ENERGY EFFICIENT COVERAGE AND CONNECTIVITY PROBLEM IN WIRELESS SENSOR NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MUSTAFA GÖKÇE BAYDOĞAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

JULY 2008

Approval of the thesis:

ENERGY EFFICIENT COVERAGE AND CONNECTIVITY PROBLEM IN WIRELESS SENSOR NETWORKS

submitted by **MUSTAFA GÖKÇE BAYDOĞAN** in partial fulfillment of the requirements for the degree of **Master of Science in Industrial Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _______________

Prof. Dr. Nur Evin Özdemirel
Head of Department, **Industrial Engineering** _______________

Prof. Dr. Nur Evin Özdemirel
Supervisor, **Industrial Engineering Dept., METU** _______________

Examining Committee Members:

Prof. Dr. Murat Köksalan
Industrial Engineering Dept., METU _____________________

Prof. Dr. Nur Evin Özdemirel
Industrial Engineering Dept., METU _____________________

Assoc. Prof. Dr. Yasemin Serin
Industrial Engineering Dept., METU _____________________

Asst. Prof. Dr. Sedef Meral
Industrial Engineering Dept., METU _____________________

Dr. Onur Tolga Şehitoğlu
Computer Engineering Dept., METU _____________________

Date: 18.07.2008

Name, Last name: Mustafa Gökçe Baydoğan

Signature :

ABSTRACT

ENERGY EFFICIENT COVERAGE AND CONNECTIVITY PROBLEM IN WIRELESS SENSOR NETWORKS

BAYDOĞAN, Mustafa Gökçe
M.S., Department of Industrial Engineering
Supervisor: Prof. Dr. Nur Evin ÖZDEMİREL

July 2008, 104 pages

In this thesis, we study the energy efficient coverage and connectivity problem in wireless sensor networks (WSNs). We try to locate heterogeneous sensors and route data generated to a base station under two conflicting objectives: minimization of network cost and maximization of network lifetime. We aim at satisfying connectivity and coverage requirements as well as sensor node and link capacity constraints. We propose mathematical formulations and use an exact solution approach to find Pareto optimal solutions for the problem. We also develop a multiobjective genetic algorithm to approximate the efficient frontier, as the exact solution approach requires long computation times. We experiment with our genetic algorithm on randomly generated problems to test how well the heuristic procedure approximates the efficient frontier. Our results show that our genetic algorithm approximates the efficient frontier well in reasonable computation times.

Keywords: Wireless sensor networks, heterogeneous sensors, energy efficiency (lifetime), network cost, connectivity, coverage, node and link capacity, location, routing, genetic algorithm, multiobjective optimization.

ÖZ

KABLOSUZ DUYGAÇ AĞLARINDA ENERJİ VERİMLİLİĞİ, KAPSAMA VE BAĞLANABİLİRLİK PROBLEMİ

BAYDOĞAN, Mustafa Gökçe
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Nur Evin ÖZDEMİREL

Temmuz 2008, 104 sayfa

Bu tezde kablosuz duygaç ağlarındaki enerji verimliliği, kapsama ve bağlanabilirlik problemi ele alınmıştır. Ağ maliyetinin minimizasyonu ve kapsama süresinin maksimizasyonu gibi birbiriyle çelişen iki amaç doğrultusunda, farklı tiplerde duygaç yerleşimi ve üretilen duygaç verisinin ana istasyona iletilmesi planlanmıştır. Etkin çözümleri tam olarak bulmak için matematiksel formülasyonlar önerilmiş ve kesin çözüm veren bir yaklaşım kullanılmıştır. Kesin çözüm veren yaklaşım çok fazla uzun çözüm zamanı gerektirdiğinden, etkin çözümleri yaklaşık olarak bulacak çok amaçlı bir genetik algoritma geliştirilmiştir. Genetik algoritmanın performansı rassal olarak üretilmiş problemler üzerinde deneysel olarak test edilmiştir. Sonuçlar genetik algoritmanın makul çözüm sürelerinde etkin çözümlere yaklaşabildiğini göstermektedir.

Anahtar Kelimeler: Kablosuz duygaç ağları, farklı duygaç tipleri, enerji verimliliği, ağ maliyeti, bağlanabilirlik, kapsama, düğüm ve kanal kapasitesi, yerleşim, rotalama, genetik algoritma, çok amaçlı optimizasyon

To my family

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and appreciation to Prof. Dr. Nur Evin Özdemirel for her encouragement and support throughout this study. Without her brilliant guidance, the thesis would not have been possible.

I would like to thank Prof. Dr. Murat Köksalan and Assoc. Prof. Dr. Haldun Süral for their support throughout this thesis. I would also like to convey thanks to jury members for their valuable comments on this thesis.

I am deeply grateful to my father Mehmet Baydoğan, my mother Perihan Baydoğan, my sister Banu Gökçen Baydoğan for their love and support. Without them, this work could not have been completed.

I am deeply indebted to my friends, Utku Lokman, Banu Lokman, Tülin İnkaya, Bora Kat, Pelin Bayındır, İsmail Bakal, Pınar Bakal, Burcu Özdemir, Melih Özlen and Baykal Hafızoğlu, whose help, stimulating suggestions and encouragement helped me in all the time of research for and writing of this thesis. I am also grateful to my dear friend Ezgi Özkapıcı for her love, continued moral support and welcomed presence.

I would also thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing the financial means throughout this study.

Finally, my special thanks go to my friends Mustafa Yüksel, Çağlar Kılcıoğlu and Çağlar Ata for their help, support and cheerful presence through the course of this study. Thanks for giving me a shoulder to lean on whenever I need.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The development of distributed networks that are capable of sensing, computation, and wireless communication has emerged from recent advances in processor, memory and radio technology. Today wireless sensor networks (WSN) have a wide variety of applications such as battlefield surveillance, biological detection, home security and inventory tracking. Therefore, the design of wireless sensor networks has started to attract a great deal of research attention.

A wireless sensor network consists of sensor devices deployed in a region of interest. Each sensor has processing and wireless communication capabilities, which enable it to gather information about the monitoring area and to generate and transmit the data to a base station. The base station aggregates and analyzes the data received and decides whether there is an unusual event occurrence in the monitoring area.

In wireless sensor networks, the energy source provided for sensors is usually battery power. Hence, sensors cannot operate for a long time without recharging. It is undesirable or impossible to replace the battery power of all sensors since they often work in remote or hostile area such as battlefields or disaster areas. However, a long system lifetime is expected by most of the monitoring applications. The lifetime of the network, which is measured by the time until the network no longer provides an acceptable event detection ratio, directly affects network usefulness. Therefore, conserving the energy resource and prolonging the system lifetime is an important issue in the design of wireless sensor networks.

Given possible locations where heterogeneous sensors can be deployed and a base station together with the available energy for each sensor type, we are interested in the deployment of the sensors in an efficient manner. The data sensed should be collected from all the sensors and transmitted to the base station such that total cost of sensors deployed is minimized and lifetime of the network is maximized. Sensor deployment is a critical issue because it affects the cost, coverage (detection) capability and energy efficiency (lifetime) of a wireless sensor network. Connectivity is another issue as sensors should be able to communicate in order to transmit the data to the base station.

The WSN design studies in the literature are generally limited with single objective formulations. However, the problem of energy efficient coverage and connectivity of WSN has a multiobjective nature. Taking these into account, we try to handle two conflicting objectives, minimization of network cost and maximization of lifetime, together. We try to make both location and routing decisions under these two objectives. Connectivity and coverage requirements together with application specific constraints are not taken into consideration explicitly in most of the studies. Patel et al. (2004) emphasize that wireless channel capacity and finite sensor capacities should be taken into consideration in order to prevent routing of the data packets over highly congested links and paths since congestion increases the delay and packet losses, which will increase the energy consumption because of retransmission of the packets. Our study takes all of these aspects into account simultaneously for WSN design. We try to investigate the tradeoff between cost and lifetime objectives while deciding on sensor deployment and data routing. We consider locating sensors at given possible locations resulting in an adhoc network and try to model the data communication under the connectivity, coverage, node capacity and link capacity constraints.

We propose mathematical formulations and use an exact solution approach to find the Pareto optimal solutions for the energy efficient coverage and connectivity problem in wireless sensor networks. The exact solution approach is based on solving the mathematical models iteratively with different objective levels. This approach requires a high computational effort. Therefore, we propose a multiobjective genetic algorithm to approximate the efficient frontier in reasonable computation times. Our approach is similar to NSGA-II (nondominated sorting GA) proposed by Deb et al. (2002). However, we have some modifications which address our problem specifics.

The rest of the thesis is organized as follows. In Chapter 2, we discuss the related literature. Chapter 3 defines the single objective problems and their formulations as well as the exact bicriteria solution approach. Chapter 4 reports the results of the exact solution approach. Then, we describe the multiobjective genetic algorithm to approximate the efficient frontier in Chapter 5. Chapter 6 provides the computational results of the genetic algorithm. We conclude in Chapter 7 by pointing our main findings and suggestions for future research.

CHAPTER 2

LITERATURE REVIEW

2.1 Properties of Wireless Sensor Networks

Design space of the wireless sensor networks is very large since applications and systems differ much with varying requirements and characteristics. Taking this fact into account, Römer and Mattern (2004) try to point out the design issues in wireless sensor network design. A similar discussion is also made by Akyildiz et al. (2002). They conclude that determination of hardware and software requirements is problematic in a multidisciplinary research area such as wireless sensor networks. Interaction between users, application domain experts, hardware designers and software developers is needed for an efficient design. Analyzing the projects that had been conducted, they conclude that sensor network design space and its various dimensions should be characterized. They complete their analysis with the following dimensions.

Deployment

Deployment of the sensors may take different forms. Sensor nodes can be located at predetermined locations or they can be dispersed randomly, e.g. dropping from aircraft on to a disaster area. This can be a one time activity (sensor nodes are deployed only once) or a continuous process (after first batch is deployed, additional nodes are deployed to replace failed ones or to improve coverage during monitoring). Type of deployment affects the decisions that will change the performance of the network.

Mobility

The initial location of the sensor nodes can change because of several factors. Sensor nodes can be carried by mobile devices or they may have capability to move themselves. In addition to these, environmental factors, like wind or water flow, can change the initial position of the sensor nodes. Mobility has an important impact on the network dynamics and hence influences the networking protocols and algorithms proposed for the design of the sensor network. Sensor nodes can be mobile or immobile considering the equipment and requirements of the sensor network.

Cost, Size, Resources, and Energy

Size of the sensors changes depending on the actual needs of the application; it varies from the size of a shoebox to a microscopically small particle. Costs of the sensors can vary widely considering their properties. Powerful nodes can be required for small sized networks and these cost hundreds of Euros whereas the cost can be only a few cents for very simple sensors. Energy availability and resources for computing, storage and communication are directly related to the size and the cost of the sensor.

Heterogeneity

Nodes may differ in the type and number of attached sensors; some computationally powerful nodes may collect, process and route data from many more limited sensing nodes; some nodes may act as gateways to long range data communication networks. Heterogeneity is important since it affects the management of the whole system.

Communication modality

Common modality is radio waves since they do not require free line of sight, and communication over medium ranges can be implemented with relatively small antennas. Light beams and sound are also used for communication in different applications.

Infrastructure

There are two common forms for infrastructure of the wireless sensor network design which are infrastructure based or ad hoc. In infrastructure based networks, nodes can communicate with only a base station. In ad hoc form, nodes can communicate with each other so that they can send data to the base station (or sink) over other nodes. Deployment of the former has higher cost therefore ad hoc networks are generally preferred.

Network Topology

There are several network topologies like star, tree and mesh considering the design of a sensor network. Topology is very crucial since network characteristics like capacity, latency, and robustness are directly affected by the choice of topology. Moreover, important decisions such as routing and processing of the data sensed should be made according to the network topology.

Coverage

Coverage is about the sensing capability of the wireless sensor network. Therefore it is related with the sensing ranges of the sensor nodes used for monitoring. In a monitoring area, only some regions may be of interest or some specific locations need to be sensed by the sensor nodes. In some cases, area of interest may have to be completely covered by sensors. There are different coverage models discussed by Gosh and Das (2006). These are blanket coverage, barrier coverage and sweep coverage. In barrier coverage, a static arrangement of the sensor nodes which minimizes the probability of undetected penetration is tried be achieved. In blanket coverage, the aim is to arrange the sensor nodes so that total detection area is maximized. Sweep coverage is the dynamic arrangement addressing a balance between maximizing the detection rate and minimizing the number of missed detections per unit area.

Connectivity

The nodes of the sensor network have to be connected in order to forward the sensing information to a base station or a sink node. A network is said to be connected if each sensor can communicate with at least one other sensor and there exists at least one node that can communicate with a sink node. Communication ranges of the sensors are important for the design of a connected network, as they determine the connectivity. In general, two sensors are connected if the distance between the two is less than the minimum of their communication ranges.

Network Size

Size of the network is determined by the size of the area of interest, the number of nodes, sensor characteristics and sensing requirements such as coverage and connectivity.

Lifetime

Lifetime of the network is determined by the properties of the sensor nodes. It can change from hours to years for different applications. Energy efficiency is an important issue for network lifetime. Sources of energy consumption are discussed in detail by Heinzelman et al. (2000).

Quality of Service Requirements

Quality of service aspects to be considered are real time constraints (degree of coverage, required time for reporting data etc.), robustness, resistance and other issues.

2.2 Design Issues in Wireless Sensor Networks

Studies in the literature generally concentrate on the deployment of the sensor nodes. The problem of deployment in wireless sensor networks emerged as the base station location problem for cellular phone networks in early 1990s, as stated in Jourdan and Weck (2004). The problem was to find the optimal location of base stations (transmitters) in order to cover subscribers. This problem is different in some aspects from the wireless sensor network (WSN) planning problems. Sensor nodes in WSNs can also transmit the data to other nodes in addition to their own sensing tasks, therefore sensor nodes need to communicate with each other (connectivity). Base station location problems are similar to facility location network design problems, where location of each facility needs to be determined and the network connecting the facilities must be optimized. In WSN design it is important to consider sensor deployment and network design together since location of the sensors determines the network topology. In this type of problems, sensors are manually deployed across the monitoring area. Random sensor deployment is generally preferred for military applications or inhospitable areas where deployment cannot be done manually as stated in Cardei and Wu (2006). The sensors are deployed over the monitoring area without human, e.g. by dropping from aircraft.

Data routing is another decision in WSN design. Obviously, connectivity is a requirement for data routing. In some of the studies, little or no attention is paid to the communication requirement between sensors. Some of the studies assume that connectivity is achieved if communication range of the sensor is sufficiently larger than the sensing range as in Meguerdichian and Potkonjak (2003). This assumption is not realistic since area to be covered can be disjoint, some physical obstacles like mountains and buildings can block communication. In the literature there are different objectives considered in the optimization of data routing, which are also discussed by Fabregat et al. (2004). Several routing techniques and protocols are addressed by Karaki and Kamal (2004). Algorithms for connectivity are discussed in Watfa (2007).

Another important concept for WSNs is energy efficiency. Sensor nodes are often tiny devices equipped with one or more sensors, one or more transceivers, processing and storage resources. Akyildiz et al. (2002) state that sensors have a small and finite source of energy, and they are limited in computational capacity and memory, therefore it is important to take wireless channel bandwidth limitations and sensors' processing capacities into consideration while minimizing the energy consumed in communication. This is directly related with efficient routing of the data.

Taking all of these into account, we categorize the related studies according to decisions considered. To start with deployment, sensor nodes may be deployed manually or randomly. If the sensor nodes are deployed randomly, there is not a location decision to make. Most of the studies considering given or random deployment deal with energy efficiency problems. Data routing is another decision that must be taken into account. In the literature there are some studies that determine the data flow between sensor nodes. Hence, studies are also categorized as “no routing” and “routing” according to determination of data flows between sensor nodes. These studies are summarized in Table 2.1 and briefly discussed below.

Table 2.1 Characteristics of the studies considering the random or given deployment

	Authors	Title	Properties	Decisions	Objectives	Constraints	Model	Solution Approach
Routing	Heinzelman et al. (2000)	Energy-efficient communication protocol for wireless microsensor networks	Homogeneous	Determine data flow	Minimize global energy usage	Connectivity	-	Adaptive clustering algorithm to distribute the energy load
	Srinivasan et al. (2004)	Optimal rate allocation for energy-efficient multipath routing in wireless ad hoc networks	Heterogeneous	Determine data flow	Maximize lifetime	Connectivity Node capacity	NLP	Heuristics based on flow algorithms proposed
	Xue et al. (2005)	Maximizing lifetime for data aggregation in wireless sensor networks	Heterogeneous	Determine data flow	Maximize lifetime	Connectivity Energy	LP	Tree based approximation algorithm proposed
	Carle and Simplot-Ryl (2004)	Energy efficient area monitoring for sensor networks	Area coverage Heteregenous	Select active nodes	Maximize lifetime	Connectivity	-	Heuristic based on area dominating sets proposed
	Fabregat et al. (2005)	Multi-objective optimization scheme for multicast flows: a survey, a model and a MOEA solution	Heterogeneous	Determine data flow	Multiple objectives	Connectivity	-	A multiobjective genetic algorithm
	Patel et al. (2004)	Energy-efficient capacity-constrained routing in wireless sensor networks	Heterogeneous	Determine data flow	Maximize lifetime	Connectivity Node capacity Energy Link capacity	MIP	Heuristics based on flow algorithms proposed
	Zussman and Segall (2003)	Energy-efficient routing in ad hoc disaster recovery networks	Heterogeneous	Determine data flow	Maximize lifetime	Connectivity Node Capacity	LP	Iterative algorithm relaxing capacity constraint
No Routing	Potkonjak and Slijepcevic (2001)	Power efficient organization of wireless sensor networks	Area coverage Homogeneous	Assignment of a sensor node to disjoint sets	Maximize the number of mutually exclusive sets	K-coverage	-	Simulated annealing based heuristic
	Cardei et al. (2005)	Energy-efficient target coverage in wireless sensor networks	Target coverage Homogeneous	Assignment of a sensor node to sets (not necessarily disjoint)	Maximize number of sets	1-Coverage	IP (set cover)	Math model based heuristic
	Zou and Chakrabarty (2005)	A distributed coverage- and connectivity- centric technique for selecting active nodes in wirelesssensor networks	Target coverage Homogeneous	Select active nodes for both sensing and communicating	Minimize total number of active nodes	Coverage Connectivity	IP (coverage)	Heuristic based on formation of connected set

Given or Random Deployment - No Routing

Potkonjak and Slijepcevic (2001) consider energy efficiency problem for stochastically placed sensor network. They introduce a heuristic that selects mutually exclusive sets of sensor nodes, where members of each set together completely cover the monitoring area. Significant energy savings is achieved by allowing only one of the sets to be active at any time.

The study of Cardei et al. (2005) is very similar to Potkonjak and Slijepcevic (2001). In this study, the objective is to maintaining coverage of the targets as long as possible. Their approach is based on finding maximum number of sets as in Potkonjak and Slijepcevic (2001). The difference is that a sensor node can be included in different sets for different time intervals. Their approach is to schedule the node activation times so that lifetime of the network is maximized. They proposed an integer programming formulation for finding the schedule and a greedy approach using LP relaxation of this integer program.

Zou and Chakrabarty (2005) address the problem of selecting a subset of randomly deployed nodes that are active for both sensing and communication. They propose an integer programming model for minimizing the number of sensors activated subject to the coverage constraint. However, connectivity is not considered in this model. They propose a node selection algorithm based on connected dominating set. The active nodes form a connected dominating set and act as a backbone for both sensing and communication. Energy saving is achieved by reducing the number of active nodes.

Cardei and Wu (2006) provide a survey on contributions addressing energy efficient coverage problems in static wireless sensor networks. They describe coverage formulations for different network requirements such as connectivity and minimum energy together with assumptions and solution approaches.

Given or Random Deployment - Routing

Heinzelman et al. (2000) propose a clustering based routing protocol (LEACH) that minimizes total energy usage by distributing the load to all nodes at different points in time. The objective is to provide a low energy, ad hoc, distributed protocol for given location of sensor nodes.

Srinivasan et al. (2004) focus on an ad hoc network with a set of sources, communication with their destinations using multiple routes. They present a formulation which maximizes lifetime of the network and uses penalty function approach for system constraints. Capacity of the sensor nodes is also taken into consideration in this study.

Xue et al. (2005) address the problem of maximizing the lifetime of the sensor network, given energy constraint on each sensor. Using linear programming, they formulate the problem as a multicommodity flow problem and propose a fast approximate algorithm.

Carle and Simplot-Ryl (2004) propose a method for energy efficient routing based on neighborhood graphs and classical spanning tree algorithms.

Fabregat et al. (2005) consider the multi-objective approaches proposed for optimization of multicast flows. Moreover, they propose a multi-objective evolutionary algorithm inspired by the Strength Pareto Evolutionary Algorithm (SPEA). Mathematical formulations are provided for load balancing techniques and studies considering different conflicting objectives are classified and discussed.

Zussman and Segall (2003) focus on the energy efficient routing problem in disaster networks. The problem is formulated as a routing problem in which the objective is to maximize the lifetime until the first battery drains out. They propose a nonlinear programming model which tries to maximize the minimum lifetime of the sensor nodes subject to flow conservation and processing capacity constraints. An equivalent linear programming formulation is also proposed and tried to be solved by iterative algorithms to find the optimal solution to the problem for distributed implementations.

In Patel et al. (2004), a minimum cost capacity constrained routing protocol is proposed, which minimizes the total energy consumed in routing while guaranteeing that total load on each sensor node and on each wireless link does not exceed its capacity. They consider an integer programming model in order to find optimal data routing for maximizing lifetime with given sensor locations. In the model, variables corresponding to data flow amounts are considered to be integer, but they discuss the impact of relaxing integrality of the data flow amounts on the lifetime of the network. They propose a heuristic approach based on minimum cost flow algorithms to solve the integer flow version of this problem.

When deployment decision is considered, there are some studies where the sensor nodes can be located anywhere over the monitoring area which implies "location on continuous space". Another alternative is to locate the sensors at previously defined locations across the monitoring area which means "location at given possible locations". Some studies focus only on the location decision whereas others simultaneously consider location and routing decisions. Summary of the former studies is provided in Table 2.2 whereas the latter ones are summarized in Table 2.3.

Deployment Decision - Only Location

Chakrabarty et al. (2002) address the grid coverage strategies for effective surveillance and target location in distributed sensor networks. Monitoring area is divided into grids and they are referred to as targets to be detected at any time. An integer linear programming model is proposed for minimizing total cost of sensors for complete coverage of the monitoring area. Large problems are tried to be solved by a divide-and-conquer approach.

Ke et al. (2007) try to solve the sensor deployment problem by dividing sensor field into grids consisting of squares or equilateral triangles. Using this format, they find the minimum number of sensors required to be deployed on grid points in order to construct a wireless sensor network that fully covers the chosen critical grids. Connectivity is also considered in this paper. They conclude that the critical grid coverage problem is NP-Complete.

Table 2.2 Characteristics of the studies considering only location

Authors	Title	Properties	Decisions	Objectives	Constraints	Model	Solution Approach
Chakrabarty et al. (2002)	Grid coverage for surveillance and target location in distributed sensor networks	Target Coverage Heterogeneous 2D and 3D monitoring	Place sensors	-	K-Coverage	IP	Mathematical model solution
Ke et al. (2007)	Constructing a wireless sensor network to fully cover critical grids by deploying minimum sensors on grid points is NP-complete	Target coverage Homogeneous	Place sensors	Place at most k sensors ensuring constraints are not violated	K-coverage Connectivity	-	Approach proposed for solving the problem in reasonable time (using triangular grids)

Table 2.3 Characteristics of the studies considering location and routing

	Authors	Title	Properties	Decisions	Objectives	Constraints	Model	Solution Approach
Continous Space	Jourdan and Weck (2004)	Layout optimization for a wireless sensor network using a multi-objective genetic algorithm	Area coverage Homogeneous	Place sensors Determine data flow	Maximize total coverage Maximize lifetime	Connectivity Energy	-	A multiobjective genetic algorithm
	Wang et al. (2007)	Device placement for heterogeneous wireless sensor networks: minimum cost with lifetime constraints	Heterogeneous	Place relay nodes given location of sensing nodes Determine data flow	Minimize total cost	ConnectivityEnergy	-	Two phased algorithm based on minimum set covering in the first stage and different heuristics in the second stage
	Hou et al. (2005)	On energy provisioning and relay node placement for wireless sensor networks	Heterogeneous	Place relay nodes given location of sensing nodes Determine data flows and energy provision	Max lifetime	Connectivity Energy	MINLP	Math model based heuristic
	Pan et al. (2003)	Topology control for wireless sensor networks	Heterogeneous	Place base stations given location of the sensor nodes Select node for relaying Determine data flow	Max lifetime	Connectivity Energy	-	Two phased algorithm where base stations placed in the forst ohase and relay nodes and data flows determined in the second stage
	Krause et al. (2006)	Near-optimal sensor placements: maximizing information while minimizing communication cost	Heterogeneous	Place sensor nodes Determine data flow	Maximize sensing quality Minimize communication cost	Connectivity	-	Iterative algorithm based on performance obtained from initial deployment of sensor nodes

Table 2.3 continued

	Authors	Title	Properties	Decisions	Objectives	Constraints	Model	Solution Approach
Given Possible Locations	Zongheng et al. (2004)	Connected K-coverage problem in sensor networks	Target coverage Homogeneous	Select the sensor node to be used in network	minimize total number of sensors	K-coverage K-connectivity	-	A greedy algorithm based on set covering
	Patel et al. (2005)	Energy efficient sensor, relay and base station placements for coverage, connectivity and routing	Target Coverage Heterogeneous	Place sensors, relays, base stations Determine data flow	Minimize total cost Minimize total number of sensors Maximize lifetime Maximize utilization	Coverage Connectivity Node capacity Energy Link capacity	MIP	Mathematical models with single objective
	Quintao et al. (2007)	Evolutionary algorithms for combinatorial problems in the uncertain environment of the wireless sensor networks	Target coverage Homogeneous	Place sensors Determine data flow	Minimize (energy consumption + uncovered points)	K-coverage Connectivity Energy	MIP	Solve coverage with genetic algorithm, penalize connectivity with a functio, determine data flow with MST algoritms
	Cheng et al. (2004)	Energy-aware node placement in wireless sensor networks	Homogeneous	Place sensors Determine data flow	Maximize lifetime Minimize application specific total cost (replacing depleted nodes)	Connectivity Energy	NLP	A greedy placement algorithm providing that all nodes run out of energy at the same time
	Pandey et al. (2007)	A hybrid approach to optimize node placements in hierarchical Heterogeneous networks	Heterogenous	Place relay nodes given sensor nodes Determine data flow	Minimize number of relay nodes	Connectivity Data traffic	IP	A greedy and genetic algorithm a hybrid of these approaches
	Chang and Chang (2008)	Energy-aware node placement, topology control and MAC scheduling for wireless sensor networks	Homogeneous	Place relays given sensor nodes Determine data flow Schedule nodes for data flow	Balance energy consumption Avoid packet collision	Connectivity	-	Heuristic algorithm for deployment on a constructed tree based on distance and density for balacing power consumption
	Ferentinos and Tsiligiridis (2006)	Adaptive design optimization of wireless sensor networks using genetic algorithms	Homogeneous Area Coverage	Determine signal range and status of sensor nodes	Minimize energy consumption	Connectivity	-	Genetic algorithm

Deployment Decision - Location and Routing

Continuous Space

Jourdan and Weck (2004) work on the optimization of wireless sensor network layouts. A multiobjective genetic algorithm for sensor deployment problem is proposed where two competing objectives are total sensor coverage and lifetime of the sensor network. It is assumed that the number of sensors to be deployed is given and each gene in a chromosome represents the coordinates of each sensor. Routing of the data is done by using Dijkstra's shortest path algorithm. They also investigate the influence of the ratio between sensing range and communication range on the optimal layout with the best coverage. They state that if this ratio is lower that ½, layout is formed of polygons and crowded whereas hub-and-spoke type layouts become optimal in the opposite case.

Wang et al. (2007) discuss relay node deployment in WSNs. They develop a formulation in which the number and position of sensing points are known. Three types of devices, sensor nodes (SN), relay nodes (RN) and base stations (BS) can be installed at chosen points. Given sensor node locations, they concentrate on connectivity oriented deployment. They state their problems as follows: Given a specific sensing task, determine the number and positions of heterogonous devices so that total network cost is minimized while the constraints of lifetime and connectivity are satisfied. Basic optimization problem is to find a minimum number of RNs and their positions so that each SN can reach at least one RN in a single hop. They solve this problem by considering it as a minimum set covering problem. In the second case, they assume that RNs have a limited energy and fixed transmission range. They pose a two phase RN deployment approach. In the first phase, minimum number of RNs are placed to ensure the connectivity of SNs. In the second phase, additional relay nodes are placed to provide the connectivity of the previously placed RNs to the sink node. For the second phase they present three heuristics based on load balancing considering the capacity of RNs and distance to the sink. In the first algorithm, starting from the RN farthest from a BS, data is routed to closest existing the RN. When existing neighboring RNs cannot handle the traffic load, a new RN is added. In the second algorithm, two different approaches are proposed. In the first one, starting from the RN farthest from the sink node, the workload is distributed to RN’s adjacent neighbors, by first filling up the capacity of the node nearest to BS. In the second approach, workload of a given RN is distributed starting from the one with the maximum residual capacity. In the last algorithm, when a new RN is added, the workload is distributed to other RNs and the newly added node whereas it is only distributed to newly added RN in the previous algorithms. Moreover the location of a new RN is chosen as close to the BS as possible. They evaluate the performance of the proposed algorithms on grid networks and random networks. They conclude that the last algorithm performs best considering their metrics which are the number of RNs added, energy cost and average capacity utilization.

Hou et al. (2005) address the problem of energy provisioning and relay node deployment for a two-tiered wireless sensor network. For some applications, even under optimal flow routing, it may not be able to meet the mission requirements. In such cases energy provisioning may solve the problem of the network lifetime. They propose a mixed integer nonlinear programming model to solve the joint problem of energy provisioning and relay node deployment. In addition to flow decisions, decision of deployment of relay nodes over the monitoring area is also considered. The deployment is done in the continuous manner meaning that a relay can be located at any location on the monitoring area. A decision variable is introduced for energy provisioning so that the energy of a certain number of relay or sensor nodes can be increased during monitoring. Total energy available is known and the aim is to distribute this energy to the nodes in order to increase the lifetime. The model is solved using BARON for small instances, but it is not computationally efficient for larger problems. A two phase heuristic algorithm is proposed. In the first phase, locations of the relay nodes are found by a heuristic algorithm. In the second phase an LP is solved for the energy provisioning problem since the energy provisioning problem turns out to be an LP with given locations of relay nodes.

Pan et al. (2003) consider a two-tiered wireless sensor network consisting of sensor clusters deployed around strategic locations. The aim is to locate base stations in a continuous manner over the monitoring area. The authors propose approaches to maximize network lifetime, by arranging the location of the base station and relaying between sensor nodes and base station. After locating base station, flow assignments are determined.

Krause et al. (2006) focus on a unified approach for deploying sensor nodes. They try to optimize location of sensor nodes using expert knowledge obtained by initial deployment. They propose a polynomial time algorithm which deploys sensors at informative and cost effective locations. They introduce a temperature measurement example where the quantity of information is taken as the expected amount of temperature change that can not be sensed.

Given possible locations

Zongheng et al. (2004) address the problem of constructing a minimum size connected network with K-coverage. This means selecting a set of sensors such that each point is covered by at least K different sensors and sensors are connected. The idea is to keep the minimum set of sensors active to provide the necessary coverage and connectivity, resulting in an energy conservation technique. They propose a greedy algorithm considering the number of times each point is covered by the same sensors in the selected set. The greedy algorithm returns a connected set. A distributed version of the algorithm provides larger size solutions.

Patel et al. (2005) consider a wireless sensor network with sensor nodes capable of sensing and communication, relay nodes capable of communication, and base stations responsible for collecting data generated by sensor nodes. They consider the problem of placing sensor nodes, relay nodes and base stations in the monitoring area such that each point of interest in the monitoring area is covered by a subset of sensors of desired cardinality (K-coverage), the resulting sensor network is connected and the sensor network has sufficient bandwidth. Several deployment strategies are proposed to determine optimal deployment of sensor nodes, relay nodes, and base stations for guaranteed coverage, connectivity, bandwidth and robustness. Different objectives are considered such as minimizing the number of sensor nodes deployed, minimizing the total cost, minimizing the energy consumption, maximizing the network lifetime and maximizing the network utilization. The problems for reliable as well as probabilistic detection models are formulated as integer linear programs. In addition to deployment decision, data routing is also considered. Capacity of the nodes is also taken into account. Problems are solved using CPLEX by introducing an upper bound on the number of base stations and relay nodes to be deployed. Solution characteristics, deployment properties are discussed in this study.

Quintao et al. (2007) address the problem of activating the minimal number of nodes for maintaining coverage and connectivity with network lifetime consideration. They state that this problem is known as the coverage and connectivity problem in wireless sensor networks and can be modeled as a mixed integer linear programming. They propose a mathematical model but try to solve it by a two phase heuristic solution method because the exact solution requires high computational effort. They pose the problem as follows: given a monitoring area, a set of demand points, a set of sensor nodes and a sink node, assure that at least n sensor nodes cover each demand point in the monitoring area, there is a path between these nodes and the sink node, and battery energy of the activated sensor nodes is not depleted. Connectivity is taken into consideration as stated in the second part of the problem but data routing is not considered. It is assumed that energy consumption for receiving and transmitting is independent of the data amount.

Quintao et al. (2007) decompose the problem into two subproblems. The first problem, finding the minimal number of nodes needed to cover all demand points, is solved by a genetic algorithm . In the second phase the best solution found in the first phase is modified to ensure the connectivity between active nodes. In the genetic algorithm, binary encoding is used to represent the activated nodes. Fitness is taken as the number of uncovered points together with total cost of the paths from all nodes to the sink node. They try to consider both energy efficiency and coverage. Using solutions obtained from the genetic algorithm, they apply Prim's minimum spanning tree algorithm. The condition for connectivity is taken as follows: two nodes can communicate with each other if distance between them is shorter than the maximum communication range of the nodes, which is not generally the case for

WSN applications. Given this condition, some active nodes may be disconnected from the tree constructed. Dijkstra's shortest path algorithm is applied starting from each disconnected node; and some inactive nodes on the path found are also activated. They compare their results with optimal results obtained by solving the mathematical model using CPLEX. The deviation of their heuristic approach from optimal is nearly 20% with better run times. Important issues like sensor capacities and data routing are not considered in this study.

Cheng et al. (2004) formulate a constrained nonlinear programming problem to determine both locations of the sensor nodes and data flow between the nodes considering two objectives: maximize network lifetime and minimize total application cost. A heuristic approach is proposed to solve this nonlinear program with single objective. They claim that for a given time horizon, both objectives can be considered by minimizing the total power consumption.

Pandey et al. (2007) consider the problem of placing the minimum number of relay nodes to handle the traffic of previously deployed sensor nodes. The problem is formulated as an optimization problem and three different approaches are proposed to solve the problem. In the first one, the problem is modeled as a binary integer linear programming model without connectivity constraints, and the solution is modified by a greedy Steiner tree algorithm to have a connected network. A greedy deployment algorithm based on clustering and a genetic algorithm are also proposed. The constraint violations are penalized in the fitness calculation. They have also considered hybridizing these algorithms.

Chang and Chang (2008) propose efficient node deployment, topology control and scheduling mechanism to prolong the sensor network lifetime, balance power consumption of sensor nodes, and avoid collision. Topology is first constructed based on grid based WSNs. Then, two different sensor node deployment schemes trying to balance the power consumption of sensor nodes are applied. Finally, a scheduling protocol is used to avoid packet collision.

Ferentinos and Tsiligiridis (2006) focus on a multi-objective optimization method for self-organizing, adaptive wireless sensor network design and energy management. They propose a genetic algorithm with a fitness function incorporating different objectives of the network optimization problem. The decision is the status of the sensor nodes deployed (active or inactive) and the signal range of the active sensors (high or low). Their GA tries to optimize sensor activation and range selection from the set of distributed sensor nodes on the given grid layout of the monitoring area. Although this seems to be a problem where sensor locations are given, activation decision and range selection are analogous to deployment of two types of sensors. They also consider the dynamic version of the problem. In the dynamic case, sensor nodes activated in GA solution work for some time and then battery energy of the sensor nodes are updated. Then a new GA run provides a new solution which will work for some given period.

2.3 Discussion

The studies in the literature and their characteristics are summarized in Table 2.4 according to the decisions, objectives and constraints they consider. About half of the studies take both location and routing decisions into consideration. Network cost, lifetime (or energy efficiency) and coverage may be objective or constraint. For example, it is possible to maximize lifetime or coverage subject to the number of sensors available (or cost). One may also minimize cost subject to coverage or energy (battery life) constraints. The studies are generally limited with single objective formulations. However the problem of energy efficient coverage and connectivity of WSN has a multiobjective nature. Several objectives are discussed in Fabregat et al. (2005) and Patel et al. (2005). Taking these into account, we try to handle two conflicting objectives, minimization of network cost and maximization of lifetime, together. We try to make both location and routing decisions under these two objectives.

Connectivity and coverage requirements together with application specific constraints are not taken into consideration explicitly in most of the studies. In Patel et al. (2004), it is emphasized that wireless channel capacity and finite sensor capacities should be taken into consideration in order to prevent routing of the data packets over highly congested links and paths since congestion increases the delay and packet losses, which will increase the energy consumption because of retransmission of the packets. Moreover most studies deal with homogeneous wireless sensor network design. In our study, we consider heterogeneous wireless sensor network design by taking application specific constraints such as connectivity, coverage, node capacity and link capacity into account.

Our study takes all of these aspects into account simultaneously for WSN design. As stated by Cheng et al. (2004), analyzing the impact of data sampling, aggregation techniques and node deployment on network lifetime and power consumption is a challenging research direction in a more general network with nonuniform data. We try to investigate the tradeoff between cost and lifetime objectives while deciding on sensor deployment and data routing. We use a probabilistic coverage model for the detection of the targets. We consider locating sensors at given possible locations resulting in an ad hoc network and try to model the data communication under the connectivity, coverage, node capacity and link capacity constraints.

Table 2.4 Summary of the characteristics of the studies in the literature and our study

	Decisions			Objectives (O) and/or constraints (C)						
Study	*Location*	*Routing*	*Other*	*Cost*	*Lifetime (energy efficiency)*	*Connectivity*	*Coverage*	*Node Capacity*	*Link Capacity*	*Other*
Potkonjak and Slijepcevic (2001)			✓		O		O			
Cardei et al. (2005)			✓		O		O			
Zou and Chakrabarty (2005)			✓			C	O			
Heinzelman et al. (2000)		✓			O	C				
Srinivasan et al. (2004)		✓			O	C		C		
Xue et al. (2005)		✓			O	C				
Carle and Simplot-Ryl (2004)		✓	✓		O	C				
Fabregat et al. (2005)		✓				C				O/C
Zussman and Segall (2003)		✓			O	C		C		
Patel et al. (2004)		✓			O	C		C	C	
Chakrabarty et al. (2002)	✓			O			C			
Ke et al. (2007)	✓			O		C	C			
Jourdan and Weck (2004)	✓	✓			O	C	O			
Wang et al. (2007)	✓	✓		O	O	C	C			
Hou et al. (2005)	✓	✓	✓		O	C				
Pan et al. (2003)	✓	✓			O	C				
Krause et al. (2006)	✓	✓		O		C				O
Zongheng et al. (2004)			✓	O		C	C			
Patel et al. (2005)	✓	✓		O/C	O	C	C	C	C	
Quintao et al. (2007)	✓	✓		O		C	C			
Cheng et al. (2004)	✓	✓		O	O	C				
Pandey et al. (2007)	✓	✓		O		C				C
Chang and Chang (2008)	✓	✓	✓		O	C				C
Ferentinos and Tsiligiridis (2006)	✓				O	C	C			
Our study	✓	✓		O	O	C	C	C	C	

CHAPTER 3

PROBLEM FORMULATION

The problem of energy efficient coverage and connectivity in wireless sensor networks is formulated mathematically. After stating our assumptions and introducing the notation of the mathematical model, we propose two formulations. We then modify these formulations to obtain the Pareto optimal solutions for the bicriteria problem.

3.1 Assumptions

Before going into detail of the formulations, the assumptions made about the system are stated as follows.

1. A wireless sensor network can have different types of sensor (or relay) nodes and a base station.
2. Base station or sink is responsible for gathering the data generated at the sensor nodes and it has sufficient power supply.
3. Relay nodes are responsible of transmitting data from one to another. They do not sense, therefore they do not generate data to be routed.
4. Sensor nodes can have different sensing, communication and process characteristics (capacities).
5. Sensor nodes are deployed in an ad hoc basis and they are immobile.
6. The data is transmitted from one node to another over a wireless channel. MAC (Medium Access Control) protocol determines the mean rate at which a sensor node can transmit data to its neighbor. This rate is the channel or link capacity.
7. All sensor types have the same battery energy. (This assumption is relaxed in the genetic algorithm)

8. Possible sensor locations are fixed and known. For convenience, the monitoring area is divided into grids and possible locations at which sensor nodes can be located are taken as these grids.
9. Targets to be sensed are located at random points over the monitoring area,

3.2 Notation

Indices:

i, j	Location index, $i, j = 1, \ldots, n$.
k	Sensor type index, $k = 1, \ldots, s$, $s+1$ is the relay.
p	Point (target) index, $p = 1, \ldots, m$.

Parameters:

$dist_{ij}$	Euclidean distance between locations i and j.
c_k	Cost of locating a sensor of type k.
sr_k	Sensing range of a sensor of type k.
cr_k	Communication range of a sensor of type k.
cap_k	Capacity of a sensor of type k.
d_p	Rate of data generated at point p.
pr_{ikp}	Probability of sensing point p with a sensor of type k located at location i.
h_p	Coverage probability threshold for sensing point p.
eg_k	Energy required to generate unit data (sense) with a sensor of type k.
et_{ij}	Energy required to transmit unit data from location i to location j.
er_{ij}	Energy required to receive unit data from location i to location j.

cap_{ij}	Capacity of wireless communication link (i, j).
e_k	Battery energy of a sensor of type k.
M	A very large value.

Sets:

NC_{ik}	Set of (location, sensor type) pairs that can communicate with a sensor of type k located at location i.
A_{ik}	Set of points p that are in the sensing range of a sensor of type k located at location i.
B_p	Set of (location, sensor type) pairs that can sense point p.

Decision variables:

$$x_{ik} = \begin{cases} 1, & \text{if a sensor } k \text{ is located at location } i \\ 0, & \text{otherwise} \end{cases}.$$

f_{ikjm}	Rate of data flow from a sensor of type k located at location i to a sensor of type m located at location j.
l	Lifetime of the network.
q	Inverse of the network lifetime to be used in linearization.

3.3 Single criterion formulations

We consider two objectives for the energy efficient coverage and connectivity problem, one of which is to minimize the total sensor cost located and the other is to maximize the network lifetime under system constraints. Before going into detail of the single objective formulations, coverage and connectivity definitions are discussed below.

Connectivity of the network is defined by the communication ranges and locations of individual sensors. In a sensor network, sensor (and relay) nodes must communicate in order to transmit the data gathered to a destination which is generally a base station. A sensor at location i can communicate with another sensor at location j if each sensor is in the communication range of the other. This means that the Euclidean distance between these sensors must be less than or equal to the minimum of the communication ranges of both sensors, as stated by Gosh and Das (2006). In Figure 3.1 (a) the sensor at location i is not within the communication range of the sensor at location j, therefore the two sensors cannot communicate (are not connected). The sensor with larger communication radius must also fall within the communication range of the other so that the sensors can communicate. Figure 3.1 (b) illustrates a case where the two sensors are connected.

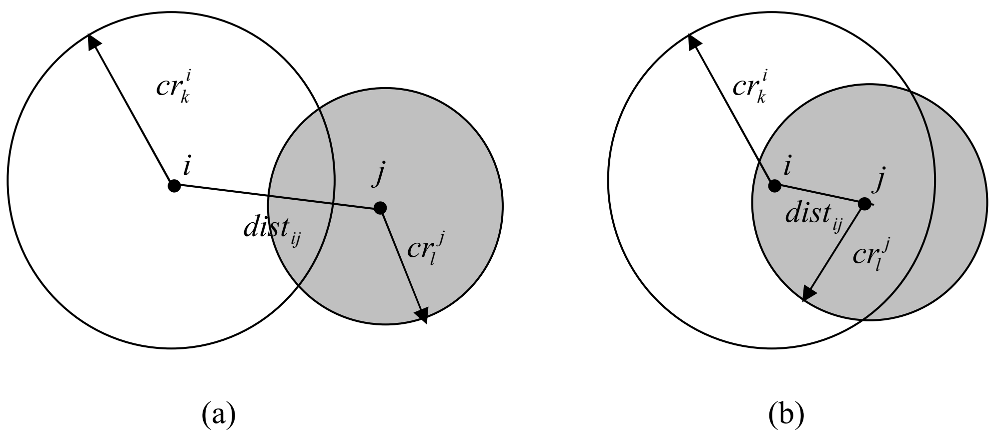


Figure 3.1 (a) Disconnected and (b) connected sensors located at locations i and j

Coverage is a measure of quality of sensing. The main aim in WSNs is to monitor the physical space as well as possible to sense any target. In our problem, the aim is to sense each target point in the physical space of interest with at least some threshold probability. Hence, coverage is defined as the probability that a target at point p is sensed (detected) by a sensor at location i. The detection probability is related with the strength of the sensor signal and the distance between the sensor and the target. As in Zou and Chakrabarty (2005), the following function is used for representing the confidence level in the received sensing signal.

$$pr_{ikp} = \begin{cases} e^{-\beta_k dist_{ip}}, & \text{if dist}_{\text{ip}} \leq sr_k \\ 0, & \text{otherwise} \end{cases}$$

The parameter β_k depends on the sensor characteristics and yields different probabilities for different sensor types at the same location. Note that the longer the distance is, the lower the confidence level or the probability of detection is. Points beyond the sensing range of the sensor are considered too noisy and the sensor cannot detect the targets at larger distances. A point can fall in the sensing range of several sensors. The probability of detection of a target at point p is then calculated as follows.

$$pr_p = 1 - \prod_{(i,k) \in B_p} \left(1 - pr_{ikp}\right)^{x_{ik}}$$

Overall coverage of point p, pr_p, is taken as the complement of the probability that point p is not sensed (detected) by any of the sensors that can sense point p, which means point p is covered jointly by its neighboring sensors.

Overall coverage of each point p must be greater than a threshold h_p. This threshold can be interpreted as the importance of the target located or the event occurring at that point. Hence, in our probabilistic coverage model, the coverage constraint can be defined as follows.

$$1 - \prod_{(i,k) \in B_p} \left(1 - pr_{ikp}\right)^{x_{ik}} \geq h_p$$

This is a nonlinear inequality but we can obtain a linear inequality by taking the logarithm of both sides.

$$\sum_{(i,k) \in B_p} \ln(1 - pr_{ikp}) x_{ik} \leq \ln(1 - h_p)$$

Cost minimization

After introducing the above definitions, we present our first single criterion formulation for the problem of minimizing total sensor placement cost denoted as PC1 (**P**roblem with **C**ost objective, version **1**).

Problem PC1

Min $\sum_{i=1}^{n}\sum_{k=1}^{s+1} c_k x_{ik}$ (1)

s.t. $\sum_{k=1}^{s+1} x_{ik} \le 1,\ \forall i$ (2)

$$\sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{ikjm} - \sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{jmik} = \sum_{k=1}^{s+1}\sum_{p\in A_{ik}} d_p x_{ik}\ ,\ \forall i \quad (3)$$

$$\sum_{(i,k)\in NC_{\text{sink}}} f_{ik,\text{sink}} = \sum_{i=1}^{n}\sum_{k=1}^{s+1}\sum_{p\in A_{ik}} d_p x_{ik} \quad (4)$$

$$f_{ikjm} \le Mx_{ik}\ ,\ \forall i,k,(j,m)\in NC_{ik} \quad (5)$$

$$f_{ikjm} \le Mx_{jm}\ ,\ \forall j,m,(i,k)\in NC_{jm} \quad (6)$$

$$\sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{ikjm} + \sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{jmik} \le \sum_{k=1}^{s+1} cap_k x_{ik}\ ,\ \forall i \quad (7)$$

$$\sum_{k=1}^{s+1}\sum_{m=1}^{s+1} f_{ikjm} \le cap_{ij}\ ,\ \forall i,j \quad (8)$$

$$\sum_{(i,k)\in B_p} \ln(1-pr_{ikp})x_{ik} \le \ln(1-h_p)\ ,\ \forall p \quad (9)$$

$$x_{ik} \in \{0,1\}\ ,\ \forall i,k \quad (10)$$

$$f_{ikjm} \in Z^+, \forall i,k,(j,m)\in NC_{ik} \quad (11)$$

Objective function (1) represents the total cost of locating sensors and relays. Constraint set (2) ensures that at most one type of sensor is placed at each location. Set (3) provides data flow conservation at each sensor node located. Set (4) guarantees that all data packets are sent to the sink node. Constraint sets (5) and (6) guarantees that data flow can be realized between two sensors only if they are located at the given locations. Set (7) ensures that data processing capacity of the sensor node is not exceeded. Set (8) is the channel capacity constraint which restricts the amount of data packets to be transmitted over a link. Set (9) provides that coverage thresholds are met. Set (10) represents the binary location decision and (11) ensures that flow values are integer.

Although our formulation assumes point or target coverage, area coverage can be approximated by choosing points to be sensed such that they represent the monitoring area uniformly. The grid structure representing possible sensor locations can be used for this purpose, provided that the distance between grid points is less than the minimum sensing range given in the problem. Therefore, the formulation given above is capable of handling area coverage as well.

PC1 resembles the location and flow allocation problems known in the operations research literature with one exception. Each target or point p to be sensed generates data at a constant rate, d_p, similar to the demand in the location and flow allocation problems. However, every sensor close enough to sense point p receives d_p and tries to transmit it to the base station. That is, the same data is generated over and over again by multiple sensors that can sense the same point. Therefore, the total flow in the WSN is in general not constant but increases as the number of sensors increases. This is why relays are needed in WSNs. Relay nodes do not sense (do not generate new flow) but only relay data from one sensor to another. They provide connectivity without increasing the total flow, thereby preventing infeasibility.

Lifetime maximization

Network lifetime is another consideration. Lifetime is highly dependent on the power supply of the sensors. There are various means of power supply such as battery and solar panel. Sensors are assumed to use batteries as power supplies in our study. A sensor consumes energy for generating data from targets, receiving from other sensors and transmitting to the other sensors. We adopt the communication power consumption model used by Heinzelman et al. (2000). The energy required for generating and receiving data is constant per unit data; however it is not constant for transmission. In the adaptive transmission power model, energy required for transmitting unit data increases with distance. In order to send the data to a sensor over longer distances, an acceptable signal to noise ratio should be achieved by consuming more energy. Energy consumed in transmitting a bit from location i to j is given as

$$et_{ij} = \delta + \lambda \times dist_{ij}^{m}$$

where δ is a distance independent constant parameter and λ is a coefficient associated with the distance dependent term. $dist_{ij}$ is the distance between two locations and m is the path loss index.

Considering these definitions, maximization of lifetime problem denoted as PL1 (**P**roblem with **L**ifetime objective, version **1**) can be formulated as follows.

Problem PL1

$$\max \quad l \tag{12}$$

$$\text{s.t.} \quad \sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} et_{ij} f_{ikjm} l + \sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} er_{ji} f_{jmik} l + \sum_{k=1}^{s+1} \sum_{p\in A_{ik}} d_p eg_k x_{ik} l \le \sum_{k=1}^{s+1} e_k x_{ik} \ , \forall i \tag{13}$$

(2) - (11)

Objective function (12) maximizes the lifetime of the network. Constraint set (13) ensures that battery energy of a sensor is not exceeded. The first term of the constraint represents the total energy consumed in transmission, the second term stands for the total energy consumed while receiving, and the last term is the total energy consumed by the sensor for generating the data during the lifetime of the network.

The constraint on total energy consumption of a sensor node is nonlinear since it involves the product of lifetime and location or flow variables. In order to linearize this constraint, both sides are divided by l and the constraint turns out to be

$$\sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} et_{ij} f_{ikjm} + \sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} er_{ji} f_{jmik} + \sum_{k=1}^{s+1} \sum_{p\in A_{ik}} d_p eg_k x_{ik} \le \frac{1}{l} \sum_{k=1}^{s+1} e_k x_{ik} \qquad (14)$$

This is not enough to linearize the constraint, since there is a nonlinear term left on the right hand side of (14). Let $q = \frac{1}{l}$ and assume that battery energy e_k is the same for all sensor types and equals e. Then we can replace the right hand side with eq since we know that at most one of the binary variables can be equal to one. Since we have used $q = \frac{1}{l}$ for linearization purposes, the problem of lifetime maximization turns out to be minimization of q. The linearized problem is as follows.

Problem PL1

$$\min \quad q \qquad (15)$$

$$\text{s.t.} \quad \sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} et_{ij} f_{ikjm} + \sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} er_{ji} f_{jmik} + \sum_{k=1}^{s+1} \sum_{p\in A_{ik}} d_p eg_k x_{ik} \le e.q\ ,\ \forall i \qquad (16)$$

(2) - (11)

The assumption of equal battery energy for all sensor types is not reasonable since different sensors can have different battery energy levels in a heterogeneous network. Locating sensors with higher battery energy level at the locations closer to the sink and using low energy ones for the farther areas is more reasonable considering the problem characteristics. However, in order to solve the problem exactly, we have to linearize the constraint with this assumption. If the network is heterogeneous in terms of energy levels, we can set e as the minimum of e_k's which yields a conservative solution. Note that we relax this assumption in our genetic algorithm.

Relaxing integrality of flow variables

In our formulations flow variables representing the number of data packets transmitted per unit time are positive integers. Relaxing integrality of flow variables will naturally result in a longer lifetime. The effect of this relaxation is also discussed by Patel et al. (2004). The important point is that the difference between the lifetimes computed using integer and continuous flow variables should not be significantly high.

Patel et al. (2004) conclude that the tradeoff between inaccuracy of lifetime and increased complexity due to integrality should be examined carefully. The decision of using integer or continuous flow variables should be made after a detailed analysis of node capacities and implementation issues. They also mention a technique known as data scaling to find the optimal lifetime to any desired degree of accuracy when integrality of flow variables is relaxed. Patel et al. (2004) state that detailed information for data scaling is provided by Ahuja et al. (1993). The main point is that, when the flow variables have a large order of magnitude, the error in lifetime due to relaxation becomes negligible.

Based on the discussion by Patel et al. (2004), we relax the integrality of flow variables in both PC1 and PL1 in order to reduce the number of integer variables and therefore the solution time.

3.4 Formulations for Bicriteria Solution

Maximization of lifetime and minimization of total sensor cost are conflicting objectives as can be seen from the formulations. Lifetime of the network can be increased by deploying more sensors so that workload and energy consumption per sensor can be reduced. This will obviously increase the total sensor cost. There will probably be more than one efficient (nondominated) solution for this problem because of its bicriteria nature. These efficient solutions are called Pareto optimal solutions and they can be found by the ε-constraint approach proposed by Haimes et al. (1971). We give the definition of the domination as follows: A solution $\left(z_1^{'}, z_2^{'}\right)$

dominates $\left(z_1^{''}, z_2^{''}\right)$ if $z_1^{'} < z_1^{''}$ and $z_2^{'} \geq z_2^{''}$ or $z_1^{'} \leq z_1^{''}$ and $z_2^{'} > z_2^{''}$ where z_1 is the cost objective and z_2 is the lifetime objective.

In this approach, one of the criteria is taken as a constraint and some parts of the criteria space are restricted with the help of this constraint. In order to generate all efficient solutions for our problem, PL1 can be solved with the cost constraint. Also, the lifetime objective is augmented with a small positive coefficient, ρ, times the value of the total cost to exclude weakly efficient but inefficient solutions. This version of the lifetime formulation is called PL2. Optimal lifetime can be found by solving PL2 iteratively with different cost levels. Formulation of PL2 is as follows.

Problem PL2

$$\min \quad q + \rho \sum_{i=1}^{n} \sum_{k=1}^{s+1} c_k x_{ik} \tag{17}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \sum_{k=1}^{s+1} c_k x_{ik} \leq \varepsilon_t \tag{18}$$

(2) - (11), (16)

where ε_t is an upper bound on the total sensor cost in iteration t and $\rho > 0$ is sufficiently small. Constraint (18) ensures that the total sensor cost does not exceed the given upper bound ε_t. Suppose that we find a lifetime l for a certain total cost level C_1. It is possible that the lifetime cannot be improved and is still l when a larger cost level C_2 is introduced. In such a case, even if we set the upper bound for cost to $\varepsilon_t = C_2$, the left hand side of (18) will take a value equal to C_1 with the help of the augmentation in objective function (17).

Assuming integer values for c_k, we have a finite number of levels for ε_t. The minimum value that ε_t can take can be found by simply solving PC1. In order to find the maximum value for ε_t, a new problem called PC2 must be defined to minimize cost under the lifetime constraint.

Problem PC2

$$\min \quad \sum_{i=1}^{n}\sum_{k=1}^{s+1} c_k x_{ik} \tag{1}$$

$$\text{s.t.} \quad \sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} et_{ij} f_{ikjm} + \sum_{k=1}^{s+1} \sum_{(j,m)\in NC_{ik}} er_{ji} f_{jmik} + \sum_{k=1}^{s+1} \sum_{p\in A_{ik}} d_p eg_k x_{ik} \le e.q_{given}, \ \forall i \tag{19}$$

(2) - (11)

Constraint set (19) guarantees that all sensors located must operate for some given lifetime $\frac{1}{q_{given}}$. Let q_{given} be the objective value found by solving PL1. It corresponds to the maximum possible lifetime regardless of any cost considerations. Then, the maximum value of ε_t for PL2 can be found by solving PC2 with this q_{given}. The objective value of PC2 solved with the maximum lifetime of PL1 gives the maximum value that ε_t can take.

Efficient solutions can then be found by the algorithm given in Figure 3.2.

Step 1. Solve PC1, let the resulting objective value be $\text{cost}_{\min}$ (minimum cost)

Step 2.1. Solve PL1, let the resulting objective value be $q_{\min}$ (maximum lifetime)

Step 2.2. Solve PC2 by letting the bound on lifetime be $q_{given} = q_{\min}$, let the resulting objective be $\text{cost}_{\max}$

Step 3. Solve PL2 for small incremental cost values between $\text{cost}_{\min}$ and $\text{cost}_{\max}$ to find Pareto optimal solutions

Figure 3.2 Algorithm for finding efficient solutions by ε-constraint approach

Another method to solve bicriteria problems is the two phase method proposed by Ulungu and Teghem (1994). They use this approach to solve the bicriteria knapsack problems. In this method, the idea is to find the supported efficient solutions by solving a single objective problem with appropriate weight vectors in the first phase. When we find the solutions with highest cost and lifetime and lowest cost and lifetime, we can formulate a single objective problem consisting of both objectives with appropriate weights. The solutions that can be found by solving this single objective problem are called supported efficient solutions. In the second phase, the unsupported efficient solutions are found by problem specific methods such as using bounds and reduced costs. We have also considered this approach but not implemented it, since it needs iterative use of the single objective method which does not guarantee to find all efficient solutions and requires problem specific approaches to find all efficient solutions.

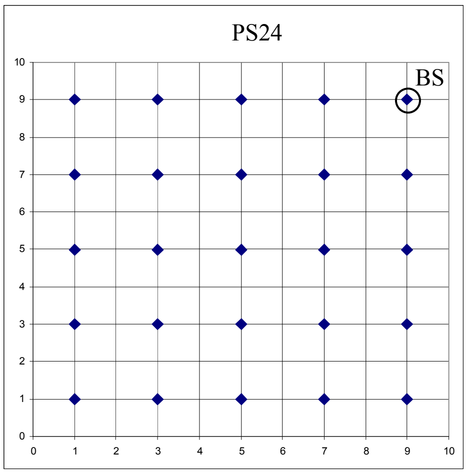
CHAPTER 4

EXACT SOLUTION RESULTS OF THE BICRITERIA PROBLEM

4.1 Problem Generation

There are no benchmark problem instances in the literature in accordance with the system characteristics and assumptions we use. Hence, we determine the instance characteristics considering the problem settings discussed in the literature. Some of the properties such as sensor characteristics and energy consumption parameters are determined by analyzing the studies in the literature. Monitoring area properties are determined by doing some preliminary analysis on the problem parameters.

There are two problem categories considered according to the grid representation of the monitoring area. A monitoring area of size 10 m $\times$ 10 m is represented by 5 $\times$ 5 = 25 locations in the first category. We divide the same area into 41 locations in the second category. For both problem sizes, we assume that a single base station is located at the upper right corner of the monitoring area. Hence 24 and 40 possible locations are left for locating the sensor or relay nodes in two problem categories respectively. The problem categories will be referred to as PS24, PS40. The grid locations are illustrated in Figure 4.1.



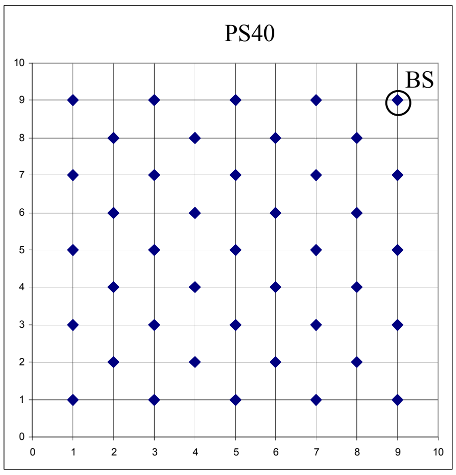


Figure 4.1 Possible grid location structures

Sensor and relay properties are taken as in Table 4.1. The communication range of the base station is taken as 6 m. We disperse 50 targets to be sensed across the monitoring area where coordinates of the targets are generated randomly. Each target has a random coverage threshold uniformly distributed between 0.7 and 1. The rate of data generated from each target is a random integer between 1 Kbps and 3 Kbps. The channel capacity of each wireless link is 150 Kbps.

Table 4.1 Sensor and relay properties for the loose capacity (LC) case

	Sensor type $k=1$	Sensor type $k=2$	Relay $k=3$
cap_k	100 Kbps	200 Kbps	150 Kbps
sr_k	2 m	3 m	0 m
cr_k	3 m	5 m	3 m
c_k	1	2	1
β_k	0.15	0.1	-

Considering the targets, the total data generation rate in the network will be 100 Kbps on the average, assuming that each target is covered by only one sensor. We know that some targets will be covered by multiple sensors because of the coverage threshold requirements, and this will increase the total data volume received by the base station. Selection of sensor characteristics, channel capacity and data generation rates results in instances whose channel and node capacity constraints are loose. However, we adopt most of these properties from Patel et al. (2005). We also consider the tight capacity constrained case by setting the sensor properties as in Table 4.2 and the channel capacity as 40 Kbps for the same set of problems. The problem types are categorized as tight capacity *(TC),* and loose capacity *(LC)*. We expect that the LC problem instances will take longer to solve than the TC instances, as there will be more alternative feasible solutions in the former case.

Table 4.2 Sensor and relay properties for the tight capacity (TC) case

	Sensor type $k=1$	Sensor type $k=2$	Relay $k=3$
cap_k	40 Kbps	80 Kbps	40 Kbps
sr_k	2 m	3 m	0 m
cr_k	3 m	5 m	3 m
c_k	1	2	1
β_k	0.15	0.1	-

The energy consumption model parameters are as in Table 4.3.

Table 4.3 Energy consumption model parameters

eg_k	10^{-8} EnergyUnits
et_{ij}	$(10+0.1\times dist_{ij})\times 10^{-8}$ EnergyUnits
er_{ij}	10^{-8} EnergyUnits
e	10^{-5} EnergyUnits

We generate 30 different problem instances. These instances differ from each other in terms of random target characteristics. Coordinates of the targets and the generation rate of each target are different for each instance.

4.2 Results for Small Problems

We summarize CPU time information in Table 4.4 and solution characteristics in Table 4.5. We were able to find the Pareto optimal solutions for all 30 PS24 instances (with both loose and tight capacity constraints) within reasonable CPU time. PS24-LC instances take about 20 minutes on the average and the maximum CPU time for these problem instances is 80 minutes. The average time for PS24-TC is about one minute and maximum time is under 3 minutes.

CPU times are acceptable for PS40-TC case. It is about 3.5 hours on the average and it takes over 11 hours for the problem instance with maximum CPU time. However, PS40-LC instances take too long. Therefore, we were able to obtain exact Pareto optimal solutions for only 10 instances. Finding these solutions takes about 28 hours on the average and as long as 265 hours for one of the 10 instances. For the remaining 20 instances, we use an approximation. We set a 10% optimality gap and a time limit of three hours for each cost level in the ε-constraint approach. Therefore, the best solution found for each cost level is reported when 10% gap is reached or three hours of CPU time is exceeded, whichever occurs first. This approach may yield dominated solutions. After elimination of dominated solutions, we take the remaining ones as the approximation of the Pareto optimal front. The results for these instances are also included in Table 4.4 and Table 4.5.

Table 4.4 CPU times for small problems

			CPU time (s)			
Problem size	Constraint tightness	# of problems	Avg	Min	Max	Avg % gap
PS24	LC	30	1118	86	4694	-
	TC	30	70	4	167	-
PS40	LC(1)	10	100088	15137	953945	-
	LC(2)	20	85983	50870	144746	15.23
	TC	30	12865	1544	40449	-

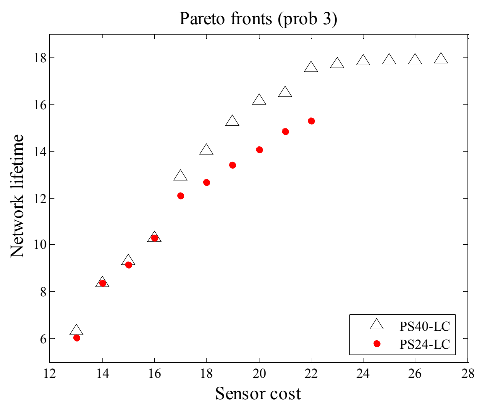
(1)Results across 10 instances that are solved exactly.
(2)Results across 20 instances that are solved approximately.

Table 4.5 Solution characteristics for small problems

			# of Pareto optimal solutions				
Problem size	Constraint tightness	# of problems	Avg	Min	Max	Avg of cost range	Avg of lifetime range
PS24	LC(1)	10	9.20	6	12	14.2 - 23.0	6.3 - 14.6
	LC(2)	20	10.60	8	14	14.3 - 24.4	6.2 - 15.6
	TC	30	7.47	4	12	19.1 - 25.6	9.9 - 13.7
PS40	LC(1)	10	13.60	11	17	13.5 - 26.1	5.5 - 18.1
	LC(2)	20	13.00	9	18	13.8 - 26.4	5.9 - 19.1
	TC	30	11.60	7	16	17.9 - 29.0	9.4 - 18.0

(1) Results across 10 instances that are solved exactly for both PS24 and PS40 cases.
(2) Results across 20 instances that are solved exactly for PS24 case but solved approximately for PS40.

When the solution characteristics in Table 4.5 are examined, the number of Pareto optimal solutions increases as the number of possible locations increases. For both constraint tightness levels, the average number of Pareto optimal solutions found for PS24 instances is about 60% of that found for PS40 instances since the same monitoring area is represented by more possible locations in the latter case. This provides more alternatives and possibility of locating sensors in a more precise manner. We plot Pareto fronts of sample LC and TC problem instances in Figure 4.2 to illustrate the effect of the problem size.



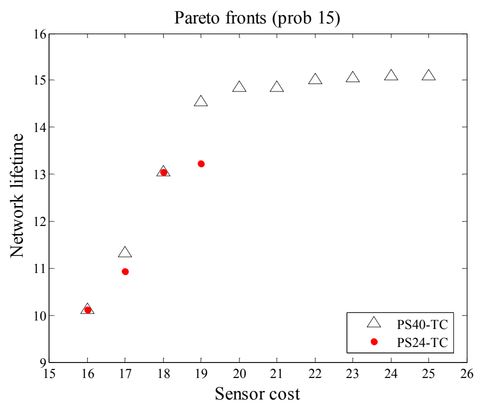


Figure 4.2 Pareto fronts for different problem sizes

For the ranges of cost and lifetime observed in the fronts, similar trends are observed at both loose and tight capacity levels. Minimum cost levels decrease with increasing problem size as given in Table 4.5. The reason is that we have more alternatives for placing the sensors and we can satisfy the given requirements with fewer sensors with more precise sensor locations. Minimum lifetime also decreases with decreasing number of sensors. Maximum lifetime and maximum cost increase with increasing problem size because of the same reason. Increased number of possible locations makes deployment of more sensors possible in the monitoring area to achieve longer lifetimes. Hence, lifetime increases with increasing cost, as expected.

When the capacity constraints are tight, the number of Pareto optimal solutions decreases compared to the loose capacity case for both problem sizes. The reason is the restriction of the solution space. Cost increases with tight capacity since more sensors should be deployed in order to satisfy the capacity requirements. The reason for the decrease in the maximum lifetime is the transmission of data to the base station over long ranges. Data load of the nodes that can directly communicate with the base station is generally high. In tight capacity case, farther nodes cannot transmit to the nodes closer to the base station because of capacity limitations. Instead, they have to transmit directly to the base station although the distance between them is large. This increases the energy consumption and which decreases the maximum lifetime. On the other hand, the minimum lifetime is longer in the tight capacity case compared to the loose capacity case, again because more sensors are deployed. We plot Pareto fronts of a sample problem instance with 24 and 40 locations in Figure 4.3 to show the effect of constraint tightness.

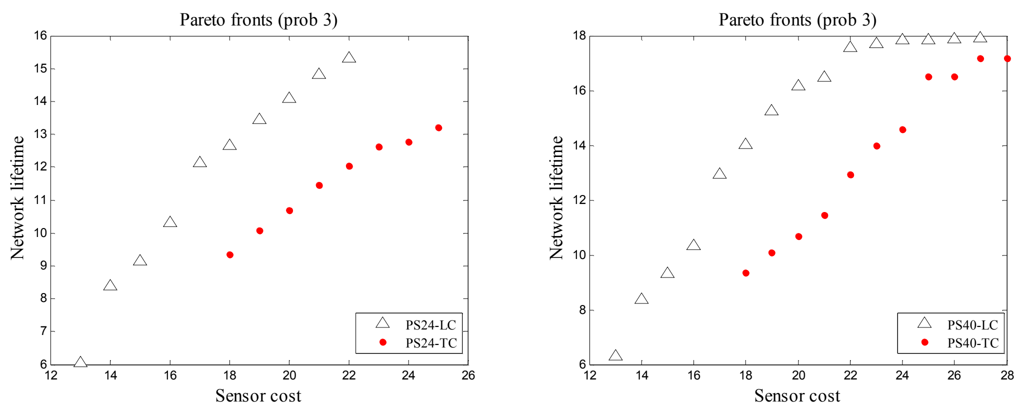


Figure 4.3 Pareto fronts for tight and loose capacity cases

Following are some important characteristics of the solutions in the Pareto front common to all problem categories.

- Higher lifetimes are obtained by placing more relays around the base station. The nodes close to the base station are overloaded with forwarded data which decreases the lifetime of these nodes. Relays located in this area help reduce the load and energy consumption per node, thereby increasing the network lifetime.

- Multiple sensors may be needed to meet the coverage threshold requirement of a target. However, it is important to sense each target with no more than sufficient number of sensors. If a target is sensed by more sensors than required to meet its coverage threshold, multiple sensors sense the same data generated from the target and transmit it over and over again to the base station. This will increase the data volume to be transmitted across the network, which decreases the lifetime. It would be better to have disjoint sensor nodes in the sense that their coverage regions do not overlap, but this might violate coverage thresholds and connectivity requirements. On the other hand, if the sensors are close to each other, the energy required for transmission decreases, which is in conflict with having disjoint sensor nodes.

- Distance of the sensors to the base station is important since energy required for transmission increases as this distance increases. It is better to place the nodes as close to the base station as possible.

CHAPTER 5

GENETIC ALGORITHM FOR THE BICRITERIA PROBLEM

In the previous chapters Pareto optimal solutions for the bicriteria problem are found using the ε-constraint approach. As stated in Deb (2001), classical search and optimization methods such as ε-constraint find a single Pareto optimal solution in each iteration. Finding the entire frontier requires repetitive use of a single objective optimization method, and the computation time is usually unacceptably long. Therefore, a population based metaheuristic approach is more suitable for multiobjective problems. In evolutionary algorithms, we use a population of solutions in every generation, and we find and maintain multiple good solutions for the next generations. Making use of appropriate mechanisms, we can emphasize nondominated solutions in the population and preserve a diverse set of multiple nondominated solutions.

5.1 Basic Approach

There are two decisions to be made as given in our formulations for this problem. The first one is to locate sensors and the second one is to determine the data flow between pairs of sensors. Handling both decisions in a single representation is a fairly complicated task for a genetic algorithm. Therefore, the algorithm is built on the idea that sensor locations are determined through evolution, and data flows are determined by solving the mathematical model with given sensor locations. In addition to two different decisions, constraints that may cause infeasibility during the search process should also be handled effectively. Actually, constraint handling is a major concern for the genetic algorithm proposed to solve this problem. We develop special mechanisms to handle the infeasibilities in the search process.

Deb (2001) states that the idea of nondominated sorting is first proposed by Goldberg in 1989 and implemented by Srinivas and Deb in 1994. We illustrate the nondominated sorting in Figure 5.1 for the bicriteria case.

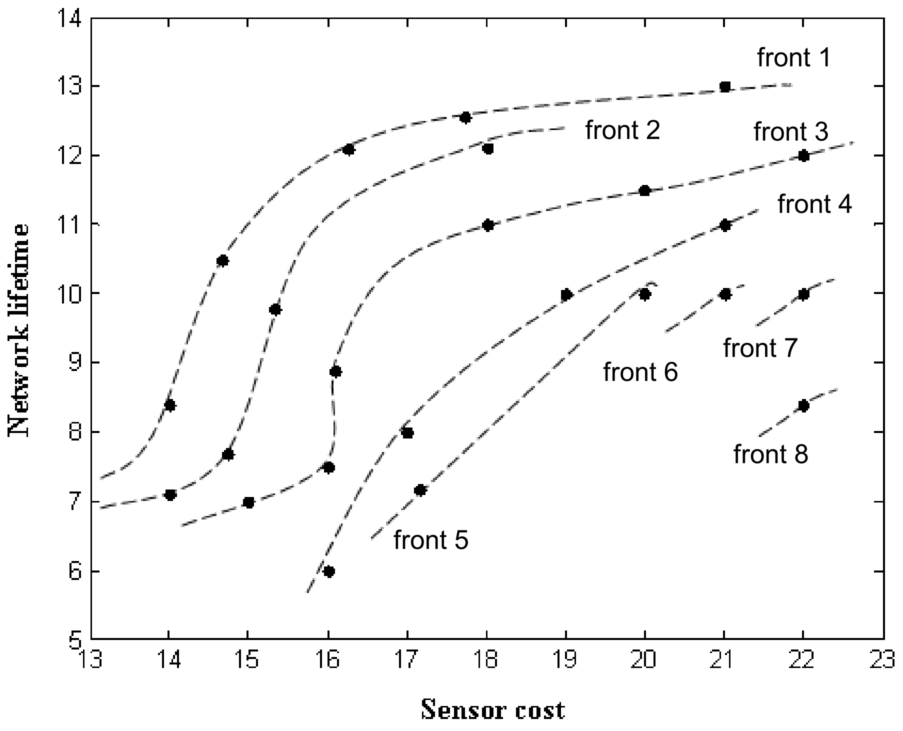


Figure 5.1 Solutions classified into different nondominated fronts

As can be seen from Figure 5.1, the population is classified= into eight fronts. Front 1 contains solutions that are not dominated by any other solutions. Solutions in the second front are found by temporarily disregarding the solutions in the first front. Second front are not dominated by any of the solutions in lesser fronts. This procedure is repeated until all population members are classified. This classification is crucial in fitness assignment. Our GA is similar to NSGA-II (nondominated sorting GA) proposed by Deb (2002). We have some modifications which address our problem specifics.

Taking the mathematical formulation into account, there can be four different types of constraint violation. These are connectivity violation, node capacity violation, channel (link) capacity violation and coverage violation. We use the amount of violation of each type in handling the constraints.

- Connectivity violation: This is taken as the total amount of violation of flow balance constraints (3) and (4). For each sensor node that cannot send data to the other nodes, the flow balance constraint will be violated for the amount of data to be transmitted. The sum of these violation amounts is taken as the connectivity violation.

- Node capacity violation: This is the total violation amount of constraint set (7). For all located sensors, the sum of capacity violations is taken as the node capacity violation.

- Channel capacity violation: The sum of violations for constraint set (8) represents the link capacity violation.

- Coverage violation: This is the violation amount of constraint set (9) in the formulation. The coverage violation is summed up over all targets to be sensed.

In addition to the cost and lifetime objectives, the overall constraint violation is also considered as a third objective to be minimized. The aggregation of different violations to obtain an overall violation will be discussed in Section 5.3.

5.2 Solution representation

Given n possible locations, a solution is represented by a chromosome composed of n genes each of which represents the sensor type assigned to location i. The representation is schematized in Figure 5.2.

1	2	3		i	$i+1$		$n-2$	$n-1$	n
0	**1**	**2**	**-------**	**1**	**3**	**-------**	**0**	**3**	**0**

Figure 5.2 Representation of a solution

In Figure 5.2, a sensor of type 1 is assigned to location 2, a sensor of type 2 is assigned to location 3, and a relay (type 3) is placed at location $i+1$. A gene value of zero means that the respective location is empty. Using this representation, total sensor cost and total coverage violation can be calculated with simple algorithms, but determination of data flow to find the lifetime is not straightforward. In order to determine the data flow, we solve the maximization of lifetime problem with the locations given in the chromosome. When locations are known, maximization of lifetime problem turns out to be a linear program (LP). However, this brings the complexity that an LP must be solved for each offspring to determine the data flow and lifetime. On the other hand, all violation amounts can be obtained by adding slack variables to respective constraints when solving these LPs.

5.3 Fitness calculation

We need to sort the population by taking three objectives into account including the constraint violation. In order to calculate the lifetime objective and violations, maximization of lifetime problem with given sensor locations, PL3, is solved. Before giving the formulation of PL3, we introduce some additional variables.

S_i	Flow balance constraint (connectivity) violation amount for a sensor located at location i.
K_i	Sensor capacity violation amount for a sensor located at location i.
L_{ij}	Channel capacity violation amount for the link between locations i and j.
T_p	Coverage violation amount for point p.

Problem PL3

$$\min \quad q + M\left(\sum_{i=1}^{n+1} \overline{S_i} + \sum_{i=1}^{n} \overline{K_i} + \sum_{i=1}^{n} \sum_{i=1}^{n+1} \overline{L_{ij}} + \sum_{p=1}^{m} \overline{T_p} \right) \qquad (20)$$

s.t. $$\sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{ikjm} - \sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{jmik} - S_i = \sum_{k=1}^{s+1}\sum_{p\in A_{ik}} d_p x_{ik}\ ,\ \forall i \qquad (21)$$

$$\sum_{(i,k)\in NC_{\text{sink}}} f_{ik,\text{sink}} - S_{\text{sink}} = \sum_{i=1}^{n}\sum_{k=1}^{s+1}\sum_{p\in A_{ik}} d_p x_{ik} \qquad (22)$$

$$f_{ikjm} \le M x_{ik} x_{jm}\ ,\ \forall i,k,(j,m)\in NC_{ik} \qquad (23\ )$$

$$\sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{ikjm} + \sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} f_{jmik} - K_i \le \sum_{k=1}^{s+1} cap_k x_{ik}\ ,\ \forall i \qquad (24)$$

$$\sum_{k=1}^{s+1}\sum_{m=1}^{s+1} f_{ikjm} - L_{ij} \le cap_{ij}\ ,\ \forall i,j \qquad (25)$$

$$\sum_{(i,k)\in B_p} \ln(1-pr_{ikp})x_{ik} - T_p \le \ln(1-h_p)\ ,\ \forall p \qquad (26)$$

$$\sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} et_{ij} f_{ikjm} + \sum_{k=1}^{s+1}\sum_{(j,m)\in NC_{ik}} er_{ji} f_{jmik} + \sum_{k=1}^{s+1}\sum_{p\in A_{ik}} d_p eg_k x_{ik} \le e_k q\ ,\ \forall i \qquad (27)$$

$$f_{ikjm} \ge 0\ ,\forall i,k,(j,m)\in NC_{ik} \qquad (28)$$

As can be seen from the formulation, additional variables are used to compute the constraint violations. The sum of the normalized violations multiplied by a sufficiently large value is added to the objective function in order to guarantee obtaining the minimum possible constraint violation values. We normalize the violations amounts using the maximum possible violation amounts for the corresponding constraints. Note that we can relax the equal battery energy assumption and use e_k instead of e in constraint (27), as this does not cause nonlinearity any longer.

The algorithm for computing fitness of an individual is given in Figure 5.3.

```
Algorithm **compute fitness**
Begin
    Calculate total sensor cost for sensors located in the chromosome
    Solve PL3 with given sensor locations to find the lifetime, flow variables, and
        violations for connectivity, sensor capacity, link capacity, and coverage
        constraints
End
```

Figure 5.3 Algorithm for computing fitness

The algorithm in Figure 5.3 yields the raw violation amounts. For an individual in GA, the overall violation to be used as the third objective is found by summing the normalized violation amounts for all types of violation. Normalization is based on the maximum and minimum violations in the population. For instance, we find the normalized coverage violation of individual z in generation t, NT_t^z, as follows.

Let T_{pt}^z be the coverage violation of point p for individual z in generation t. Let N be the population size. Then,

$$T_t^z = \sum_{p=1}^{m} T_{pt}^z$$

$$T_t^{\min} = \min_z \{T_t^z, z = 1,\dots,N\}$$

$$T_t^{\max} = \max_z \{T_t^z, z = 1,\dots,N\}$$

$$NT_t^z = \frac{T_t^z - T_t^{\min}}{T_t^{\max} - T_t^{\min}}$$

Normalized violations for connectivity (NS_t^z), node capacity (NK_t^z) and channel capacity (NL_t^z) are calculated in a similar manner. We then take the weighted sum of four violations as the overall violation, which becomes the third objective to be minimized. Initially the violations are equally weighted, but weights for violations change throughout the generations according to the average violations of the population members. We provide the weight update procedure in Figure 5.4 where

WS_t, WK_t, WL_t and WT_t are the respective weights of connectivity, node capacity, link capacity and coverage violations in generation t. Weight update is carried out at the end of every new generation using the procedure given in Figure 5.4.

If a certain type of violation has a larger share in the population average of overall violation, then it has a larger weight and it is emphasized more in computing the violation of an individual. We incorporate the previous generation's violation information with the current one while determining the weights as part of this adaptive procedure.

```
Algorithm **update weights of violations**
Begin
    Calculate $\overline{NS}_t = \frac{1}{N}\sum_{z=1}^{N} NS_t^z$, $\overline{NK}_t = \frac{1}{N}\sum_{z=1}^{N} NK_t^z$, $\overline{NL}_t = \frac{1}{N}\sum_{z=1}^{N} NL_t^z$,
        $\overline{NT}_t = \frac{1}{N}\sum_{z=1}^{N} NT_t^z$
    Let $sum = \overline{NS}_t + \overline{NS}_{t-1} + \overline{NK}_t + \overline{NK}_{t-1} + \overline{NL}_t + \overline{NL}_{t-1} + \overline{NT}_t + \overline{NT}_{t-1}$
    Set $WS_t = \left(\overline{NS}_t + \overline{NS}_{t-1}\right)/sum$
        $WK_t = \left(\overline{NK}_t + \overline{NK}_{t-1}\right)/sum$
        $WL_t = \left(\overline{NL}_t + \overline{NL}_{t-1}\right)/sum$
        $WT_t = \left(\overline{NT}_t + \overline{NT}_{t-1}\right)/sum$
End
```

Figure 5.4 Algorithm for updating weights of violations

5.4 Initial population generation

Initial population is generated in two stages. Relays are located after deployment of sensor nodes since relay locations are highly dependent on sensor locations. The algorithm for initial population generation is given in Figure 5.5 where we use the additional notation below.

N	Population size
gpr_{ik}	Probability of locating a sensor of type k at location i
T_z	Set of points p whose coverage thresholds are not met and $B_p \neq \varnothing$ for individual z

In generating individual z, for each point p whose coverage threshold is not met, we use the (location i, sensor type k) combinations that can sense p, B_p. We start with point p that can be sensed by the least number of such combinations. We find the amount of data flow that each combination can handle by itself. If the combination can satisfy the coverage threshold, we add the data amount of p, d_p to the numerator of H_{ik}. If this is not the case, we multiply d_p by $\frac{\ln(1-pr_{ikp})}{\ln(1-h_p)}$ which stands for the fraction of the coverage threshold of point p met by combination (i,k). We find the amount of data sensed per unit cost, H_{ik}, by dividing the total data amount by the cost of the sensor. We then normalize H_{ik} values for each $(i,k) \in B_p$ and set the probability of placing a sensor of type k at location i, gpr_{ik}, as this normalized value. After a sensor is placed, we update set T_z after updating set B_p for all $p \in T_z$ and coverage threshold requirements using the current assignments. T_z is recomputed using the updated B_p's . We repeat the same process to locate sensors until there are no points left in T_z.

```
Algorithm **generate initial population**
Begin
    For each individual $z = 1,...,2N$
        Compute $B_p$ for all $p = 1,...,m$
        Let $T_z$ include all $p = 1,...,m$
        While $T_z \neq \varnothing$
            Find point $p \in T_z$ with minimum $|B_p|$
            For each $(i,k) \in B_p$
                Let $H_{ik} = \frac{\sum_{p \in A_{ik}} \min\left(\frac{\ln(1 - pr_{ikp})}{\ln(1 - h_p)}, 1\right) d_p}{c_k}$
            End for
            For each $(i,k) \in B_p$
                $gpr_{ik} = \frac{H_{ik}}{\sum_{(i,k) \in B_p} H_{ik}}$
            End for
            Select an $(i,k) \in B_p$ with probability $gpr_{ik}$ and locate a sensor of type $k$
                at location $i$
            Update $B_p$ for all $p \in T_z$ considering the newly located sensor
            Compute $T_z$ using the updated $B_p$'s
        End while
    End for
    Sort $2N$ individuals in ascending order of coverage violation
    Take the first $N$ individuals as $P_o$
    For each individual $z \in P_o$
        **Locate relays**
    End for
End
```

Figure 5.5 Algorithm for initial population generation

This procedure provides that $(i,k) \in B_p$ with the higher amount of data sensed per unit cost has a higher probability of assignment, which is reasonable for both minimizing the total cost and maximizing the lifetime. When the amount of data that a sensor can handle by itself is high, we expect that each point will be sensed by fewer sensors and the amount of oversensing of the points will be low. Sensing a point with more than the required number of sensors (oversensing) increases the total amount of data flow to be transmitted. If the total amount of data flow increases in the network, network lifetime will decrease. We expect to avoid this with the above procedure.

In applying this procedure, we restrict the solution space for location decision by locating sensors according to the point coordinates. We take into account all (i,k) combinations that can cover points but leave others out, therefore the locations that cannot sense any point are not assigned.

After determining the sensor locations, we deploy relays considering the sensors located and the amount of data sensed. The algorithm for locating relays is given in Figure 5.7, which can be explained with the help of Figure 5.6.

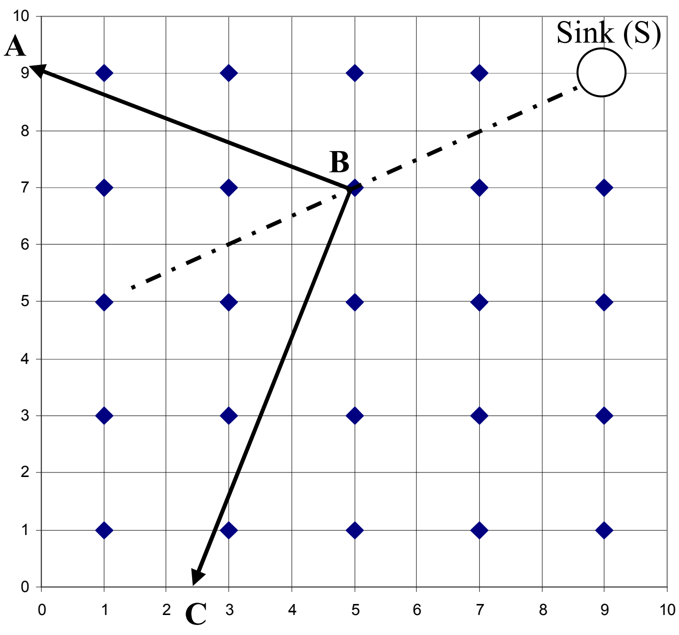


Figure 5.6 Sample monitoring area for locating relays

We know that the data amount to be transmitted is higher for the nodes that are closer to the base station. In order to balance the data flow for these nodes, relays should be placed at the appropriate locations. Therefore, location of the base station becomes important in locating relays. We take the location of the base station into account using a line of sight approach as can be seen in Figure 5.6. We draw line SB to the potential location i or B from the base station. Two perpendicular lines, whose bisector is SB, represent the "line of sight" of location i. L_i is the set of (j,k) combinations that fall in the line of sight of i. The probability of locating a relay at location i is computed by dividing the total data generated by the previously located sensors that are in the line of sight of location i by the total data generated in the monitoring area. If location i is closer to the base station, it has more sensors and more data generated in its line of sight, and probability of placing relay at i is higher.

```
Algorithm **locate relays**
Begin
    For each location $i$
        If no sensor is located at location $i$ (denoted as $B$ in Figure 5.6)
            Let $SB$ be the bisector of two rays, $BA$ and $BC$, where angle $ABC$ is
                90°
            Let $L_i$ be the set of (location $j$, sensor type $k$) combinations that fall in
                the area covered by $ABC$ and the limits of the monitoring area
            $gpr_{i3} = \frac{\sum_{(j,k)\in L_i} \sum_{p\in A_{jk}} d_p}{\sum_{p=1}^{m} d_p}$
            Locate a relay at location $i$ with $gpr_{i3}$
        End if
    End for
End
```

Figure 5.7 Algorithm for locating relays

We generate $2N$ individuals in total and select N of them as the initial population. The selection is made according to the coverage violation amounts of the individuals since we can compute only the coverage violation with the given solution representation. Solving PL3 in order to determine the other violations would require high computation time. Therefore, we introduce a procedure to start the search with the individuals having less coverage violation.

5.5 Parent Selection and Replacement

Tournament selection is used for selecting parents from the current population P_t as shown in Figure 5.8. In the selection procedure, among the two individuals we favor the one that is in the better nondominated front. If two randomly selected individuals are in the same front, we select the one with less overall violation in order to guide the search towards the feasible solutions. If the tie cannot be broken considering the fronts and the violations, we use a selection rule based on nondomination. We simply select the individual that is dominated by fewer solutions. When an individual is dominated by fewer solutions, it is likely to be in a less crowded region of the search space. This selection rule gives more chance to such individuals to become parents, thereby increasing the diversity of search. If the tie cannot be broken in this selection, we select one of the individual randomly.

Parent selection is repeated until C acceptable offspring are generated. Set R_t is formed as the union of parent population P_t and offspring population Q_t. To form the population for the next generation, P_{t+1}, the $N+C$ solutions in R_t are sorted by using the fast nondominated sorting algorithm due to Deb (2000). R_t is sorted into fronts F_j for $j = 1,2,3,...$. Then, the algorithm given in Figure 5.9 is applied for parent replacement.

```
Algorithm **select parents**
Begin
    Repeat
        Select two individuals from $P_t$ randomly with replacement
        If they are in different fronts
            Select the one that is in the better front
        Else
            If their overall violations are different
                  Select the one with less overall violation
            Else
                  Select the one that is dominated by fewer individuals
            End if
        End if
    Until two parents are selected
End
```

Figure 5.8 Algorithm for parent selection

According to the algorithm, P_{t+1} is formed by selecting N individuals from the better fronts. If adding all individuals from the last of these fronts causes the population to have more than N members, only a subset is selected. In NSGA-II, the crowding measure is used for this selection. Our algorithm does not have an explicit crowding mechanism. Instead we select from the last front those individuals that are dominated by fewer solutions in the better fronts. With this rule similar to the one we use for parent selection, we favor individuals in less crowded regions in an attempt to preserve diversity. If the ties considering the number of solutions dominating the individuals cannot be broken, we select individuals randomly.

```
Algorithm **form new population**
Begin
    Set $P_{t+1} = \phi$
    Set $j = 1$
    While $|P_{t+1}| + |F_j| \leq N$
        Set $P_{t+1} = P_{t+1} \cup F_j$
        Set $j = j + 1$
    End while
    Select $N - |P_{t+1}|$ individuals from $F_j$ that are dominated by fewer individuals in
        $P_{i+1}$, insert them in $P_{i+1}$
End
```

Figure 5.9 Algorithm for replacement

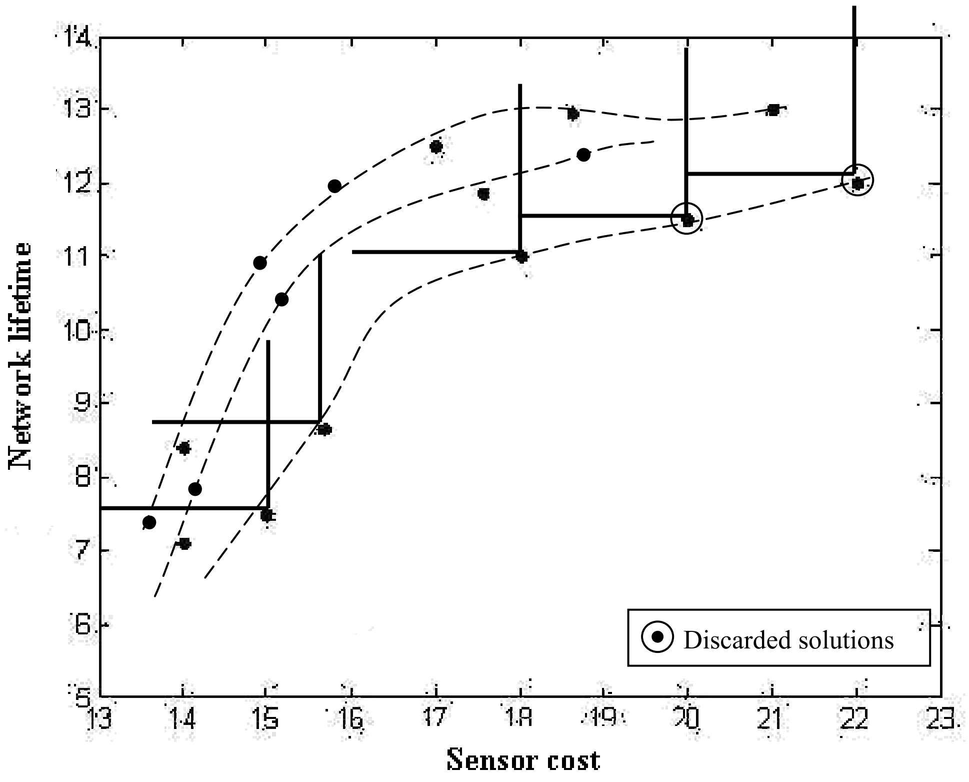


Figure 5.10 Sample population at the end of generation t

The selection rule from the last front is illustrated in Figure 5.10. Suppose there are 30 individuals in R_t and $N = 15$. We select seven individuals from the first front and five individuals from the second front in the population and $|P_{t+1}| = 12$. We have five solutions in the third front, therefore we have to choose three of them in order to have $|P_{t+1}| = 15$. We discard the individuals circled in Figure 5.10 since they are dominated by a larger number of individuals than the others. The region of search space that contains discarded individuals is already represented by individuals in the better fronts.

5.6 Crossover and Mutation

The classical crossover operators are considered for generating offspring in the GA. These are one point, two point and uniform crossover operators (Michalewicz and Fogel, 2004). Uniform crossover is eliminated after analyzing the results of preliminary runs since using it does not yield good results.

We aim to repair and improve solutions with the mutation operator. Mutation is applied to each offspring with a given probability, pr_m. The algorithm for mutation is given in Figure 5.11. It makes use of the sets T_z and B_p that are also used in initial population generation. In the first stage of the mutation, we try to improve the solution in terms of the coverage violation. This is similar to the first phase of our initial population generation scheme. The difference is that we do not consider the amount of data to be sensed or the cost of sensors. We assign equal probability to each (i,k) combination that can sense point p in mutation. Consequently, we next have an improvement phase which removes the redundant sensors. An individual may have redundant sensor assignments, which can be removed without increasing the coverage violation of the individual. While removing the redundant sensors, we start from the farthest sensor to the base station and continue with the removal attempts by considering the distance of the sensor to the base station. If there are multiple sensors with the same distance to the base station, one of them is selected randomly. The aim is to improve the network lifetime by locating sensors closer to the base station. Lifetime is also improved by eliminating oversensing of the targets, which will decrease the data amount to be transmitted to the base station. The algorithm also works for the cost objective since cost is decreased by removing the sensors.

```
Algorithm **mutation**
Begin
    If individual $z$ has coverage violation
        Update set $B_p$ for all $p = 1,...,m$
        Compute $T_z$
        While $T_z \neq \varnothing$
            Find point $p \in T_z$ with minimum $|B_p|$
            For each $(i,k) \in B_p$
                $gpr_{ik} = \frac{1}{|B_p|}$
            End for
            Select an $(i,k) \in B_p$ with probability $gpr_{ik}$ and locate a sensor of type $k$
                at location $i$
            Update $B_p$ for all $p \in T_z$ considering the newly located sensor
            Compute $T_z$ using the updated $B_p$'s
        End while
    End if
    Sort locations in descending order of Euclidean distance to the base station
    For each location $i$ with sensor of type $k$ in the ordered set
        Remove sensor $k$ if the total coverage violation does not increase
    End for
    **Locate relays**
End
```

Figure 5.11 Algorithm for mutation

In the second stage, we place relay nodes as we do in the initial population generation algorithm. We use the same relay location procedure in the mutation. After locating additional relay nodes, we intend to remove the redundant relays from the individual using the procedure described in Figure 5.12. If a relay does not have outgoing data flow, we can remove it since it does not bring any contribution to the lifetime objective. We can decrease the total network cost by this removal. To determine the redundant relays to be removed, we need to solve PL3. As will be explained in the next section, not every offspring is accepted in our GA and we do not want to spend PL3 computation time for an individual that will eventually be discarded. Therefore, we postpone the removal phase of the mutation until after the individual passed the acceptance test and PL3 is solved to compute its fitness.

```
Algorithm **remove redundant relays**
Begin
    For each location $i$ with relay
        If total outgoing data flow from the relay at location $i$ is zero
            Remove relay
        End if
    End for
End
```

Figure 5.12 Algorithm for removing redundant relays

5.7 The Main Algorithm

The outline of our algorithm, which is similar to NSGA-II (Deb, 2002) is presented in Figure 5.13. The algorithm runs for a certain number of generations, G, determined by preliminary experiments. We start by generating the initial population. Note that removal of redundant relays is also applied to the initial population after computing fitness. In every generation, we select parents and generate offspring using the crossover and mutation operators until the total number of acceptable offspring reaches C. Coverage violation is easily calculated for each offspring generated. If coverage violation of the offspring is larger than a factor of the average coverage violation of the population or if the offspring is stillborn, then the offspring is discarded. (If an offspring has the same assignments as a current population member or previously generated offspring, it becomes stillborn.) Otherwise, the offspring is accepted and PL3 is solved for the offspring. The purpose of discarding an offspring based on a coverage violation condition is to decrease the number of LPs solved. The accepted offspring are promising in terms of coverage violation, and solving PL3 only for these offspring significantly reduces the number of LPs to be solved and the computation time. Pilot runs encourage using this filtering procedure. For example, a sample GA run with 40 grids takes 600.47 seconds, and 568.26 seconds of this time is spent for solving LPs. Time for solving LPs constitutes about 95% of the total algorithm time. Therefore filtering promising offspring becomes crucial, otherwise we can achieve the same solution quality in longer run times. The factor $(1-bt/G)$ used in filtering is a decreasing function of bt when G is fixed. When the parameter $b \in [0,1]$ is larger, the acceptance criterion becomes tighter. The coverage violation criterion also becomes tighter with increasing iteration count t so that solutions with less and less coverage violation are accepted towards the end of the run. There is a tradeoff between run time and solution quality in this case. The condition added can affect the diversity of the solutions negatively, however we can search for better solutions in the time gained by such a procedure.

After a certain number of offspring is generated, we sort the union of the parent and offspring population, R_t, and we select N of them for the next generation, P_{t+1}.

At the end of the run, we eliminate the infeasible solutions and apply nondominated sorting to the feasible solutions in the final R_t. We take the solutions in the first front as the approximation of the Pareto optimal front.

```
Algorithm **main**
Begin
    **Generate initial population**, $P_0$
    **Compute fitness** for each individual in $P_0$
    **Remove redundant relays** for each individual in $P_0$
    Apply nondominated sorting to $P_0$ to find $F_j$, $j = 1,2,\ldots$
    Set $t = 0$
    Repeat
        Set $Q_t = \phi$
        Repeat
            **Select parent**s from $P_t$ by binary tournament selection
            Perform crossover to generate two offspring
            For each offspring
                Apply **mutation** to offspring with probability $pr_m$
                If coverage violation of offspring $\leq (1 - bt/G) \times$ average coverage violation
                    of current population, and if offspring is not stillborn
                    **Compute fitness** of offspring
                    **Remove redundant relays**
                    Insert offspring in $Q_t$
                End if
            End for
        Until $|Q_t| \geq C$
        Set $R_t = P_t \cup Q_t$
        For each individual in $R_t$
            Normalize constraint violations
            Calculate overall violation
        End for
        Apply nondominated sorting to $R_t$ to find $F_j$, $j = 1,2,\ldots$
        **Form new population**, $P_{t+1}$
        **Update weights of violations** using members of $P_{t+1}$
        Set $t = t + 1$
    Until $t = G$
    Discard infeasible individuals, apply nondominated sorting to feasible individuals
    Take the individuals in the first front
End
```

Figure 5.13 Main algorithm

CHAPTER 6

EXPERIMENTATION

We design an experiment to test the performance of our genetic algorithm and to investigate the effects of parameters on the performance of the algorithm. The algorithm is tested on the same randomly generated problem instances described in Section 4.1 for which Pareto optimal solutions are found except some of the PS40-LC instances. In addition to these problems, we also test our algorithm on larger problem instances.

We first discuss the parameter settings. Then, we present the performance measures. Finally we analyze the results of our experiments.

6.1 Algorithm Parameters and Experimental Settings

The parameters of our algorithm together with the experimental settings are summarized in Table 6.1. The levels for the parameters are determined after the analysis of pilot run results and Pareto optimal frontiers.

Two levels for the population size, N, are decided by analyzing the number of solutions found in the Pareto optimal frontier for PS24 and PS40 instances.

The number of fitness computations, which is equal to the total number of acceptable offspring generated, is the second parameter. It is the product of the number of generations and the number of acceptable offspring generated in each generation, $G \times C$. We have tried different levels for C during the pilot runs, however we have observed that setting it equal to N as in NSGA-II gives better results. Therefore, we take $C = N$. Hence, the number of fitness computations is determined by the number of generations once the population size is set.

Offspring are generated using crossover and mutation operators in each generation. We consider two crossover operators for our experiments. These are one point and two point crossover operators as given in Section 5.6.

We apply mutation to each offspring with a mutation probability. We set levels for the mutation probability, pr_m, as given in Table 6.1 based on the results of the pilot runs.

Another parameter, b, is used in determining the maximum acceptable coverage violation for an offspring as a factor of the average population coverage violation. We have tried different levels for b such as 1 and $1/2$, but we have seen that the results are not as good as when b is equal to zero. We have observed that restricting the search by discarding more offspring causes premature convergence and reduces the solution quality. Therefore, we set b equal to zero after pilot runs, which means that an offspring is accepted when its coverage violation is less than the average coverage violation of the population.

Table 6.1 Algorithm parameters

	PS24-LC and PS24-TC		PS40-LC and PS40-TC		
Parameter	Level 1	Level 2	Level 1	Level 2	Level 3
N	100	200	200	400	-
$G \times C$	5000	10000	10000	16000	20000
CT	one point	two point		two point	
pr_m	0.1	0.2	0.2	0.3	-
b	0			0	

For PS24 we solve 30 problems with all 16 combinations of the parameters. Based on these results, we restrict the combinations for PS40.

6.2 Performance Measures

We use three performance measures to test our algorithm. The first measure is the proximity indicator, PI. In this measure, we compute the normalized Tchebychev distance between each solution found by the GA and the closest Pareto optimal solution found by the ε-constraint approach. Distance calculation for a GA solution is illustrated in Figure 6.1. Suppose that Z^{GA} and Z^{Pareto} denote the sets of GA and Pareto optimal solutions. For each solution $s \in Z^{GA}$, we find the closest solution $t \in Z^{Pareto}$. Let the normalized distance (deviation) between s and t for cost and lifetime objectives be nd_{st}^{c} and nd_{st}^{l}, respectively. Then PI is calculated by equation (29).

$$\mathrm{PI}_{Z^{GA} \rightarrow Z^{Pareto}} = \frac{1}{\left|Z^{GA}\right|} \sum_{s \in Z^{GA}} \frac{\left(nd_{st}^{c} + nd_{st}^{l}\right)}{2} \qquad (29)$$

This measure provides information about closeness of the GA solutions to the Pareto optimal front.

The second measure is the reverse proximity indicator, RPI. In this measure, we start from a Pareto optimal solution and find the closest representative solution found by GA. This involves the symmetric case of PI calculation. Let solution $s \in Z^{GA}$ be the closest solution to the Pareto optimal solution $t \in Z^{Pareto}$. Then, RPI is calculated by equation (30). RPI calculation is also illustrated in Figure 6.1.

$$\mathrm{RPI}_{Z^{Pareto} \rightarrow Z^{GA}} = \frac{1}{\left|Z^{Pareto}\right|} \sum_{t \in Z^{Pareto}} \frac{\left(nd_{ts}^{c} + nd_{ts}^{l}\right)}{2} \qquad (30)$$

This measure provides information about closeness of the GA solutions to the Pareto optimal front as well as the diversity of the GA solutions along the front.

Last measure is the hypervolume indicator, HI, which also measures both closeness and diversity of the GA solutions. We find the area bounded by the maximum cost and minimum lifetime (nadir point) for the Pareto optimal solutions and the GA solutions. The area calculated for GA solutions is illustrated in Figure 6.1. Let A_{Pareto} and A_{GA} be the areas calculated for Pareto optimal solutions and GA solutions. Then HI is calculated by equation (31).

$$\mathrm{HI} = \frac{A_{Pareto} - A_{GA}}{A_{Pareto}} \tag{31}$$

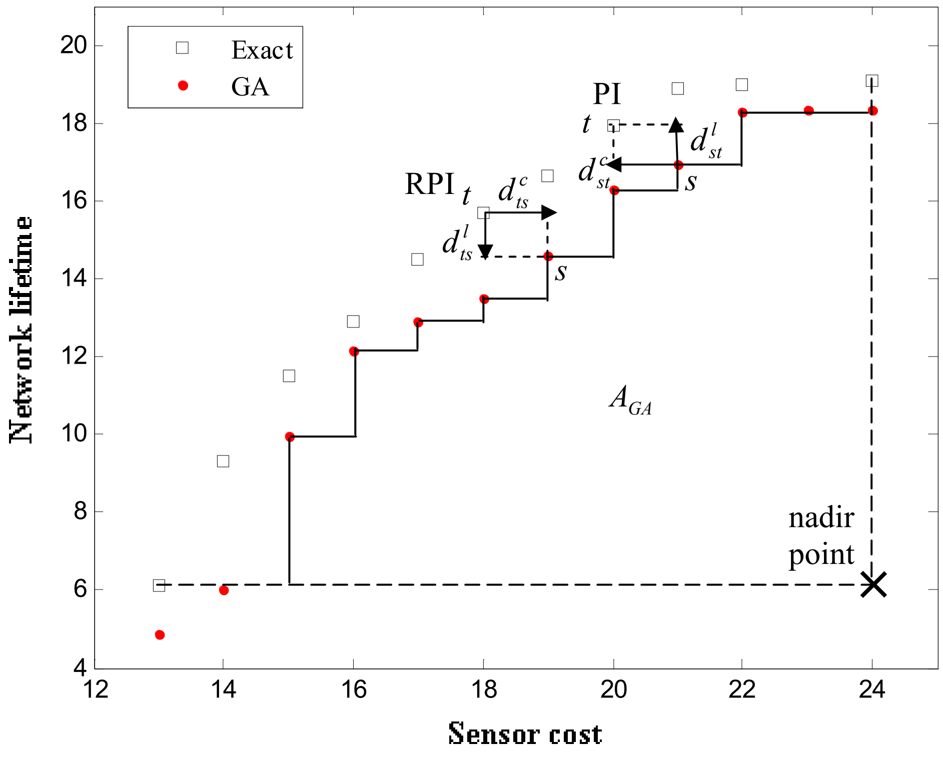


Figure 6.1 Sample GA and Pareto optimal solutions

6.3 Computational Results for Small Problems

Our algorithm is coded using C programming language and we conduct our experiments on a personal computer with 1GB DDR2 RAM and Intel Core2 Duo 2.33 Ghz processor. We use CPLEX 10.1 to find the Pareto optimal solutions by the ε-constraint approach and to solve PL3 in the genetic algorithm.

Fine Tuning the GA for PS24 Instances

We start experimentation with the PS24 instances. According to the levels of the parameters provided in Table 6.1, there are $2 \times 2 \times 2 \times 2 = 16$ combinations. Results of GA runs for all PS24 instances with loose (LC) and tight (TC) capacity constraints are summarized in Table 6.2. We use RPI as our primary performance measure in evaluating the performance of GA since it gives an idea about both convergence and diversity. Therefore, we also provide RPI information for the initial population in order to figure out the progress throughout the evolution. For most PS24-TC instances, the initial population has no feasible solutions, therefore we cannot calculate RPI and the other performance measures for these instances. We provide the number of instances where the initial population has at least one feasible solution. Our algorithm cannot find any feasible solutions for some PS24-TC instances at the end of the evolution. We also provide the number of instances for which the algorithm ends with some feasible solutions. The average number of solutions GA finds as well as the average number of GA solutions that are the same as Pareto optimal solutions (GA=Exact) are also given. We report the CPU times in the last column.

Table 6.2 Computational results for PS24 instances

					# of feasible problems		RPI				# of solutions		
Constraint tightness	Population size	Fitness Computations	Crossover type	Mutation prob.	Initial	Final	Initial	Final	PI	HI	GA	GA=Exact	GA CPU time (s)
	100	5000	one point	0.1	30/30	30/30	0.2567	0.0536	0.0341	0.1113	8.77	2.67	50
	100	5000	one point	0.2	30/30	30/30	0.2591	0.0531	0.0359	0.1082	8.67	2.83	54
	100	5000	two point	0.1	30/30	30/30	0.2699	0.0460	0.0351	0.0971	9.13	2.90	47
	100	5000	two point	0.2	30/30	30/30	0.2652	0.0502	0.0334	0.1066	9.10	2.90	51
	100	10000	one point	0.1	30/30	30/30	0.2567	0.0462	0.0280	0.0992	8.97	2.83	103
	100	10000	one point	0.2	30/30	30/30	0.2591	0.0436	0.0297	0.0900	8.67	2.93	110
	100	10000	two point	0.1	30/30	30/30	0.2699	0.0381	0.0298	0.0753	9.50	2.93	99
LC	100	10000	two point	0.2	30/30	30/30	0.2652	0.0431	0.0290	0.0828	9.13	3.03	105
	200	5000	one point	0.1	30/30	30/30	0.2498	0.0496	0.0324	0.1008	8.87	3.20	53
	200	5000	one point	0.2	30/30	30/30	0.2328	0.0413	0.0287	0.0787	9.00	3.60	56
	200	5000	two point	0.1	30/30	30/30	0.2425	0.0456	0.0353	0.0963	9.17	2.83	51
	200	5000	two point	0.2	30/30	30/30	0.2375	0.0452	0.0310	0.0911	8.93	3.57	54
	200	10000	one point	0.1	30/30	30/30	0.2498	0.0418	0.0264	0.0817	9.07	3.07	112
	200	10000	one point	0.2	30/30	30/30	0.2328	0.0344	0.0228	0.0657	9.60	3.60	118
	200	10000	two point	0.1	30/30	30/30	0.2425	0.0366	0.0294	0.0731	9.40	3.10	105
	200	10000	two point	0.2	30/30	30/30	0.2375	0.0317	0.0220	0.0558	9.27	3.70	110
	100	5000	one point	0.1	5/30	28/30	0.5349	0.1214	0.0921	0.2763	5.70	1.20	55
	100	5000	one point	0.2	1/30	29/30	0.5513	0.1258	0.0910	0.2837	5.90	1.20	56
	100	5000	two point	0.1	1/30	29/30	0.6494	0.1309	0.0953	0.2913	6.00	1.40	54
	100	5000	two point	0.2	3/30	29/30	0.4464	0.1118	0.0820	0.2519	6.20	1.33	55
	100	10000	one point	0.1	5/30	29/30	0.5349	0.1132	0.0817	0.2647	5.83	1.13	103
	100	10000	one point	0.2	1/30	29/30	0.5513	0.1145	0.0878	0.2677	6.20	1.20	106
	100	10000	two point	0.1	1/30	29/30	0.6494	0.1067	0.0727	0.2431	5.97	1.47	101
TC	100	10000	two point	0.2	3/30	29/30	0.4464	0.0998	0.0802	0.2217	6.60	1.33	103
	200	5000	one point	0.1	4/30	29/30	0.5670	0.1156	0.0827	0.2710	6.20	1.57	57
	200	5000	one point	0.2	4/30	29/30	0.5592	0.1059	0.0791	0.2405	6.10	1.57	57
	200	5000	two point	0.1	2/30	29/30	0.5219	0.0979	0.0696	0.2103	5.90	1.80	55
	200	5000	two point	0.2	4/30	29/30	0.6462	0.1025	0.0753	0.2339	6.00	1.90	58
	200	10000	one point	0.1	4/30	29/30	0.5670	0.0846	0.0766	0.1923	6.87	1.57	107
	200	10000	one point	0.2	4/30	29/30	0.5592	0.0898	0.0640	0.1976	6.23	1.57	111
	200	10000	two point	0.1	2/30	29/30	0.5219	0.0688	0.0587	0.1608	6.43	1.80	105
	200	10000	two point	0.2	4/30	29/30	0.6462	0.0761	0.0574	0.1734	6.57	1.93	110

We also plot the interaction effects of the parameters on RPI given in Figures 6.2 and 6.3 for PS24-LC and PS24-TC instances.

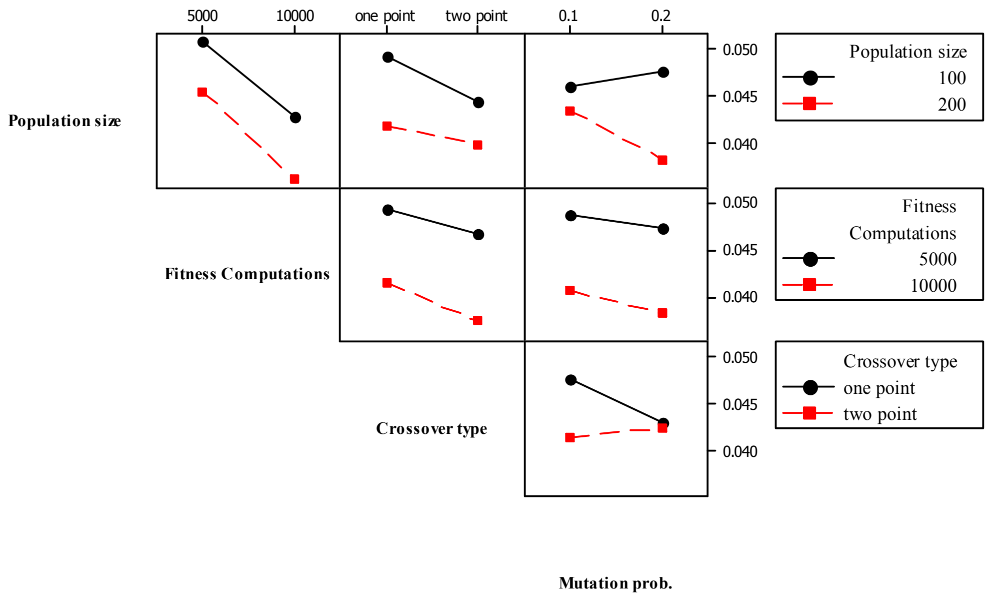


Figure 6.2 Interaction effect plots for final RPI for PS24-LC instances

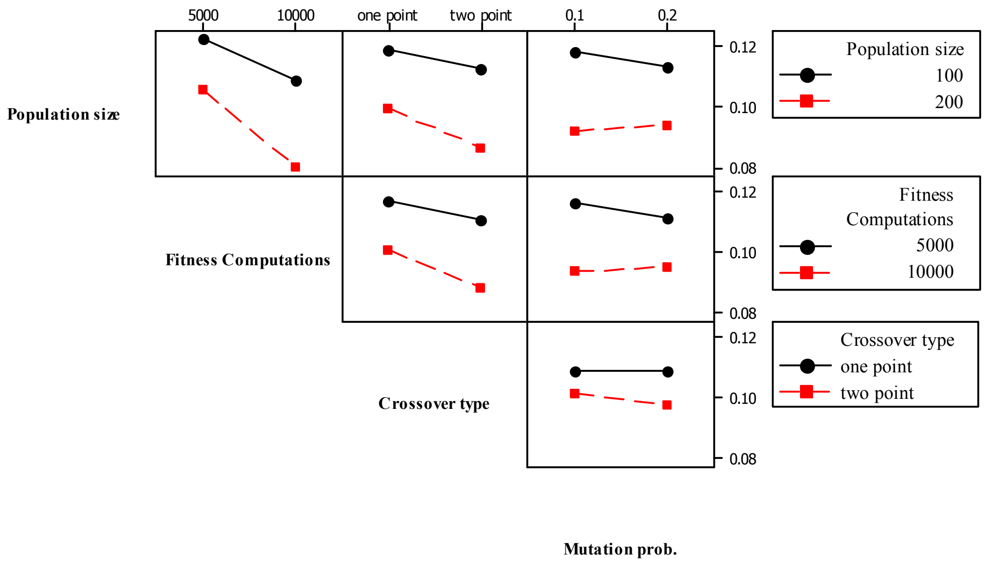


Figure 6.3 Interaction effect plots for final RPI for PS24-TC instances

Numerical results and interaction plots show that better results are obtained for population size of 200 and 10000 fitness computations. Two point crossover generally outperforms one point crossover. It seems that increasing the mutation probability provides better solutions. It increases the effectiveness of one point crossover significantly for PS24-LC as can be seen from Figure 6.2. A higher mutation probability of 0.2 results in better RPI in general.

In Figure 6.4, population averages of PI, RPI and HI are plotted against the number of fitness computations to show their progress throughout the generations. The values are the averages over 30 instances of PS24-LC. It seems that convergence is achieved after 8000 fitness computations, and 10000 fitness computations are sufficient for these instances. We cannot provide this information for PS24-TC since the number of instances having feasible solutions is too small during the initial stages of evolution.

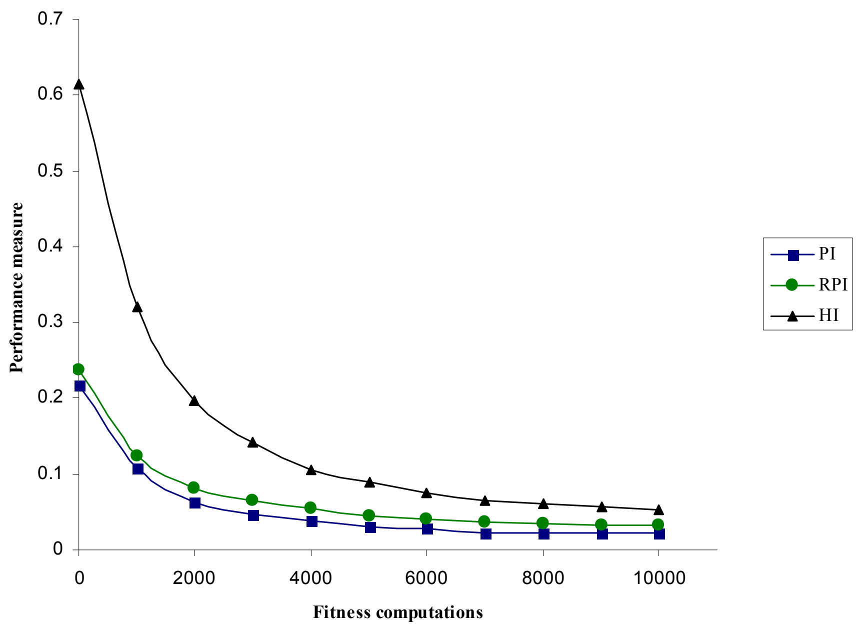


Figure 6.4 Progress of PI, RPI and HI over generations for PS24-LC

($N = 200$, $pr_m = 0.2$, two point crossover)

Hence, the best parameter settings for PS24 are a population size of $N = 200$, $G \times C = 10000$ fitness computations, two point crossover, and a mutation probability of $pr_m = 0.2$.

Fine Tuning the GA for PS40 Instances

We next experiment with the problems of size 40. As the problem size gets larger, we consider larger values for the population size (200 and 400) and the number of fitness computations (10000, 16000 and 20000). Considering the results for PS24 instances, we observe that two point crossover provides better solutions than one point crossover in general. Hence, only the two point crossover is used in the remaining experiments. For PS24 instances, we see that larger mutation probability generally provides better results. Therefore, we set two levels for the mutation probability as 0.2 and 0.3. We have 10 instances that are solved exactly for PS40-LC and 30 instances for PS40-TC, hence we experiment with our algorithm considering these instances.

The results for all PS40 instances are given in Table 6.3. The information provided in this table is found across 10 problem instances for PS40-LC and 30 problem instances for PS40-TC.

We again plot the interaction effects of the parameters on RPI in Figures 6.5 and 6.6 for PS40-LC and PC40-TC instances. According to these plots, the best settings for PS40-LC are a population size of $N = 400$, $G \times C = 20000$ fitness computations, and a mutation probability of $pr_m = 0.2$. Increasing the mutation probability does not affect the performance significantly for PS40-LC. However, for PS40-TC, $pr_m = 0.3$ seems to yield better results.

Table 6.3 Computational results for PS40 instances

					# of feasible problems		RPI				# of solutions		
Constraint tightness	Population size	Fitness Computations	Crossover type	Mutation prob.	Initial	Final	Initial	Final	PI	HI	GA	GA=Exact	GA CPU time (s)
	200	10000	two point	0.2	10/10	10/10	0.3141	0.0631	0.0562	0.1472	11.70	1.40	376
	200	10000	two point	0.3	10/10	10/10	0.3094	0.0637	0.0550	0.1739	12.40	1.20	392
	200	16000	two point	0.2	10/10	10/10	0.3141	0.0540	0.0469	0.1167	11.90	1.40	565
	200	16000	two point	0.3	10/10	10/10	0.3094	0.0559	0.0517	0.1511	12.80	1.20	591
	200	20000	two point	0.2	10/10	10/10	0.3141	0.0510	0.0457	0.1058	12.40	1.60	690
LC	200	20000	two point	0.3	10/10	10/10	0.3094	0.0545	0.0503	0.1345	12.60	1.10	723
	400	10000	two point	0.2	10/10	10/10	0.3047	0.0715	0.0716	0.1992	12.50	1.30	458
	400	10000	two point	0.3	10/10	10/10	0.2818	0.0649	0.0665	0.1835	12.30	0.80	455
	400	16000	two point	0.2	10/10	10/10	0.3047	0.0502	0.0499	0.1261	12.10	1.30	662
	400	16000	two point	0.3	10/10	10/10	0.2818	0.0514	0.0516	0.1260	12.60	0.80	667
	400	20000	two point	0.2	10/10	10/10	0.3047	0.0464	0.0489	0.1164	12.80	1.30	798
	400	20000	two point	0.3	10/10	10/10	0.2818	0.0484	0.0458	0.1110	12.00	0.80	809
	200	10000	two point	0.2	4/30	30/30	0.4914	0.1030	0.1033	0.2910	10.53	0.70	328
	200	10000	two point	0.3	8/30	30/30	0.5210	0.0996	0.0944	0.2853	9.73	0.87	337
	200	16000	two point	0.2	4/30	30/30	0.4914	0.0824	0.0925	0.2355	11.23	0.70	503
	200	16000	two point	0.3	8/30	30/30	0.5210	0.0863	0.0838	0.2390	10.23	0.87	518
	200	20000	two point	0.2	4/30	30/30	0.4914	0.0796	0.0855	0.2270	11.10	0.73	619
	200	20000	two point	0.3	8/30	30/30	0.5210	0.0812	0.0769	0.2214	10.20	0.83	638
TC	400	10000	two point	0.2	8/30	30/30	0.5432	0.1148	0.1103	0.3426	9.83	1.13	364
	400	10000	two point	0.3	9/30	30/30	0.5723	0.1081	0.1067	0.3073	9.80	0.90	378
	400	16000	two point	0.2	8/30	30/30	0.5432	0.0908	0.0872	0.2682	10.33	1.13	547
	400	16000	two point	0.3	9/30	30/30	0.5723	0.0850	0.0894	0.2437	10.50	0.90	566
	400	20000	two point	0.2	8/30	30/30	0.5432	0.0856	0.0813	0.2443	10.37	1.13	669
	400	20000	two point	0.3	9/30	30/30	0.5723	0.0777	0.0784	0.2176	10.63	0.90	690
	400	24000	two point	0.3	7/30	30/30	0.5048	0.0744	0.0780	0.1957	11.10	1.13	797

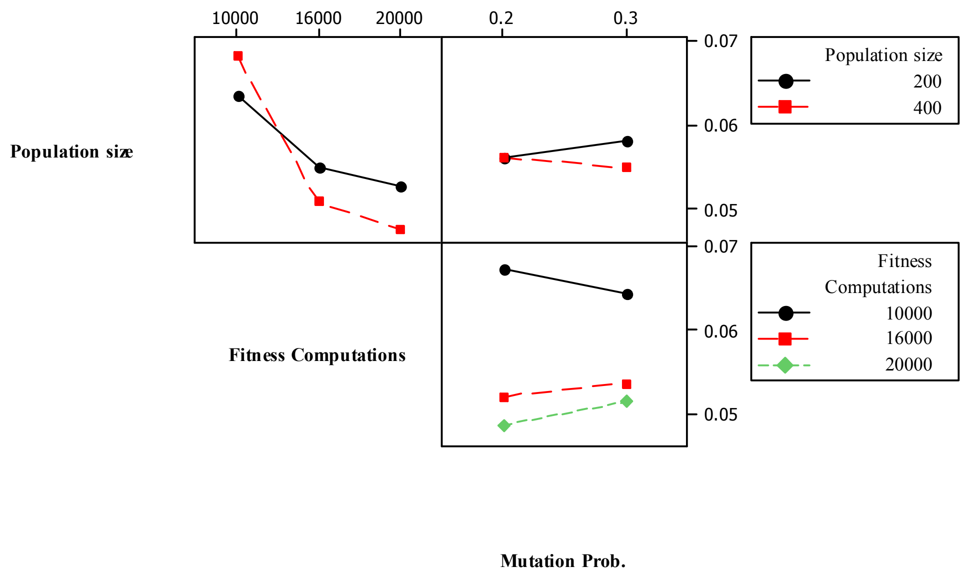


Figure 6.5 Interaction effect plots for final RPI for PS40-LC instances

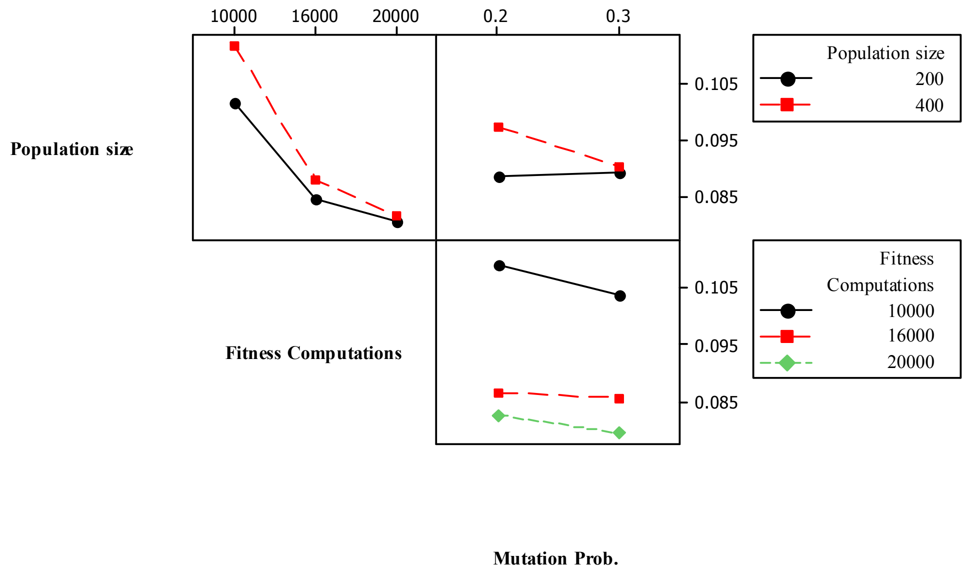


Figure 6.6 Interaction effect plots for final RPI for PS40-TC instances

The progress of the performance measures over generations is presented in Figure 6.7 for PS40-LC instances. It seems that convergence is achieved after 18000 fitness computations, and 20000 fitness computations are indeed sufficient for these instances. However, we have seen that 20000 fitness computations are not sufficient for PS40-TC instances. We have increased this number to 24000 to achieve convergence as seen in Figure 6.8.

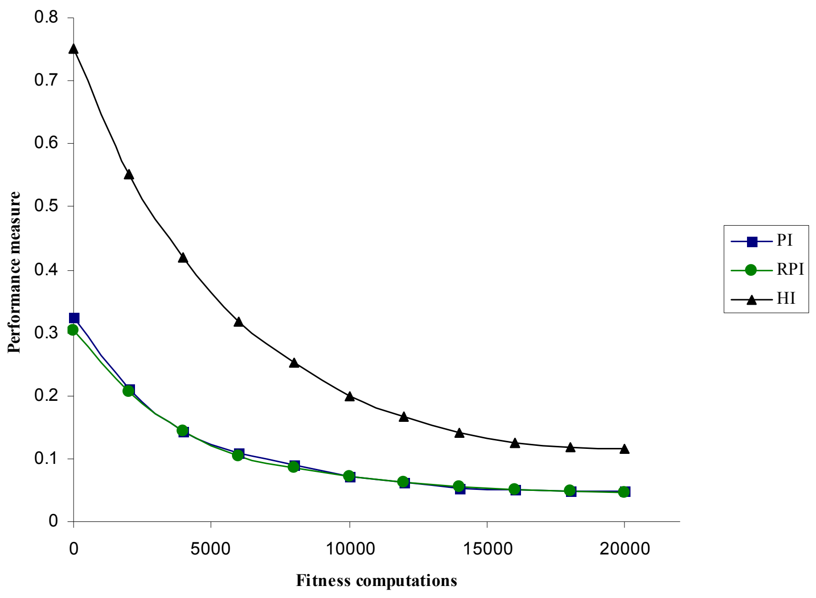


Figure 6.7 Progress of PI, RPI and HI over generations for PS40-LC ($N = 400$, $pr_m = 0.2$, two point crossover)

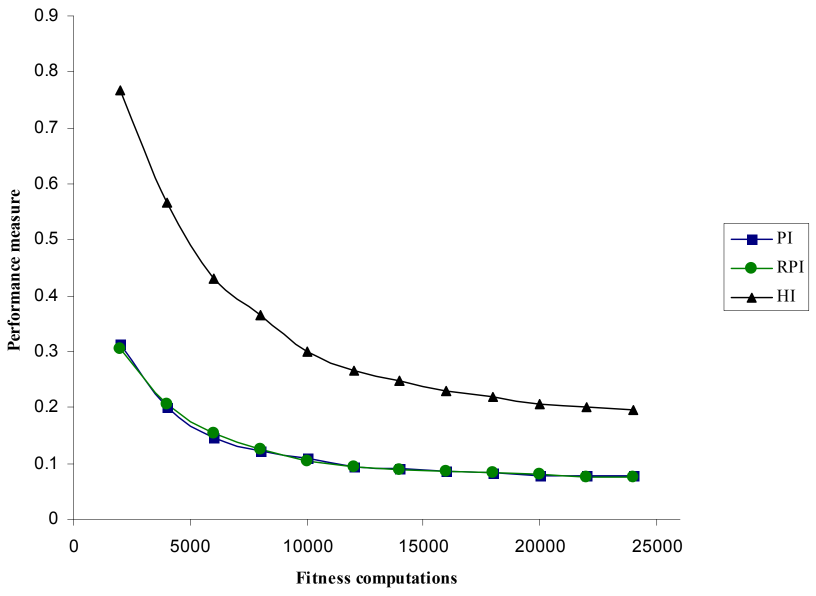


Figure 6.8 Progress of PI, RPI and HI over generations for PS40-TC ($N = 400$, $pr_m = 0.3$, two point crossover)

Hence, the best settings for PS40-TC are a population size of $N = 400$, $G \times C = 24000$ fitness computations, and a mutation probability of $pr_m = 0.3$.

Comparison of GA with ε-Constraint Approach

A comparison of the GA with the ε-constraint approach is given in Table 6.4.

Table 6.4 Comparison of ε-constraint approach and GA with the best setting

$N = 200$, $G \times C = 10000$, $pr_m = 0.2$ for PS24

$N = 400$, $G \times C = 20000$, $pr_m = 0.2$ for PS40-LC

$N = 400$, $G \times C = 24000$, $pr_m = 0.3$ for PS40-TC

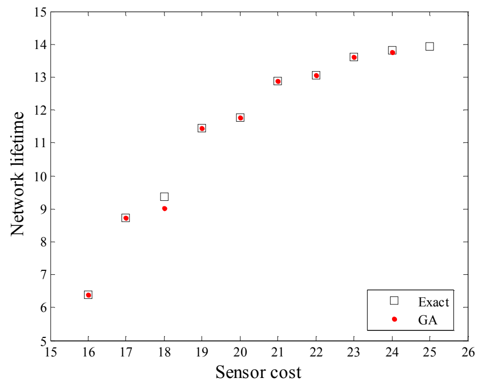
			GA performance measures			Number of solutions			CPU time (s)	
Problem size	Constraint tightness	# of feasible problems	RPI	PI	HI	ε-constraint	GA	GA=Exact	ε-constraint	GA
PS24	LC	30/30	0.0317	0.0220	0.0558	10.20	9.27	3.70	1118	110
	TC	29/30	0.0761	0.0574	0.1734	7.47	6.57	1.93	70	110
PS40	LC(1)	10/10	0.0464	0.0489	0.1164	13.60	12.80	1.30	100088	798
	LC(2)	20/20	-	-	-	13.00	14.20	-	85983	821
	TC	30/30	0.0744	0.0780	0.1957	11.60	11.10	1.13	12865	797

(1) Results across 10 instances that are solved exactly by the ε-constraint approach.

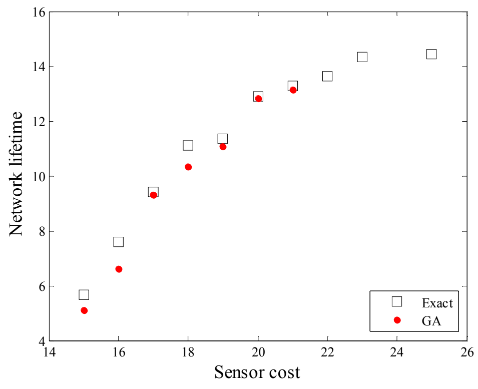
(2) Results across 20 instances that are solved approximately by the ε-constraint approach.

The average number of Pareto optimal solutions of PS24-LC instances is 10.2 with an average CPU time of 1118 seconds. CPU time for PS24-TC instances is 70 seconds where the average number of Pareto optimal solutions is 7.47. With the best settings, our GA finds an average of 9.27 and 6.57 solutions per instance for PS24-LC and PS24-TC, respectively. The average solution time is 110 seconds for both cases. This means that, for PS24-LC, the number of feasible GA solutions is 91% of the number of Pareto optimal solutions, and these are found in one tenth of the exact solution time. The same figure is 88% for PS24-TC, however the GA solution time is slightly longer than the exact solution time. GA can find 3.70 of 10.2 (36%) Pareto optimal solutions for PS24-LC, and 1.93 of 7.47 (26%) solutions for PS24-TC. PI, RPI and HI measures are also larger for PS24-TC than those for PS24-LC. Therefore, we conclude that GA performs worse when the capacity constraints are tight, in terms of both the solution quality and the CPU time. Moreover, it can be seen from Table 6.4 that the problem of infeasibility for PS24-TC instances can be solved by increasing the number of possible locations.

We illustrate GA and Pareto optimal fronts in Figures 6.9 and 6.10 for sample instances of PS24-LC and PS24-TC. In both figures, (a) represents an instance with better performance measures and (b) with worse.

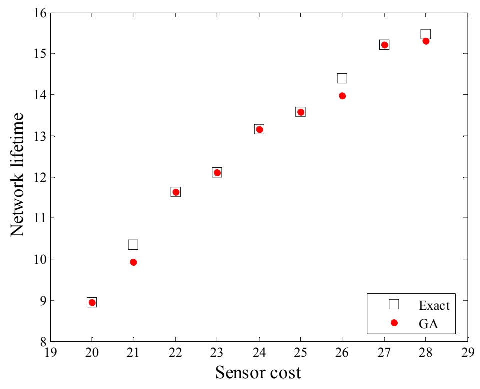


(a) Instance 19
RPI=0.0096 PI=0.0031 HI=0.0098

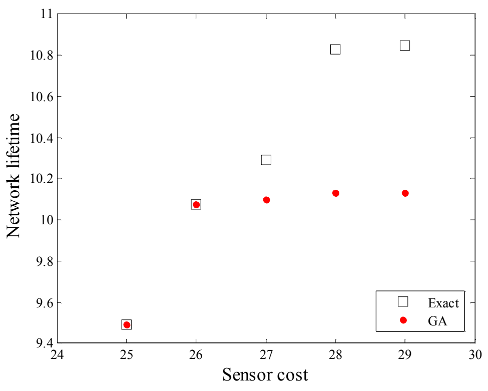


(b) Instance 10
RPI=0.0686 PI=0.0308 HI=0.0919

Figure 6.9 Plots of fronts for sample PS24-LC instances



(a) Instance 4
RPI=0.0087 PI=0.0087 HI=0.0307



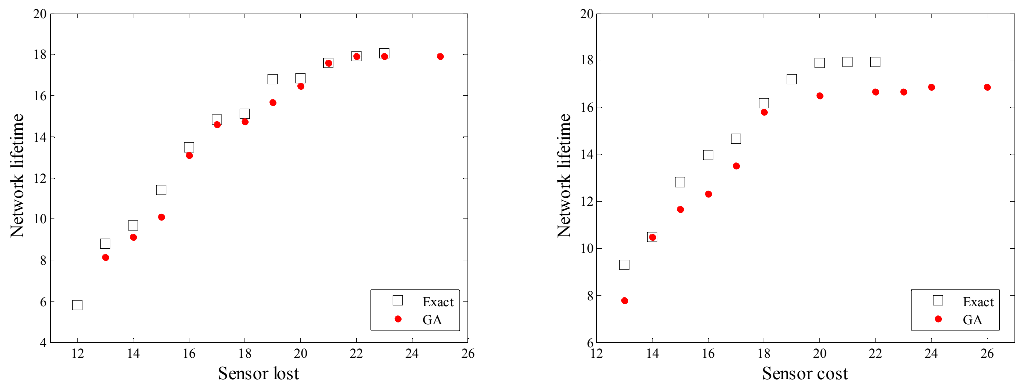
(b) Instance 27
RPI=0.1938 PI=0.1792 HI=0.3295

Figure 6.10 Plots of fronts for sample PS24-TC instances

When we examine the PS40-LC results given in Table 6.4, we see that PI, RPI and HI measures are larger (worse) compared to PS24-LC, but smaller (better) compared to PS24-TC for the 10 problems with exact solutions. These measures cannot be calculated for the 20 problems solved approximately, because some of the approximate ε-constraint solutions are dominated by GA solutions. When we consider all 30 instances of PS40-LC, the average number of frontier solutions found by GA is relatively closer to that found by ε-constraint approach. However, GA can find only 1.30 of 13.6 (10%) Pareto optimal solutions for the first 10 problems. The GA CPU times, on the other hand, are about 100 times shorter than the ε-constraint CPU times.

As in the case of PS24, the solution quality of GA is worse for PS40-TC than it is for PS40-LC. The CPU time of GA is only 15 times shorter than the ε-constraint CPU time. In general, problems with tight capacity constraints prove to be harder for the GA and easier for the ε-constraint approach.

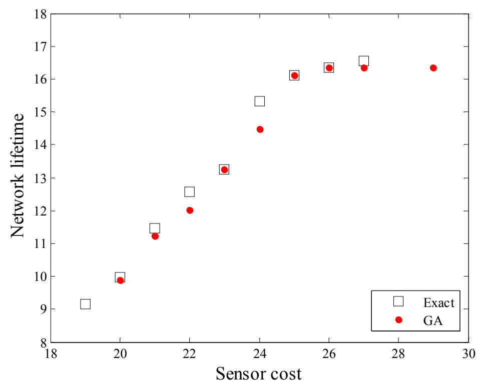
GA and Pareto optimal fronts are shown in Figures 6.11 and 6.12 for sample instances of PS40-LC and PS40-TC.



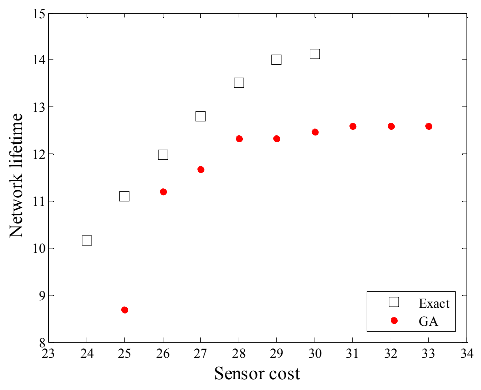
(a) Instance 15
RPI=0.0290 PI=0.0261 HI=0.0590

(b) Instance 18
RPI=0.0749 PI=0.0874 HI=0.1440

Figure 6.11 Plots of fronts for sample PS40-LC instances



(a) Instance 29
RPI=0.0266 PI=0.0295 HI=0.0544



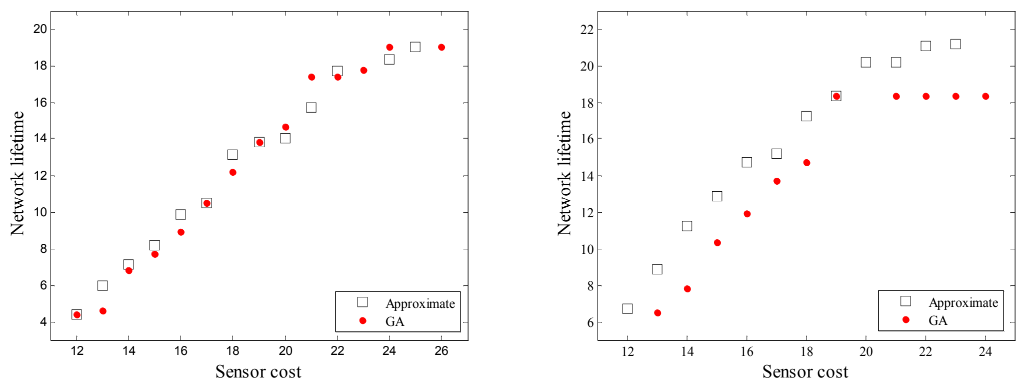
(b) Instance 23
RPI=0.2059 PI=0.2260 HI=0.4304

Figure 6.12 Plots of fronts for sample PS40-TC instances

Finally, we compare GA and approximate ε-constraint results for PS40-LC in Table 6.5 and in Figures 6.13 and 6.14. For the first 10 problem instances with known Pareto optimal fronts, we can calculate PI, RPI and HI for the GA and the approximate ε-constraint solutions. Approximate ε-constraint measures are still better than GA measures for these instances. Also, approximate ε-constraint can find 5.90 of 13.6 (43%) Pareto optimal solutions, whereas this figure is only 10% for the GA. However, the approximation has much longer CPU times than GA. For the remaining 20 problems, although GA finds more solutions in the front, the hypervolume measure of GA is smaller (worse) than that of the approximate approach. Sample front plots are given in Figures 6.13 and 6.14 for some PS40-LC and PS40-TC instances.

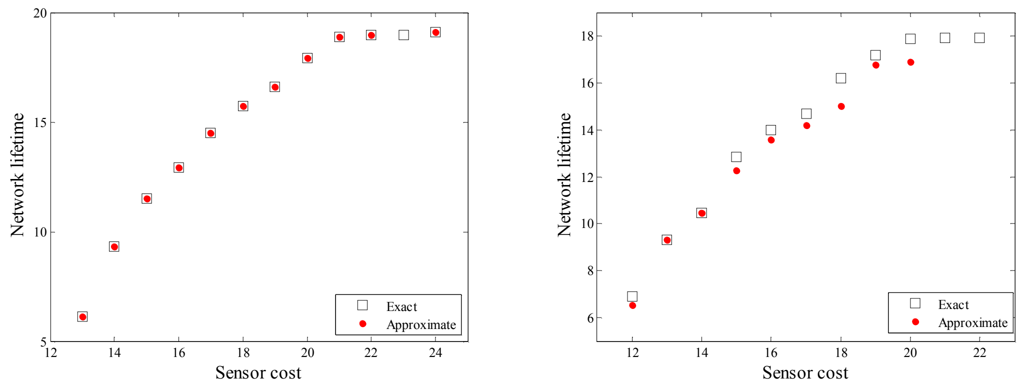
Table 6.5 Comparison of GA with approximate ε-constraint approach

						# of Pareto optimal solutions		
Algorithm (A)	# of instances	RPI	PI	HI	Hypervolume	(A)	(A)=Exact	CPU time (s)
GA	10	0.0464	0.0489	0.1164	-	12.80	1.30	798
Approximate	10	0.0253	0.0122	0.0442	-	10.90	5.90	54268
GA	20	-	-	-	0.6034	14.20	-	821
Approximate	20	-	-	-	0.6667	13.00	-	85983



(a) Instance 4 (b) Instance 27

Figure 6.13 Plots of approximate and GA fronts for sample PS40-LC instances



(a) Instance 18
RPI=0.0038 PI=0 HI=0

(b) Instance 20
RPI=0.0422 PI=0.0233 HI=0.0746

Figure 6.14 Plots of exact and approximate fronts for sample PS40-LC instances

6.4 Computational Results for Large Problems

We also test our algorithm on larger problem instances. We combine four problem instances to obtain a monitoring area of size 20 m × 20 m represented by 10 × 10 = 100 locations. After combining the instances, we have totally 200 targets dispersed on the monitoring area. The sensor and link capacities and battery energy are doubled compared to the loose capacity case since total demand increases significantly as total number of targets increases. We generate three different PS99 instances in terms of target characteristics (coordinates, coverage thresholds, demand rates).

We know that the number of Pareto optimal solutions depends on the number of possible locations close to the base station. The amount of data to be transmitted close to the base station increases as the sensors become closer to the base station as discussed in Chapter 5. Hence, the number of Pareto optimal solutions is highly dependent on the number of possible locations around the base station. When the number of possible locations around the base station is limited, the number of sensor types that can communicate with the base station decreases. This decreases the number of Pareto optimal solutions significantly. Therefore, we increase the number of possible locations as in Figure 6.15 around the base station to obtain problem instances with 112 possible locations.

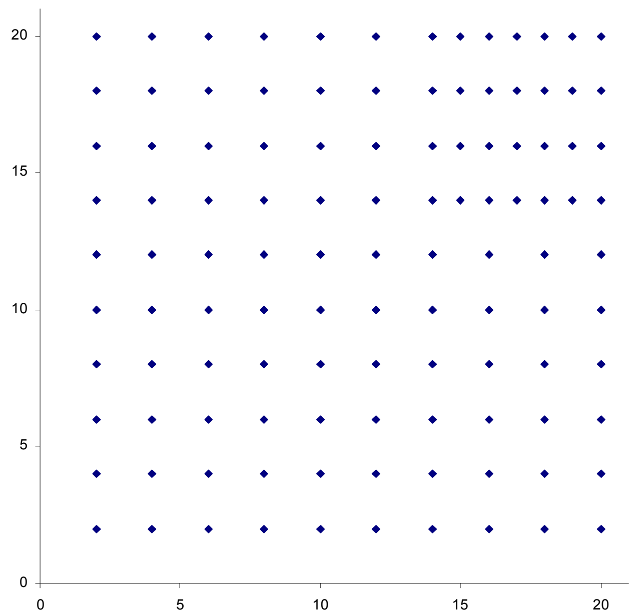


Figure 6.15 Possible grid location structure for PS111

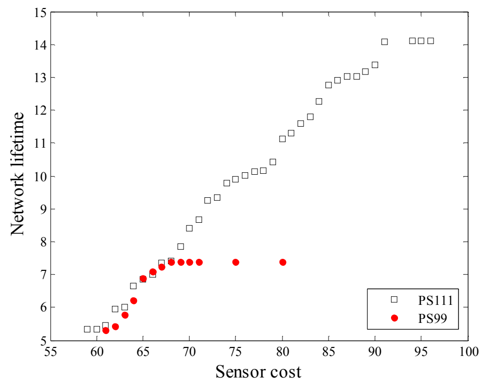
Exact solution for PS99 and PS111 instances cannot be found because of the problem size. However, we find the maximum and minimum sensor costs by solving single objective problems in order to be able to estimate maximum possible number of solutions in the Pareto optimal front. We also run the GA to approximate the Pareto optimal front. We set the parameters of GA as $N = 800$, $G \times C = 40000$, $pr_m = 0.2$ by considering the results of PS24 and PS40 instances. We summarize the results in Table 6.6.

Table 6.6 Summary of the results for PS99 and PS111 instances

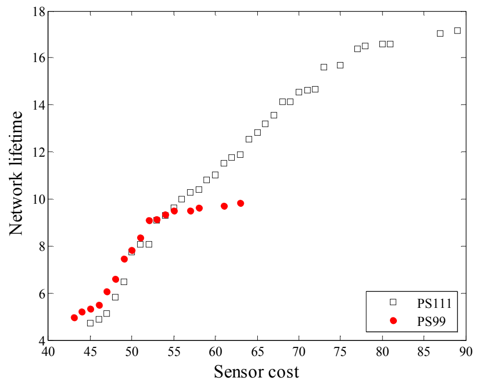
				GA	
Problem size	Instance no	Range of cost	Exact CPU time (s)*	# of solutions	CPU time (s)
PS99	1	57 - 78	50599	13	7675
	2	42 - 67	23955	17	7225
	3	55 - 74	24879	12	7447
PS111	1	56 - 94	72368	36	10502
	2	41 - 87	30595	36	11864
	3	53 - 91	32671	30	9534

* CPU time to find the minimum and maximum cost by solving the single objective formulation

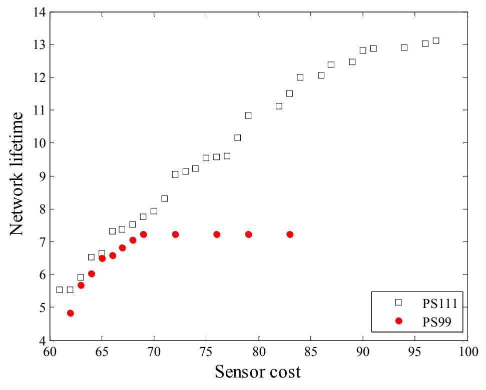
For PS111 instances maximum possible number of Pareto optimal solutions is significantly larger than that of PS99 instances as in Table 6.6. The number of feasible GA solutions for PS99 is about 60% of the maximum possible number of Pareto optimal solutions and these are found in reasonable CPU times compared to the total CPU times for finding the minimum and maximum cost levels. For PS111 instances, the number of feasible GA solutions is about 80% of the maximum possible number of Pareto optimal solutions and these are also found in reasonable CPU times compared to the total CPU times for finding the minimum and maximum cost levels. We also plot the nondominated solutions for PS99 and PS111 instances in Figure 6.16.



(a) Instance 1



(b) Instance 2



(c) Instance 3

Figure 6.16 Plots of approximate fronts for sample PS99 and PS111 instances

CHAPTER 7

CONCLUSION

In this thesis, we study the problem of energy efficient coverage and connectivity problem in wireless sensor networks. We formulate two single objective problems for minimization of total sensor cost and maximization of network lifetime. Our formulations satisfy connectivity, coverage and capacity constraints. We use the ε-constraint approach for solving the bicriteria problem exactly which depends on iterative solution of single objective problems. This approach requires long CPU times to generate the Pareto optimal solutions. CPU time increases significantly with this approach when the problem size increases.

We also propose a multiobjective genetic algorithm for solving the problem. Our GA is similar to NSGA-II of Deb (2002). Constraint handling is one of the challenging issues in our GA. We develop mechanisms to overcome the problem of infeasibility. We show that the GA approximates the efficient frontier well in reasonable time in most of our test problems, for which nondominated solutions are generated by the help of the exact solution approach. Furthermore, we provide the computational results for large sized problems.

Our experimentation with problems of size 24 and 40 having loose or tight capacity constraints lead to the following conclusions.

- Our performance measures, which consider convergence of GA solutions to the Pareto optimal front and diversity of solutions along the front, worsen (approximately doubled) as the problem size increases from 24 to 40 when capacity constraints are loose.
- Problems with tight capacity constraints are harder to solve for the GA compared to the loose capacity problems, whereas they are easier for the ε-constraint approach. Performance measures for tight capacity are about twice

as large as those for loose capacity. However, the problem size has less effect on the performance measures when the capacity constraints are tight.

- When the capacity constraints are loose, the GA solves problems of size 24 in one tenth of the ε-constraint CPU times. For problems of size 40, GA CPU time is about 100 times shorter than ε-constraint time.

- For the tight capacity case, GA CPU times are slightly longer than ε-constraint times with 24 possible locations, but they are 15 times shorter with 40 possible locations.

- For problems with 99 and 111 possible locations, the GA converges to a solution in about 160 minutes.

As a future research topic, one can modify ε-constraint approach to find Pareto optimal solutions by using some method as in the second phase of the two phase method. For example, the information from solutions of the previous iterations can be used in the current iteration. Efficient sensor deployments can be introduced in each iteration in order to reduce the problem complexity.

We can use the results of LP3 to increase efficiency of our GA. Sensitivity analysis results obtained by solving LP3 provides valuable information. For instance, capacity constraint of a deployed sensor may have a negative shadow price, and then we may think of deploying a sensor with a higher capacity in order to increase the lifetime. We may also use this idea as part of mutation or recombination operator during the search.

Considering multiobjective nature of the problem, focusing on a selected region by incorporating decision maker's preferences may be a good idea. The ε-constraint approach can be modified easily for this purpose. One may try to develop mechanisms to guide the search to the preferred regions for GA.

Instead of taking coverage as a constraint, one may consider it as a third objective to be maximized. Maximizing the area to be covered would clearly be in conflict with minimizing cost and maximizing lifetime. Different objectives such as minimization of total delay, total hop count or average path length can also be taken into consideration in the design of WSNs. According to the requirements of the application, our formulation and genetic algorithm can be modified for different objectives.

Special network requirements such as K-coverage or K-connectivity can be considered in our formulations. Integration of these properties requires additional constraints in the formulations. We should introduce additional mechanisms to handle these requirements in GA.

Finally, locating sensors in continuous space can be studied with similar objectives and constraints. This may require substantial changes in formulations and the GA starting with the solution representation.

REFERENCES

Al-Karaki J. N. and Kamal A. E., (2004), “Routing techniques in wireless sensor networks: a survey”, IEEE Wireless Communications, 11(6), pp. 6–28.

Akyildiz, F., Su W., Sankarasubramaniam Y., Cayirci E., (2002), “Wireless sensor networks: A survey”, Computer Networks, 38(4), pp. 393-422

Cardei, M., Thai, M.T., Yingshu Li, Weili Wu, (2005), "Energy-efficient target coverage in wireless sensor networks," 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), 3, pp. 1976-1984

Carle J., Simplot-Ryl D., (2004), “Energy efficient area monitoring for sensor networks”, IEEE Computer, 37(2), pp. 40-46.

Chakrabarty K., Iyengar S.S., Qi H., Cho E., (2002), “Grid coverage for surveillance and target location in distributed sensor networks”, IEEE Trans. Computers, 51, pp. 1448-1453.

Chang J. and Tassiulas L., (2000), “Energy conserving routing in wireless ad hoc networks”, 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), pp. 22-31.

Chang C., Chang H., (2008), “Energy-aware node placement, topology control and MAC scheduling for wireless sensor networks”, Computer Networks, 52(27)

Cheng P., Chuah C., Liu X., (2004), "Energy-aware node placement in wireless sensor networks", IEEE Global Telecommunications Conference (GLOBECOM), 5(3), pp. 3210-3214

Chunhua Z., Zhiqiang Yu., Peng C., (2007), "Optimal deployment of nodes based on genetic algorithm in heterogeneous sensor networks", International Conference on Wireless Communications, Networking and Mobile Computing (WiCom), pp. 2743-2746.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2002). “A fast and elitist multiobjective genetic algorithm: NSGA-II”, IEEE Transactions on Evolutionary Computation, 6(2), 182-197

Deb, K. (2001), “Multi-objective optimization using evolutionary algorithms“, Wiley.

Fabregat R., Donoso Y., Baran B., Solano F., (2005), “Multi-objective optimization scheme for multicast flows: a survey, a model and a MOEA solution”, 3rd international IFIP/ACM Latin American conference on Networking (LANC), pp. 73-86.

Ferentinos K. P., Tsiligiridis T. A., (2007), “Adaptive design optimization of wireless sensor networks using genetic algorithms”, Computer Networks 51(4), pp. 1031-1051

Gosh A., Das S. K., (2006), “Coverage and connectivity issues in wireless sensor networks”, Book Chapter: Mobile, Wireless and Sensor Networks: Technology, Applications and Future Directions, John Wiley & Sons.

Haimes Y.Y., Lasdon L.S., Wismer D.A., (1971), “On a bicriterion formulation of the problems of integrated system identification and system optimization”, IEEE Transaction on Systems, Man, and Cybernetics, 1(3), pp. 296-297.

Heinzelman. W.R., Chandrakasan, A., Balakrishnan, H., (2000), "Energy-efficient communication protocol for wireless microsensor networks", 33rd Annual Hawaii International Conference on System Sciences, pp. 2-10.

Hou Y.T., Yi Shi, Sherali H.D., Midkiff S.F., (2005), "On energy provisioning and relay node placement for wireless sensor networks", IEEE Transactions on Wireless Communications, 4(5), pp. 2579-2590.

Jourdan, D.B.; de Weck, O.L., (2004), "Layout optimization for a wireless sensor network using a multi-objective genetic algorithm", IEEE 59th Vehicular Technology Conference (VTC), 5, pp. 2466-2470.

Ke W., Liu B., Tsai M., (2007), "Constructing a wireless sensor network to fully cover critical grids by deploying minimum sensors on grid points is NP-Complete", IEEE Transactions on Computers, 56(5), pp. 710-715.

Krause, A., Guestrin, C., Gupta, A., Kleinberg, J., (2006), "Near-optimal sensor placements: maximizing information while minimizing communication cost", The Fifth International Conference on Information Processing in Sensor Networks (IPSN), pp. 2-10.

Michalewicz Z. and Fogel D. B., (2004), “How to solve it: Modern heuristics”, Springer

Pan J., Hou Y. T., Cai L., Shi Y., Shen S. X., (2003), “Topology control for wireless sensor networks”, 9th annual international conference on Mobile computing and networking San Diego, CA, USA, pp. 286 - 299

Pandey, S., Shaoqiang Dong, Agrawal, P., Sivalingam, K., (2007), "A hybrid approach to optimize node placements in hierarchical heterogeneous networks", Wireless Communications and Networking Conference (WCNC), pp.3918-3923.

Patel M., Chandrasekaran, R., Venkatesan, S., (2005), “Energy efficient sensor, relay and base station placements for coverage, connectivity and routing", 24th IEEE International Performance, Computing, and Communications Conference (IPCCC) , pp. 581-586

Patel M., Chandrasekaran, R., Venkatesan, S., (2006), “Energy-efficient capacity-constrained routing in wireless sensor networks", International Journal of Pervasive Computing and Communications, 2(2), pp. 69-80

Romer, K., Mattern, F., (2004), "The design space of wireless sensor networks", Wireless Communications, IEEE, 11(6), pp. 54-61.

Quintao F.P., Nakamura F. G., Mateus G. R., (2007), “Evolutionary algorithms for combinatorial problems in the uncertain environment of the wireless sensor networks”, Book Chapter: Evolutionary Computation in Dynamic and Uncertain Environments: Studies in Computational Intelligence, Springer.

Slijepcevic S., Potkonjak M., (2001), “Power efficient organization of wireless sensor networks”, IEEE International Conference on Communications, Helsinki, Finland, pp. 472-476.

Srinivasan, V., Chiasserini, C.-F., Nuggehalli, P.S., Rao, R.R., (2004), "Optimal rate allocation for energy-efficient multipath routing in wireless ad hoc networks", IEEE Transactions on Wireless Communications, 3(3), pp. 891-899.

Tubaishat, M., Madria, S., (2003), "Sensor networks: An overview," Potentials, IEEE , 22(2), pp. 20-23.

Ulungu E. L., Teghem J., Fortemps P. H., Tuyttens D., (1999), "MOSA method: A tool for solving multiobjective combinatorial optimization problems", Journal of Multi-criteria Decision Analysis, 8, pp.221-236

Wang Q., Xu K., Takahara, G., Hassanein, H., (2007), "Device placement for heterogeneous wireless sensor networks: Minimum cost with lifetime constraints", IEEE Transactions on Wireless Communications, 6(7), pp.2444-2453

Watfa M., (2007), “Practical applications and connectivity algorithms in future wireless sensor networks”, International Journal of Information and Technology (IJIT), 4(1), pp. 18-28.

Xue Y., Cui Y., Nahrstedt K., (2005), “Maximizing lifetime for data aggregation in wireless sensor networks”, Special Issue on Energy Constraints and Lifetime Performance in Wireless Sensor Networks in Mobile Networks and Applications (MONET),, 10(6), pp. 853-864.

Yi Z., Chakrabarty, K., (2005), "A distributed coverage- and connectivity-centric technique for selecting active nodes in wireless sensor networks", IEEE Transactions on Computers, 54(8), pp. 978-991.

Zongheng Z., Das, S., Gupta, H., (2004), "Connected K-coverage problem in sensor networks", 13th International Conference on Computer Communications and Networks (ICCCN), pp.373-378.

Zussman G., Segall, A., (2003), “Energy efficient routing in ad hoc disaster recovery networks ", Ad Hoc Networks, 1(4), pp. 405-421