS, while the data decreased more slowly. For PEGASIS, it comparatively had more nodes die at much earlier stage because it used a simple algorithm to form the chain and some nodes have high possibility to have far away “permanent” neighbours that drilled all of it energy in transmission.

Fig 6-9. Data Integrity comparison at different sanity threshold: when 20%, 50% and 80% nodes died; Sensor area=100x100; base station at (0,0)

The data quality metric that we compared in this section provides an overall quality measure for the sensor network data gathering for each algorithm. The results were consistent with the detailed node death distribution we discussed in last section.

Chapter 7

Conclusion and Discussion

As wireless sensor networks have been used in the civilian or industry applications more and more, our AntChain algorithms provide a promising solution to face the new challenges on high industry requirements of energy-efficiency, data quality and lifetime predictability.

7.1 Summary of contribution

First of all, AntChain algorithm divided the work between sensor nodes and base station and thus simplifies sensor node computation and lowers its communication work load. It also provides more flexibility for applications to choose from different schemes according to different requirements, including Bi-AntChain and Uni-AntChain. The data-gathering in our centralized approach can be held over the whole network or just part sensor nodes when required.

AntChain can construct more elegant structures for the random deployed WSNs. It provides least total length (cost) and better positioning for each node. The centralized optimization shows a strong advantage of the high quality of output chain. It also shows strong advantage on flexibility. All the constraints and requirements can be easily handled by the ACO optimization on base station. For example, it can easily control the chain direction or conveniently define or change the cost of each link when required. This property suits the characters of WSN very well.

AntChain provides an energy-efficient solution for the application in which base station is not far away. From the comparison of the simulation results, we can conclude that for the energy-efficiency, AntChain out-perform LEACH and PEGASIS by about 8 times and 2 times when the sensor area is 100m*100m. The advantages of AntChain increase if the distance between the sensor nodes increases. In another words, if an application has low density of sensor nodes, which is quite true for many civilian applications, AntChain algorithm can provide much better performance results than LEACH and PEGASIS.

AntChain can not only provide longer lifetime, it also shows much higher reliability of the sensor nodes lifetime. In AntChain, all sensor nodes have highest possibility to have very similar and predictable lifetime, while in PEGASIS, there is a large range of the length of lifetime. For LEACH, the lifetime is much shorter.

From the data quality point of view, the AntChain has the nearly idea performance where it can provide maximum data quality during 99% of the sensor network's lifetime in the 100m*100m setting. While PEGASIS and LEACH start to lose node at very early stage and thus affects its overall data quality.

In a summary, we can conclude that AntChain algorithm can provides a flexible, reliable and energy-efficient data-gathering and communication solution for sensor networks, especially for the civilian applications that require low cost, steady performance and high data quality.

7.2 Future work

Future work can be done in the further research on the reactive behavior of AntChain algorithm and compare it with other reactive WSNs. Other work also can include the research of part-chain schemes in different occasion and the performance comparison.

Another interesting area is to conduct further research in the ACO optimization in the particular WSN setting. For example, more constraints and dynamics such as limited transmission range, different initial sensor node energy level or few (or one) mobile nodes can be added in the consideration.

References

- [1] T. Stutzle and S. Linke, Experiments with Variants of Ant Algorithms, *Mathware and Soft Computing*, 9(2-3), pp.193–207, 2002.
- [2] T.,Stutzle, A. Gre, S. Linke, and M. Rotger, A Comparison of Nature Inspired Heuristics on the Travelling Salesman Problem.” In Deb et al, editors, *Proceedings of PPSN-VI, Sixth International Conference on Parallel Problem Solving from Nature*, volume 1917 of LNCS, pages 661-670, 2000.
- [3] Di Caro G. and M. Dorigo, AntNet, A Mobile Agents Approach to Adaptive Routing. *Tech. Rep. IRIDIA/97-12*, Université Libre de Bruxelles, Belgium, 1997
- [4] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothcrantz, Ant-based Load Balancing in Telecommunication Networks”, *Adaptive Behavior*, 5, pp. 169-207,1996.
- [5] L.M , Dorigo M, An Ant Colony System Hybridized with a New Local Search for the Sequential Ordering Problem, *INFORMS Journal on Computing*, vol.12(3), pp. 237-255, 2000.
- [6] G. Di Caro, and M. Dorigo, Mobile Agents for Adaptive Routing, *Proceedings of the 31st Hawaii International Conference on Systems Science*, pp. 74-83, 1998.
- [7] Wireless Sensor Networks for Medical Care
<http://www.eecs.harvard.edu/~mdw/proj/vitaldust/>, April. 2005
- [8] J. Elson, L. Girod, and D. Estrin. Fine-Grained Network Time Synchronization using Reference Broadcasts, In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002)*, Boston, MA. December 2002.
- [9] S. Ganeriwal, R. Kumar, and M.B. Srivastava. Timing-Sync Protocol for Sensor Networks *SenSys'03*, Los Angeles, California, USA. November 5-7, 2003.
- [10] J.L. Deneubourg, S. Goss, J. Pasteels, D. Fresneau, and J.P. Lachaud, "Self-Organization Mechanisms in Ant Societies (II): Learning in Foraging and Division of Labor," in *From Individual to Collective Behavior in Social Insects*, J.M. Pasteels, and J.L. Deneubourg, eds., *Experimentia Supplementum*, vol. 54, Birkhauser Verlag, Basel, pp. 177-196, 1990
- [11] J.L. Deneubourg, and S. Goss "Collective Patterns and Decision Making," *Ethology, Ecology and Evolution I*, 1989, pp.295-311.

- [12] J.L. Deneubourg et al, The Dynamics of collective Sorting: Robot-Like Ants and Ant-Like Robots, in From Animals to Animats: Proc. First Int. Conference on Simulation of Adaptive Behavior, J-A. Meyer, and S.W. Wilson, eds., Paris, France, pp. 356-363, 1990.
- [13] L. M. Gambardella, E. Taillard and M. Dorigo Ant Colonies for the Quadratic Assignment Problem. *Journal of the Operational Research Society*, 50:167-176, 1999.
- [14] V. Maniezzo and A. Colomi The Ant System Applied to the Quadratic Assignment Problem. *IEEE Transactions on Knowledge and Data Engineering*, to appear, 1999.
- [15] Bullnheimer B., R.F. Hartl and C. Strauss. Applying the Ant System to the Vehicle Routing Problem. In: Voss S., Martello S., Osman I.H., Roucairol C. (eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer:Boston, 1999
- [16] Forsyth P. and A. Wren, An Ant System for Bus Driver Scheduling. Presented at the *7th International Workshop on Computer-Aided Scheduling of Public Transport*, Boston, August 1997.
- [17] M. Dorigo, V. Maniezzo & A. Colomi. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26(1):29-41, 1996
- [18] Thomas Stutzle and Marco Dorigo: The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, Kluwer Academic Publishers, Norwell, MA, 2002.
- [19] Schoonderwoerd R., O. Holland, J. Bruten and L. Rothkrantz. Ant-based Load Balancing in Telecommunications Networks. *Adaptive Behavior*, 5(2):169-207, 1997
- [20] Di Caro G. and M. Dorigo (). Mobile Agents for Adaptive Routing. *Proceedings of the 31st Hawaii International Conference on System*, IEEE Computer Society Press, Los Alamitos, CA, 74-83, 1998.
- [21] Navarro Varela G. and M.C. Sinclair (1999). Ant Colony Optimisation for Virtual-Wavelength-Path Routing and Wavelength Allocation. *Proceedings of the Congress on Evolutionary Computation (CEC'99)*, Washington DC, USA, July 1999.
- [22] B. Bullnheimer, R. F. Hartl, and C. Strauss. A new rank-based version of the Ant System: A computational study. *Central European Journal for Operations Research and Economics*, 7(1):25-38, 1999.
- [23] M. Dorigo and L. M. Gambardella. Ant colonies for the traveling salesman problem. *BioSystems*, 43:73-81, 1997.

- [24] M. Dorigo and L. M. Gambardella. Ant Colony System: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66, 1997.
- [25] T. Stutzle and H. H. Hoos. The Max-Min Ant System and local search for the traveling salesman problem. In T. Bick, Z. Michalewicz, and X. Yao, editors, *Proceedings of the 1997 IEEE International Conference on Evolutionary Computation (ICEC'97)*, pages 309–314. IEEE Press, Piscataway, NJ, 1997.
- [26] T. Stutzle and H. H. Hoos. Max-Min Ant System. *Future Generation Computer Systems*, 16(8):889–914, 2000.
- [27] T. Stutzle, MAX-MIN Ant System for the Quadratic Assignment Problem. *Technical Report AIDA-97-04*, Darmstadt University of Technology, Computer Science Department, Intellectics Group.
- [28] M. Guntsch, M. Middendorf, and H. Schmeck. An Ant Colony Optimization Approach to Dynamic TSP. In: L. Spector et al. (eds.) *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, San Francisco, CA: Morgan Kaufmann Publishers, 860–867, 2001.
- [29] T. Stutzle and H. Hoos. The MAX-MIN Ant System and Local Search for Combinatorial Optimization Problems: Towards Adaptive Tools for Combinatorial Global Optimization. In S. Voss, S. Martello, I.H. Osman, and C. Roucairol, editors, *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, pages 313–329, 1998.
- [30] F. Mondinelli, Z.M. Kovacs-Vajna. Self-Localizing Sensor Network Architectures. *Instrumentation and Measurement*, IEEE Transactions on, volume: 53, issue: 2, Pages: 277 – 283, April. 2004.
- [31] A. Lim. Distributed Sensor Networks for Real-Time Systems with Adaptive Reconfiguration. *J. Franklin Inst.*, 2000.
- [32] R. H. Ware. Real-time national GPS networks for atmospheric sensing. *Journal of Atmospheric and Solar-Terrestrial Physics* Volume 63, Issue 12, Pages 1315–1330, August 2001,
- [33] G. Varela and M. Sinclair. Ant Colony Optimization for Virtual-Wavelength-Path Routing and Wavelength Allocation", *Proceedings of the Congress on Evolutionary Computation*, Washington DC, pp. 1809–1816, 1999.
- [34] T. White, B. Paturek, and F. Oppacher. Connection Management by Ants: An Application of Mobile Agents in Network management, *Proceedings of Combinatorial Optimization*,

Brussels, 1998

- [35] M. Guntsch and M. Middendorf, Pheromone Modification Strategies for Ant Algorithms applied to Dynamic TSP. In: E.J.W. Boers et al. (Eds.) *Applications of Evolutionary Computing: Proceedings of EvoWorkshops 2001*, Lake Como, Italy, Springer Verlag, LNCS 2037 213-222; 2001
- [36] X. Ji, H. Zha, Robust sensor localization algorithm in wireless ad-hoc sensor networks, *Proceedings of Computer Communications and Networks*, the 12th International Conference (ICCCN) on Pages:527 – 532, 20-22 Oct. 2003
- [37] S. Capkun, M. Hamdi, J.-P. Hubaux, GPS-free positioning in mobile ad-hoc networks. *System Sciences*, 2001. *Proceedings of the 34th Annual Hawaii International Conference on* Pages:10 pp, 3-6 Jan. 2001
- [38] A. Manjeshwar, D.P. Agrawal, TEEN: a routing protocol for enhanced efficiency in wireless sensor networks. *Parallel and Distributed Processing Symposium*, *Proceedings 15th International* , 23-27 April 2001
- [39] A. Manjeshwar, D. P. Agrawal, APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks, *International Parallel and Distributed Processing Symposium: IPDPS 2002 Workshops* ;April 15 - 19, 2002
- [40] D. Niculescu, B. Nath, Ad hoc positioning system (APS), *Global Telecommunications Conference*, 2001. *GLOBECOM '01. IEEE*, Pages:2926 - 2931 vol.5, 25-29 Nov. 2001
- [41] W. Ye, J. Heidemann, D. Estrin, An Energy-Efficient MAC Protocol for Wireless Sensor Networks, In *Proceedings of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, Vol.3, pp. 1567-1576, New York, NY, USA, June, 2002.
- [42] M. Stemm, R.H. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices," *IEICE Transactions on Communications*, vol. E80-B, no. 8, pp. 1125–1131, Aug. 1997.
- [43] L. MAN, Standards Committee of the IEEE Computer Society, *Wireless LAN medium access control (MAC) and physical layer (PHY) specification*, IEEE, New York, NY, USA, IEEE Std 802.11-1997 edition, 1997.
- [44] S. Singh, C.S. Raghavendra, "PAMAS: Power aware multi-access protocol with signalling for ad hoc networks," *ACM Computer Communication Review*, vol. 28, no. 3, pp. 5–26, July 1998.
- [45] R. Kalidindi, L. Ray, R. Kannan, and S.S. Iyengar, Distributed Energy-Aware MAC Protocol for Wireless Sensor Networks , in *International Conference on Wireless Networks* , Las Vegas, Nevada, June 2003.

- [46] K. Sohrabi, G.J. Pottie, Performance of a Novel Self-organization Protocol for Wireless Ad Hoc Sensor Networks, *Proc. of the IEEE 50th Vehicular Technology Conference*, pp.1222-1226, 1999.
- [47] J.C. Haartsen, The Bluetooth radio system, *IEEE Personal Communications Magazine*, pp. 28–36, Feb. 2000.
- [48] A.J. Viterbi, A. Wesley, *CDMA Principles of Spread Spectrum Communication*, 1995, ISBN 0201633744
- [49] M. Guntsch, M. Middendorf, Applying Population Based ACO to Dynamic Optimization Problems, Ant Algorithms, Proceedings of Third International Workshop ANTS 2002, Brussels, Belgium, Springer Verlag, LNCS 2463, 2002, 111-122.
- [50] M. Guntsch and M. Middendorf, A Population Based Approach for ACO, Proc. 2nd European Workshop on Evolutionary Computation in Combinatorial Optimization (EvoCOP-2002), Kinsale, Ireland, Springer Verlag, LNCS 2279, 2002, 72-81.
- [51] G.J. Pottie, W.J. Kaiser, *Wireless Integrated Network Sensors*, Communications of the ACM, vol. 43, no. 5, pp. 551-8, May 2000.
- [52] D. Estrin et al., Next Century Challenges: Scalable Coordination in Sensor Networks, MobiCom 1999.
- [53] S.S. Iyengar, D.N. Jayasimha, D. Nadig, A versatile architecture for the distributed sensor integration problem. *IEEE Trans. Comput.* 1994, 43(2), 175-185.
- [54] D.N. Jayasimha, S.S. Iyengar, and R.L. Kashyap, Information integration and synchronization in distributed sensor networks. *IEEE Trans. Syst., Man, Cybern.*, SMC21 (21):1032–1043, Sept./Oct. 1991.
- [55] S. Lindsey, C. Raghavendra, and K.M. Sivalingam, Data Gathering Algorithms in Sensor Networks Using Energy Metrics, *IEEE Transactions on Parallel and Distributed Systems*, 13(9):924-935, September 2002
- [56] E. Shih et al., Physical layer driven algorithm and protocol design for energy-efficient wireless sensor networks, *Proc. MobiCOM 2001*.
- [57] J. Rabaey, M. Ammer, J. Silva jr, D. Patel, and S. Roundy, Picoradio supports ad hoc ultra-low power wireless networking. In *IEEE Computer Magazine*, July 2000. pp42-48
- [58] A. Mainwaring et al., Wireless Sensor Networks for Habitat Monitoring. *ACM International Workshop on Wireless Sensor Networks and Applications*, September 28, 2002.

- [59] J. Hill, R. Szewczyk, A. Woo, S. Hollar, and D.C. K. Pister, System architecture directions for networked sensors. In ASPLOS(Architectural Support for Programming Languages and Operating Systems), November 2000.
- [60] L. Schwiebert, S.K.S. Gupta, and J. Weinmann, Research challenges in wireless networks of biomedical sensors, in Proc. ACM/IEEE Conf. on Mobile Computing and Networking (MOBICOM), pp. 151–165, 2001.
- [61] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, *Energy efficient communication protocol for wireless micro-sensor networks*, In Proc. IEEE Hawaii Int. Conf. on System Sciences, pages 4–7, January 2000.
- [62] J.Kulik, W.R. Heinzelman, and H. Balakrishnan, *Negotiation-based Protocols for Disseminating Information* in Wireless Sensor Networks. In ACM Wireless Networks, Vol. 8. 2002.
- [63] H. Qi, S. Iyengar, K. Chakrabarty, *Multi-resolution data integration using mobile agents in distributed sensor networks*, IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews, vol. 31, no. 3, pp383-391, August, 2001
- [64] R. Wesson et al, *Network structure for distributed situation assessment*, IEEE transaction on System, man, cybernetics, SMC-11(1), Jan 1981
- [65] D.C. Steere, A. Baptista, D. McNamee, C. Pu, J. Walpole, Research challenges in environmental observation and forecasting systems. MOBICOM 2000: 292-299
- [66] SMART DUST, Autonomous sensing and communication in a cubic millimeter <http://robotics.eecs.berkeley.edu/~pister/SmartDust/>, April, 2004
- [67] C.J. Eyckelhof, M. Snoek, Ant Systems for a Dynamic TSP; Ant Algorithms : Third International Workshop, ANTS 2002, Brussels, Belgium, September 12-14, 2002. Proceedings
- [68] D. Li, K.D. Wong, Y.H. Hu, A.M. Sayeed, *Detection, classification and tracking of targets*, IEEE Signal Processing Magazine 19 (2002) 17–29
- [69] M. Ridley, E. Nettleton, S. Sukkarieh, H. Durrant-Whyte, Tracking in decentralised air-ground sensing networks, *Information Fusion*, 2002. Proceedings of the Fifth International Conference on , Volume: 1 , 8-11 July 2002 Page(s): 616 -623 vol.1
- [70] J. Nemeroff, L. Garcia, D. Hampel, S. DiPierro, *Application of sensor network communications* Military Communications Conference, 2001. MILCOM 2001.

Communications for Network-Centric Operations: Creating the Information Force. IEEE , Volume: 1 , Page(s): 336 -341 vol.1, 28-31 Oct. 2001

- [71] C. Intanagonwiwat, R. Govindan, and D. Estrin, *Directed diffusion: A scalable and robust communication paradigm for sensor networks*. In *MobiCOM*, Boston, MA, August 2000.
- [72] J. Elson, D. Estrin, Time Synchronization for Wireless Sensor Networks. *Parallel and Distributed Processing Symposium., Proceedings 15th International* , 23-27 April 2001.
- [73] K. Romer, Time Synchronization in Ad Hoc Networks. *Proc. 2001 ACM Int. Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'01)*, Long Beach, CA, October 4 -5, 2001.
- [74] J. Elson, K. Romer, Wireless Sensor Networks: A New Regime for Time Synchronization. *In Proceedings of the First Workshop on Hot Topics in Networks (HotNets)*, Princeton, New Jersey, USA. 28-29 October 2002.
- [75] M.L. Sichitiu, C. Veerarittiphan, Simple, accurate time synchronization for wireless sensor networks. *Wireless Communications and Networking (WCNC'03)*, IEEE , Volume: 2, 16-20 March 2003.
- [76] R. Karp, J. Elson, D. Estrin, and S. Shenker, Optimal and Global time Synchronization in Sensor networks. *CENS Technical Report 0012*, April 10, 2003.
- [77] J.V. Greunen, J. Rabaey. Lightweight Time Synchronization for Sensor Networks. *WSNA'03*, San Diego, California, USA. September 19, 2003.
- [78] A. Hu, S.D. Servetto, Asymptotically Optimal Time Synchronization in Dense Sensor Networks. *WSNA'03*, San Diego, California, USA. September 19, 2003.
- [79] The Network Simulator - ns-2; <http://www.isi.edu/nsnam/ns/>, Sept. 2004
- [80] The Pacific Northwest Seismograph Network; http://www.pnsn.org/INFO_GENERAL/INFOSHEET/welcome.html, Feb. 2005
- [81] S.J. Dyke, J.M. Caicedo and E.A. Johnson, Monitoring of a Benchmark Structure for Damage Identification, *Proc. of the Engineering Mechanics Specialty Conference*, 2000.
- [82] Y. Fujino and M. Abe, Structural Health Monitoring in Civil Infrastructures and Research on SHM of Bridges at the University of Tokyo, *Proc. of the Third World Conference on Structural Control*, Vol.1: 125-140, 2002.
- [83] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, D. Estrin, A Wireless Sensor Network for Structural Monitoring *In Proceedings of the ACM Conference on Embedded Networked Sensor Systems(Sensys04)*, November 2004.

- [84] D. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*. Brooks/Cole Pub Co., 1996
- [85] B. Parkinson, P. Axelrad, and P. Enge, Eds., *Global Positioning System: Theory and Applications*: AIAA, 1996.
- [86] P. Enge, R.M. Kalafus, M.F. Ruane, Differential Operation of the Global Positioning System, *IEEE Communication Magazine*, Vol. 26, No. 7, pp48-60, 1988.
- [87] Ed. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The travelling salesman problem - A guided tour of combinatorial optimization*. John Wiley & Sons, 1985.
- [88] Eric W. Weisstein. "NP-Hard Problem." From *MathWorld*—A Wolfram Web Resource. <http://mathworld.wolfram.com/NP-HardProblem.html>, May. 2005
- [89] W. Ye, "Radio Propagation Model", ns Manual, Chapter 17, http://www.isi.edu/~weiyu/pub/propagation_ns.pdf, May. 2005