COVERAGE, SINK LOCATION AND ROUTING PROBLEMS IN WIRELESS SENSOR NETWORKS

by

Evren Güney

BS, in IE, Boğaziçi University, 1999

MS, in IE, Boğaziçi University, 2002

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in FBE Program for which the Thesis is Submitted

Boğaziçi University

2009

ACKNOWLEDGEMENTS

First of all, I would like to thank to my thesis supervisor and long time advisor Prof. İ.Kuban Altınel for his extensive support and guidance during this thesis. His unique style of discipline and diligence lead and inspired me all throughout this study.

I am deeply grateful to Assoc. Prof. Necati Aras. As my thesis co-supervisor he had extensive contribution in this work. His attentive and delicate observations and smart solutions for complicated cases greatly improved the overall quality of the thesis.

I have a lot to thank to Prof. Cem Ersoy because of his deep knowledge on the thesis topic and also for his valueable recommendations and efforts to make this study more concrete.

I would like to express my gratitude to Prof. Taner Bilgiç and Assist. Prof Ayşegül Gençata Yayımlı for taking part in my thesis jury.

I am very gratefull to my family for all their support and motivation during this long academic journey. I want to thank to my fellow companions Nuray, Tayga, Hande, Mehmet, Burak and Onur for their support and days of accompany in the Grad. Lab.

I have to mention my thanks to my partner Ibrahim and all Akampüs family for backing me up in the days of my physical and mental absence at work.

This research has been partially supported by TÜBİTAK Grant no: 107M250 and by Boğaziçi University Research Fund Grant No: 05HA303.

ABSTRACT

COVERAGE, SINK LOCATION AND ROUTING PROBLEMS IN WIRELESS SENSOR NETWORKS

Recent advances in electronics and telecommunications have enabled the development of low-cost, low-power multifunctional sensor devices that are powered by batteries and communicating through wireless channels over small distances. When a large number of these devices work collaboratively for certain purposes, they form a network which is called a wireless sensor network (WSN). Due to their benefits and inherent characteristics, WSNs give rise to a broad-range of many real-life applications.

There are various design issues in the construction of WSNs. In this study, we focus on three important classes of problems, which are Point Coverage, Sink Location and Data Routing Problems. We propose a Binary Integer Programming (BIP) formulation to model the coverage problem and develop various heuristics to solve it. Both the Routing Problem (RP), which involves finding the most energy efficient sensor-to-sink routes and Sink Location Problem (SLP), which is determining the optimal sink locations are important design issues to extend the lifetime of a WSN. However, the joint optimization of these two problems results in a more efficient network design, so we develop several Mixed Integer Linear Programming (MILP) formulations and propose optimum solution techniques and heuristics to solve them. Finally, we solve the complete integrated model, which consists of the coverage, sink location and data routing problems simultaneously. We call this hard problem as the Coverage, Location and Routing Problem (CLRP) and develop new mathematical programming formulations. We propose a nested solution procedure, where in the first level of this approach, good sensor locations are sought by using some metaheuristics. In the second level, we solve the remaining sink location and routing problem using efficient heuristics.

ÖZET

KABLOSUZ DUYGAÇ AĞLARINDA KAPSAMA, ÜS YERİ YERLEŞİMİ VE YÖNLENDİRME PROBLEMLERİ

Elektronik ve iletişim dünyasındaki yeni gelişmeler sonucunda, düşük enerji ve maliyetli, çok işlevli, pille çalışan ve kablosuz iletişim kanallarını kullanan duygaçlar ortaya çıkmıştır. Bu cihazların topluca belli amaçlar için bir araya gelmesiyle oluşan ağa Kablosuz Duygaç Ağı (KDA) denmektedir. Çeşitli avantaj ve içsel özellikleri sebebiyle KDA'ların pek çok gündelik hayat uygulaması bulunmaktadır.

Bir KDA oluşturulurken çeşitli tasarım konularına dikkat etmek gerekir. Bu çalışmada KDA'lar ile ilgili üç önemli tasarım problemi olan Nokta Kapsama, Üs Yeri Yerleşimi ve Veri Yönlendirme Problemleri incelenmektedir. Kapsama problemi için 0-1 tamsayılı programlama gösterimi geliştirilmekte ve çeşitli sezgisellerle bu problem çözülmektedir. Duygaçlarla üs yerleri arasındaki en etkin rotaları bulan Yönlendirme Problemi ile en iyi üs yerini arayan Üs Yeri Yerleşimi Problemi, KDA'ların yaşam sürelerini uzatmak açısından önemli tasarım konularıdır. Ancak bu iki problemin tek bir eniyileme problemi olarak bir arada çözülmesi daha etkin bir ağ tasarımı sağlayacağı için, bu iki problemi içeren Karmaşık Tamsayılı Doğrusal Programlama gösterimleri geliştirilmekte, sezgisel ve en iyi çözümü bulan yöntemlerle etkin çözümler elde edilmektedir. Son olarak bu üç problemi birden içeren bütünleşik bir KDA tasarım problemi incelenmektedir. Bu çok zor problem Kapsama, Üs Yeri Yerleşimi ve yönlendirme Problemi (KÜYP) olarak adlandırılmış ve çeşitli matematiksel programlama modelleri geliştirilmiştir. KÜYP'ü çözmek için iki seviyeli bir içiçe çözüm yöntemi önerilmiş olup, birinci seviyede metasezgiseller ile en iyi duygaç yerleri aranırken, ikinci aşamada geriye kalan Üs Yeri Yerleşimi ve Yönlendirme problemi çözülerek ana problemimiz için etkin çözümler üretilmektedir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES ix
LIST OF TABLES x
LIST OF SYMBOLS/ABBREVIATIONS xii
1. INTRODUCTION 1
1.1. Basics of Wireless Sensor Networks 2
1.2. Important Design Issues 3
2. LITERATURE SURVEY 6
2.1. Coverage 6
2.1.1. Controlled Deployment 9
2.1.2. Random Deployment 13
2.2. Sink Location and Data Routing 16
2.2.1. Data Routing 17
2.2.2. Sink Location 18
2.2.3. Sink Location and Routing 19
3. COVERAGE PROBLEM IN WIRELESS SENSOR NETWORKS 22
3.1. Integer Programming Formulation for Differentiated Coverage 22
3.1.1. Effective Coverage Problem for Perfect Detection 24
3.1.2. Effective Coverage Problem for Imperfect Detection 25
3.1.3. Effective Coverage Problem for Uncertain Detection 26
3.2. Efficient Methods to Solve Effective Coverage Problem 29
3.2.1. Approximation algorithms to solve Effective Coverage Problem . 29
3.2.1.1. Linear programming relaxation and rounding 30
3.2.1.2. Greedy approximation 32
3.2.1.3. Special cases and refinements 44
3.2.2. Heuristics to Solve Effective Coverage Problem 48
3.2.2.1. Greedy Heuristic 48

3.2.2.2. Lagrangean Heuristic 50
3.2.3. Simple Improvement Strategies 53
3.2.3.1. Sweeping 53
3.2.3.2. Variable Fixing 53
4. SINK LOCATION AND ROUTING PROBLEMS IN WIRELESS SENSOR NETWORKS 55
4.1. Mixed-integer Linear Programming Formulations 56
4.1.1. Energy-aware Sink Location and Routing 57
4.1.1.1. A Single-commodity Flow Formulation 57
4.1.1.2. A Multi-commodity Flow Formulation 60
4.1.1.3. A p-median Formulation 64
4.1.2. Cost-aware Sink Location and Routing 66
4.1.2.1. A Single-commodity Flow Formulation 67
4.1.2.2. A Multi-commodity Flow Formulation 67
4.1.2.3. An Uncapacitated Facility Location Formulation . . . 68
4.2. Solution Methods 68
4.2.1. Lowerbounds Using Lagrangean Relaxation 69
4.2.1.1. Lagrangean Relaxation for Single Commodity Flow Formulations 69
4.2.1.2. Lagrangean Relaxation for Multi Commodity Flow Formulations 71
4.2.1.3. Lagrangean Relaxation for p-median Formulations . . 73
4.2.2. Computing Upper Bounds Using Largrangean Heuristics 76
4.2.3. Lower Bounds using DUALOC for p-median Formulations . . . 79
4.2.4. Lower Bounds Using Canonical Reduction for p-median Formulations 81
5. INTEGRATED MODELS FOR COVERAGE, SINK LOCATION AND ROUTING PROBLEMS 85
5.1. Mixed-integer Linear Programming Formulations 85
5.1.1. A Single-Commodity Flow Formulation 86
5.1.2. A Multi-Commodity Flow Formulation 88
5.1.3. A Complete Network Formulation 89

5.1.4. An Assignment Formulation 91
5.2. Solution Methods 93
5.2.1. A Nested Solution Approach 93
5.2.2. Metaheuristic Solution Procedures to Determine Sensor Locations 94
5.2.2.1. Initial Sensor Locations 95
5.2.2.2. Variable Neighbourhood Search 95
5.2.2.3. Tabu Search 96
5.2.2.4. Neighborhood Structures 97
5.2.2.5. Determining a Local Minimum 97
6. COMPUTATIONAL RESULTS 100
6.1. Coverage Problems 100
6.1.1. Test Environment 100
6.1.2. Perfect Detection 102
6.1.3. Uncertain Detection 102
6.2. Sink Location and Routing Problems 104
6.2.1. Test Environment 105
6.2.2. Quality of the Formulations 108
6.2.3. Performance of the Solution Methods 111
6.2.3.1. Small and Medium Size Instances 112
6.2.3.2. Large Instances 115
6.3. Integrated Coverage, Sink Location and Routing Problems 115
6.3.1. Test Environment 117
6.3.2. Experimental Results for Variable Neighborhood Search 119
6.3.3. Experimental Results for Tabu Search 119
7. CONCLUSIONS 126
REFERENCES 130

LIST OF FIGURES

Figure 1.1. A wireless sensor network on an agricultural field 2

Figure 3.1. Optimal sensor locations with and without constraints (3.4) 25

Figure 3.2. Algorithm RELAX_AND_ROUND 31

Figure 3.3. Algorithm GREEDY_APPROXIMATION 33

Figure 3.4. Algorithm GREEDY_HEURISTIC 50

Figure 3.5. Algorithm LAGRANGEAN_HEURISTIC for ECP 52

Figure 4.1. Algorithm LAGRANGEAN_HEURISTIC for SLRP 78

Figure 4.2. Algorithm NESTED_DUAL_HEURISTIC 81

Figure 4.3. Algorithm CANONICAL_REDUCTION 83

Figure 5.1. Algorithm VNS_COVER 95

Figure 5.2. Algorithm TS_COVER 98

LIST OF TABLES

Table 4.1. List of subgradient functions for different formulations 79

Table 6.1. Sensor specifications used in the Coverage Problems 101

Table 6.2. Percent deviations for perfect detection 103

Table 6.3. CPU times for perfect detection in seconds 104

Table 6.4. Percent deviations for uncertain detection 105

Table 6.5. CPU times for uncertain detection in seconds 106

Table 6.6. Key features of the models and methods 107

Table 6.7. Comparison of the formulations, $p = 1$ 109

Table 6.8. Comparison of the formulations, $p = 2$ 110

Table 6.9. Comparison of the formulations, $p = 3$ 111

Table 6.10. Comparison of the upper bounds: % deviation from the optimum value 112

Table 6.11. Comparison of the lower bounds: % deviation from the optimum value 113

Table 6.12. Comparison of the CPU times 114

Table 6.13. Comparison of the heuristics for large instances 116

Table 6.14. Properties of Sensors 118

Table 6.15. Results for VNS, p=1 120

Table 6.16. Results for VNS, p=2 121

Table 6.17. Efficiency of VNS 122

Table 6.18. Results for Tabu Search, p=1 123

Table 6.19. Results for Tabu Search, p=2 124

Table 6.20. Efficieny of Tabu Search 125

LIST OF SYMBOLS/ABBREVIATIONS

a_{ijk}	Coverage of a point i by a type k sensor at point j
b_i	Coverage requirement of point i
B	Available Budget
c_{jk}	Cost of deploying a type k sensor at point j
c_{ijk}	Routing cost of sending unit data by a type k sensor at point i to point j
$cov(i)$	Coverage amount of point i
d	Destination
d_{ij}	Distance between point i and point j
f_j	Fixed cost of installing a sink at point j
g_{ijk}	Assignment cost of a type k sensor at point i to the sink at point j
G_j	Subgradient function corresponding to dual parameter j
h_k	Fixed cost of type k sensor
i	Index of points in the sensor field
I	Number of sensors in the network
$\mathcal{I}$	Set of sensors in the network
j	Index of points in the sensor field
k	Index of sensor types
K	Number of sensor types
$\mathcal{K}$	Set of sensor types
m_{ij}	Miss probability at point i by a sensor at point j
n	grid size
N	Number of points in the sensor field
$\mathcal{N}$	Set of points in the sensor field
o	Origin
p	Number of sinks to be installed
p_{ij}	Coverage probability at point i by a sensor at point j
p_{ijk}	Coverage probability at point i by a type k sensor at point j

r_c	Coverage range of a sensor
r_k^c	Coverage range of a type k sensor
r_s	Sensing range of a sensor
r_k^s	Sensing range of a type k sensor
$\mathcal{S}$	Set of commodities
s_{ik}	Binary variables representing the availablity of a type k sensor at point i
s_{ik}^q	Binary variables / parameters representing the availablity of a type k sensor at point i generating commodity q
t	Iteration counter
$\mathcal{T}$	Set of sensors after canonical reduction
T_i	Coverage threshold for point i
u_{lm}^{ijk}	Binary variables representing the availability of arc *(l,m)* when a type k sensor at point i is assigned to the sink at point j
v_{ijk}^q	Amount of flow of commodity q from type k sensor at point i to sink at point j
w_i^q	Amount of flow of commodity q collected by the sink at point i
x_{jk}	Binary variable representing the availablity of a type k sensor at point j
x_{ijkl}	Amount of flow from a type k sensor at point i to type l sensor at point j
x_{ijkl}^q	Amount of flow of commodity q from a type k sensor at point i to type l sensor at point j
y_{ijk}	Amount of flow from a type k sensor at point i to sink at point j
y_{ijk}^q	Amount of flow of commodity q from a type k sensor at point i to sink at point j
z_j	Binary variable representing the availablity of a sink at point j
z_{GA}	Objective function value obtained by the Greedy Approximation
z_{GH}	Objective function value obtained by the Greedy Heuristic

z_{IP}	Objective function value of an integer programming problem
z_{LP}	Objective function value of a linear programming relaxation of a problem
z_{LR}	Objective function value obtained by solving a Lagrangean Relaxation of a problem
z_{RR}	Objective function value obtained by the Relax and Round Heuristic
α_k	Exponential decay parameter of a type k sensor
δ_k	Coverage decrease parameter of a type k sensor
η_t	Step Length parameter of subgradient optimization at iteration t
γ_{jk}	Gain factor by placing a type k sensor at point j
λ_k	Coverage ability of a type k sensor
μ_{jt}	Lagrangean multipliers at iteration t corresponding to constraint j
π_i	Dual parameters corresponding to coverage constraints
ρ_{jk}	Dual parameters corresponding to feasibility constraints
WSN	Wireless Sensor Network
Qos	Quality of Service
BIP	Binary Integer Program
RP	Routing Problem
SLRP	Sink Location and Routing Problem
MILP	Mixed Integer Linear Program
CLRP	Coverage, Location and Routing Problem
OR	Operations Research
ILP	Integer Linear Program
AGP	Art Gallery Problem
SLP	Sink Location Problem
ALA	Alternating Location Allocation
LP	Linear Program
ECP	Effective Coverage Problem

LR	Lagrangean Relaxation
SO	Sub-gradient Optimization
ELRP	Energy-aware Sink Location and Routing Problem
CLRP	Cost-aware Sink Location and Routing Problem
SELRP	Single-Commodity Energy-aware Sink Location and Routing Problem
SCF	Single-Commodity Flow
MELRP	Multi-Commodity Energy-aware Sink Location and Routing Problem
MCF	Multi-Commodity Flow
PELRP	p-median Energy-aware Sink Location and Routing Problem
SCLRP	Single-Commodity Cost-aware Sink Location and Routing Problem
MCLRP	Multi-Commodity Cost-aware Sink Location and Routing Problem
UCLRP	Uncapacitated Facility Location Cost-aware Sink Location and Routing Problem
UFLP	Uncapacitated Facility Location Problem
CR	Canonical Reduction
NDH	Nested-Dual Heuristic
LH	Lagrangean Heuristic
ECLRP	Energy-aware Coverage, Location and Routing Problem
SECLRP	Single-commodity Energy-aware Coverage, Location and Routing Problem
MECLRP	Multi-commodity Energy-aware Coverage, Location and Routing Problem
CECLRP	Complete Network Energy-aware Coverage, Location and Routing Problem
PECLRP	p-median Energy-aware Coverage, Location and Routing Problem
VNS	Variable Neighbourhood Search
TS	Tabu Search
GH	Greedy Heuristic
RR	Relax and Round

GA	Greedy Approximation
LB	Lower Bound
UB	Upper Bound

1. INTRODUCTION

Recent advances in mircoelectromechanical-systems, digital electronics and wireless communications have enabled the development of low-cost, low-power multifunctional sensor devices. These tiny devices are typically powered by batteries and communicate through wireless channels over small distances [1, 2]. Each sensor node is usually equipped with one or more sensors, one or more transceivers, processing, storage units and actuators. A sensor node by itself has severe resource constraints, such as limited memory, battery power, signal processing, computation and communication capability; hence it can sense only a small portion of its environment. When a large number of these devices work collaboratively for certain purposes, they form a network to form a wireless sensor network (WSN). Some of the many benefits of WSNs are their ability to:

- Build large scale networks
- Implement sophisticated protocols
- Reduce the amount of communication required to perform tasks by distributed and/or local pre-computations
- Implement complex power saving modes of operation depending on the environment and the state of the network

Due to these benefits and inherent characteristics of wireless sensor networks, they give rise to a broad-range of many real-life applications such as the applications in military, homeland security, health care, environment, agriculture, manufacturing industry, logistics, smart home or office design and other areas. An ageicultural application is shown in Figure 1.1. The detection or sensing ability of a wireless sensor can be of different types such as seismic, acoustic, chemical, optical with respect to the application.

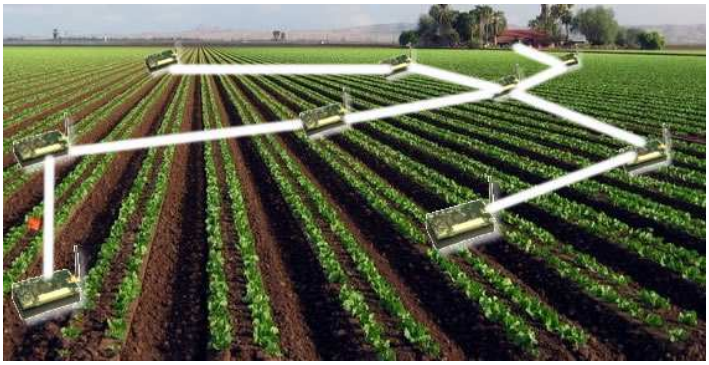

Figure 1.1. A wireless sensor network on an agricultural field

1.1. Basics of Wireless Sensor Networks

The purpose of deploying a WSN is to collect relevant data for processing and/or reporting. There are two types of reporting: event-driven and on-demand. In an event driven reporting case, when one or more sensors detect an event, they collaboratively work to further observe the event and report it to a central unit. This is generally used for intrusion detection, surveillance or natural disaster (fire, earthquake, tsunami) applications. In an on-demand driven case, usually the sensors wake up at predefined intervals, observe their environment and report back to a central unit. These are usually data heavy applications in agriculture, climate control or inventory control [3].

In wireless sensor networks, the energy source provided for a sensor is usually battery power which has not yet reached the stage for the sensors to operate for a very long time without being recharged. Moreover, in many applications sensors are intended to work in remote or hostile environments, such as in a battlefield area, over a volcano or in a forest. As a consequence, it is undesirable or impossible to recharge or replace the battery of a sensor. However, a long system lifetime is always expected by many applications. The system lifetime, which can be measured by the time until first/some/all nodes have been drained out of their energy, or the network no longer provides an acceptable level of operating quality, directly affects the network usefulness. Therefore, conserving energy resource and prolonging system lifetime is an important issue in the design of large-scale wireless sensor networks [4].

The area of interest where the wireless sensors are desired to be operating is called

a “sensor field”. A sensor field can be in any dimensions. It can be an agricultural field of two dimensions or could be a segment of an ocean of three dimensions. It can even be modeled as a very thin linear area (such as a border) and assumed to be in single dimension.

Since sensors have different physical properties, there exists different ways of modeling their detection capabilities [5, 6]. “Perfect detection” is the simplest case, where a sensor can precisely detect any target within its sensing range. This assumption is usually assumed to be unrealistic for real life cases, because it is generally true that a sensor can detect better if the target is closer. “Imperfect detection” models are used to include this scenario. A third option is to incorporate the inherent probabilistic nature of detecting a certain target with a certain probability. This kind of detection model is called “uncertain detection”.

The transmission range of wireless sensors are limited as a consequence of energy and size constraints, hence they cannot communicate through large distances. Therefore, each sensor needs to transmit its data to a nearby central unit which is called a “sink” or “base station”. Sinks are assumed to be larger devices with unlimited energy supply and long-range transmission capabilities, such as internet or satellite communication.

1.2. Important Design Issues

There are various design issues in the construction of a wireless sensor network. One of the most important problems addressed in the literature is the sensor coverage problem. The problem is centered around a fundamental question: “How well do the sensors observe the physical space?” As pointed out in [7] the coverage problem is a measure of the quality of service (QoS) of the sensing function and it is subject to a wide range of interpretation due to large variety of sensors and applications. Overall the ultimate goal is to have each location in the physical space of interest to be within the sensing range of at least one or more sensors [3]. We propose a Binary Integer Programming (BIP) formulation to model the point coverage problem and develop

various heuristics to solve it for large networks.

In a typical WSN, each sensor collects and processes data and tries to send this information to a sink. Since the sensors' communication ranges are limited, the data packets carrying the sensed information usually have to follow multihop paths. The Routing Problem (RP), which involves finding the most energy efficient sensor-to-sink routes, given the sensor and sink locations, is an essential problem in WSNs, because data transmission is an energy consuming task [5].

Determining the optimal sink locations is also an important design issue to extend the lifetime of a WSN. The average number of hops to reach the data sink becomes a critical factor and the sink location dramatically affects the duration of the effective operational period of a WSN. In the majority of the research studies, sink locations are either fixed or assumed to be known a priori. However, this is not the best choice, because locations of the sinks considerably effect the optimum routes. Hence determining the optimum sink locations plays an important role for maximizing the network lifetime due to improved energy consumption [8].

The joint optimization of sink locations and data routing, namely the Sink Location and Routing Problem (SLRP) has also received considerable attention as it results in a more efficient network design [9]. We develop various mathematical formulations to model this problem as a Mixed Integer Linear Program (MILP) and propose optimum solution techniques and heuristics to solve them. Very efficient solution techniques to design large-scale WSN's is possible with these techniques.

Finally, we solve the complete integrated model, which consists of the differentiated (point) coverage, sink location and data routing problems on heterogeneous sensor networks. We name this integrated problem as Coverage, Location and Routing Problem (CLRP). We develop new mathematical programming formulations, which consider these three design issues simultaneously in a single model. However, these formulations can be solved using commercial solvers only for small-sized problems. For medium and large sized problems, we propose a nested solution procedure. In the first

level of this approach, we try to find good sensor locations that satisfy coverage and budget constraints, by using some metaheuristics. In the second level, we efficiently solve the remaining sink location and routing problem using Lagrangean Heuristic and Nested-Dual Heuristic. To the best of our knowledge, this approach is new in this area of research.

There are seven chapters in this thesis. We give a brief survey of the related literature in the following chapter. In Chapter 3, the Coverage Problem is analyzed. Chapter 4 is devoted to the Sink Location and Routing Problem, whereas in Chapter 5, we combine the problems of the previous two chapters to formulate them as a single-integrated model. Chapter 6 presents the Computational Results and finally, Chapter 7 concludes the thesis.

2. LITERATURE SURVEY

There is an immense literature addressing many issues related to Wireless Sensor Networks. WSNs are in the research domain of Computer Sciences, Electrical-Electronical Engineering and to some extend Operations Research and Mathematical Programming. In this chapter only the ones that are most relevant to our approach are reported.

2.1. Coverage

A typical sensor have an omnidirectional antennae and can monitor a disk around itself, whose radius is referred to as sensing range [3]. In this study various different coverage formulations, their assumptions and solution techniques are overviewed. Before getting into details of them, we present some more basic definitions about WSNs.

The surveillance area considered in WSN applications, which is called as the "sensor field" is application specific. Some applications model the sensor field as a two or three dimensional subspace, while others discretize it to a possibly large but finite set of points. In the finite space case a popular method is to use a grid structure of two or three dimensions as the sensor field. In this work, we study on a finite space domain where the sensor field is made up of finite number of points, which can be on any dimension.

Efficient resource management and providing reliable QoS are two of the most important requirements in sensor networks as mentioned in [10]. However, due to severe resource constraints and hostile environmental conditions, it is non-trivial to design efficient node deployment strategies that would minimize cost, reduce computation and communication overhead, provide a high degree of coverage, and maintain a globally-connected-network simultaneously. Challenges also arise because of the fact that most often topographical information about the monitoring region is unavailable, and also such information may change over time due to the presence of obstacles [11].

Coverage problems can be classified in the following two types: point coverage and area coverage [3]. In this study, we focus on the point coverage which is actually a subset of the area coverage, where the objective is to deploy the sensors in strategic ways, such that required coverage is achieved with minimum deployment cost according to the needs of any underlying application.

A typical WSN can be homogeneous, consisting of identical sensors, where all the sensors have exactly the same technical properties. It becomes heterogeneous, when the sensors have different characteristics. Although most of the studies consider homogenous networks, few studies including ours extend the state of the art to deal with heterogeneous WSNs [5].

A sensor's primary duty is sensing and transmitting its data, but due to limited transmission ranges and energy considerations, sensors can send information through other sensors. This is called relaying, where the relay sensors do not only transmit their data, but also data sent by other sensors. In most of the applications, each sensor can sense, transmit and relay. However, in some specific applications relay sensors are deployed to improve the communication [4].

The sensing ability of sensors is also a research issue. In the basic models, it is assumed that sensing is "perfect" where a sensor always detects a target remaining within its sensing range. This is called binary (0/1) sensing model [12]. Most of the early work on WSNs assume perfect detection, although it is not very realistic. As a consequence of technical incapabilities or environmental conditions the closer a sensor to its target the better it provides coverage. Hence it is much realistic to use a sensor field intensity approach, where the coverage amount decreases with distance [13]; this is known as the "imperfect" detection model. A third approach is to use "uncertain" detection which models sensing probabilistically, because of the uncertainty associated with sensor readings in real life [14].

The coverage requirements is also an issue of research. Most of the works assume a non-differentiated coverage for the sensor field, which implies all the area of interest

should be covered equally, but in some of the real life applications it can be required to have a higher degree of coverage in certain regions of the sensor field, especially in the security or surveillance applications. A differentiated coverage paradigm should be applied in such cases and we include this issue in our work.

Each individual sensor is desired to be reachable in terms of communication in a fully-functional WSN, where any sensor in the network is within the communication range of some other sensors without forming any disconnected subsets. When all the sensors are connected to the sink (probably via other sensors) and able to communicate with the sink, then the WSN is said to be "connected". The degree of connectivity is also a design issue. A WSN is called k-connected, when each sensor can find at least k different paths to communicate with its neighbours [3].

Since wireless sensor networks is a new area of research, the literature does not go back to more than a decade. Earliest works on coverage mainly focus on sensor communication, localization and data fusion for a given sensor network topology [5]. In most of these early works, optimization has not been the primary concern and mostly simulation studies are applied for performance assessment.

First mathematical models and applying Operations Research (OR) techniques to solve these models appear in [12]. Before getting into details of previous works, it is important to mention two main lines of research on the coverage problem. Since there are many different real life applications of WSNs, each line of research fulfills the requirements of the application areas in its domain. The first line of research, which is also the one we follow, assumes that sensors can be deployed in a controlled way to the sensor field. Agricultural monitoring, industrial control or some military base surveillance applications lie in this area, where it is possible to deploy sensors on exactly the required locations. Here, the primary purpose is to cover the required area with a desired level of coverage with minimum resources. In this line of research, usually decision making is centralized.

The second line of research, on the contrary, assumes that it is usually impossible

to deploy sensors on exact/predetermined locations, because of hostile or unreachable environmental conditions. Hence the deployment process is uncontrolled, where the sensors are dropped from an aeroplane, or the locations of the sensors are not static or the conditions in the sensor filed is so dynamic that sensor locations can change in time. Environmental monitoring, battlefield observation or border surveillance applications fit into this category. In uncontrolled deployment scenarios, the number of sensors deployed is much more than the minimum necessary amount, so there is a favorable redundancy. The main purpose is to determine disjoint sensor sets, where each set can collectively cover the whole sensor field. Each set gets into active status to accomplish its task, while the other sensors in the remaining sets, to conserve energy, wait in a sleep mode until their turn comes. The primary objective in these type of applications is to maximize system lifetime, where decision making is desired to be decentralized in most cases.

2.1.1. Controlled Deployment

Chakrabarty *et al.* [12] formulate the controlled-deployment heterogeneous coverage problem as a non-linear integer program where the sensor field is represented by a two-dimensional grid, with two types of sensors and perfect detection ability. They apply some well-known linearization techniques to cope with the non-linearity in their model, but their mathematical model is inefficient and they can only solve small problems to optimality.

Extending this work, Dhillon *et al.* [14] develop a greedy heuristic algorithm to minimize sensor deployment costs. Their heuristic is capable of incorporating obstacles and differentiated coverage requirement in the sensor field, where the sensing ability of the sensors are based on imperfect detection following an exponential probability distribution. Since the coverage is defined probabilistically, they define the miss probability of a sensor $m_{ij} = 1 - p_{ij}$ where p_{ij} is the coverage probability of point j by a sensor located at point i. Assuming independence of sensors the overall miss probability of a point is the product of individual miss probabilities. Their greedy heuristic deploys sensors on the points such that the sum of the all miss probabilities decreases

the most. The algorithm terminates when all the points are covered sufficiently. In their following work [15], they propose a different point of view and propose two new heuristics. The procedures, which are called *MAX_AVG_COV* and *MAX_MIN_COV*, determine the locations of the candidate sensors such that maximum average coverage and maximum of the minimum point coverages are the selection criteria.

A similar work is by Zou *et al.* [16], where they develop a sensor placement algorithm to determine the minimum number of sensors and their locations such that every grid point on the sensor field is covered with a minimum confidence level. Their work differentiates from the previous ones by the uncertainty of the sensor locations, which are modeled as random variables following a Gaussian probability distribution. Polynomial-time greedy algorithms *MIN_MISS* and *MAX_MISS* are proposed to deploy the sensors such that the decisions are given according to the minimum miss probability or maximum miss probability of each point. In their previous work [17], they propose a Virtual Force Algorithm, which is originally developed for the deployment of mobile robots, but applied to wireless sensor networks. After an initial deployment of the sensors, a one-time repositioning is carried out to improve coverage. Based on repulsive and attractive forces a balanced network design is obtained.

Jourdan *et al.* [18], use genetic algorithm to determine locations of homogeneous sensors with perfect sensing, where they optimize two objectives: sensor coverage and system lifetime. Since these two objectives conflict with each other they consider pareto optimal solutions. Elba *et al.* [19], follow the same line to develop a new model, which they solve using genetic algorithm and simulated annealing. They report that their genetic algorithm performs better.

In Zhang *et al.* [20], a distributed solution that utilizes a small number of mobile sensors to improve coverage performance on a sensor field with randomly pre-deployed sensors is proposed. Three algorithms are presented, where the first one is an Integer Linear Programming (ILP) formulation to solve small size problems. The remaining two, namely the spiral-in algorithm and the geometry algorithm are used for larger instances. Seo *et al.* [21], work on a similar problem where the sensors are first located

randomly then moved to their optimum locations to maximize coverage. They develop a hybrid-steady state genetic algorithm to determine the best sensor locations and they test their model with real life inputs, such as sensor capabilities, terrain features, target identification parameters and direction of target movements of real military vehicles.

Wang and Zhong [22] propose a set covering formulation, where their objective is to cover the sensor field with minimum number of heterogeneous sensors. They provide a Linear Programming (LP) relaxation based heuristic with a proven approximation ratio. Although theoretical lower bounds are weak, they declare that actual solutions provide better lower bounds. They assume perfect sensing over a sensor field with non-differentiated coverage requirements. Although they deal with the same problem, this formulation is more efficient than the formulation given in [12]. The formulation in this work is also the closest to our work, where we generalize it by handling different sensing types and differentiated coverage requirements.

Under differentiated detection requirements a MILP is developed to formulate the sensor deployment problem in Zhang *et al.* [23]. They assume a homogeneous WSN and uncertain detection. A greedy deployment algorithm, *DIFF_DEPLOY*, is proposed to solve the problem efficiently. This algorithm makes use of the Linear Shift Invariant property, which justifies the representation of the relationship among the deployment strategy, detection model and collective miss probability distribution by matrix multiplication. At each iteration of the algorithm the difference between the remaining coverage requirement and the logarithmic collective miss probability distribution provided by the nodes that are already deployed is calculated and a sensor is placed at the location with the maximum difference. According to the authors *DIFF_DEPLOY* overperforms the *MIN_MISS* algorithm of [16].

Qishi *et al.* [24] propose a sensor deployment problem defined on a planar grid sensor field, which is similar to a knapsack problem with minimum coverage and maximum budget constraints. The objective is to maximize the coverage probability of each grid point. A heterogenous probabilistic sensing model with different sensor capabilities in terms of coverage range, detection quality and costs are incorporated in

their model. They show that this problem is NP-complete and provide approximate solutions using a two-dimensional genetic algorithm.

A similar model of the coverage problem with a minimax objective function is formulated in Cavalier *et al.* [25]. Their formulation suffers from the non-linearity of the objective function and non-convexity of the constraints. Hence they propose a heuristic utilizing Voronoi polygons, where the sensor locations are optimized using a so called "Toward the Largest Peak"approach. At each iteration of this algorithm, sensor locations are fixed and the remaining problem can be solved effectively using Voronoi polygons. The algorithm relocates the sensors and calculates the maximum miss probability. The procedure continues until the maximum miss probability cannot be further minimized.

Lazos *et al.* [26] work on the deployment problem of maximizing the detection probability of a mobile target in heterogeneous WSNs. They derive analytic formulas by mapping the target detection problem into line-set intersection problem. The mathematical expressions for the maximum detection probability are not practical for large networks, hence the authors provide lower and upper bounds.

The sensor deployment problem resembles the famous art-gallery problem (AGP) [27]. In the AGP, the objective is to determine the minimum number of guards required to cover the interior of an art gallery. Several variants of art gallery problems have been studied, including mobile guards, exterior visibility and polygons with holes [28]. WSN coverage problem differs from the AGP in two fundamental ways: First sensors can have different ranges (heterogeneous property), second the energy concern of WSNs is not available in the AGP.

There are also some studies which do not provide any mathematical models to formulate the coverage problem. These models mostly utilize geometrical analysis to develop various deployment strategies. Meguerdicihian *et al.* [7] study coverage in built-large-scale wireless sensor networks. The definition of coverage as a quality of service is initially mentioned in their work. Using Voronoi diagrams and Delauney

triangles they develop a polynomial time worst and best-case coverage algorithms, which correspond to the construction of maximal breach path and maximal support paths in WSNs. They conclude that by placing additional sensors on these paths coverage can be improved. In Biagioni *et al.* [29], three types of geometrical deployment strategies, namely square, triangular and hexagonal tiling strategies are compared to provide maximum coverage with minimum number of sensors. They point out that hexagonal tiling performs the best among them. Kar *et al.* [30] provides a geometric interpretation of the coverage problem to develop a near optimal connected cover where the sensors are deployed such that the entire sensor field is covered. They assume that sensing range is equivalent to transmission range which is too restrictive.

Another interesting point of view on the coverage problem is presented in the work of Toumpis [31]. He studies the massively dense WSNs in a macroscopic level where individual sensor information is not important. Instead, density functions representing the sensors are considered. An optimal sensor deployment strategy so that the minimum number of sensors (minimum sensor density per unit area) are deployed to deliver the created traffic by the sensors to the sinks. Some analogies with electrostatics are provided to model and solve the optimization problem to obtain the desired sensor densities.

2.1.2. Random Deployment

Up to this point, controlled sensor deployment for the coverage problem is presented. Now we will briefly cover the second line of research, where the sensors are pre-deployed randomly or according to a probabilistic distribution. Using some localization techniques, the exact locations of the sensors are determined. Here, it is assumed that a large number of sensors is deployed in order to build a redundant network, so that only a portion of the sensors are in active mode to fulfill their coverage and transmission tasks. The remaining sensors are in sleep mode to conserve energy and wake up when their turn comes.

The work by Slijepcevic and Potkonjak [32] is one of the first studies in this

context. They develop a most-constrained least-constraining heuristic to obtain mutually exclusive sensor sets over a randomly deployed homogeneous WSN with non-differentiated coverage requirements. Each set covers the sensor field and the aim is to find the maximum number of disjoint sets so as to maximize the system lifetime.

The coverage problem is formulated as a binary integer programming problem in Meguerdichian and Potkonjak [13]. In that work the aim is to find a set of sensors among the all deployed ones with minimum cardinality, satisfying the coverage requirements. They compare the performance of their BIP formulation with a greedy heuristic. Li *et al.* [33] follows the same direction and develop an efficient distributed algorithm to solve the same problem. They extend [13] by considering the imperfect detection model.

The relationship between coverage and connectivity is an important issue, because if a WSN cannot provide connectivity, then the data gathered at some points in the network may not be used, hence even the coverage task of the disconnected area is accomplished, it may not be communicated to the decision center. In Wang *et al.* [34] and Zhang *et al.* [35] coverage and connectivity is related with each other. They both proved that, if the communication range of the sensors, r_c is at least twice the range of sensing, r_s, then coverage implies connectivity in homogeneous WSNs. This result is not valid for heterogeneous networks [36]. In real life most sensors have their communication range larger than their sensing range, so in many cases it is reasonable to assume that, $r_c \geq 2r_s$ and only deal with coverage. It is also proved in [34] that with the same range properties, k-coverage implies k-connectivity, which means that if each point is covered by at least k sensors, then each sensor can send their data through k different paths. Tian *et al.* [37] provide the complete proof of this relationship and show that, it is a sufficient condition and the tight lowerbound to ensure complete coverage implies connectivity.

Carle *et al.* [38] consider the coverage problem for a homogeneous sensor network with perfect detection. They develop a method to select and update energy-efficient connected active sensor sets to extend the network lifetime while maintaining full cov-

erage, full connectivity and data aggregation.

The coverage problem with the objective of maximum coverage with fault tolerance over a homogeneous sensor network is analyzed in Ishizuka *et al.* [39]. They develop three types of stochastic deployment models, which are simple diffusion, constant placement and r-random placement. All these models follow certain probability distributions and try to model the inherited uncertainty.

The work of *Lin et al.* [40] considers a controlled deployment together with a determination of active and sleeping sensors. They provide a set covering based MILP formulation with some additional constraints to handle the sensor discrimination. Their formulation assumes heterogeneous sensors with perfect sensing ability. Instead of solving the MILP to optimality, they provide a simulated annealing heuristic to obtain good solutions.

Onur *et al.* [41] work on the breach path detection problem where a coverage scheme with uncertain sensing ability of the sensors is considered. Given the sensor locations, the breach path detection problem can be formulated as a Linear Programming Problem. Dijkstra's shortest path algorithm is used to solve the LP. With various different settings, the trade off between the coverage quality and the number of necessary sensors is analyzed.

Cardei *et al.* assume the sensor locations are given in [42] and develop a MILP formulation to determine a collection of the subset of sensors that maximize the system lifetime while providing sufficient coverage. They prove this problem to be NP-hard and propose two heuristics. The first one is an LP relaxation based heuristic with a performance guarantee. The second one is a greedy heuristic which runs faster, but does not provide a performance guarantee.

Zou *et al.* [43] work on a similar setting with the objective of determining the minimum number of active nodes on a pre-deployed homogeneous WSN with uncertain detection. They formulate this problem as a MILP with coverage constraints. They

use some linearization techniques to handle the non-linearity in their initial formulation. Since the remaining problem is still intractable for large instances, they develop a heuristic solution technique, which they call as the coverage centric active nodes selection. This heuristic utilizes a distributed approach, where each sensor has only the neighborhood information and the algorithm can construct a network with both providing required coverage and connectivity.

Yang *et al.* [44] also formulate the minimum number of sensor selection problem with coverage constraints. Although the formulation is very similar to our model, the fundamental difference is that our model is a generalization of theirs with the addition of heterogeneous sensor capabilities and costs, differentiated coverage requirement and incorporating the probabilistic sensor coverage. They prove that the MILP to formulate this problem is NP-hard and provide an approximation algorithm based on the LP relaxation of the MILP. They also develop a cluster-based solution protocol with the objective of minimizing energy consumption to obtain distributed solutions satisfying both k-coverage and k-connectivity.

A recent work of Cao *et al.* [45] studies a power-aware node placement scheme in linear wireless multi-media sensor networks, where the data traffic is relatively large. Given the deployed homogeneous sensors, the aim is to determine the role of the sensors. While each sensor is sensing and relaying, some sensors are selected to be clusterheads and aggregate the data of the sensors that are prior to them in the network. Mathematical formulations are provided to determine the scheme to minimize the average energy consumption per node and maximize the network lifetime by designing the optimal node numbers and node distances in the network. Although the problem is formulated, because of the complex mathematical terms in both the objective and constraints, a simulation analysis is carried out.

2.2. Sink Location and Data Routing

Each sensor processes the data it collects within its sensing range to generate the information transmitted to sinks either directly or through other sensors within

the communication range. Data transmission is an energy consuming sensor activity, which makes the determination of energy efficient sensor-to-sink message flow paths or routes one of the fundamental issues in the design of WSNs. This is known as the Routing Problem (RP) in which both sensor and sink locations are assumed to be given. Also, the location of the sinks are an important issue in the design of a WSN, because it dramatically affects the energy consumption in the network. Hence, determining the optimum sink location, namely the solution of the Sink Location Problem (SLP), is crucial for minimizing the energy consumption and maximizing the network lifetime. Although these two problems can be tackled separately, it will result in a more efficient WSN design when they are modeled under a single framework, where both the sink locations and data routes are optimized simultaneously. We call this joint optimization problem as the Sink Location and Data Routing Problem (SLRP).

2.2.1. Data Routing

Although early research in data routing mostly concentrates on protocols, which are surveyed in [46, 47], a considerable amount of effort is also devoted to the determination of optimal routes recently. Related formulations are usually linear or nonlinear network flow models with additional constraints maximizing the network lifetime or minimizing total routing energy. For example, Younis *et al.* [48] consider a hierarchical network structure in their earlier work and formulate the problem as a one-to-many shortest (i.e. least energy) path problem after determining a weight that is related to the unit energy usage for every pair of nodes. Another example is Sankar and Liu's multi-concurrent flow formulation [49] which maximizes the network lifetime. It is solved approximately by a distributed routing algorithm.

Kim *et al.* [50] use the hierarchical network structure proposed by Hou *et al.* [51], and give a linear programming formulation. Although the constraints of the RP models are usually standard flow balance constraints or their extensions with additional energy inequalities, the objective functions are not. For example, in [52, 53, 54], the network lifetime is defined as the minimum of individual sensor lifetimes, which is directly maximized in the first one for the situations when the information generated

by the sensors is constant or variable. Lifetime maximization is achieved by using feasible aggregations of sensor-to-sink routes in the second one, and by minimizing the maximum normalized power consumption of the bottleneck sensors in the third one.

The objective function of the models given in [55, 56] also maximizes the network lifetime; but this is defined as the time until the first sensor of an already deployed WSN runs out of energy. The optimization model developed by Madan and Lall [55] is close to the one of Chang and Tassiulas [52]; but the solution methodology is based on the Lagrangean dual of the linear programming formulation resulting in efficient partially and fully distributed subgradient algorithms. The formulation and the algorithms are also extended to handle more realistic situations in WSNs. Chang and Tassiulas' work is also important, because most of the optimization models that jointly locate sinks and find data routes originate from their network flow formulation. Ergen and Varaiya [56] start with a similar linear programming model in their recent work; the main difference is that the energy inequalities include also terms for energy spent during sensing and sleeping. Then they revise this model and make it capable to determine not only the energy efficient routes but also the ones with given delay guarantee.

The work of Pham *et al.* [57] can be seen as a major deviation since their model maximizes the time WSN covers an area subject to coverage and connectivity restrictions in addition to flow balance equations. The mathematical programming model of Ciciriello *et al.* is also different; they formulate the routing problem as a multi-commodity design problem with the objective of minimizing the number of links exploited [58].

2.2.2. Sink Location

The determination of optimal sink locations considerably affect the routes and therefore the energy spent on data transmission. Optimal placement of single or multiple sinks in a sensor network is addressed in a growing number of works. For example, Bogdanov *et al.* [8] show that sink locations affect the flow data rate which in turn has an influence on the power efficiency of the WSN. In a recent paper, Vincze *et*

al. [59] give a mathematical programming model that determines the locations of the sinks minimizing the average Euclidean distance between the sinks and sensors. This is eventually the so-called multi-facility Weber problem [60], which can be solved approximately using the Alternating Location-Allocation (ALA) heuristic [61]. ALA is very similar to K-means algorithm [62], but the latter one uses squared Euclidean distance to measure distances. A K-means type approach is also used by Oyman and Ersoy [63] to locate a given number of sinks in their earlier work where they treated SLP as a clustering problem.

2.2.3. Sink Location and Routing

The joint optimization of sink locations and data routing, namely the Sink Location and Routing Problem (SLRP) has also received considerable attention. According to Gandam *et al.* [64] using multiple sinks and periodically changing their locations increases the network lifetime. They split the lifetime of the WSN into equal periods called rounds each of which consists of a given number of time frames and assume that every sensor transmits one packet of data at the beginning of every time frame. They formulate two mixed-integer linear programming problems (MILPs) to determine sink locations at the beginning of each round and data flows over the time frames so that the maximum of sensor energy consumptions and the total sensor energy consumption are minimized. The work of Gandam *et al.* is also important since it is one of the earliest studies about the effect of sink mobility on the network lifetime: their MILP models calculate optimal locations of multiple sinks for every round.

The work by Kim *et al.* deals with sink location as well and includes an MILP formulation [50] that maximizes the amount of data sent from sensors to sinks. Luo and Hubaux [65] formulate the problem of lifetime maximization as a min-max problem by combining sink mobility together with data routing. An optimal solution of the corresponding MILP is load balancing.

Efrat *et al.* [66] study joint sink location and routing for a WSN with a single moving sink. Their approximation algorithm is based on the discretization of the search

space of the optimal sink location network. It uses the fact that for fixed sink location SLRP reduces to RP, which can be formulated as a flow problem and solved efficiently when the objective is to maximize the network lifetime. This is also the main idea of the approximation algorithm due to Shi and Hou [67]. They consider the same optimal routing problem but they propose a more efficient approximation strategy. They discretize the cost parameters associated with energy consumption with tight lower and upper bounds. The strategy by Efrat *et al.* is based on the modification of the optimal transmission scheme such that each sensor transmits to the sink using less than possible levels of battery energy allocation. Recently Shi *et al.* [68] improved this method further not only to obtain a faster approximation algorithm but also to propose a design procedure for approximation algorithms that can be applied to solve a larger class of problems. They also show that how this procedure can be used for SLRP, to maximize the network capacity objective instead of the network lifetime.

After the early attempt by Gandam *et al.* [64] to determine specific sink movements for energy minimization, Wang *et al.* [69] study the joint problems of controlled sink mobility and the determination of sink sojourn times at certain sensor nodes in order to maximize the network lifetime. They assume that the sensor field is a two-dimensional grid and formulate the network lifetime as the sum of sojourn times after defining it as the time until the first sensor is drained out of energy, which results in an efficient LP model. This model is generalized by Basagni *et al.* in [70, 71] to determine an optimal route through a set of sites the sink may visit and its sojourn times. They first develop a MILP which is difficult to solve. It is used as a benchmark to assess the performance of the new distributed routing protocols they propose. Another extension is given in [72], where the authors investigate how to control the concurrent and controlled mobility of multiple sinks. The resulting LP model whose optimal solution provides an upperbound on the maximum lifetime of a WSN is similar to the one of [69] but its dual has an exponential number of constraints. As a result, they propose a centralized heuristic that runs in polynomial time. All these solution approaches are routing independent (excluding RP model of Gandam *et al.*) and optimal routing of the sink is presented independently of the routing of the data packets.

Papadimitriou and Georgiadis [73] address this issue and show that the maximum lifetime can be achieved by solving the scheduling problem that determines the sojourn times of the sink at different locations, and routing problem in order to send the sensed data to the sink in an energy efficient way. They formulate an LP problem to maximize the network lifetime, which is defined as the sum of sojourn times subject to network flow and energy constraints. This is a combination of the model of Wang *et al.* [69] with the LP formulation for the maximum lifetime data routing given in [52]. Their model jointly considers the problem of determining the optimal sink sojourn times at the given sink sojourn sites, and the routing of data packets to the current position of the sink by redefining the power consumptions at sensor nodes as variables, which are originally assumed given constants in [69].

In a very recent work, the work by Papadimitriou and Georgiadis [73] is revisited and an efficient distributed algorithm is developed by Gatzianas and Georgiadis [74] as an alternate to the centralized solution proposed in the preceding one. The distributed algorithm exploits the fact that the new formulation reduces to the one given in [55] when the sink is stationary and benefits from their use of Lagrangean duality and subgradient algorithm; the Lagrangean dual obtained by relaxing the flow conservation constraints is solved using subgradient optimization .

3. COVERAGE PROBLEM IN WIRELESS SENSOR NETWORKS

One of the fundamental issues in the design of wireless sensor networks is to determine an effective sensor placement strategy so that the desired tasks are accomplished efficiently. Coverage is one of the primary tasks that is expected to be provided from a WSN. Actually it is a primary indicator of the QoS of the network. Early research in sensor networks mostly concentrates on sensor communication and sensor data fusion for a given sensor network topology. However, as the number of sensors used in a sensor network increases, efficient sensor deployment becomes crucial. In this chapter we propose a binary integer programming (BIP) formulation of the minimum cost point coverage problem for heterogeneous networks with differentiated detection. It is well known that BIP is NP-Complete [75] and computationally difficult to solve exactly even for moderate-sized instances. Therefore, we propose new polynomial approximation algorithms and heuristics for realistic problem sizes. We should also point out that although minimum cost point coverage models based on set covering formulations do not seem to have a direct concern with efficient energy usage, the work of Yang *et al.* provides a counter example [44]; they propose a BIP set covering formulation for energy efficient k-connected coverage sets, which is in fact a special case of the BIP formulation as will be explained in the next section. Moreover, some of the energy-aware coverage models are set covering BIP formulations with additional restrictions [42, 76]. As a result, the solution methods introduced in this chapter can be used within a relaxation and/or decomposition framework obtained after relaxing the complicating constraints.

3.1. Integer Programming Formulation for Differentiated Coverage

Let us consider a sensor field consisting of N points. They may belong to a two (three) dimensional grid with $N = m \times n$ ($N = m \times n \times k$) points in case the field is an $m \times n$ ($m \times n \times k$) grid. They may also form a set of discrete points approximating a sensor field where their granularity in the space is determined by

the desired sensor placement accuracy. Let d_{ij} denote the Euclidean distance between points i and j. We assume that there are K different types of sensors with known sensing (r_k^s) and communication (r_k^c) ranges such that $r_k^c \geq 2r_k^s$ for $k = 1, ..., K$. This is not a strong assumption since it is a technological fact for almost all sensor brands nowadays. Besides, coverage implies connectivity for homogeneous sensor networks under this assumption [77]. This result is extended by Wang *et al.* [34], where they show that k-coverage implies k-connectivity for homogeneous WSNs. Further analysis in connectivity maintenance and coverage preservation, including the effect of sensing scheduling, can be found in [37]. We also assume that there is a unit cost c_{jk} of placing a type k sensor on point j. This enables the formulation of a more general BIP. Usually the more expensive the sensor is, the larger is its sensing and/or communication range. Then the objective becomes the determination of an optimal sensor placement strategy, namely the determination of sensor types and locations so that every point in the sensor field is covered with certain coverage restrictions (i.e. it is within the sensing range of a minimum number of sensors) at the minimum total cost.

Let a_{ijk} be the (nonnegative) coverage coefficient which determines the contribution of a type k sensor to the coverage of point j when placed at point i. There are KN^2 such coefficients and their value can be determined by using sensor ranges and inter-point distances. We allow the deployment of sensors of different types at the same point j. Then the effective coverage problem (ECP) can be formulated as follows.

$$\text{ECP} : \min \sum_{j=1}^{N} \sum_{k=1}^{K} c_{jk} x_{jk} \tag{3.1}$$

$$\text{s.t.} \sum_{j=1}^{N} \sum_{k=1}^{K} a_{ijk} x_{jk} \geq b_i \qquad i = 1, \ldots, N \tag{3.2}$$

$$x_{jk} = 0, 1 \qquad j = 1, \ldots, N, k = 1, \ldots, K. \tag{3.3}$$

Here x_{jk} are the decision variables and have value 1 if a sensor of type k is placed at point j, and 0 otherwise, and c_{jk} is the unit cost of placing a type k sensor at point j. Coverage inequalities (3.2) guarantee that nonnegative coverage requirements b_i, $i = 1, \ldots, N$ are satisfied; they are not necessarily equal and depend on the points in

the sensor field, which makes ECP an appropriate model for differentiated coverage. When the coverage is uniform, $b_i = b \geq 1$ for $i = 1, \ldots, N$. ECP can easily be adapted for three major detection types as explained in the following sections.

3.1.1. Effective Coverage Problem for Perfect Detection

In the basic model, we assume that sensing is perfect and a sensor always detects a target remaining in its range. Thus, a_{ijk} is set to 1 if a sensor of type k located at point j covers (senses) point i, and 0 otherwise. Technically speaking, $a_{ijk} = 1$ if $d_{ij} \leq r_k^s$ and 0 otherwise. Moreover, ECP becomes the formulation of the minimum cost set covering problem [78] when sensors are identical, namely $K = 1$, and $b_i = 1$, $i = 1, \ldots, N$. Notice that ECP generalizes the BIP model by Meguerdichian and Potkonjak [13], which does not discriminate between sensor types. It is also equivalent to one of the formulations of Chakrabarty *et al.* [12] except that sensors of different types can be located at the same point. However, it is more efficient since it does not include any nonlinear term to be linearized. k-coverage set problem formulation of Yang *et al.* [44] is a special case of ECP, since the latter reduces to the former for $K = 1$, $c_{jk} = 1$, and $b_i = b \geq 1$. This is in fact the first of the formulations given in [13].

As in Chakrabarty *et al.* [12], in some of the formulations, the number of sensors is limited to one at every point in the sensor field. For example, BIP formulations given in [22] include the constraint set

$$\sum_{k=1}^{K} x_{jk} \leq 1 \quad j = 1, \ldots, N \tag{3.4}$$

for this purpose. However, since the model is relaxed without these constraints, allowing more than one sensor at a point results in a smaller optimal objective value; this is expected since it is a relaxation of the formulation which also includes (3.4). We illustrate this fact with an example. Let us consider a 7×7 grid sensor field (i.e. $N = 49$) and assume that three types of sensors are available with characteristics given

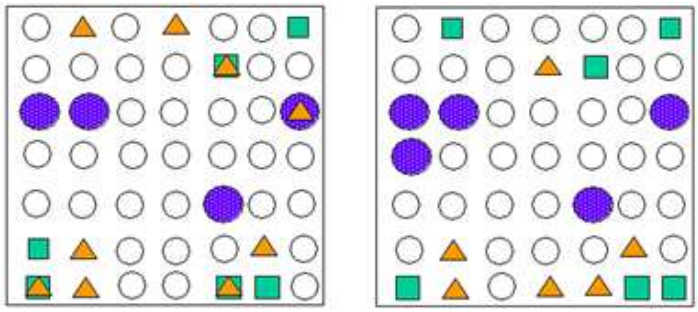

Figure 3.1. Optimal sensor locations with and without constraints (3.4)

in Table 6.1 in Chapter VI, and uniform coverage requirements $b_i = 4$ for $i = 1, \ldots, 49$. Optimal deployments with and without constraints (3.4) are given in Figure 3.1. Three types of sensors are shown by "■", "▲", and "◯", respectively. As can be observed, there are more than one sensor on some of the grid points (e.g. point 7 from left on line 3 and the leftmost point on line 7), which is not the case on the right for the same cost and range values. Besides, the optimal cost for the restricted formulation is 4000; and it is larger than 3950, which is the optimal cost of the unrestricted formulation. This is expected because (3.1) – (3.3) is a relaxation of (3.1) – (3.4).

3.1.2. Effective Coverage Problem for Imperfect Detection

In the ECP formulation, a sensor either detects a point in the sensor field or not, which makes it somewhat restrictive. We introduce an extension of it without harming its structure very much. It uses the sensor field intensity based approach introduced by Meguerdichian and Potkonjak [13] to deal with this restriction. This approach seems to be more realistic since for each point of the sensor field, closer sensors generally provide better coverage.

Numerous models with varying complexities have been proposed for sensors and their sensing intensity. The one introduced by Meguerdichian and Potkonjak [13] is general since it is based on the common characteristics of the existing models, such as the decrease of sensing ability with increasing distance. They use it to improve their original coverage model suffering from the same restriction as ECP. We therefore define

the sensing intensity at point i of a type k sensor placed at point j as

$$a_{ijk} = \frac{\lambda_k}{(d_{ij})^{\delta_k}}. \tag{3.5}$$

Here d_{ij} is the Euclidean distance between point i and point j. Positive constants λ_k and δ_k are sensor specific technological parameters. We prefer to use the total sensor field intensity instead of the average field intensity of the sensors at point i, and force it to be larger than a threshold, say b_i.

3.1.3. Effective Coverage Problem for Uncertain Detection

Although (3.5) provides a model which can somewhat handle the case where sensor detection is imperfect, it is still incapable to provide a solution when there is inherent uncertainty associated with sensor readings due to some obstacles, and thus sensor detections must be formulated probabilistically as pointed in [79].

We let p_{ijk} denote the probability of sensing point i with a sensor of type k located at point j. These probabilities can be calculated for each (i, j, k) triplet by adopting an appropriate sensor terrain model. A common approach is to assume that a target at distance d from a sensor is detected by that sensor with probability $e^{-\alpha d}$ [14, 15]. Exponential decay parameter α determines the rate at which a sensor's detection probability decreases with the distance and thus used to model the quality of the sensor. Sensors with higher α values are of lower quality. Setting

$$p_{ijk} = e^{-\alpha_k d_{ij}} \tag{3.6}$$

is a possibility. This model assumes that sensors have infinite ranges since the sensing probability approaches asymptotically to zero. More realistic sensor detection models can be found in the works by Zou and Chakrabarty [16, 43] and Zhang *et al.* [23]. It is possible to compute p_{ijk} values by using any one of them. Notice that in case there are obstacles in the sensor field these probabilities cannot be symmetrical, which requires the modification of the detection probabilities appropriately. Some placement

algorithms are also proposed in these works, which are heuristics and do not guarantee an optimal solution.

The decision variables x_{jk} of the ECP formulation have the same definition as before. Then $(1 - p_{ijk}x_{jk})$ determines the probability of missing a target located at point i by a sensor of type k at point j. This value is clearly $1 - p_{ijk}$ if there is a sensor of type k at point j, namely $x_{jk} = 1$. It is one when $x_{jk} = 0$. Hence, the overall miss probability for point i can be constrained as

$$\prod_{j=1}^{N}\prod_{k=1}^{K}(1 - p_{ijk}x_{jk}) \leq T_i, \tag{3.7}$$

where $0 < T_i < 1$ is the given maximum allowable miss probability at point i. The left hand side of this inequality is the overall miss probability and obtained under the assumption that sensing probabilities p_{ijk} are independent. Inequality (3.7) is highly nonlinear in binary variables x_{jk}; but it is possible to obtain a more friendly expression. Let us consider the probability $\Pr(uR \leq r)$ where R is a random variable, r a given value and u a binary variable. Clearly, it is 1 if $u = 0$, and $\Pr(R \leq r)$ if $u = 1$. Then,

$$\ln\left(\Pr(uR \leq r)\right) = u\ln\left(\Pr(R \leq r)\right) = u\ln\left(1 - \Pr(R > r)\right) \tag{3.8}$$

can be written as long as $\Pr(R > r) \neq 1$ or $\Pr(R \leq r) \neq 0$. Let us now consider the probabilistic inequality

$$\prod_{i=1}^{s}\Pr(u_iR_i \leq r_i) \leq \theta, \tag{3.9}$$

where R_i are independent random variables, r_i are given values and u_i are binary variables. When we take the logarithm of both sides,

$$\sum_{i=1}^{s} u_i\ln\left(1 - \Pr(R_i > r_i)\right) \leq \ln\theta \tag{3.10}$$

as a consequence of (3.8) with the condition that $\Pr(R_i > r_i) \neq 1$. Notice that

$\ln(1 - \Pr(R_i > r_i))$ and $\ln\theta$ are constants and we have an equivalent binary linear inequality. This approach and its relatives are often used to obtain deterministic equivalents of probabilistic inequalities involving binary deterministic variables within different contexts [43, 80, 81]. As can be seen, (3.7) and (3.9) have the same form, and

$$\sum_{j=1}^{N}\sum_{k=1}^{K} \ln(1 - p_{ijk}x_{jk}) \leq \ln T_i, \tag{3.11}$$

and thus

$$\sum_{j=1}^{N}\sum_{k=1}^{K} \ln(1 - p_{ijk})\, x_{jk} \leq \ln T_i, \tag{3.12}$$

follows as a consequence of (3.10). (3.12) can be transformed into the form of constraints (3.2) by redefining the coefficients a_{ijk} and b_i as $a_{ijk} = -\ln(1 - p_{ijk})$ and $b_i = -\ln T_i$. Observe that these values are positive since they are the logarithms of numbers between zero and one. Our last model aims at minimizing the total deployment cost so that a certain maximum level of miss probability is not exceeded. In other words, it is again possible to use ECP with the new definitions of a_{ijk} and b_i. However, a_{ijk} is not necessarily 0 or 1, and b_i is not necessarily a nonnegative integer this time. Recall that this is also true for imperfect detection.

Although uncertain detection is a more realistic model, (3.12) is not well-defined for $i = j$ since $p_{iik} = 1$ for any sensor of type k, and the condition $\Pr(R > r) \neq 1$ or $\Pr(R \leq r) \neq 0$ of (3.8) is violated. In other words, the logarithm cannot be evaluated for a point in the sensor field which has a sensor located on itself. A remedy is to add additional constraints handling this situation. For the sake of brevity we do not explain them here. Unfortunately, the number of the additional constraints and auxiliary binary variables used in their formulations are very large, and increase the intractability of the model enormously. Yet another remedy is to perturb p_{iik} and set it to a value very close, but not exactly equal, to 1 (e.g. 0.999). According to our numerical experiments we can say that perturbation works well, since the optimal solution is not sensitive for p_{ijk} close enough to 1.

An interesting variant of ECP can be obtained by interchanging the objective function and constraints. This can be done by maximizing the total coverage subject to some budget constraint, which is done by Qishi *et al.* [24] for an uncertain detection model involving sensors with different sensing capabilities in terms of coverage range and detection quality as well as with different costs. Another possibility in this direction is to maximize the minimum of the coverage requirements within the limits of a fixed budget. The new strategy is basically a shift from a set covering type paradigm to a knapsack type one and results in a difficult combinatorial optimization problem as shown by the authors.

3.2. Efficient Methods to Solve Effective Coverage Problem

ECP with a_{ijk} and b_i defined according to one of the detection models explained above is a BIP and can be solved by using one of the available commercial BIP solvers. Unfortunately, it becomes the set covering problem in its simplest form ($c_{jk} = 1$, $a_{ijk} = 0, 1$ and $b_i = 1$), which is known to be not only NP-hard in the strong sense [75], but also difficult from the point of view of theoretical approximation [82]. As a consequence, solving ECP becomes computationally prohibitive to obtain an exact optimal sensor placement plan for networks of realistic size. We therefore propose efficient and accurate approximation algorithms and heuristics.

3.2.1. Approximation algorithms to solve Effective Coverage Problem

We assume that $c_{jk} \geq 0$, $a_{ijk} \geq 0$ and $b_i \geq 0$, which is not a strong assumption since unit costs, coverage contributions and requirements are nonnegative values. There will be some additional assumptions, such as the rationality of the data during the analysis of some of the methods, which will be pointed out when it is necessary. We will also explain how some refinements in the methods and worst case bounds can be obtained by using these additional assumptions.

3.2.1.1. Linear programming relaxation and rounding. As mentioned previously, a_{ijk} and b_i are nonnegative numbers. By using the fact that computer arithmetic is rational we can assume that they are rational numbers (or approximated by the closest rational numbers). Then, we can scale coverage constraints (3.2) to obtain equivalently

$$\sum_{j=1}^{N}\sum_{k=1}^{K}\overline{a}_{ijk}x_{jk} \geq \overline{b}_i \quad i = 1, \ldots, N, \tag{3.13}$$

where $\overline{a}_{ijk}$ and $\overline{b}_i$ are nonnegative integers. Rationality assumption is necessary for the proof of Proposition 1 where we show that the cover obtained RELAX_AND_ROUND satisfies the coverage constraints of ECP.

The LP relaxation of ECP is obtained by replacing integrality requirements (3.3) with simple bounds $0 \leq x_{jk} \leq 1$. We first solve the LP relaxation to obtain an optimal solution $\overline{x}$ with the optimal objective value z_{LP}, and then round its entries with fractional values in order to obtain a feasible solution $\widehat{x}$ for the ECP with objective value z_{RR}. The algorithm is given below with ρ defined as

$$\rho = \max_{1\leq i\leq N}\left\{\sum_{j=1}^{N}\sum_{k=1}^{K}\overline{a}_{ijk}\right\},$$

and C denoting the set of index pairs forming a cover; if a pair $(j,k) \in C$, then a type k sensor is placed at point j in the sensor field, (i.e. $x_{jk} = 1$, $(j,k) \in C$).

RELAX_AND_ROUND is clearly a polynomial time algorithm since its cost is dominated by the complexity of Step 2 where the LP relaxation of ECP is solved and there are known polynomial time algorithms to solve the linear programming problem [83].

Yang *et al.* [44] use also this "relax and round" approach to solve approximately the "k-coverage problem" in homogeneous WSNs. The BIP they consider is a special case of ECP: Since the network is homogeneous there is no sensor type index k. Besides, the cost in the objective function are all equal to 1, the coverage coefficients a_{ijk} are all

1. Set $C = \emptyset$;
2. Let $\overline{x}$ be an optimal solution of the LP relaxation;
3. For all pairs $(j,k) \in \{(1,2,\ldots,N) \times (1,2,\ldots,K)\}$
   do begin
   if $\overline{x}_{jk} \geq 1/\rho$
   then $\widehat{x}_{jk} = 1$ and set $C := C \cup \{(j,k)\}$;
   if $\overline{x}_{jk} < 1/\rho$
   then $\widehat{x}_{jk} = 0$;
   end;
4. Set $z_{RR} = \sum_{(j,k)\in C} c_{jk}$;
5. Output C and z_{RR};

Figure 3.2. Algorithm RELAX_AND_ROUND

0,1 and the coverage requirements b_i are all equal to the same positive integer. In fact their formulation is the (unweighted) set multi-cover problem, whose generalizations have already been studied within the rounding context by Bertsimas and Vohra [84]. We use the argument they use in their rounding results in order to show the correction of RELAX_AND_ROUND in Proposition 1. It is also used by Yang *et al.* [44] in the correction proof of their so called "LP-based Algortihm (LPA)".

Proposition 1. *RELAX_AND_ROUND is a ρ–approximation algorithm for ECP.*

Proof. From Step 3 of the algorithm it is clear that $\widehat{x}_{jk} \leq \rho\overline{x}_{jk}$. Hence, $z_{RR} = \sum_{j=1}^{N}\sum_{k=1}^{K} c_{jk}\widehat{x}_{jk} \leq \rho\sum_{j=1}^{N}\sum_{k=1}^{K} c_{jk}\overline{x}_{jk} = \rho z_{LP} \leq \rho z_{IP}$, where z_{IP} denotes the optimal objective value of ECP. Then it suffices to show that $\widehat{x}$ is a feasible solution of ECP. After scaling the data, cover inequalities (3.2) reduce to (3.13) with nonnegative integer coefficients and right hand sides.

Observe that $\widehat{x}_{jk} \in \{0,1\}$ by definition. Thus we have to show that $\sum_{j=1}^{N}\sum_{k=1}^{K} \overline{a}_{ijk}\widehat{x}_{jk} \geq \overline{b}_i$. Let us consider the set of index pairs $C = \{(j,k) : \overline{x}_{jk} \geq 1/\rho\}$ and its complement $\overline{C} = \{(j,k) : \overline{x}_{jk} < 1/\rho\}$. Then, for any $i = 1,\ldots,N$ we have $0 \leq \sum_{(j,k)\in\overline{C}} \overline{a}_{ijk}\overline{x}_{jk} <$

$\frac{1}{\rho}\sum_{(j,k)\in\overline{C}}\overline{a}_{ijk} \leq \frac{1}{\rho}\sum_{j=1}^{N}\sum_{k=1}^{K}\overline{a}_{ijk} \leq 1$ by the definition of ρ, implying that $0 \leq \sum_{(j,k)\in\overline{C}}\overline{a}_{ijk}\overline{x}_{jk} < 1$. In addition, $\sum_{j=1}^{N}\sum_{k=1}^{K}\overline{a}_{ijk}\widehat{x}_{jk} \geq \sum_{(j,k)\in C}\overline{a}_{ijk}\overline{x}_{jk} \geq \overline{b}_i - \sum_{(j,k)\in\overline{C}}\overline{a}_{ijk}\overline{x}_{jk} > \overline{b}_i - 1$. Since both $\sum_{j=1}^{N}\sum_{k=1}^{K}\overline{a}_{ijk}\widehat{x}_{jk}$ and $\overline{b}_i$ are integers, $\sum_{j=1}^{N}\sum_{k=1}^{K}\overline{a}_{ijk}\widehat{x}_{jk} \geq \overline{b}_i$ follows. Therefore, C is a feasible cover and $\widehat{x}$ is a feasible solution. □

3.2.1.2. Greedy approximation. The second approximation algorithm is based on a greedy selection criterion. Let $I = \{1, 2, \ldots, N\}$ be the set of rows and $J = \{(1, \ldots, N) \times (1, \ldots, K)\}$ be the set of columns each represented by a pair (j, k), namely a point and a sensor type. Then, $I^{(t)}$ and $J^{(t)}$ are subsets of I and J, respectively, at iteration t of the algorithm. We also define a subset of rows $I_{jk} = \{i \in I : a_{ijk} > 0\}$ and

$$s_{jk}^{(t)} = \sum_{i\in I_{jk}\cap I^{(t)}} a_{ijk}, \tag{3.14}$$

which is the total contribution of a type k sensor located at point j of the sensor field. Then, the ratio $\frac{c_{jk}}{s_{jk}^{(t)}}$ is the coverage cost per contribution of a type k sensor located at point j. The greedy algorithm uses these ratios and places at iteration t a type $k^{(t)}$ sensor at point $j^{(t)}$ for which the following ratio is minimum:

$$(j^{(t)}, k^{(t)}) = \arg\min_{(j,k)\in J^{(t)}} \left\{ \frac{c_{jk}}{s_{j^{(t)}k^{(t)}}} \right\}. \tag{3.15}$$

GREEDY_APPROXIMATION is finite since row i is deleted at Step 7 when $\sum_{(j,k)\in C^{(t)}} a_{ijk} \geq b_i$, which happens in finitely many steps since $\sum_{(j,k)\in J} a_{ijk} \geq b_i$ for all $i \in I$ by the feasibility of ECP formulation. Also, notice that the adjustement of the coefficients a_{ijk} during the initializations at Step 1 does not effect the optimal objective value and solutions.

Proposition 2. *GREEDY_APPROXIMATION is a polynomial time algorithm.*

1. Set $C^{(1)} = \emptyset$, $I^{(1)} = I$, $J^{(1)} = J$, $b_i^{(1)} = b_i$, $a_{ijk} = min\{a_{ijk}, b_i\}$ for all $i \in I^{(1)}$ and $(j,k) \in J^{(1)}$, and $t = 1$;
2. If $I^{(t)} = \emptyset$
   then STOP, $C^{(t)}$ is a feasible cover,
   $x_{jk} = 1$ for all $(j,k) \in C^{(t)}$, $x_{jk} = 0$ for all $(j,k) \notin C^{(t)}$,
   and $z_{GA} = \sum_{(j,k)\in J^{(t)}} c_{jk}$;
3. else Calculate $s_{jk}^{(t)}$ using (3.14) for all $(j,k) \in J^{(t)}$;
   Determine $(j^{(t)}, k^{(t)})$ using (3.15);
4. Set $C^{(t+1)} = C^{(t)} \cup \left\{(j^{(t)}, k^{(t)})\right\}$ and
   $J^{(t+1)} = J^{(t)} \diagdown \left\{(j^{(t)}, k^{(t)})\right\}$;
5. For all $i \in I_{j^{(t)}k^{(t)}} \cap I^{(t)}$
   do begin
6. Set $b_i^{(t+1)} = b_i^{(t)} - a_{ij^{(t)}k^{(t)}}$;
7. If $b_i^{(t+1)} \leq 0$, then set $I^{(t+1)} = I^{(t)} \diagdown \{i\}$;
   end;
8. Set $t := t + 1$ and go to 2;

Figure 3.3. Algorithm GREEDY_APPROXIMATION

Proof. Step 3 performs sorting of $|J^{(t)}|$ values at iteration t and dominates the cost of the algorithm. It is repeated $N \times K$ times in the worst case since one (j,k) pair is deleted at each step, which makes $(N\times K)$, $(N\times K-1)$, $(N\times K-2)$,...,1 comparisons to determine the minimum ratio each time, which gives an overall complexity of $O(N^2K^2)$. This bound can be reduced down to $O((NK)\log(NK))$ by using an efficient data structure (e.g., heap) and sorting method (e.g. heapsort) to determine $(j^{(t)}, k^{(t)})$. For an $m \times n$ grid the bounds become $O(m^2n^2K^2)$ and $O((mnK)\log(mnK))$. □

Our analysis of GREEDY_APPROXIMATION is somehow involved and uses linear programming duality. Besides, no rationaliy assumption is needed for the data.

The linear programming (LP) relaxation of ECP can be stated as

$$\text{P} : z_P = \min \sum_{(j,k)\in J} c_{jk}x_{jk} \tag{3.16}$$

$$\text{s.t.} \sum_{(j,k)\in J} a_{ijk}x_{jk} \geq b_i \qquad i \in I \tag{3.17}$$

$$-x_{jk} \geq -1 \qquad (j,k) \in J \tag{3.18}$$

$$x_{jk} \geq 0 \qquad (j,k) \in J. \tag{3.19}$$

Let π_i and ρ_{jk} be the dual variables associated with constraints (3.17) and (3.18), respectively. The last two inequality sets replace bounds $0 \leq x_{jk} \leq 1$, $(j,k) \in J$ obtained after relaxing binary restrictions (3.3). This LP is in standard form and its dual is

$$\text{D} : z_D = \max \sum_{i\in I} b_i\pi_i - \sum_{(j,k)\in J} \rho_{jk} \tag{3.20}$$

$$\text{s.t.} \sum_{i\in I} a_{ijk}\pi_i - \rho_{jk} \leq c_{jk} \quad (j,k) \in J \tag{3.21}$$

$$\pi_i \geq 0 \qquad i \in I \tag{3.22}$$

$$\rho_{jk} \geq 0 \qquad (j,k) \in J. \tag{3.23}$$

Let $(\pi, \rho) = (\overline{\pi}, \overline{\rho})$ be a feasible solution of D with an objective value $\overline{z}_D$. Then, $z_{GA} \geq z_{IP} \geq z_P = z_D \geq \overline{z}_D$. The first inequality follows since GREEDY_APPROXIMATION finds a feasible solution of ECP, the second one follows from the fact that P is a relaxation of ECP. z_P is equal to z_D as a consequence of LP duality and $z_D \geq \overline{z}_D$ because $(\overline{\pi}, \overline{\rho})$ is just a feasible solution of D. Hence, we can combine them as

$$\frac{z_{GA}}{z_{IP}} \leq \frac{z_{GA}}{\sum_{i\in I} b_i\overline{\pi}_i - \sum_{(j,k)\in J} \overline{\rho}_{jk}}. \tag{3.24}$$

In short if we can find a vector $(\overline{\pi}, \overline{\rho})$ such that $\overline{\pi}_i \geq 0$ $i \in I$, $\overline{\rho}_{jk} \geq 0$ $(j,k) \in J$ and $\sum_{i\in I} a_{ijk}\overline{\pi}_i - \overline{\rho}_{jk} \leq c_{jk}$ $(j,k) \in J$ (i.e. $(\overline{\pi}, \overline{\rho})$ is dual feasible), and $\sum_{i\in I} b_i\overline{\pi}_i - \sum_{(j,k)\in J} \overline{\rho}_{jk} \geq z_G/\eta$, then the GREEDY_APPROXIMATION algorithm would be an η–approximation algorithm.

Let

$$\theta^{(t)} = \frac{c_{j^{(t)}k^{(t)}}}{s^{(t)}_{j^{(t)}k^{(t)}}} = \min_{(j,k)\in J}\left\{\frac{c_{jk}}{s^{(t)}_{jk}}\right\} \tag{3.25}$$

and assume that GREEDY_APPROXIMATION runs for $t = 1, \ldots, r$ iterations. Then, it is clear that

$$I^{(t)} \supseteq I^{(t+1)} \text{ for } t = 1, 2, \ldots, r$$

and

$$0 \leq \theta^{(1)} \leq \theta^{(2)} \leq \ldots \leq \theta^{(r)}.$$

We set $\theta^{(r+1)} = \infty$ for convenience since $I^{(r+1)} = \emptyset$ and hypothesize the following dual solution

$$\begin{aligned}
\overline{\pi}_i &= \frac{\theta^{(t)}}{\eta} \quad i \in I^{(t)} \diagdown I^{(t+1)} && t = 1, 2, \ldots, r \\
\overline{\rho}_{j^{(t)}k^{(t)}t)} &= \begin{cases} \sum\limits_{i\in I^{(t)}} a_{ij^{(t)}k^{(t)}}\left(\overline{\pi}_i - \frac{\theta^{(t)}}{\eta}\right) & t = 1, 2, \ldots, r-1 \\ -\frac{\theta^{(r)}}{\eta}\sum\limits_{i\in I^{(r)}} a_{ij^{(r)}k^{(r)}} & t = r \\ 0 & t = r+1, \ldots, m \end{cases} .
\end{aligned} \tag{3.26}$$

Here $\eta = H(\lceil d \rceil) + H(\lfloor d \rfloor)$ with $d = \max_{(j,k)\in J}\{s_{jk}\}$ and

$$H(q) = \sum_{i=1}^{q} \frac{1}{i}$$

for $q \in \mathbb{Z}_+$. We will first show that (3.26) is dual feasible for which the following general purpose proposition will be helpful.

Proposition 3. *Let u and v be two n-dimensional real vectors with nonnegative entries.*

If $u_1 \leq u_2 \leq \ldots \leq u_n$ *and* $v_1 \geq v_2 \geq \ldots \geq v_n$ *then*

$$\sum_{i=1}^{n-1} u_i \left(v_i - v_{i+1}\right) + u_n v_n \leq \max_{i=1,\ldots,n} \left(u_i v_i\right) \left(H(\lceil v_1 \rceil) + H(\lfloor v_1 \rfloor)\right).$$

Proof. Let us assume that $v_n > 0$ and define $v_{n+1} = 0$ for convenience. Then,

$$\begin{aligned}
\sum_{i=1}^{n-1} u_i \left(v_i - v_{i+1}\right) + u_n v_n &= \sum_{i=1}^{n-1} u_i v_i \left(\frac{v_i - v_{i+1}}{v_i}\right) + u_n v_n \\
&= \sum_{i=1}^{n-1} u_i v_i \left(\frac{v_i - v_{i+1}}{v_i}\right) + u_n v_n \left(\frac{v_n - v_{n+1}}{v_n}\right) \\
&= \sum_{i=1}^{n} u_i v_i \left(\frac{v_i - v_{i+1}}{v_i}\right) \\
&\leq \max_{i=1,\ldots,n} \left(u_i v_i\right) \sum_{i=1}^{n} \left(\frac{v_i - v_{i+1}}{v_i}\right).
\end{aligned}$$

Claim 4. *Let* a, b *are two nonnegative real numbers and assume that* $a \geq b$. *Then,*

$$\frac{a-b}{a} \leq H(\lceil a \rceil) - H(\lfloor b \rfloor).$$

Proof. First,

$$\frac{a-b}{a} = 1 - \frac{b}{a} \leq 1 - \frac{\lfloor b \rfloor}{\lceil a \rceil} = \frac{\lceil a \rceil - \lfloor b \rfloor}{\lceil a \rceil}$$

trivially holds. Then,

$$\frac{\lceil a \rceil - \lfloor b \rfloor}{\lceil a \rceil} = \underbrace{\frac{1}{\lceil a \rceil} + \cdots + \frac{1}{\lceil a \rceil}}_{\lceil a \rceil - \lfloor b \rfloor} \leq \frac{1}{\lfloor b \rfloor + 1} + \frac{1}{\lfloor b \rfloor + 2} + \cdots + \frac{1}{\lfloor b \rfloor + (\lceil a \rceil - \lfloor b \rfloor)}$$

follows, which completes the proof since

$$\begin{aligned}\frac{1}{\lfloor b\rfloor+1}+\frac{1}{\lfloor b\rfloor+2}+\cdots+\frac{1}{\lfloor b\rfloor+(\lceil a\rceil-\lfloor b\rfloor)} &= \left(1+\frac{1}{2}+\cdots+\frac{1}{\lceil a\rceil}\right)\\ &- \left(1+\frac{1}{2}+\cdots+\frac{1}{\lfloor b\rfloor}\right)\\ &= H(\lceil a\rceil)-H(\lfloor b\rfloor).\end{aligned}$$

□

Claim 5. *Let v be an n-dimensional real vector with nonnegative entries and assume $v_1 \geq v_2 \geq \ldots \geq v_n$. Then,*

$$\sum_{i=1}^{n}\left(\frac{v_i - v_{i+1}}{v_i}\right) = H(\lfloor v_1\rfloor)+H(\lceil v_1\rceil).$$

Proof. By Claim 4 $\left(\frac{v_i-v_{i+1}}{v_i}\right) \leq H(\lceil v_i\rceil)-H(\lfloor v_{i+1}\rfloor)$ and hence,

$$\begin{aligned}\sum_{i=1}^{n}\left(\frac{v_i - v_{i+1}}{v_i}\right) &\leq \sum_{i=1}^{n}(H(\lceil v_i\rceil)-H(\lfloor v_{i+1}\rfloor)\\ &= H(\lceil v_1\rceil)-H(\lfloor v_2\rfloor)+\\ &\quad H(\lceil v_2\rceil)-H(\lfloor v_3\rfloor)+\\ &\qquad\vdots\\ &\quad H(\lceil v_n\rceil)-H(\lfloor v_{n+1}\rfloor)\end{aligned}$$

$$
\begin{aligned}
&= H(\lfloor v_1 \rfloor + 1) - H(\lfloor v_2 \rfloor) + \\
&\quad H(\lfloor v_2 \rfloor + 1) - H(\lfloor v_3 \rfloor) + \\
&\qquad \vdots \\
&\quad H(\lfloor v_n \rfloor + 1) - H(\lfloor v_{n_1} \rfloor) \\
&= H(\lfloor v_1 \rfloor) - H(\lfloor v_2 \rfloor) + \frac{1}{\lfloor v_1 \rfloor + 1} + \\
&\quad H(\lfloor v_2 \rfloor) - H(\lfloor v_3 \rfloor) + \frac{1}{\lfloor v_2 \rfloor + 1} + \\
&\qquad \vdots \\
&\quad H(\lfloor v_n \rfloor) - H(\lfloor v_{n+1} \rfloor) + \frac{1}{\lfloor v_n \rfloor + 1}
\end{aligned}
$$

implying that

$$
\begin{aligned}
\sum_{i=1}^{n} \left(\frac{v_i - v_{i+1}}{v_i} \right) &\leq H(\lfloor v_1 \rfloor) + \sum_{i=1}^{n} \frac{1}{\lfloor v_i \rfloor + 1} \\
&\leq H(\lfloor v_1 \rfloor) + H(\lceil v_1 \rceil)
\end{aligned}
$$

since $H(\lfloor v_{n+1} \rfloor) = 0$, and the proof of claim is complete. □

Finally,

$$
\max_{i=1,\ldots,n} (u_i v_i) \sum_{i=1}^{n} \left(\frac{v_i - v_{i+1}}{v_i} \right) \leq \max_{i=1,\ldots,n} (u_i v_i) \left(H(\lfloor v_1 \rfloor) + H(\lceil v_1 \rceil) \right)
$$

as a consequence of Claim 5 and the proof of the proposition is complete. Same arguments also follow for $v_n = 0$. □

Then, it suffices to show that $\sum_{i \in I} a_{ijk} \overline{\pi}_i - \overline{\rho}_{jk} \leq c_{jk}$ $(j,k) \in J$ for the solution (3.26). This can be done by analyzing the inequalties $\sum_{i \in I} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i - \overline{\rho}_{j^{(t)}k^{(t)}} \leq c_{j^{(t)}k^{(t)}}$ for $t = 1, \ldots, r-1$, $t = r$ and $t = r+1, \ldots, m$. Recall that "r" is the number of

iterations and "t" is the iteration counter. For $1 \leq t \leq r-1$,

$$\begin{aligned}
\sum_{i \in I} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i - \overline{\rho}_{j^{(t)}k^{(t)}} &= \sum_{i \in I} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i - \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \left(\overline{\pi}_i - \frac{\theta^{(t)}}{\eta} \right) \\
&= \sum_{i \in I \setminus I^{(t)}} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i + \frac{\theta^{(t)}}{\eta} \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \\
&= \frac{1}{\eta} \left(\sum_{p=1}^{t-1} \sum_{i \in I^{(p)} \setminus I^{(p+1)}} a_{ij^{(t)}k^{(t)}} \theta^{(p)} + \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \theta^{(t)} \right) \\
&= \frac{1}{\eta} \left(\sum_{p=1}^{t-1} \sum_{i \in I^{(p)} \setminus I^{(p+1)}} a_{ij^{(t)}k^{(t)}} \frac{c_{j^{(p)}k^{(p)}}}{s^{(p)}_{j^{(p)}k^{(p)}}} + \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \frac{c_{j^{(t)}k^{(t)}}}{s^{(t)}_{j^{(t)}k^{(t)}}} \right) \\
&\leq \frac{c_{j^{(t)}k^{(t)}}}{\eta} \left(\sum_{p=1}^{t-1} \frac{\sum_{i \in I^{(p)}} a_{ij^{(t)}k^{(t)}} - \sum_{i \in I^{(p+1)}} a_{ij^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}} \right. \\
&\quad \left. + \frac{\sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}}}{s^{(t)}_{j^{(t)}k^{(t)}}} \right) \\
&= \frac{c_{j^{(t)}k^{(t)}}}{\eta} \left(\sum_{p=1}^{t-1} \frac{s^{(p)}_{j^{(t)}k^{(t)}} - s^{(p+1)}_{j^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}} + \frac{s^{(t)}_{j^{(t)}k^{(t)}}}{s^{(t)}_{j^{(t)}k^{(t)}}} \right).
\end{aligned}$$

The inequality follows since

$$\theta^{(p)} = \frac{c_{j^{(p)}k^{(p)}}}{s^{(p)}_{j^{(p)}k^{(p)}}} \leq \frac{c_{j^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}}$$

as a consequence of the greedy selection criterion. Hence,

$$\frac{c_{j^{(t)}k^{(t)}}}{\eta} \left(\sum_{p=1}^{t-1} \frac{s^{(p)}_{j^{(t)}k^{(t)}} - s^{(p+1)}_{j^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}} + \frac{s^{(t)}_{j^{(t)}k^{(t)}}}{s^{(t)}_{j^{(t)}k^{(t)}}} \right) \leq$$

$$\frac{c_{j^{(t)}k^{(t)}}}{\eta} \left\{ \max_{1 \leq p \leq t} \left(\frac{1}{s^{(p)}_{j^{(t)}k^{(t)}}} s^{(p)}_{j^{(t)}k^{(t)}} \right) \left(H(\lceil s^{(1)}_{j^{(t)}k^{(t)}} \rceil) + H(\lfloor s^{(1)}_{j^{(t)}k^{(t)}} \rfloor) \right) \right\}$$

holds by Proposition 3 for $u_p = \frac{1}{s^{(p)}_{j^{(t)}k^{(t)}}}$ and $v_p = s^{(p)}_{j^{(t)}k^{(t)}}$. Notice that

$$\frac{c_{j^{(t)}k^{(t)}}}{\eta}\left\{\max_{1\le p\le t}\left(\frac{1}{s^{(p)}_{j^{(t)}k^{(t)}}}s^{(p)}_{j^{(t)}k^{(t)}}\right)\left(H(\lceil s^{(1)}_{j^{(t)}k^{(t)}}\rceil)+H(\lfloor s^{(1)}_{j^{(t)}k^{(t)}}\rfloor)\right)\right\}=$$

$$c_{j^{(t)}k^{(t)}}\frac{H(\lceil s^{(1)}_{j^{(t)}k^{(t)}}\rceil)+H(\lfloor s^{(1)}_{j^{(t)}k^{(t)}}\rfloor)}{H(\lceil d\rceil)+H(\lfloor d\rfloor)}\le c_{j^{(t)}k^{(t)}}$$

since $s^{(1)}_{j^{(t)}k^{(t)}}\le d$ by the definition of d.

For $t=r$,

$$\begin{aligned}
\sum_{i\in I}a_{ij^{(r)}k^{(r)}}\overline{\pi}_i-\overline{\rho}_{j^{(r)}k^{(r)}} &= \sum_{i\in I}a_{ij^{(r)}k^{(r)}}\overline{\pi}_i+\sum_{i\in I^{(r)}}a_{ij^{(r)}k^{(r)}}\frac{\theta^{(r)}}{\eta}\\
&= \sum_{p=1}^{r-1}\sum_{i\in I^{(p)}\setminus I^{(p+1)}}a_{ij^{(r)}k^{(r)}}\frac{\theta^{(p)}}{\eta}+\sum_{i\in I^{(r)}}a_{ij^{(r)}k^{(r)}}\frac{\theta^{(r)}}{\eta}\\
&= \frac{1}{\eta}\left(\sum_{p=1}^{r-1}\sum_{i\in I^{(p)}\setminus I^{(p+1)}}a_{ij^{(r)}k^{(r)}}\theta^{(p)}+\sum_{i\in I^{(r)}}a_{ij^{(r)}k^{(r)}}\theta^{(r)}\right)\\
&= \frac{1}{\eta}\left(\sum_{p=1}^{r-1}\sum_{i\in I^{(p)}\setminus I^{(p+1)}}a_{ij^{(r)}k^{(r)}}\frac{c_{j^{(p)}k^{(p)}}}{s^{(p)}_{j^{(p)}k^{(p)}}}+\sum_{i\in I^{(r)}}a_{ij^{(r)}k^{(r)}}\frac{c_{j^{(r)}k^{(r)}}}{s^{(r)}_{j^{(r)}k^{(r)}}}\right)\\
&\le \frac{c_{j^{(r)}k^{(r)}}}{\eta}\left(\sum_{p=1}^{r-1}\frac{\sum_{i\in I^{(p)}}a_{ij^{(r)}k^{(r)}}-\sum_{i\in I^{(p+1)}}a_{ij^{(r)}k^{(r)}}}{s^{(p)}_{j^{(r)}k^{(r)}}}\right.\\
&\quad \left.+\frac{\sum_{i\in I^{(r)}}a_{ij^{(r)}k^{(r)}}}{s^{(r)}_{j^{(r)}k^{(r)}}}\right)\\
&= \frac{c_{j^{(r)}k^{(r)}}}{\eta}\left(\sum_{p=1}^{r-1}\frac{s^{(p)}_{j^{(r)}k^{(r)}}-s^{(p+1)}_{j^{(r)}k^{(r)}}}{s^{(p)}_{j^{(r)}k^{(r)}}}+\frac{s^{(r)}_{j^{(r)}k^{(r)}}}{s^{(r)}_{j^{(r)}k^{(r)}}}\right).
\end{aligned}$$

The inequality follows since

$$\theta^{(p)} = \frac{c_{j^{(p)}k^{(p)}}}{s^{(p)}_{j^{(p)}k^{(p)}}} \leq \frac{c_{j^{(r)}k^{(r)}}}{s^{(p)}_{j^{(r)}k^{(r)}}}$$

as a consequence of the greedy selection criterion. Thus,

$$\frac{c_{j^{(r)}k^{(r)}}}{\eta}\left(\sum_{p=1}^{t-1}\frac{s^{(p)}_{j^{(r)}k^{(r)}} - s^{(p+1)}_{j^{(r)}k^{(r)}}}{s^{(p)}_{j^{(r)}k^{(r)}}} + \frac{s^{(r)}_{j^{(r)}k^{(r)}}}{s^{(r)}_{j^{(r)}k^{(r)}}}\right) \leq$$

$$\frac{c_{j^{(r)}k^{(r)}}}{\eta}\left\{\max_{1\leq p\leq r}\left(\frac{1}{s^{(p)}_{j^{(r)}k^{(r)}}}s^{(p)}_{j^{(r)}k^{(r)}}\right)\left(H(\lceil s^{(1)}_{j^{(r)}k^{(r)}}\rceil) + H(\lfloor s^{(1)}_{j^{(r)}k^{(r)}}\rfloor)\right)\right\}$$

holds by Proposition 3 for $u_p = \frac{1}{s^{(p)}_{j^{(r)}k^{(r)}}}$ and $v_p = s^{(p)}_{j^{(r)}k^{(r)}}$. Notice that

$$\frac{c_{j^{(r)}k^{(r)}}}{\eta}\left\{\max_{1\leq p\leq r}\left(\frac{1}{s^{(p)}_{j^{(r)}k^{(r)}}}s^{(p)}_{j^{(r)}k^{(r)}}\right)\left(H(\lceil s^{(1)}_{j^{(r)}k^{(r)}}\rceil) + H(\lfloor s^{(1)}_{j^{(r)}k^{(r)}}\rfloor)\right)\right\} =$$

$$c_{j^{(r)}k^{(r)}}\frac{H(\lceil s^{(1)}_{j^{(r)}k^{(r)}}\rceil) + H(\lfloor s^{(1)}_{j^{(r)}k^{(r)}}\rfloor)}{H(\lceil d\rceil) + H(\lfloor d\rfloor)} \leq c_{j^{(r)}k^{(r)}}$$

since $s^{(1)}_{j^{(r)}k^{(r)}} \leq d$ by the definition of d.

Finally, for $r+1 \leq t \leq m$,

$$\begin{aligned}
\sum_{i \in I} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i - \overline{\rho}_{j^{(t)}k^{(t)}} &= \sum_{i \in I} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i \\
&= \sum_{p=1}^{r} \sum_{i \in I^{(p)} \setminus I^{(p+1)}} a_{ij^{(t)}k^{(t)}} \frac{\theta^{(p)}}{\eta} \\
&= \frac{1}{\eta} \sum_{p=1}^{r} \sum_{i \in I^{(p)} \setminus I^{(p+1)}} a_{ij^{(t)}k^{(t)}} \frac{c_{j^{(p)}k^{(p)}}}{s^{(p)}_{j^{(p)}k^{(p)}}} \\
&\leq \frac{c_{j^{(t)}k^{(t)}}}{\eta} \left(\sum_{p=1}^{r} \frac{\sum_{i \in I^{(p)}} a_{ij^{(t)}k^{(t)}} - \sum_{i \in I^{(p+1)}} a_{ij^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}} \right) \\
&= \frac{c_{j^{(t)}k^{(t)}}}{\eta} \left(\sum_{p=1}^{r-1} \frac{s^{(p)}_{j^{(t)}k^{(t)}} - s^{(p+1)}_{j^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}} + \frac{s^{(r)}_{j^{(t)}k^{(t)}}}{s^{(r)}_{j^{(t)}k^{(t)}}} \right).
\end{aligned}$$

The inequality follows since

$$\theta^{(p)} = \frac{c_{j^{(p)}k^{(p)}}}{s^{(p)}_{j^{(p)}k^{(p)}}} \leq \frac{c_{j^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}}$$

as a consequence of the greedy selection criterion. Hence,

$$\frac{c_{j^{(t)}k^{(t)}}}{\eta} \left(\sum_{p=1}^{r-1} \frac{s^{(p)}_{j^{(t)}k^{(t)}} - s^{(p+1)}_{j^{(t)}k^{(t)}}}{s^{(p)}_{j^{(t)}k^{(t)}}} + \frac{s^{(r)}_{j^{(t)}k^{(t)}}}{s^{(r)}_{j^{(t)}k^{(t)}}} \right) \leq$$

$$\frac{c_{j^{(t)}k^{(t)}}}{\eta} \left\{ \max_{1 \leq p \leq r} \left(\frac{1}{s^{(p)}_{j^{(t)}k^{(t)}}} s^{(p)}_{j^{(t)}k^{(t)}} \right) \left(H(\lceil s^{(1)}_{j^{(1)}k^{(1)}} \rceil) + H(\lfloor s^{(1)}_{j^{(1)}k^{(1)}} \rfloor) \right) \right\}$$

holds by Proposition 3 for $u_p = \frac{1}{s^{(p)}_{j^{(t)}k^{(t)}}}$ and $v_p = s^{(p)}_{j^{(t)}k^{(t)}}$. Notice that

$$\frac{c_{j^{(t)}k^{(t)}}}{\eta} \left\{ \max_{1 \leq p \leq r} \left(\frac{1}{s^{(p)}_{j^{(t)}k^{(t)}}} s^{(p)}_{j^{(t)}k^{(t)}} \right) \left(H(\lceil s^{(1)}_{j^{(t)}k^{(t)}} \rceil) + H(\lfloor s^{(1)}_{j^{(t)}k^{(t)}} \rfloor) \right) \right\} =$$

$$c_{j^{(t)}k^{(t)}} \frac{H(\lceil s^{(1)}_{j^{(t)}k^{(t)}} \rceil) + H(\lfloor s^{(1)}_{j^{(t)}k^{(t)}} \rfloor)}{H(\lceil d \rceil) + H(\lfloor d \rfloor)} \leq c_{j^{(t)}k^{(t)}}$$

since $s^{(1)}_{j^{(r)}k^{(r)}} \leq d$ by the definition of d.

At sum, the hypothesized solution (3.26) is dual feasible and the dual objective value becomes

$$\begin{aligned}
\overline{z}_D &= \sum_{i \in I} b_i \overline{\pi}_i - \sum_{t=1}^{r} \overline{\rho}_{j^{(t)}k^{(t)}} \\
&= \sum_{i \in I} b_i \overline{\pi}_i - \sum_{t=1}^{r-1} \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i + \sum_{t=1}^{r-1} \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \frac{\theta^{(t)}}{\eta} + \sum_{i \in I^{(r)}} a_{ij^{(r)}k^{(r)}} \frac{\theta^{(r)}}{\eta} \\
&\geq \sum_{t=1}^{r} \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \frac{\theta^{(t)}}{\eta} \\
&= \frac{1}{\eta} \sum_{t=1}^{r} \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \frac{c_{j^{(t)}k^{(t)}}}{s^{(t)}_{j^{(t)}k^{(t)}}} = \frac{1}{\eta} \sum_{t=1}^{r} s^{(t)}_{j^{(t)}k^{(t)}} \frac{c_{j^{(t)}k^{(t)}}}{s^{(t)}_{j^{(t)}k^{(t)}}} \\
&= \frac{\sum_{t=1}^{r} c_{j^{(t)}k^{(t)}}}{\eta} = \frac{z_{GA}}{\eta}.
\end{aligned}$$

The inequality follows from the fact that $\sum_{t=1}^{r-1} \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i \leq \sum_{i \in I} b_i \overline{\pi}_i$. Please notice that $\sum_{t=1}^{r} \sum_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i \leq \sum_{i \in I} b_i \overline{\pi}_i$ (i.e. when the upper limit of the first summation is the number of iterations r) is not necessarily true. Therefore,

$$\frac{z_{GA}}{z_{IP}} \leq \frac{z_{GA}}{\overline{z}_D} \leq H(\lceil d \rceil) + H(\lfloor d \rfloor)$$

and the following proposition is shown:

Proposition 6. *The worst case bound for a solution of GREEDY_APPROXIMATION is*

$$\frac{z_{GA}}{z_{IP}} \leq H(\lceil d \rceil) + H(\lfloor d \rfloor).$$

Although this is a stronger result then the one reported in an earlier version of this part of the thesis [100], we have not been able to show that it is tight. Similar results have been also obtained before within the context of "general covering problem" by Dobson [85] and Rajagopalan and Vazirani [86].

3.2.1.3. Special cases and refinements. The worst case bound of GREEDY_APPROXIMATION given in Proposition 6 becomes tighter for particular assumptions on the coverage coefficients a_{ijk} and coverage requirements b_i.

Nonnegative rational coverage coefficients (a_{ijk}) and requirements (b_i). This is actually the most realistic situation we can face. For our purpose, having nonnegative coverage coefficients and requirements is equivalent to having them nonnegative integers since we can rescale the coverage constraints as explained in Subsection 3.2.1.1. Thus, we assume that $a_{ijk} \in \mathbb{Z}_+$ and $b_i \in \mathbb{Z}_+$ without loss of generality. Then Proposition 3 reduces to Proposition 7. Observe that the entries of v are now assumed to be nonnegative integers.

Proposition 7. *Let u be an n-dimensional real vector and v be an n-dimensional integer vector both with nonnegative entries. If $u_1 \leq u_2 \leq \ldots \leq u_n$ and $v_1 \geq v_2 \geq \ldots \geq v_n$ then*

$$\sum_{i=1}^{n-1} u_i \left(v_i - v_{i+1}\right) + u_n v_n \leq \max_{i=1,\ldots,n} \left(u_i v_i\right) H(v_1).$$

Proof. Let us assume that $v_n > 0$ and define $v_{n+1} = 0$ for convenience. Then,

$$\begin{aligned}
\sum_{i=1}^{n-1} u_i (v_i - v_{i+1}) + u_n v_n &= \sum_{i=1}^{n-1} u_i v_i \left(\frac{v_i - v_{i+1}}{v_i} \right) + u_n v_n \\
&= \sum_{i=1}^{n-1} u_i v_i \left(\frac{v_i - v_{i+1}}{v_i} \right) + u_n v_n \left(\frac{v_n - v_{n+1}}{v_n} \right) \\
&= \sum_{i=1}^{n} u_i v_i \left(\frac{v_i - v_{i+1}}{v_i} \right) \\
&\leq \max_{i=1,\ldots,n} (u_i v_i) \sum_{i=1}^{n} \left(\frac{v_i - v_{i+1}}{v_i} \right).
\end{aligned}$$

Claim 8. *Let a, b are two nonnegative integer numbers and assume that $a \geq b$. Then,*

$$\frac{a-b}{a} \leq H(a) - H(b).$$

Proof. Trivially

$$\frac{a-b}{a} = \underbrace{\frac{1}{a} + \cdots + \frac{1}{a}}_{a-b} \leq \frac{1}{b+1} + \frac{1}{b+2} + \cdots + \frac{1}{b+(a-b)}$$

holds, which completes the proof since

$$\begin{aligned}
\frac{1}{b+1} + \frac{1}{b+2} + \cdots + \frac{1}{b+(a-b)} &= \left(1 + \frac{1}{2} + \cdots + \frac{1}{a}\right) - \left(1 + \frac{1}{2} + \cdots + \frac{1}{b}\right) \\
&= H(a) - H(b).
\end{aligned}$$

□

Claim 9. *Let v be an n-dimensional integer vector with nonnegative entries and assume $v_1 \geq v_2 \geq \ldots \geq v_n$. Then,*

$$\sum_{i=1}^{n} \left(\frac{v_i - v_{i+1}}{v_i} \right) = H(v_1).$$

Proof. By Claim 8 $\left(\frac{v_i - v_{i+1}}{v_i}\right) \leq H(v_i) - H(v_{i+1})$ and hence

$$\begin{aligned}\sum_{i=1}^{n}\left(\frac{v_i - v_{i+1}}{v_i}\right) &\leq \sum_{i=1}^{n}(H(v_i) - H(v_{i+1})) \\ &= H(v_1) - H(v_2) + \\ &\quad H(v_2) - H(v_3) + \\ &\qquad \vdots \\ &\quad H(v_n) - H(v_{n+1}) \\ &= H(v_1)\end{aligned}$$

since $H(v_{n+1}) = 0$ and the proof of claim is complete. □

Finally,

$$\max_{i=1,\ldots,n}(u_i v_i)\sum_{i=1}^{n}\left(\frac{v_i - v_{i+1}}{v_i}\right) \leq \max_{i=1,\ldots,n}(u_i v_i)\, H(v_1)$$

as a consequence of Claim 5 and the proof of the proposition is complete. Same arguments also follow for $v_n = 0$.

As consequence

$$\sum_{i\in I} a_{ij^{(t)}k^{(t)}}\overline{\pi}_i - \overline{\rho}_{j^{(t)}k^{(t)}} \leq c_{j^{(t)}k^{(t)}}\frac{H(s^{(1)}_{j^{(t)}k^{(t)}})}{H(d)} \leq c_{j^{(t)}k^{(t)}} \quad t = 1,\ldots,m$$

follows for dual variable $(\overline{\pi}, \overline{\rho})$ redefined as

$$\begin{aligned}\overline{\pi}_i &= \frac{\theta^{(t)}}{H(d)} \quad i \in I^{(t)} \diagdown I^{(t+1)} \qquad t = 1,2,\ldots,r \\ \overline{\rho}_{j^{(t)}k^{(t)}} &= \begin{cases} \sum\limits_{i\in I^{(t)}} a_{ij^{(t)}k^{(t)}}\left(\overline{\pi}_i - \frac{\theta^{(t)}}{H(d)}\right) & t = 1,2,\ldots,r-1 \\ -\frac{c_{j^{(r)}k^{(r)}}}{H(d)} & t = r \\ 0 & t = r+1,\ldots,m \end{cases}.\end{aligned} \tag{3.27}$$

Hence, $(\overline{\pi}, \overline{\rho})$ is feasible, and the worst case bound given in Proposition 6 becomes tighter:

$$\frac{z_{GA}}{z_{IP}} \leq H(d).$$

□

It is shown by Dobson [85] that this bound is tight within the "general set covering" context.

$a_{ijk} = 0, 1$, $b_i \geq 1$ *and integer.* This is exactly what we face while dealing with perfect detection and differentiated coverage. First of all

$$s_{jk}^{(t)} = \left| I_{jk} \cap I^{(t)} \right|,$$

since $a_{ijk} = 1$ $i \in I_{jk}$. Then, it follows that

$$\sum_{i \in I} a_{ij^{(t)}k^{(t)}} \overline{\pi}_i - \overline{\rho}_{j^{(t)}k^{(t)}} \leq c_{j^{(t)}k^{(t)}} \frac{H(s_{j^{(t)}k^{(t)}}^{(1)})}{H(d)} \leq c_{j^{(t)}k^{(t)}} \quad t = 1, \ldots, m,$$

and

$$\overline{z}_D = \frac{1}{H(d)} \underbrace{\sum_{(j,k) \in C^{(t)}} c_{jk}}_{z_{GA}}$$

for

$$\begin{aligned} \overline{\pi}_i &= \tfrac{\theta^{(t)}}{H(d)} \quad i \in I^{(t)} \diagdown I^{(t+1)} \qquad t = 1, 2, \ldots, r \\ \overline{\rho}_{j^{(t)}k^{(t)}} &= \begin{cases} \sum\limits_{i \in I^{(t)}} a_{ij^{(t)}k^{(t)}} \left(\overline{\pi}_i - \tfrac{\theta^{(t)}}{H(d)} \right) & t = 1, 2, \ldots, r \\ 0 & t = r+1, \ldots, m \end{cases} . \end{aligned} \tag{3.28}$$

As a result, the worst case bound becomes

$$\frac{z_{GA}}{z_{IP}} \leq H(d).$$

In particular for the ordinary weighted set covering problem, namely for $a_{ijk} = 0, 1$ and $b_i = 1$, the same bound has been obtained and proven to be tight by Chvatal [87].

3.2.2. Heuristics to Solve Effective Coverage Problem

We introduce two heuristics in this section. The first one has a greedy behavior; but it differs from GREEDY_APPROXIMATION in the criterion it uses. The second one is based on the Lagrangean relaxation of ECP obtained by relaxing coverage constraints (3.2).

3.2.2.1. Greedy Heuristic. The basic idea is very simple. At every step, we only consider the available (grid) points. A point of the sensor field is available if no sensor is placed on it. Therefore, every point in the sensor field is available at the beginning. The consideration of available points guarantees the feasibility with respect to constraints (3.2). We determine point $j^{(t)}$ and sensor type $k^{(t)}$ with the largest gain factor and place a sensor of type $k^{(t)}$ on point $j^{(t)}$. Henceforth, point $j^{(t)}$ becomes unavailable. We repeat these steps until there are no remaining uncovered points. A point i in the sensor field is uncovered if it is not covered by at least b_i sensors, which means that one of the inequalities of the constraint set (3.2) is not satisfied. It is implicitly assumed that the value of b_i and/or sensor field's resolution is such that coverage constraints (3.2) are satisfied by placing a sensor of the largest range on every point in the sensor field in the worst case. The gain factor combines the infeasibility information on constraints (3.2) with the cost of different sensor types and has slightly different definitions for perfect, imperfect, and uncertain detection.

For perfect detection, we define the coverage $cov(i)$ of point i as the number of (already installed) sensors that have point i in their sensing range. A point i is said to be *uncovered* if its coverage is (strictly) less than b_i. Then, $\max(0, b_i - cov(i))$ can be regarded as the *uncoverage* of point i. For any such point, the corresponding coverage constraint (3.2) is clearly violated and $b_i - cov(i)$ is a measure of infeasibility. In fact, for a given placement strategy, namely a zero-one assignment of x_{jk} variables, $cov(i) = \sum_{(j,k)\in C} a_{ijk}$, where C denotes a cover over the sensor field. In our greedy heuristic, we prefer to place a sensor on an available point j with the largest gain factor. The gain factor for point j and sensor type k can be expressed as

$$\gamma_{jk} = \frac{\sum_{i\in I_{jk}\cap I} a_{ijk} \max(0, b_i - cov(i))}{c_{jk}}. \tag{3.29}$$

Here, the numerator determines the total uncoverage at points that can be covered by a sensor of type k placed at point j, and the denominator is the unit cost of sensor of type k placed at point j. Thus the gain factor γ_{jk} measures the potential decrease in the total infeasibilities of coverage constraints (3.2) per unit cost when a sensor of type k is placed at point j.

The gain factor can be defined similarly for imperfect detection. The only difference is in the definition of the coefficients a_{ijk}; they represent the sensor field intensity and are not necessarily binary. However, the definition of the gain factor for uncertain detection is different since a_{ijk} is related with sensing probabilities. We define the miss probability of point i (relative to cover C), which we denote as $miss(i)$, as $\prod_{(j,k)\in C} (1 - p_{ijk})$. Then, the quantity $\max(0, miss(i) - T_i)$ becomes the uncoverage of point i, and the gain factor can be defined as

$$\gamma_{jk} = \frac{\sum_{i\in I_{jk}\cap I} p_{ijk} \max(0, miss(i) - T_i)}{c_{jk}}. \tag{3.30}$$

Here, the numerator measures the expected total uncoverage at points that can be covered by a sensor of type k placed at point j, and $I_{jk} = \{i \in I : p_{ijk} > 0\}$ with the assumption that $p_{ijk} < 1$. The steps of the generic greedy heuristic are given in Figure

3.4.

---

1. Set $C^{(1)} = \emptyset$, $cov(i) = 0$ $i \in I$ and $t = 1$;
2. If there is not any available point
   then STOP, Go to 8;
3. If there is an uncovered point i
   then
4. Calculate the gain factor γ_{jk} for each
   available grid point j and sensor type k;
5. Let $(j^{(t)}, k^{(t)})$ be the grid point and sensor
   type with the largest gain factor $\gamma_{j^{(t)}k^{(t)}}$;
6. Set $C^{(t+1)} = C^{(t)} \cup \left\{(j^{(t)}, k^{(t)})\right\}$ and update the coverage of
   the points remaining within the range of sensor type $k^{(t)}$
   placed on point $j^{(t)}$, set $t := t + 1$ and go to 2;
7. else every point i is covered by at least b_i sensors and STOP;
8. $C^{(t)}$ is a feasible cover,
   Set $x_{jk} = 1$ for all $(j, k) \in C^{(t)}$, $x_{jk} = 0$ for all $(j, k) \notin C^{(t)}$, and
   $z_{GH} = \sum_{(j,k)\in J^{(t)}} c_{jk}$;

Figure 3.4. Algorithm GREEDY_HEURISTIC

3.2.2.2. Lagrangean Heuristic. Lagrangean relaxation [88] and subgradient optimization [89] have been important techniques for generating lower (upper) bounds for integer programming and combinatorial optimization problems with a minimization (maximization) objective. They can also be used, however, in almost exactly the same way, to develop efficient and accurate heuristic algorithms, known as Lagrangean heuristics. In a Lagrangean heuristic designed for a minimization type problem, we generate a sequence of Lagrange multipliers producing lower bounds, and a sequence of feasible solutions giving upper bounds on the optimal objective value z_{IP} of the original problem. When the Lagrangean heuristic terminates, the best feasible solution found is a heuristic solution, and the objective value obtained with this solution is an upper bound on z_{IP}. In fact, this upper bound is usually quite tight even though the lower bound is weak.

We consider now the ECP formulation of the sensor coverage problem and relax coverage constraints (3.2) to obtain the Lagrangean subproblem

$$\mathrm{LR}(\mu) : z_{LR}(\mu) = \min \sum_{(j,k)\in J} c_{jk}x_{jk} + \sum_{i\in I} \mu_i(b_i - \sum_{(j,k)\in J} a_{ijk}x_{jk}). \tag{3.31}$$

$$\text{s.t.} \quad x_{jk} = 0, 1 \qquad (j,k) \in J. \tag{3.32}$$

The only constraints are the integrality restrictions and the Lagrangean subproblem $\mathrm{LR}(\mu)$ has integrality property. Hence, one should not expect Lagrangean lower bound to be tighter than LP lower bound. However, an accurate upper bound can be calculated efficiently. Notice that the objective function (3.31) can be equivalently stated as

$$\min \sum_{(j,k)\in J} \left(c_{jk} - \sum_{i\in I} \mu_i a_{ijk} \right) x_{jk} + \sum_{i\in I} \mu_i b_i \tag{3.33}$$

after simple algebraic manipulations. Here μ is the Lagrange multiplier vector. Clearly, for any given value of μ the Lagrangean subproblem can be solved easily by inspection: Set $x_{jk} = 1$ if $C_{jk} < 0$ where $C_{jk} = c_{jk} - \sum_{i\in I} \mu_i a_{ijk}$. This simple procedure can be used in a subgradient algorithm. Notice that the relaxed solution obtained at every step by solving the Lagrangean subproblem is not necessarily feasible for the original problem. We therefore use GREEDY_HEURISTIC initialized with this infeasible solution to obtain a feasible one. The objective value of the final greedy solution gives an upper bound on z_{IP}. In short, by keeping track of these upper bounds and choosing the smallest of them, we end up with a tight upper bound on z_{IP} at the end of the subgradient algorithm.

We set the multipliers arbitrarily to 1 at the beginning. However, the best results for the set covering problem are obtained when multipliers are initially set to the dual solution of the LP relaxation [78]. The same approach can also be used here, which will probably improve the quality of the heuristic. Initially the value of the step size

1. INITIALIZATION:

   1.1 Find an initial solution with GREEDY_HEURISTIC
   and let z_{GH} be its objective value;

   1.2. Set the upper bound $z_{UB} = z_{GH}$, the lower bound $z_{LB} = -\infty$;

   1.3. Set $\mu^{(0)} = 1$ and $t = 0$;

2. MAIN STEP:

   2.1 Solve the Lagrangean subproblem LR($\mu^{(t)}$) by inspection with the current values of $\mu^{(t)}$ to obtain a relaxed solution and calculate $z_{LR}(\mu^{(t)})$;

   2.2 If $z_{LR}(\mu^{(t)}) > z_{LB}$ then set $z_{LB} = z_{LR}(\mu^{(t)})$;

   2.3 If the relaxed solution is a feasible solution of the original problem

   2.4 then Determine the objective value for this feasible solution let $\overline{z}$ denote this value and go to 2.6;

   2.5 else Use greedy heuristic initialized at the current relaxed solution in order to find a feasible solution for the original problem. Let z_{GH} be the objective value obtained with GREEDY_HEURISTIC and set $\overline{z} = z_{GH}$;

   2.6 If $\overline{z} < z_{UB}$ then Set $z_{UB} = \overline{z}$;

   2.7 Update the multipliers by setting

$$\mu_j^{(t+1)} = \mu_j^{(t)} + \eta^{(t)} \left(b_i - \sum_{(j,k)\in J} a_{ijk} x_{jk} \right)$$

   with step size

$$\eta^{(t)} = \frac{\theta \left(z_{UB} - z_{LR}(\mu^{(t)}) \right)}{\sum_{i\in I} \left(b_i - \sum_{(j,k)\in J} a_{ijk} x_{jk} \right)^2}.$$

   2.8 Set $t := t + 1$ and go to 2.1;

Figure 3.5. Algorithm LAGRANGEAN_HEURISTIC for ECP

parameter θ is set to 2 as suggested in the literature [90], which is then halved every 20 iterations unless there is an improvement.

3.2.3. Simple Improvement Strategies

Although Lagrangean heuristic performs well, it can be improved both in terms of accuracy and efficiency through simple and computationaly inexpensive strategies. The first one is sweeping, which is used to increase the quality of the final solution. The second one is variable fixing, which is applied to increase the efficiency of the subgradient algorithm.

3.2.3.1. Sweeping. Greedy and Lagrangean heuristics usually generate non-optimal solutions and one way to quickly improve their solution is to sweep over the sensor field to remove redundant sensors without violating the coverage constraints. The algorithm simply checks all the points with sensors and deletes any redundant sensor. Sweeping is a linear time operation in the number of points of the sensor field.

3.2.3.2. Variable Fixing. Variable fixing is a widely used strategy to reduce the size of the optimization problems. For example, in the case of perfect detection, fixing $x_{jk} = 1$ for some $j \in J' \subseteq \{1, \ldots, N\}$ and $k \in K' \subseteq \{1, \ldots, K\}$ permits the deletion of columns $(j, k) \in J' \times K'$ and all rows $i \in I' \subseteq \{1, \ldots, N\}$ covered by these columns, namely $\sum_{j \in J'} \sum_{k \in K'} a_{ijk} \geq b_i$ for some $i \in I'$, since once rows (grid points) have been covered by at least b_i columns they need no longer be considered. In addition, fixing $x_{jk} = 0$ for some (j, k) enables the deletion of column (j, k) from the problem.

Then we can fix $x_{jk} = 0$ if the value of x_{jk} is zero in the optimal solution of the current Lagrangean subproblem and $z_{LB} + C_{jk} > z_{UB}$. Recall that we have defined $C_{jk} = (c_{jk} - \sum_{i \in I} \mu_i a_{ijk})$, z_{LB} as the current best lower bound for the Lagrangean relaxation and z_{UB} as the best available upper bound for our original problem. This means that variable x_{jk} with $z_{LB} + C_{jk}$ (lower bound corresponding to an optimal solution with variable x_{jk}) larger than z_{UB} cannot be in the optimal solution. We

can also fix $x_{jk} = 1$ if the value of x_{jk} is one in the optimal solution of the current Lagrangean subproblem and $z_{LB} - C_{jk} > z_{UB}$. This means that variable x_{jk} with $z_{LB} - C_{jk}$ (lower bound corresponding to an optimal solution without variable x_{jk}) larger than z_{UB} cannot be in the optimal solution.

The quality of the bounds is an important issue in variable fixing; and tighter bounds are more likely to increase the solution efficiency. Unfortunately, for our models, the bounds are not always tight. For example, lower bounds are quite loose in the case of uncertain detection and upper bounds are not tight enough for perfect detection. As a result, variable fixing becomes beneficial for only few instances. This can be observed through out the computational experiments reported in chapter 6.

4. SINK LOCATION AND ROUTING PROBLEMS IN WIRELESS SENSOR NETWORKS

Each sensor transmits its or some other sensors' data to the sinks either directly or through other sensors within the communication range. Since data transmission is an energy consuming sensor activity, the determination of energy efficient sensor-to-sink message flow paths or routes is one of the fundamental issues in the design of WSNs, which is known as the Routing Problem (RP) in which both sensor and sink locations are assumed to be given. Also, the location of the sinks is an important issue in the design of a WSN, because it affects the energy consumption in the network. Hence, determining the optimum sink location, namely the solution of the Sink Location Problem (SLP), is crucial for minimizing the energy consumption. Although these two problems can be tackled separately, a more efficient WSN design can be obtained when they are modeled within the same frame, where both the sink locations and data routes are optimized simultaneously. We call this joint optimization problem as the Sink Location and Data Routing Problem (SLRP).

In this section we develop various different mathematical models to solve SLRP. The mathematical models we develop can be roughly classified into two groups: those with energy concern and those with financial concern. The models in the first group usually aim to minimize the total energy spent on sensor activities or maximize the network lifetime. The main objective of the second group is the minimization of the total cost. We propose new models for the Energy-aware Sink Location and Routing Problem (ELRP) which minimize the total routing energy, and the Cost-aware Sink Location and Routing Problem (CLRP) which minimize the sum of the total data routing and sink location costs. We also provide efficient solution algorithms to solve these problems for large instances.

4.1. Mixed-integer Linear Programming Formulations

We again consider a sensor field $\mathcal{N}$ consisting of N points, which can be a two (three) dimensional grid with $N = m \times n$ ($N = m \times n \times k$) points. This time we are given S sensors placed on a subset $\mathcal{I}$ of $\mathcal{N}$. We assume that there are S_i sensors placed at point i, with known technical characteristics, and denote them by the set $\mathcal{S}_i$. This implies that we allow the possibility of having more than one sensor at a point. They are possibly nonidentical, which makes the WSN heterogeneous; it is homogeneous if the sensors are identical. Notice that the total number of sensors in the WSN is $S = \sum_{i \in \mathcal{I}} S_i$, which is the size of the sensor set $\mathcal{S} = \bigcup_{i \in \mathcal{I}} \mathcal{S}_i$.

New cost-aware formulations have binary variables in their objective functions. They are used to represent fixed sink location costs in addition to nonnegative flow variables, which is not the case for the energy-aware MILPs; only continuous data flow variables are used in their objectives. This is similar to only the MILP model of Gandam *et al.* [64] . New commodity flow ELRP and CLRP formulations include flow conservation equalities at sensor nodes as the previous SLRP formulations given in [64], [52], [73] and [74]; but there are also additional inequalities limiting the total number of sinks and relating binary sink location variables with data inflows. Variants of such inequalities exist also in the constraint set of models by Gandam *et al.* [64]; but the latter also include energy restrictions at sensor nodes, which also exist in the formulations given in [52], [73] and [74]. Second formulation (i.e. the one with the total energy minimization objective) in Gandam *et al.* is very close to our single commodity ELRP formulation (SELRP). However, authors consider only homogeneous sensor networks with multiple mobile sinks which are relocated at the beginning of rounds each of which consists of a given number of time frames, and assume that each sensor generates one packet of data at the beginning of each time frame. As a result data flow at each sensor node is balanced in a given round by means of flow conservation equalities.

4.1.1. Energy-aware Sink Location and Routing

The *Energy-aware Sink Location and Routing Problem* (ELRP) can be briefly defined as the determination of sink locations and optimal sensor-to-sink data flow paths minimizing the total energy spent on routing. The total number of sinks to be located is given. We use three different paradigms resulting in three different formulations: a single-commodity flow (SELRP), a multi-commodity flow (MELRP) and a *p-median* (PELRP) formulation of the ELRP. We should point out that three formulations resulting from three paradigms are equivalent in their solutions: they give optimal solutions with the same optimal objective values.

4.1.1.1. A Single-commodity Flow Formulation. Two sets of continuous decision variables are used to represent sensor-to-sensor and sensor-to-sink data flows: x_{ijkl} denotes the amount of data flow from sensor $k \in \mathcal{S}_i$ at point $i \in \mathcal{I}$ to sensor $l \in \mathcal{S}_j$ at point $j \in \mathcal{I}$, and y_{ijk} denotes the amount of data flow from sensor $k \in \mathcal{S}_i$ at point $i \in \mathcal{I}$ to the sink at point $j \in \mathcal{N}$. Notice that no variables are defined for $i \in \mathcal{N} \setminus \mathcal{I}$, because no flow can originate at a point without a sensor. Binary variables are used to formulate the sink location decisions: z_j is equal to one if a sink is located at point j; otherwise it is set to zero. Observe that we do not differentiate between data flows with respect to the sensor originating the flow. Then, the *Single Commodity Flow* (SCF) formulation of the ELRP can be given as the following MILP:

SELRP:

$$\min \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} c_{ijk} x_{ijkl} + \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{j\in\mathcal{N}} c_{ijk} y_{ijk} \tag{4.1}$$

$$\text{s.t.} \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{jikl} + \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} y_{jil} + S_i = \sum_{j\in\mathcal{N}} \sum_{k\in\mathcal{S}_i} y_{ijk} + \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{ijkl} \qquad i \in \mathcal{I} \tag{4.2}$$

$$\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i} y_{ijk} \le S z_j \qquad j \in \mathcal{N} \tag{4.3}$$

$$\sum_{j\in\mathcal{N}} z_j = p \tag{4.4}$$

$$x_{ijkl} \ge 0 \qquad i,\ j \in \mathcal{I};\ j \ne i,\ k \in \mathcal{S}_i,\ l \in \mathcal{S}_j \tag{4.5}$$

$$y_{ijk} \ge 0, z_j \in \{0,1\} \qquad i \in \mathcal{I},\ j \in \mathcal{N},\ k \in \mathcal{S}_i \tag{4.6}$$

The objective function (4.1) is the total routing energy and consists of the sum of the energy spent on sensor-to-sensor and sensor-to-sink data flows. Here, c_{ijk} is obtained by using the relation $c_{ijk} = \gamma_k d_{ij}^{\alpha}$, where γ_k is the amount of energy spent for sending one unit of data per unit of distance by sensor k, d_{ij} is the Euclidean distance between points i and j, and α is the path loss factor which is usually assumed to be between 2 and 4 [2]. Equalities (4.2) are the flow balance constraints written for each point i. They guarantee that the total inflow to point i from other points with a sensor and the data generated by the sensors at this point must be equal to the total outflow from point i either to a point (including itself) with a sink or to another point with sensors. Constraints (4.3) ensure that data flows from a point with a sensor to point j with a sink can only occur if there is a sink at that point. We use the fact that an upper bound on the total outflow from all sensors to a sink at point j can be given by the total number of sensors S. Constraint (4.4) sets the number of sinks to be located equal to p, which is a predetermined value. Constraints (4.5) and (4.6) are the integrality and nonnegativity constraints. In this formulation, there are $S(S+N)$ continuous variables, N binary variables and $S+N+1$ constraints. Before proceeding further, we would like to list some observations on this formulation.

1. Variables x_{iikl}, $i \in \mathcal{I}$, $k \in \mathcal{S}_i$, $l \in \mathcal{S}_i$ imply a flow from sensor k to sensor l both at point i, and thus they are dropped from the formulation. However, variables y_{iik}, $i \in \mathcal{I}$, $k \in \mathcal{S}_i$ represent data flow from sensor k to a sink both at point i (a sink is placed at a point i with sensor k), and they can have positive values. Therefore, they are dropped on the left-hand side of equalities (4.2), which is the total inflow to sensor k at point i; but they are kept on the right-hand side, which

represents the total outflow.

2. $d_{ii} = 0$ by definition, which makes $c_{iik} = 0$. As a consequence, there will always be an alternative optimal solution with respect to flow variables if the location of a sink at a point with a sensor is optimal: the total flow from other sensors to such a sink can be either direct or through a sensor placed at that point. Namely, either $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{jilk} = 0$ and $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} y_{jil} > 0$, or $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{jilk} > 0$ and $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} y_{jil} = 0$ on the left-hand side of equality set (4.2). An extreme situation occurs when no data is sent from other sensors to point i, and sensors at that point are the only ones with data outflow to the sink at point i. In short, $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{jilk} = 0$, $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{ijlk} = 0$, $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} y_{jil} = 0$, and $\sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y_{ijk} = 0$, which forces $y_{iik} = S_i$.
3. For any $i \in I$, $\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} y_{jil} > 0$ is possible if a sink is located at point i. However, in this case, this amount of data can also be sent through the sensor located at this point, which makes the second summation on the left-hand side of (4.2) redundant; it can be replaced with

$$\sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{jikl} + S_i = \sum_{j\in\mathcal{N}} \sum_{k\in\mathcal{S}_i} y_{ijk} + \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{l\in\mathcal{S}_j} \sum_{k\in\mathcal{S}_i} x_{ijkl} \qquad i \in \mathcal{I}. \tag{4.7}$$

We use this version in our computations. Notice that this replacement is not possible if $c_{iik} \neq 0$.

4. At first it may seem that an equivalent formulation can be obtained after disaggregating inequality (4.3) as

$$\sum_{k\in\mathcal{S}_i} y_{ijk} \leq S_i z_j \qquad i \in \mathcal{I},\, j \in \mathcal{N}\ . \tag{4.8}$$

Unfortunately, this is not possible since it is based on the implicit assumption of $y_{ijk} \leq 1$, which is not always true for relay sensors and cases with $\sum_{k\in\mathcal{S}_i} y_{ijk} > S_i$ and $z_j = 1$ can occur. Replacing S_i with S on the right-hand side is a remedy for (4.8); but this results in a larger polyhedron weakening the formulation (i.e. a weaker LP bound).

5. It may be more realistic to assume that each time an information packet sent by

sensor k at point i to sensor l at point j, the receiver spends also energy r_l, which does not depend on the distance d_{ij}. As a result, $c_{ijkl} = \gamma_k d_{ij}^{\alpha} + r_l$ can be a more appropriate expression to determine the unit energy expense for sensor-to-sensor flows x_{ijkl}. We can still use the previous expression to determine the unit energy expense for sensor-to-sink flows since the energy spent by a sink to receive an information packet does not affect the WSN's lifetime.

6. In most of the existing works, SLRP models consider only WSNs with at most one sensor at the points of the sensor field, i.e. $S_i = 1$ for all $i \in \mathcal{I}$. For this case, the SELRP formulation reduces to
SELRP′:

$$\min \sum_{i\in\mathcal{I}} \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} c_{ij}x_{ij} + \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{N}} c_{ij}y_{ij} \tag{4.9}$$

$$\text{s.t.} \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} x_{ji} + 1 = \sum_{j\in\mathcal{N}} y_{ij} + \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} x_{ij} \qquad i \in \mathcal{I} \tag{4.10}$$

$$\sum_{i\in\mathcal{I}} y_{ij} \leq I z_j \qquad j \in \mathcal{N} \tag{4.11}$$

$$\sum_{j\in\mathcal{N}} z_j = p \tag{4.12}$$

$$x_{ij} \geq 0 \qquad i \in \mathcal{I},\ j \in \mathcal{I};\ j \neq i \tag{4.13}$$

$$y_{ij} \geq 0, z_j \in \{0,1\} \qquad i \in \mathcal{I},\ j \in \mathcal{N} \tag{4.14}$$

which is considerably smaller in terms of the number of variables and constraints. Variables have similar definitions with less indices.

4.1.1.2. A Multi-commodity Flow Formulation. The second formulation also includes two sets of flow variables; but it is based on the multi-commodity flow approach which is successfully used in the literature for the reformulation of some difficult combinatorial optimization problems. Examples are the work of Wong [91] for the asymmetric travelling salesman problem and the work of Melkote and Daskin [92] for the facility location and transportation network design problem. In the *Multi-commodity Flow* (MCF) formulation of the ELRP (MELRP), it is assumed that each sensor generates a

commodity. Specifically, the data packet generated by a sensor at point i is commodity q. In other words, point i with a sensor has S_i sources of S_i distinct commodities, which makes the total number of commodities equal to the number of sensors. Thus the objective of the new formulation becomes the determination of both the sink locations and the routes for each commodity $q \in \bigcup_{i \in \mathcal{I}} \mathcal{S}_i$, so as to minimize the total routing energy. The size of the multi-commodity formulation MELRP is larger than that of the single commodity formulation SERLP. However, as will be seen later, the LP-relaxation of MELRP is considerably stronger, and the MILP solution benefits from this fact and takes significantly less solution time.

Commodity q originates at some point i with a sensor and reaches a sink at point j. In order to establish the relationship between commodity q and the sensor at point i, which is the source of commodity q, we define parameters s^q_{ik}, $i \in \mathcal{I}$, $k \in \mathcal{S}_i$, $q \in \mathcal{S}$ where $s^q_{ik} = 1$ if commodity q is generated by sensor k at point i, and $s^q_{ik} = 0$, otherwise. Clearly, no commodity is generated at a point without a sensor. This implies that $s^q_{ik} = 0$ for $i \in \mathcal{N} \setminus \mathcal{I}$, $k \in \mathcal{S}_i$. Moreover, commodity q is uniquely identified by sensor k at point i, which means that the same commodity cannot be generated by two different sensors. Hence, for a point i with a sensor (i.e. $i \in \mathcal{I}$), $s^q_{ik} = 1$ for $q = k$, and $s^q_{ik} = 0$ for $q \neq k$. As is the case for the SELRP, we use two types of flow variables: x^q_{ijkl} denotes the amount of flow of commodity q from sensor k at point i to sensor l at point j, and y^q_{ijk} denotes the amount of flow of commodity q from sensor k at point i to point $j \in \mathcal{N} \setminus \mathcal{I}$ without a sensor. Observe that although the definition of x^q_{ijkl} is quite similar to that of x_{ijkl}, the definition of y^q_{ijk} is slightly different from that of y_{ijk}: they represent data flows from a sensor to a point without a sensor (with or without a sink) this time. We also define a new variable w^q_j, which represents the total amount of commodity q that reaches point j. Binary variable z_j is the location variable for the sinks, as before. Namely, $z_j = 1$ if a sink is located at point $j \in \mathcal{N}$, $z_j = 0$ otherwise. Then, the MCF formulation of the ELRP, MELRP, can be given as the following MILP:

MELRP:

$$\min \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j}\sum_{q\in\mathcal{S}} c_{ijk}x^q_{ijkl} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}\setminus\mathcal{I}}\sum_{q\in\mathcal{S}} c_{ijk}y^q_{ijk} \tag{4.15}$$

$$\text{s.t.} \sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j}\sum_{k\in\mathcal{S}_i} x^q_{jilk} + \sum_{k\in\mathcal{S}_i} s^q_{ik} =$$

$$\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j}\sum_{k\in\mathcal{S}_i} x^q_{ijkl} + \sum_{j\in\mathcal{N}\setminus\mathcal{I}}\sum_{k\in\mathcal{S}_i} y^q_{ijk} + w^q_i \qquad i\in\mathcal{I},\ q\in\mathcal{S} \tag{4.16}$$

$$\sum_{j\in\mathcal{I}}\sum_{l\in\mathcal{S}_j} y^q_{jil} = w^q_i \qquad i\in\mathcal{N}\setminus\mathcal{I},\ q\in\mathcal{S} \tag{4.17}$$

$$\sum_{j\in\mathcal{N}} w^q_j = 1 \qquad q\in\mathcal{S} \tag{4.18}$$

$$w^q_j \le z_j \qquad j\in\mathcal{N},\ q\in\mathcal{S} \tag{4.19}$$

$$\sum_{j\in\mathcal{N}} z_j = p \tag{4.20}$$

$$x^q_{ijkl} \ge 0 \qquad i,\ j\in\mathcal{I};\ j\neq i,\ k\in\mathcal{S}_i,\ l\in\mathcal{S}_j,\ q\in\mathcal{S} \tag{4.21}$$

$$y^q_{ijk} \ge 0 \qquad i\in\mathcal{I},\ k\in\mathcal{S}_i,\ j\in\mathcal{N}\setminus\mathcal{I},\ q\in\mathcal{S} \tag{4.22}$$

$$w^q_j \ge 0,\ z_j\in\{0,1\} \qquad j\in\mathcal{N},\ q\in\mathcal{S} \tag{4.23}$$

The objective function (4.15) consists of the total routing energy (i.e. the sum of the energy spent on sensor-to-sensor and sensor-to-sink data flows). Flow balance equations (4.16) are written for each point $i \in \mathcal{I}$, (i.e., points with a sensor), and each commodity $q \in \mathcal{I}$. They simply ensure that if commodity q flows from some point j to point i or it is generated by a sensor at point i, then it must be sent either to another sensor or to a sink, or has to be received by a sink located at point i. Note that when commodity q is generated at point $i \in \mathcal{I}$, then the first term on the left-hand side is zero and $s^q_{ik} = 1$. Constraints (4.17) are the flow balance equations for points $i \in \mathcal{N} \setminus \mathcal{I}$ without a sensor and each commodity q. By noting that the flow of commodity q to one of these points is only possible if there is a sink there, this set of constraints guarantee that if commodity q flows from point $j \in \mathcal{I}$ to point $i \in \mathcal{N} \setminus \mathcal{I}$, then this point must be the final destination of commodity k. Constraints (4.18) make sure that every commodity q ultimately reaches a destination point with a sink. Constraints (4.19) guarantee that

the final destination of commodity q cannot be point j unless there is a sink installed there. Equality (4.20) fixes the number of sinks to p. Constraints (4.21)–(4.23) are the nonnegativity restrictions for the flow variables x^q_{ijkl}, y^q_{ijk}, and w^q_j, and integrality restrictions for the location variables z_j.

We remark that when a zero-one assignment of location variables z_j is given, the constraint set of MELRP becomes the one of a pure multi-commodity flow problem with additional upper bounds (4.19) on variables w^q_j. In this formulation, there are $NI(I+1)$ continuous variables, and N binary variables, and $(2N+1)S+1$ constraints. The formulation simplifies considerably when there is at most one sensor at a point, i.e. $S_i = 1$ for all $i \in \mathcal{I}$. Then the MELRP formulation reduces to

MELRP′ :

$$\min \sum_{i\in\mathcal{I}} \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} \sum_{q\in\mathcal{I}} c_{ij} x^q_{ij} + \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{N}/\mathcal{I}} \sum_{q\in\mathcal{I}} c_{ij} y^q_{ij} \tag{4.24}$$

$$\text{s.t.} \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} x^q_{ji} + s^q_i = \sum_{\substack{j\in\mathcal{I}\\ j\neq i}} x^q_{ij} + \sum_{j\in\mathcal{N}/\mathcal{I}} y^q_{ij} + w^q_i \qquad i \in \mathcal{I},\ q \in \mathcal{I} \tag{4.25}$$

$$\sum_{j\in\mathcal{I}} y^q_{ji} = w^q_i \qquad i \in \mathcal{N}/\mathcal{I},\ q \in \mathcal{I} \tag{4.26}$$

$$\sum_{j\in\mathcal{N}} w^q_j = 1 \qquad q \in \mathcal{I} \tag{4.27}$$

$$w^q_j \leq z_j \qquad j \in \mathcal{N},\ q \in \mathcal{I} \tag{4.28}$$

$$\sum_{j\in\mathcal{N}} z_j = p \tag{4.29}$$

$$x^q_{ij} \geq 0 \qquad i \in \mathcal{I},\ j \in \mathcal{I},\ q \in \mathcal{I};\ j \neq i \tag{4.30}$$

$$y^q_{ij} \geq 0 \qquad i \in \mathcal{I},\ j \in \mathcal{N}/\mathcal{I},\ q \in \mathcal{I} \tag{4.31}$$

$$w^q_j \geq 0,\ z_j \in \{0,1\} \qquad j \in \mathcal{N},\ q \in \mathcal{I} \tag{4.32}$$

which is considerably smaller in terms of the number of variables and constraints. Variables and parameters have similar definitions with less indices. Notice that in this formulation, the set of commodities $\mathcal{S}$ is replaced with the set of points $\mathcal{I}$ having a sensor.

4.1.1.3. A p-median Formulation. This is a BIP formulation with two sets of binary decision variables: the routing variables x_{ijk} and the sink location variables y_j. Routing variable x_{ijk} is set to one if a sensor at point i sends its data to the sink at point j, otherwise it is zero. The location variable y_j is set to one if a sink is located at point j, it is zero otherwise. Then, the p-median formulation of the ELRP, PELRP, can be given as the following BIP.

PELRP:

$$\min \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{j\in\mathcal{N}} g_{ijk} x_{ijk} \tag{4.33}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}} x_{ijk} = 1 \qquad i \in \mathcal{I},\ k \in S_i \tag{4.34}$$

$$x_{ijk} \leq y_j \qquad i \in \mathcal{I},\ k \in S_i,\ j \in \mathcal{N} \tag{4.35}$$

$$\sum_{j\in\mathcal{N}} y_j = p \tag{4.36}$$

$$x_{ijk},\ y_j \in \{0,1\} \qquad i \in \mathcal{I},\ k \in S_i,\ j \in \mathcal{N} \tag{4.37}$$

Here, parameter g_{ijk} is the minimum energy required for routing the information generated by sensor k at point i to the sink located at point j. They can be considered as fixed assignment costs; g_{ijk} is the cost of assigning sensor k at point i to the sink at point j. Hence, the objective function (4.33) minimizes the total assignment cost. These parameters can be determined by solving a shortest path problem for every i, k and j, which can be efficiently done using Floyd-Warshall algorithm [93]. In other words, we consider a complete graph with node set $\mathcal{N} \cup \mathcal{S}$ (points of the sensor field together with the sensors deployed over the sensor field) and determine one-to-one shortest paths between the S nodes of $\mathcal{S}$, and the N nodes of $\mathcal{N}$ prior to solving PELRP. Notice that the solution of the shortest path problem requires the knowledge of the weights g_{ijk} associated with every edge of this graph. They are again determined using the relation $\gamma_k d_{ij}^{\alpha}$. Here, d_{ij}^{α} is the given distance (e.g. Euclidean distance) between nodes i and j of this graph, γ_k is the amount of energy spent for sending one unit of data per unit of distance by sensor k, and α is the path loss factor. In short, we first calculate the weights of the complete graph with nodes $\mathcal{N} \cup \mathcal{S}$. Then, we determine

one-to-one shortest paths between the sensors ($i \in \mathcal{S}$) and the candidate sink locations ($j \in \mathcal{N}$), which gives the candidate minimum-cost routes from sensor k at point i to point j and their lengths g_{ijk}. We can easily obtain sensor-to-sink routes using the shortest path information generated during the computation of the parameters g_{ijk}, when it is necessary. Assignment constraints (4.34) ensure that each sensor should send its data to one sink only. Constraint set (4.35) guarantees that in order to send data to point j there must be a sink located at that point, otherwise no data can be sent. They can be replaced by the set

$$\sum_{k \in \mathcal{S}_i} x_{ijk} \leq S_i y_j \qquad i \in \mathcal{I},\ j \in \mathcal{N} \tag{4.38}$$

or the set

$$\sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{S}_i} x_{ijk} \leq S y_j \qquad j \in \mathcal{N} \tag{4.39}$$

each resulting in a valid formulation with a smaller number of constraints. Nevertheless the strongest is the original one since it gives a tighter LP lower bound than the ones obtained with (4.38) and (4.39). Finally equality (4.36) sets the number of sinks to be located equal to p and the last set of constraints are the binary restrictions on the decision variables. There are $S(N+1)$ binary variables and no continuous variables. The number of constraints is $(N+1)S+1$, which decreases down to $S+IN+1$ and $S+N+1$ respectively after the above mentioned changes in the formulation. Again the formulation simplifies considerably for the situation where there is at most one sensor at a point; $S_i = 1$ for all $i \in \mathcal{I}$, which is given below:

PELRP′:

$$\min \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij} \tag{4.40}$$

$$\text{s.t.} \sum_{j \in \mathcal{N}} x_{ij} = 1 \qquad i \in \mathcal{S} \tag{4.41}$$

$$x_{ij} \leq y_j \qquad i \in \mathcal{S},\ j \in \mathcal{N} \tag{4.42}$$

$$\sum_{j \in \mathcal{N}} y_j = p \tag{4.43}$$

$$x_{ij},\ y_j \in \{0,1\} \qquad i \in \mathcal{S},\ j \in \mathcal{N} \tag{4.44}$$

This is exactly the formulation of the *p-median* problem. It is known to be NP-hard [94]; but the LP relaxation obtained by relaxing the binary restrictions on the variables and adding constraints $0 \leq x_{ij}, y_j \leq 1$ to the model is very strong and experimental results show that it yields integer solutions in most of the cases. Again, this fact does not guarantee that all LP relaxations give integer solutions. There are some special cases where MELRP$'$ can be solved also in polynomial time. The LP relaxation always gives integer solutions for the class of simple networks consisting of line networks, single cycle networks or a forest having these components [94]. However, these structures are too restrictive to model a realistic WSN.

4.1.2. Cost-aware Sink Location and Routing

In contrast to the *Energy-aware Sink Location and Routing Problem* (ELRP), the *Cost-aware Sink Location and Routing Problem* (CLRP) is financially driven and aims to determine the optimal sink locations and sensor-to-sink data flow paths minimizing the total routing and fixed sink investment costs. In certain situations, sensors and especially sinks can be expensive and decision makers may want to obtain optimal plans, which are also economically attractive. It is also possible to interpret the cost as penalty. As a result some of the candidate sink locations and sensor-to-sensor or sensor-to-sink connections can be penalized using higher costs. Also, fixed sink location costs can vary between candidate locations due to different regional conditions, e.g. difficult geographical characteristics.

The number of sinks to be located is not fixed but there is a fixed charge associated with them and the previous formulations can easily be adapted to obtain a single-commodity flow (SCLRP), a multi-commodity flow (MCLRP) and an *Uncapacitated Facility Location* (UCLRP) formulation of the CLRP. This can be done by simply

removing median constraints (4.4), (4.12), (4.20), (4.29), (4.36), (4.43) and adding a cost term to the objective function in order to consider the fixed charge f_j associated with sink placement.

In the cost-aware formulations, the total flow cost can be dominated by the total fixed sink placement cost due to the difference in the scale of c_{ijk} and f_j. One remedy is to incorporate sensor prices into the flow cost c_{ijk} using the fact that a sensor running out of energy can be regarded as consumed.

4.1.2.1. A Single-commodity Flow Formulation. The variables of the cost-aware version of the single-commodity flow formulation (SCLRP) have exactly the same definitions. However, the coefficients of the objective function have different meanings. The formulation can be given as follows:

SCLRP:

$$\min \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{\substack{j\in\mathcal{I} \\ j\neq i}} \sum_{l\in\mathcal{S}_j} c_{ijk} x_{ijkl} + \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{j\in\mathcal{N}} c_{ijk} y_{ijk} + \sum_{j\in\mathcal{N}} f_j z_j \tag{4.45}$$

$$\text{s.t.} \quad (4.2),\ (4.3),\ (4.5),\ (4.6).$$

Here, c_{ijk} denotes the cost of one unit data sent by sensor k at point i to a sensor at point j. It can be computed as before using the relation $c_{ijk} = \gamma_k d_{ij}^{\alpha}$. There is also a fixed cost f_j of installing a sink at point j. Notice that all costs are nonnegative.

4.1.2.2. A Multi-commodity Flow Formulation. The cost-aware version of the multi-commodity flow formulation (MCLRP) can be obtained by applying similar changes:

MCLRP:

$$\min \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j}\sum_{q\in\mathcal{S}} c_{ijk}x^q_{ijkl} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}\setminus\mathcal{I}}\sum_{q\in\mathcal{S}} c_{ijk}y^q_{ijk} + \sum_{j\in\mathcal{N}} f_j z_j \tag{4.46}$$

$$\text{s.t.} \quad (4.16)\text{–}(4.19),\ (4.21)\text{–}(4.23).$$

4.1.2.3. An Uncapacitated Facility Location Formulation. The removal of p-median equality (4.36) and the addition of the fixed cost term to the objective function results in the cost-aware version of the p-median formulation, which we call the uncapacitated facility location problem (UFLP) formulation of the CLRP (UCLRP) because of its similarity with the famous UFL problem in location theory [94];

UCLRP:

$$\min \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}} g_{ijk}x_{ijk} + \sum_{j\in\mathcal{N}} f_j y_j \tag{4.47}$$

$$\text{s.t.} \quad (4.34),\ (4.35),\ (4.37).$$

The cost coefficients are computed again by solving a many-to-many shortest path algorithm [93] as explained before.

4.2. Solution Methods

Efficiency and strength are two main criteria employed to assess the quality of an integer programming formulation. Although the number of variables and constraints give an idea about it, the efficiency is usually measured experimentally by collecting CPU statistics by means of a widely used commercial solver. The strength of a formulation depends on the structure of the polyhedron representing the feasible solution set of the formulation's LP relaxation. It is measured using the optimal objective of the objective function subject to that polyhedron: the tighter the polyhedron, the higher is the lower bound (i.e. the optimal objective value of the LP relaxation) for a mini-

mization problem. In fact, efficiency and strength are related: a stronger formulation usually requires less computation time and thus is more efficient. We provide a detailed experimental analysis given in chapter 6. As a consequence of the computational results, we can say that both the PELRP and UCLRP formulations are very efficient and strong compared to the single and multi-commodity formulations SELRP, SCLRP, MELRP and MCLRP. As a result, in this section, we concentrate on the approximate and exact solution methods only for PELRP and UCLRP, but provide Lagrangean Heuristics for all formulations.

4.2.1. Lowerbounds Using Lagrangean Relaxation

As we mentioned earlier Lagrangean Relaxation (LR) and Subgradient Optimization (SO) have effectively been used to obtain good lower bounds for minimization problems. These techniques can also be utilized to generate good and accurate upper bounds, which is called a Lagrangean Heuristic. We propose Lagrangean relaxation schemes and Lagrangean Heuristics for all six formulations.

4.2.1.1. Lagrangean Relaxation for Single Commodity Flow Formulations. When we relax the constraint set (4.3) with Lagrangean dual parameters θ_j, $j \in \mathcal{N}$, we obtain the following formulation, which is separable.

LR1(θ):

$$\begin{aligned} \min \; z_{LR1}(\theta) = & \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{\substack{j\in\mathcal{I} \\ j\neq i}} \sum_{l\in\mathcal{S}_j} c_{ijk} x_{ijkl} \\ & + \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{j\in\mathcal{N}} (c_{ijk} + \theta_j)\, y_{ijk} - \sum_{j\in\mathcal{N}} \theta_j S z_j \qquad (4.48) \\ & \text{s.t.(4.2),(4.4), (4.5), (4.6)} \end{aligned}$$

Notice that the objective function with variables x_{ijkl} and y_{ijk}, and the constraint set (4.2) together with nonnegativity constraints make up the first subproblem, which is a linear programming problem and can be solved easily.

LR1$'(\theta)$:

$$\min z_{11}(\theta) = \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j} c_{ijk}x_{ijkl} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}} (c_{ijk}+\theta_j)\, y_{ijk} \tag{4.49}$$

s.t.(4.2), (4.5), (4.6)

Also the objective function with only variables z_j and the p-median constraint (4.4) with binary restrictions constructs the second subproblem and can be easily solved by inspection. Namely we rank the $-\theta_j S$ values in a non-decreasing order and set $z_j = 1$ for the first p of them.

LR1$''(\theta)$:

$$\min z_{12}(\theta) = -\sum_{j\in\mathcal{N}} \theta_j S z_j \tag{4.50}$$

s.t.(4.5), (4.6)

The sum of the objective values of these two sub-problems will add up to the objective value of the Lagrangean subproblem $LR1(\theta)$. That is $z_{LR1}(\theta) = z_{11}(\theta)+z_{12}(\theta)$, and $z_{LR1}(\theta)$ is a lowerbound for the original problem SELRP. Hence our purpose is to maximize the objective of the Largrangean problem: $\max_\theta$-$LR1(\theta)$. The final solution may not be feasible, hence we apply a Lagrangean Heuristic to obtain a feasible solution.

The procedure is quite similar with $SCLRP$. The Lagrangean subproblem after dualizing the constraint set (4.3) with Lagrangean dual parameters θ_j $j \in \mathcal{N}$ is:

LR2(θ):

$$\min\ z_{LR2}(\theta) = \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j} c_{ijk}x_{ijkl} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}} (c_{ijk}+\theta_j)\, y_{ijk}$$
$$+ \sum_{j\in\mathcal{N}} (f_j - \theta_j) S z_j \tag{4.51}$$
$$\text{s.t.(4.2), (4.5), (4.6)}$$

Similar to the energy-aware formulation this problem decomposes into two simpler sub-problems. First sub-problem is exactly the same of the energy-aware case. The difference in the second sub-problem is in the objective function value and the absence of the p-median constraint. In the second sub-problem we set $z_j = 1$ if $f_j - \theta_j < 0$ and zero otherwise, by inspection. The sum of the objective functions of these two subproblems gives the objective function of the Lagrangean sub-problem.

4.2.1.2. Lagrangean Relaxation for Multi Commodity Flow Formulations. When we relax the constraint set (4.19) with Lagrangean dual parameters θ_j^q, $j \in \mathcal{N}$, $q \in \mathcal{S}$, we obtain the following formulation, which is again separable.

LR3(θ):

$$\min\ z_{LR3}(\theta) = \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j}\sum_{q\in\mathcal{S}} c_{ijk}x_{ijkl}^q + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}\setminus\mathcal{I}}\sum_{q\in\mathcal{S}} c_{ijk}y_{ijk}^q$$
$$+ \sum_{j\in\mathcal{N}}\sum_{q\in\mathcal{S}} \theta_j^q w_j^q - \sum_{j\in\mathcal{N}}\sum_{q\in\mathcal{S}} \theta_j^q z_j \tag{4.52}$$
$$\text{s.t.(4.16)-(4.18), (4.20)- (4.23)} \tag{4.53}$$

Notice that the objective function with variables x_{ijkl}^q, y_{ijk}^q and w_j^q and the constraint set (4.16)-(4.18), (4.21)- (4.23) make up the first subproblem, which is again a linear programming problem and can be solved efficiently.

LR3$'(\theta)$:

$$\begin{aligned}\min\ z_{31}(\theta) = &\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j}\sum_{q\in\mathcal{S}} c_{ijk}x^q_{ijkl} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}\setminus\mathcal{I}}\sum_{q\in\mathcal{S}} c_{ijk}y^q_{ijk}\\ &+\sum_{j\in\mathcal{N}}\sum_{q\in\mathcal{S}}\theta^q_j w^q_j \qquad (4.54)\\ &\text{s.t.(4.16)-(4.18),(4.21)-(4.23)}\end{aligned}$$

The second subproblem is composed of the remaining variables z_j and the p-median constraint (4.20) with binary restrictions (4.23). It can be easily solved by inspection. We rank the $-\sum_{q\in\mathcal{S}}\theta^q_j$ values in a non-decreasing order and set $z_j = 1$ for the first p of them.

LR3$''(\theta)$:

$$\begin{aligned}\min z_{32}(\theta) = &-\sum_{j\in\mathcal{N}}\sum_{q\in\mathcal{S}}\theta^q_j z_j \qquad (4.55)\\ &\text{s.t.(4.20), (4.23)}\end{aligned}$$

The objective value of the Lagrangean subproblem $LR3(\theta)$ is calculated by summing the objective values of these two sub-problems, namely $z_{LR3}(\theta) = z_{31}(\theta) + z_{32}(\theta)$. Again $z_{LR3}(\theta)$ is a lowerbound for the original problem MELRP. Hence our purpose is to maximize the objective function value of Lagrangean subproblem by determining optimal θ^q_j using sub-gradient optimization. Similar to $LR1(\theta)$ the final solution may not be feasible, so we apply a heuristic to make it a feasible solution, which is described in the next subsection.

The cost-aware case for MCLRP is again easy to formulate. The Lagrangean sub-problem is :

LR4(θ):

$$\min\ z_{LR4}(\theta) = \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{l\in\mathcal{S}_j}\sum_{q\in\mathcal{S}} c_{ijk}x^q_{ijkl} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}\setminus\mathcal{I}}\sum_{q\in\mathcal{S}} c_{ijk}y^q_{ijk}$$

$$+ \sum_{j\in\mathcal{N}}\sum_{q\in\mathcal{S}}\theta^q_j w^q_j + \sum_{j\in\mathcal{N}} f_j - \left(\sum_{q\in\mathcal{S}}\theta^q_j\right) z_j \tag{4.56}$$

$$\text{s.t.(4.16)-(4.18), (4.21)- (4.23)} \tag{4.57}$$

Similar to the energy-aware formulation this problem decomposes into two simpler sub-problems. First sub-problem is exactly the same of the energy-aware case. The difference in the second sub-problem is in the objective function value and the absence of the p-median constraint. In the second sub-problem we set $z_j = 1$ if $f_j - \left(\sum_{q\in\mathcal{S}}\theta^q_j\right) < 0$ and zero otherwise, by inspection. The sum of the objective functions of these two subproblems gives the objective function of the Lagrangean sub-problem.

4.2.1.3. Lagrangean Relaxation for p-median Formulations. We consider the PELRP formulation first and relax the assignment constraints (4.34) with Lagrangean multipliers θ_{ik}, $i \in \mathcal{I}$, $k \in S_i$, which are unrestricted, and obtain the Lagrangean subproblem

LR5(θ):

$$\min\ z_{LR5}(\theta) = \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}} g_{ijk}x_{ijk} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\theta_{ik}[1 - \sum_{j\in\mathcal{N}} x_{ijk}] \tag{4.58}$$

$$\text{s.t.} \qquad (4.35)\text{–}(4.37). \tag{4.59}$$

We let $z_{LR5}(\theta)$ denote the optimal objective value of the subproblem LR5(θ) for a given set of multipliers θ. The objective function (4.58) becomes

$$\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\sum_{j\in\mathcal{N}} \overline{g}_{ijk}x_{ijk} + \sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{S}_i}\theta_{ik} \tag{4.60}$$

after arranging the terms and defining $\overline{g}_{ijk} = g_{ijk} - \theta_{ik}$. LR5($\theta$) can be solved easily, because it can be decomposed for each $j \in \mathcal{N}$. If $y_j = 1$, the contribution to the objective function of the Lagrangean subproblem is equal to

$$a_j = \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{S}_i} \min(0, \overline{g}_{ijk}). \tag{4.61}$$

Since θ_{ik} are free, $\overline{g}_{ijk}$ and a_j can take negative values and LR5(θ) reduces to

LR5(θ):

$$\min \; z_{LR5}(\theta) = \sum_{j \in \mathcal{N}} a_j y_j + \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{S}_i} \theta_{ik} \tag{4.62}$$

$$\text{s.t.} \sum_{j \in \mathcal{N}} y_j = p \tag{4.63}$$

$$y_j \in \{0, 1\} \qquad\qquad j \in \mathcal{N} \; . \tag{4.64}$$

Observe that the objective function includes location variables, y_j, which is not the case in the original formulation. This problem can be decomposed and solved for each j separately by inspection. For this purpose, we first rank the a_j values in ascending order and choose the smallest p of them. In other words, a lower bound becomes

$$z_{LR5}(\theta) = \sum_{j=1}^{p} a_{[j]} + \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{S}_i} \theta_{ik}, \tag{4.65}$$

where $a_{[1]} \leq a_{[2]} \leq \ldots \leq a_{[N]}$, and the Lagrangean dual problem can be stated as

$$z^*_{LR5} = \max_{\theta} z_{LR5}(\theta). \tag{4.66}$$

This is a non-smooth optimization problem and can be solved by a subgradient algorithm as will be explained in the next subsection. Notice that $y_{[j]}(\theta) = 1$, $x_{i[j]k}(\theta) = 1$ if $\overline{g}_{i[j]k} < 0$ and $x_{i[j]k}(\theta) = 0$ for $i \in \mathcal{I}$ and $k \in \mathcal{S}_i$ for all $j = 1, \ldots, p$, and $y_{[j]}(\theta) = 0$, $x_{i[j]k}(\theta) = 0$ for $i \in \mathcal{I}$ and $k \in \mathcal{S}_i$ for all $j = p+1, \ldots, N$ is an optimal solution of the Lagrangean subproblem LR5(θ).

Computations for a Lagrangean lower bound for the UCLRP is similar. We again relax the assignment constraint (4.34) with Lagrangean multipliers θ_{ik}, $i \in \mathcal{I}$, $k \in S_i$, which results in the Lagrangean subproblem

LR6(θ):

$$\min \; z_{LR6}(\theta) = \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{j\in\mathcal{N}} \overline{g}_{ijk} x_{ijk} + \sum_{j\in\mathcal{N}} f_j y_j + \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \theta_{ik} \tag{4.67}$$

$$\text{s.t.} \qquad (4.35)\text{–}(4.37) \tag{4.68}$$

after rearranging the terms and defining $\overline{g}_{ijk} = g_{ijk} - \theta_{ik}$. We let $z_{LR6}(\theta)$ denote the optimal objective value of the subproblem LR6(θ) for a given set of multipliers θ. LR6(θ) can be solved easily, because it can be decomposed for each $j \in \mathcal{N}$. If $y_j = 1$, the contribution to the objective function of the Lagrangean subproblem is equal to

$$a_j = f_j + \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \min(0, \overline{g}_{ijk}). \tag{4.69}$$

Since θ_{ik} are unrestricted, $\overline{g}_{ijk}$ and a_j can take negative values and LR6(θ) reduces to

LR6(θ):

$$\min \; z_{LR6}(\theta) = \sum_{j\in\mathcal{N}} a_j y_j + \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \theta_{ik} \tag{4.70}$$

$$\text{s.t.} \;\; y_j \in \{0,1\} \qquad\qquad j \in \mathcal{N} \;, \tag{4.71}$$

which can be solved by inspection for each candidate sink location j. In other words, we set $y_j(\theta) = 1$ if $a_j \leq 0$, and $y_j(\theta) = 0$ otherwise. After determining $y_j(\theta)$, $x_{ijk}(\theta)$ can be found easily: $x_{ijk}(\theta) = 1$, when $y_j(\theta) = 1$ and $\overline{g}_{ijk} \leq 0$, and $x_{ijk}(\theta) = 0$ otherwise. Finally, a valid lower bound can be calculated as

$$z_{LR6}(\theta) = \sum_{j\in\mathcal{N}} a_j y_j + \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \theta_{ik}, \tag{4.72}$$

and the Lagrangean dual problem becomes

$$z^*_{LR6} = \max_{\theta} z_{LR6}(\theta). \tag{4.73}$$

Notice that the multipliers are unrestricted in sign because the relaxed constraints are equalities. This is again a non-smooth optimization problem and can be solved by a subgradient algorithm as explained in the next subsection.

For both formulations the Lagrangean subproblems LR5(θ) and LR6(θ) have the so-called integrality property: they have integer optimal values even if the integrality restrictions $y_j \in \{0,1\}$, $j \in \mathcal{N}$ are replaced with $0 \leq y_j \leq 1$, $j \in \mathcal{N}$. As a result, the best lower bounds obtained by LR are only as good as the bounds calculated by solving the LP relaxation of the BIP formulations PELRP and UCLRP [95].

4.2.2. Computing Upper Bounds Using Largrangean Heuristics

SO is used to solve the Lagrangean dual problem. It is basically a descent method used to minimize certain non-differentiable functions such as $z_{LR1}(\theta)$ - $z_{LR6}(\theta)$. At each iteration, a lower and upper bound on the optimal objective value are computed. They are used to determine the value of the step parameter which is necessary to update the multipliers θ. The upper bound is obtained by finding a feasible solution of the original problem and evaluating the objective value at this feasible solution. This whole process of finding lower and upper bounds, applied together in a single framework, gives a Lagrangean Heuristic [96].

Algorithm LAGRANGEAN_HEURISTIC in 4.1 is the formal definition of the generic Lagrangean Heuristic we use to compute lower and upper bounds for all these six formulations. We use Z_{LB} and Z_{UB} to denote the best lower and upper bounds. The lower bound is obtained by solving one of the Lagrangean subproblems:

$$\{LR1(\theta), ..., LR6(\theta)\}.$$

This simply requires the replacement of $z_{LR}(\theta^{(t)})$ with one of $\{z_{LR1}(\theta^{(t)}), ..., z_{LR6}(\theta^{(t)})\}$ in the listing of LAGRANGEAN_HEURISTIC. The upper bound is the objective value of a feasible solution of the original problem. The solution of the Lagrangean subproblem does not usually satisfy the relaxed constraint(s) and is infeasible for the original problem. Depending on the type of the problem, one can devise an efficient procedure in order to make it feasible. This is done at every step and the upper bound is updated if the current feasible solution results in an objective value less than the current upper bound. Let T be the counter of subgradient iterations since the last Z_{LB} update, and t be the iteration counter. T is used to update the step length parameter τ.

Step 3, Step 5 and Step 6 of LAGRANGEAN_HEURISTIC include two procedures requiring formulation-specific treatments. FEASIBLE simply tries to generate a feasible solution of the original problem based on a solution to the Lagrangean subproblem. When we consider SELRP, SCLRP, MELRP or MCLRP then FEASIBLE first fixes the sink locations then solves the remaining linear routing problem by LP techniques.

Recall that the Lagrangean subproblems LR5(θ) and LR6(θ) are obtained by relaxing the assignment constraints (4.34). Hence, they can be violated by $\mathbf{x}(\theta)$ and $\mathbf{y}(\theta)$. FEASIBLE fixes this by setting $x_{ij^*k} = 1$, $x_{ijk} = 0$ for $j \in \mathcal{N}\backslash\{j^*\}$ and $y_{j^*} = 1$, where $j^* = \arg\min\{g_{ijk} : j \in \mathcal{N},\ y_j(\theta) = 1\}$ for all $i \in \mathcal{I}$ and $j \in \mathcal{N}$. If the case $y_j(\theta) = 0,\ j \in \mathcal{N}$ occurs, then FEASIBLE sets $x_{ij^*k} = 1,\ y_{j^*} = 1$ and $x_{ijk} = 0$ $j \in \mathcal{N}\backslash\{j^*\}$ with $j^* = \arg\min\{g_{ijk} : j \in \mathcal{N}\}$. Notice that this extreme case cannot occur for the PELRP formulation, since Constraint set (4.36) always forces p of the variables $y_j(\theta)$ to be one. In short, FEASIBLE always assigns a sensor to the sink with the minimum routing cost, and sets the objective value to the cost of the feasible set $\mathcal{F} = \{j : y_j = 1, j \in \mathcal{N}\}$.

IMPROVE is only applied for PELRP or UCLRP and it is different from FEASIBLE in principle. It works on a given feasible solution and tries to improve it by applying simple operations on a feasible set $\mathcal{F} = \{j : y_j = 1, j \in \mathcal{N}\}$, whose application depends on the formulation. In case of the UCLRP formulation, IMPROVE examines

1. 1. Set $Z_{LB} = -\infty$, $Z_{UB} = \infty$, $t = 0$, $T = 0$, $\tau = 2$,
   Initialize Lagrangean multipliers $\theta^{(t)}$;
2. Solve the Lagrangean subproblem with the current Lagrangean multipliers
   and let the solution be $\mathbf{x}(\theta^{(t)})$, $\mathbf{y}(\theta^{(t)})$, $\mathbf{w}(\theta^{(t)})$ and $\mathbf{z}(\theta^{(t)})$
   with the objective value $z_{LR}(\theta^{(t)})$;
   If $z_{LR}(\theta^{(t)}) > Z_{LB}$
   then set $T = 0$, $Z_{LB} = z_{LR}(\theta^{(t)})$, and store
   $\mathbf{x}(\theta^{(t)})$, $\mathbf{y}(\theta^{(t)})$, $\mathbf{w}(\theta^{(t)})$ and $\mathbf{z}(\theta^{(t)})$ for possible use in step 5;
   else set $T = T + 1$;
3. Call FEASIBLE to obtain a feasible solution $\mathbf{x}^{(t)}$, $\mathbf{y}^{(t)}$, $\mathbf{w}^{(t)}$ and $\mathbf{z}^{(t)}$
   of the main problem using $\mathbf{x}(\theta^{(t)})$, $\mathbf{y}(\theta^{(t)})$, $\mathbf{w}(\theta^{(t)})$ and $\mathbf{z}(\theta^{(t)})$ and
   let $z_{UB}^{(t)}$ be the objective function value;
4. If $z_{UB}^{(t)} < Z_{UB}$ then set $Z_{UB} = z_{UB}^{(t)}$ and save $\mathbf{x}^{(t)}$, $\mathbf{y}^{(t)}$, $\mathbf{w}^{(t)}$, $\mathbf{z}^{(t)}$;
5. If $T = 30$ (i.e. 30 subgradient iterations without updating Z_{LB} has occurred)
   then set $\tau = \tau/2$, and call IMPROVE to improve the feasible solution
   based on the solution of Lagrangean subproblem which gives
   the current lower bound Z_{LB};
6. Calculate the subgradients $\mathbf{G}^{(t)}$ according to the formulas in Table 4.1
7. If $\sum \left(G^{(t)}\right)^2 = 0$ or $\tau < 0.005$ then Go to step 10;
8. Update the multipliers as $\theta^{(t+1)} = \theta^{(t)} + \eta^{(t)} G^{(t)}$ with step size

$$\eta^{(t)} = \frac{\tau(1.05 Z_{UB} - Z_{LB})}{\sum \left(G^{(t)}\right)^2} \ ;$$

9. Set $t := t + 1$ and Go to step 2;
10. If $Z_{LB} = Z_{UB}$
    then STOP, the current feasible solution is optimal
    else STOP, convergence with duality gap $Z_{UB} - Z_{LB}$;

Figure 4.1. Algorithm LAGRANGEAN_HEURISTIC for SLRP

the costs of sets $\mathcal{F} \cup \{k\}$ (addition), $\mathcal{F} \backslash \{j\}$ (deletion) and $\mathcal{F} \backslash \{j\} \cup \{k\}$ (swap) for all pairs $(j,k) \in \mathcal{F} \times (\mathcal{N} \backslash \mathcal{F})$. If the cost of the new set having the lowest cost is lower than the cost of $\mathcal{F}$, then IMPROVE updates the upper bound with this value and replaces $\mathcal{F}$ with this set. Then these steps are repeated until no improvement occurs in the objective value. As for the PELRP, IMPROVE performs only swap operations because of Constraint set (4.36) (i.e. p sinks must be located in every feasible solution). Besides, IMPROVE considers sink j when either $y_j = 1$, or $y_j = 0$ and a_j is one of the $\max(25, N/10)$ smallest a_k values having $y_k = 0$, $k \in \mathcal{N}$. Essentially IMPROVE seeks a good solution by applying swap operations to the subset of the sinks that are most likely to be in an optimal solution, as suggested in [96].

In Step 6 of the algorithm the subgradient values are calculated for each problem. The functions for the subgradients are given in Table 4.1.

Table 4.1. List of subgradient functions for different formulations

Problem	Subgradient
SELRP, SCLRP	$G_j^{(t)} = \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{S}_i} y_{ijk} - S z_j \quad j \in \mathcal{N}$
MELRP, MCLRP	$G_{jq}^{(t)} = w_j^q - z_j \quad j \in \mathcal{N},\ q \in \mathcal{S}$
PELRP, UCLRP	$G_{ik}^{(t)} = 1 - \sum_{j \in \mathcal{N}} x_{ijk} \quad i \in \mathcal{I},\ k \in \mathcal{S}_i$

4.2.3. Lower Bounds using DUALOC for *p*-median Formulations

Erlenkotter's DUALOC method [97] is a simple ascent adjustment procedure originally proposed for the UFLP to solve the LP relaxation of the Lagrangean dual to compute a lower bound on the optimal value. Then a heuristic can be used to obtain a feasible solution for the original problem. Exploiting the similarity between the UFLP and the UCLRP, we adapt DUALOC to work on UCLRP.

DUALOC begins with feasible dual values u_i and attempts to increase each one at a time while maintaining the bounds set by the dual constraint $\sum_{i \in \mathcal{I}} (u_i - c_{ij})^+ - f_j \leq 0$, $j \in \mathcal{N}$. This step is called *Dual Ascent Procedure* and it terminates when the dual

values cannot be increased further. The facility locations obtained by the solution of the Dual Ascent Procedure are possible locations for the sinks. Then each sensor is assigned to the sink with the minimum routing cost. The objective value obtained by the Dual Ascent Procedure is a lower bound for UCLRP and may be infeasible, because some sensors may not be assigned to a sink. Hence a *Dual Adjustment Procedure* is applied to improve the solution. This procedure, which terminates DUALOC, may not also guarantee an integer solution. Hence DUALOC can be embedded within a branch-and-bound framework to generate an optimal solution. The branch-and-bound steps are not included in our solution procedure. Instead, at the end we apply LAGRANGEAN_HEURISTIC on the DUALOC solution and obtain integer feasible solutions.

As we have already mentioned, the p-median problem and PELRP are related to each other just as the UFLP and UCLRP are. We benefit from this fact to adapt the *Nested Dual Approach* that works efficiently to solve the p-median problem. It begins dualizing the median constraint (4.36) using a dual variable θ resulting in the Lagrangean subproblem

LR7(θ):

$$\min \; z_{LR7}(\theta) = \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{j\in\mathcal{N}} g_{ijk} x_{ijk} + \theta(\sum_{j\in\mathcal{N}} y_j - p) \tag{4.74}$$

$$\text{s.t.} \quad (4.34), (4.35) \text{ and } (4.37), \tag{4.75}$$

which is equivalent to UCLRP with a sink installation cost θ for any candidate sink location point $j \in \mathcal{N}$. Since the number of sinks, i.e. p, is given, the term $-\theta p$ is a constant. The Lagrangean dual problem tries to find such a θ maximizing the objective function $z_{LR7}(\theta)$ and can be stated as

$$z^*_{LR7} = \max_{\theta} z_{LR7}(\theta).$$

We can efficiently solve subproblem LR7(θ) with DUALOC and obtain an optimal

solution $\mathbf{x}(\theta)$ and $\mathbf{y}(\theta)$ when θ is fixed. Then, we can use $\mathbf{x}(\theta)$ and $\mathbf{y}(\theta)$ to update θ. This process can be repeated until an optimal value for θ is reached or a stopping condition is satisfied. The Nested Dual Heuristic is given formally in Figure 4.2.

---

1. Set $t = 0$, $q = 1$, $r = n$, $\theta^{(0)} = \frac{z_{LR7}(q) - z_{LR7}(r)}{r-q} = \frac{z_{LR7}(1)}{n-1}$;
2. Solve LR7($\theta^{(t)}$) using DUALOC and let the solution be $\mathbf{x}(\theta^{(t)})$, $\mathbf{y}(\theta^{(t)})$;
3. Let $\sum_{j \in \mathcal{N}} y_j(\theta^{(t)}) = p^{(t)}$;
   If $p^{(t)} = p$ or $z_{LR7}(p^{(t)}) + (p^{(t)} - p)\theta^{(t)} = z_{LR7}(r) + (r-p)\theta^{(t)} = z_{LR7}(q) + (q-p)\theta^{(t)}$
   then STOP;
4. Update $\theta^{(t)}$ by $\theta^{(t+1)} = \frac{z_{LR7}(q) - z_{LR7}(r)}{r-q}$
   where if $p^{(t)} < p$ then $q = p^{(t)}$, or if $p^{(t)} > p$ then $r = p^{(t)}$;
   Set $t = t + 1$; Go to step 2;

Figure 4.2. Algorithm NESTED_DUAL_HEURISTIC

NESTED_DUAL_HEURISTIC requires at most $(n-1)$ iterations. When it terminates with $p^{(k)} = p$, we have an optimal solution due to the complementary slackness conditions of the p-median problem [94]. Otherwise, if the algorithm stops with $p^{(k)} \neq p$, then the optimal dual objective function is only a lower bound on the optimal objective value of the original problem. An upper bound (a feasible solution) can be generated by some heuristic technique such as the LAGRANGEAN_HEURISTIC, as before.

4.2.4. Lower Bounds Using Canonical Reduction for *p*-median Formulations

Canonical reduction is a method successfully applied to UFLP to reduce the size of the formulation [98, 99]. We illustrate how to use it for PELRP and UCLRP so as to reduce their size and compute a lower bound efficiently in the following. Let us consider a (row) vector say $c^T = \begin{pmatrix} 1 & 3 & -2 & 0 & 1 \end{pmatrix}$. It can be decomposed into two vectors as $c_1^T = \begin{pmatrix} -2 & -2 & -2 & -2 & -2 \end{pmatrix}$ and $c_2^T = \begin{pmatrix} 3 & 5 & 0 & 2 & 3 \end{pmatrix}$ so that $c = c_1 + c_2$.

The nonzero entries of c_1 are equal and it is said to be in *canonical form*. c_2 is not in the canonical form and can be decomposed further as $c_{21}^T = \begin{pmatrix} 2 & 2 & 0 & 2 & 2 \end{pmatrix}$ and $c_{22}^T = \begin{pmatrix} 1 & 3 & 0 & 0 & 1 \end{pmatrix}$ where $c_2 = c_{21} + c_{22}$. c_{21} is in the canonical form and c_{22} can also be decomposed as $c_{221}^T = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \end{pmatrix}$ and $c_{222}^T = \begin{pmatrix} 0 & 2 & 0 & 0 & 0 \end{pmatrix}$. Observe that $c_{22} = c_{221} + c_{222}$ and they are both in the canonical form since their nonzero entries are equal. In short, the original vector c can be expressed as the sum of c_1, c_{21}, c_{221} and c_{222}, which are in the canonical form. This procedure can be applied to the coefficients of the objective function and constraints of any linear integer programming formulation to obtain the canonical form of the cost vector and the canonical reduction matrix. The rows with identical zero entries (zeros at the same entries) of the canonical reduction matrix is then aggregated by adding them entry by entry. For example, if we have two separate rows $\begin{pmatrix} 1 & 1 & 0 & 0 & 1 \end{pmatrix}$ and $\begin{pmatrix} 3 & 3 & 0 & 0 & 3 \end{pmatrix}$ we can aggregate them as the row $\begin{pmatrix} 4 & 4 & 0 & 0 & 4 \end{pmatrix}$. The formal description of the procedure given in Figure 4.3 as Algorithm CANONICAL_REDUCTION is adapted from [98, 99].

The resulting matrix obtained by applying these steps is called a *canonical matrix*. Let $\mathcal{T}$ denote the new set of sensors (rows) corresponding to the canonical matrix and let x_{tj}, $t \in \mathcal{T}$, $j \in \mathcal{N}$ be the variables representing them. It can be shown that not only the optimal solutions of the formulations PELRP and PELRP$'$, but also the optimal solutions of their LP relaxations are equivalent [98]. Furthermore, the variables x_{tj} can be reduced to x_t, $t \in \mathcal{T}$, to represent each row (sensor) for all $j \in \mathcal{N}$ and the constraint set (4.42) becomes redundant. This reduced problem PELRP$''$ with (x_t, y_j) has the same solution with PELRP. Finally, first by transforming PELRP$''$ using the transformation $\pi_t = \sum_{j \in \mathcal{T}} y_j - x_t$, $t \in \mathcal{T}$, and then by taking the dual of the LP relaxation of the transformed problem, a very efficient dual formulation is obtained [100]:

1. *Row Reduction*:

   1.1 If all the elements of c_{ij} of row $i \in \mathcal{S}$ are equal to zero,

   then delete the row and STOP;.

   else Go to step 1.1;

   1.2 Let p be the smallest non-zero entry in row $i \in \mathcal{S}$;

   Add a new row $(q_{h1}, q_{h2}, ..., q_{hn})$ to the reduced matrix $Q = (q_{hj})$,

   where $q_{hj} = p_i$ if $c_{ij} \geq p_i$, and 0 otherwise;

   Subtract this new row from row i and Go to step 1;

2. *Matrix Reduction*:

   2.1 Define the support of a given row h as $J_h = \{j \in \mathcal{N} : q_{hj} \neq 0\}$;

   2.2 Do the row reduction for all rows $i \in \mathcal{S}$;

   2.3 If the supports of all the rows are different in the resulting matrix Q

   then STOP;

   else Set $J_{h1} = J_{h2}$ for some h_1 and h_2,

   replace rows h_1 and h_2 by their sum, and Go to step 2;

3. *Canonical Reduction:*

   3.1 Do the matrix reduction of $C = (c_{ij})$ getting $R = (r_{tj})$,

   where each row is indexed by its support;

   3.2 Delete rows with no zero entries and add the value of the

   nonzero entry to the objective function;

Figure 4.3. Algorithm CANONICAL_REDUCTION

CR:

$$\max z = \sum_{t \in \mathcal{T}} u_t + pw \tag{4.76}$$

$$\text{s.t.} \sum_{t \supseteq (j)} u_t + w \leq \sum_{t \supseteq (j)} r_t \qquad j \in \mathcal{N} \tag{4.77}$$

$$0 \leq u_t \leq r_t \qquad t \in \mathcal{T}. \tag{4.78}$$

Canonical dual formulation (CR) has at most $S(N-2)+(N+1)$ variables and N constraints and upper bounds on the variables. It is widely accepted that the number of variables has less influence on the computational speed of the simplex algorithm than the number of constraints in practice, since it dictates the size of the basis. In fact, experimental analysis suggests that the simplex method requires empirically, on the average in most instances, roughly *O(m)* iterations when the formulation has m constraints [101]. In our case, the number of constraints reduces from $(N+1)S+1$ to N, which is a considerable saving. When we solve CR, we obtain the same objective function with the LP relaxation of PELRP. This solution is a lower bound for PELRP, but because of the favorable properties of CR, a solution is obtained much faster than solving the LP relaxation of PELRP. So, when we apply canonical reduction, lower bounds can be obtained efficiently. Since the LP relaxation is strong, in most of the cases, we can even find an objective value which is the same as the objective value of an optimal integer solution of the original problem.

5. INTEGRATED MODELS FOR COVERAGE, SINK LOCATION AND ROUTING PROBLEMS

In this chapter, we focus on the design of a wireless sensor network where we consider the point coverage, sink location and data routing problems simultaneously. We name this integrated problem as the *Energy-Aware Coverage, Sink Location and Routing Problem* (ECLRP) since the objective is to minimize the total energy spent by the sensors. We propose four new mathematical programming formulations. As will be seen they are very large and can be solved by commercial solvers for small-sized networks only. Therefore, for medium and large networks we propose a nested solution procedure, which consists of a metaheuristic search of sensor locations, that is combined with the solution of sink location and routing problem (ELRP).

5.1. Mixed-integer Linear Programming Formulations

We propose four different mathematical programming formulations for ECLRP. Our first three formulations represent different network design models, where the total routing energy is arc-based and obtained by summing up the costs of arc flow energies. Hence the energy spent on each arc is proportional to the amount of flow on it. These models are relatively simple and can be used to efficiently solve small problems. The last formulation is a median model, where the total energy spent is path-based and the assignment cost of a sensor to a sink is equal to the least cost (energy) path connecting them. Although this is a more complicated model, it results in an efficient and accurate heuristic algorithm to solve ECLRP.

ECLRP, is defined on a field $\mathcal{N}$ with $|\mathcal{N}| = N$ points. Every point of $\mathcal{N}$ is a candidate location for a sensor and/or a sink. We are given a sensor type set $\mathcal{K}$ with $|\mathcal{K}| = K$. Each sensor type has a different cost, sensing and transmission range. Every point on the sensor field is to be covered by the sensors collectively, so the coverage amount of each point should be above a certain threshold. The coverage amount of

a point by a sensor is determined by a probability function of distance computed as $e^{-\alpha d}$. Parameter a_{ijk} determines the amount of coverage at point i by a type k sensor at point j. The required coverage amount (threshold) at any point i is b_i. Each sensor has a cost of h_k and the total budget of sensor investment is limited to B. We are required to cover every point i on the sensor field with coverage amount b_i, without exceeding budget B. Finally, we define binary decision variables s_{jk}, where s_{jk} is set to one if a type k sensor is located at point $j \in \mathcal{N}$, it is zero otherwise.

We assume only a single type of sink in the network and the number of sinks to be installed is equal to p. A sink can be installed at any point on the sensor field and we do not disallow a sensor and a sink to be placed at the same point. The sink locations are represented by binary decision variables z_j, where z_j is one if a sink is installed at point j and zero otherwise.

We first develop three MILP models based on arc-flow formulations. The first formulation is a single-commodity flow model, which is based on the SELRP formulation given in the previous chapter. The second formulation is a multi-commodity model and it is based on MELRP. The third formulation, which we call the complete network design formulation, is similar to the second formulation but contains less number of variables and constraints. The last formulation is a median formulation and when the sensor locations are fixed, it reduces to the PELRP.

5.1.1. A Single-Commodity Flow Formulation

In the first arc-flow formulation of ECLRP, two sets of variables are defined to represent the flows. The first set of flow variables, x_{ijkl}, represents the amount of sensor-to-sensor flow between a sensor of type k at point i to a sensor of type l at point j. The second variable set, y_{ijk}, represents sensor-to-sink flow from a sensor of type k at point i to a sink at point j. The flow cost c_{ijk} is calculated using the formula $\gamma^k d_{ij}^{\theta}$. Here, d_{ij} is the known distance between points i and j of the sensor field, γ^k is the amount of energy spent by sensor type k for one unit of data flow and θ is the path loss factor which is usually assumed to be between 2 and 4. This formulation can be

seen as the combination of our coverage model (ECP) with uncertain detection and Energy-aware Single-Commodity Flow Formulation (SELRP). The MILP formulation for this case is given below.

SECLRP:

$$\min \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{l\in\mathcal{K}} c_{ijk}x_{ijkl} + \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{N}} c_{ijk}y_{ijk} \tag{5.1}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}} a_{jik}s_{jk} \geq b_i \qquad i\in\mathcal{N} \tag{5.2}$$

$$\sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}} h_k s_{jk} \leq B \tag{5.3}$$

$$\sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{l\in\mathcal{K}} x_{jilk} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y_{jil} + s_{ik} = \sum_{j\in\mathcal{N}} y_{ijk} + \sum_{\substack{j\in\mathcal{I}\\ j\neq i}}\sum_{k\in\mathcal{K}} x_{ijkl} \qquad i\in\mathcal{N},\, l\in\mathcal{K} \tag{5.4}$$

$$\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}} y_{ijk} \leq KNz_j \qquad j\in\mathcal{N} \tag{5.5}$$

$$\sum_{j\in\mathcal{N}} z_j = p \tag{5.6}$$

$$x_{ijkl}, y_{ijk} \geq 0 \qquad i,\ j\in\mathcal{N};\ j\neq i,\, k\in\mathcal{K},\, l\in\mathcal{K} \tag{5.7}$$

$$z_j, s_{ik} \in \{0,1\} \qquad i\in\mathcal{N},\ j\in\mathcal{N},\, k\in\mathcal{K} \tag{5.8}$$

The objective function (5.1) minimizes the sum of sensor-to-sensor and sensor-to-sink routing costs. Coverage constraints (5.2) force to place a sufficient number of sensors to satisfy the desired coverage quality for all points in the sensor field. A budget constraint (5.3) limits the number of sensors to be installed. Constraints (5.4) are the flow balance equations, where the left-hand side of the equation is the sum of the total inflow from other sensors to the sensor at point i, the total inflow to the sink at point i and the data generated by the sensor of type k at point i. The right-hand side of the equation is the total outflow, which is the sum of the total outflow to the sinks or to other sensors. Feasibility constraints (5.5) ensure that there is no sensor-to-sink flow

if there is no sink at the destination. The Constraint (5.6) sets the number of sinks to be installed equal to p and finally constraints (5.7) and (5.8) are the non-negativity and binary restrictions on the variables. Observe that there are $O(N^2K^2)$ variables and $O(NK)$ constraints in the formulation.

5.1.2. A Multi-Commodity Flow Formulation

The second formulation combines the ECP and MELRP formulations introduced previously. Here we consider a multi-commodity flow formulation to model sink location and data routing. As is the case in MELRP, q is the commodity index, but this time, the commodity set $\mathcal{S}$ consists of all pairs in $\mathcal{N} \times \mathcal{K}$ since we do not know the exact locations of sensors. This is a much larger set compared to that of MELRP. In MELRP, s_{ik}^q is a parameter, but now it is a binary decision variable defined as $s_{ik}^q = 1$ if commodity q is generated by a type k sensor at point i and $s_{ik}^q = 0$ otherwise. Again there are two types of flow variables, which are x_{ijkl}^q and y_{ijk}^q. Here, x_{ijkl}^q denotes the flow amount of commodity q from a type k sensor at point i to a type l sensor at point j. y_{ijk}^q denotes the flow amount of commodity q from a type k sensor at point i to a sink at point j. Similar to PELRP, w_{ik}^q represents the amount of commodity q that is collected by the sink at point i. Although these variables are only non-negative variables, they always have binary values in an optimal solution. The multi-commodity flow formulation of the integrated problem ECLRP is given below.

MECLRP:

$$\min \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{l\in\mathcal{K}}\sum_{q\in\mathcal{S}} c_{ijk}x_{ijkl}^q + \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{q\in\mathcal{S}} c_{ijk}y_{ijk}^q \tag{5.9}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{q\in\mathcal{S}} a_{jik}s_{jk}^q \geq b_j \qquad i \in \mathcal{N} \tag{5.10}$$

$$\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{N}}\sum_{q\in\mathcal{S}} h_k s_{jk}^q \leq B \tag{5.11}$$

$$\sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{l\in\mathcal{K}} x^q_{jilk} + s^q_{ik} = \sum_{\substack{j\in\mathcal{N}\\ j\neq i}}\sum_{l\in\mathcal{K}} x^q_{ijkl} + \sum_{\substack{j\in\mathcal{N}\\ j\neq i}} y^q_{ijk} + w^q_i \qquad i\in\mathcal{N},\ k\in\mathcal{K},\ q\in\mathcal{S} \tag{5.12}$$

$$\sum_{i\in\mathcal{N}} w^q_i = \sum_{i\in\mathcal{N}}\sum_{k\in\mathcal{K}} s^q_{ik} \qquad q\in\mathcal{S} \tag{5.13}$$

$$w^q_{jk} \leq z_j \qquad j\in\mathcal{N},\ q\in\mathcal{S} \tag{5.14}$$

$$\sum_{j\in\mathcal{N}} z_j = p \tag{5.15}$$

$$x^q_{ijkl},\ y^q_{ijk},\ w^q_i \geq 0 \qquad i,\ j\in\mathcal{N};\ j\neq i,\ k\in\mathcal{K},\ l\in\mathcal{K},\ q\in\mathcal{S} \tag{5.16}$$

$$z_j, s^q_{jk} \in \{0,1\} \qquad j\in\mathcal{N},\ k\in\mathcal{K},\ q\in\mathcal{S} \tag{5.17}$$

Constraints (5.10) and (5.11) are coverage and budget constraints, whereas (5.12) are the flow balance constraints. Constraint set (5.13) forces every sensor to send its data to a sink and constraint set (5.14) prevents this unless there is a sink on that point. Constraint (5.15) ensures that a total of p sinks are located and finally, constraints (5.16) and (5.17) are non-negativity and binary restrictions on the variables. According to the observations made in previous chapter, the LP relaxation of multi-commodity flow model (MELRP) is very strong and in many test problems it yields an optimal integer solution. Although the MECLRP formulation is based on MELRP, experimental results indicate that LP relaxation of MECLRP is not strong. This is a consequence of the addition of the coverage and budget constraints, i.e., constraints (5.10) and (5.11), to the model.

5.1.3. A Complete Network Formulation

Our third formulation is based on a different perspective. Namely, we formulate the ECLRP on a complete network. Here we add two dummy nodes to the network, an **o**rigin node and a **d**estionation node. We connect these two nodes to all $n\in\mathcal{N}$ points of the original network, so as to form an extended network with $N+2$ points. This way, we can represent the sensor locations, sink locations and commodity flows by a single set of variables v^q_{ijk}, $i,j\in\mathcal{N}\cup o\cup d$, $k\in\mathcal{K}$, $q\in\mathcal{S}$. To be more precise if there is type k sensor at point i generating commodity q, then we set $v^q_{oik}=1$. Similarly, if

there is a sink at point j then, $v^q_{jdk} = 1$. The coverage parameters a_{ijk} and sensor cost parameters h_k are defined as weights for the arcs between o and $n \in \mathcal{N}$. The complete model is as follows.

CECLRP:

$$\min \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{q\in\mathcal{S}} c_{ijk}v^q_{ijk} \tag{5.18}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{q\in\mathcal{S}} a_{jik}v^q_{ojk} \geq b_i \qquad i \in \mathcal{N} \tag{5.19}$$

$$\sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}}\sum_{q\in\mathcal{S}} h_k v^q_{ojk} \leq B \tag{5.20}$$

$$\sum_{j\in\mathcal{N}\cup o\cup d} v^q_{ijk} - \sum_{j\in\mathcal{N}\cup o\cup d} v^q_{jik} \begin{cases} \leq 1, \ i = o \\ = 0, \ i \in N \\ \geq -1, \ i = d \end{cases} \quad k \in \mathcal{K}, \ q \in \mathcal{S} \tag{5.21}$$

$$v^q_{idk} \leq z_i \qquad i \in \mathcal{N}, \ k \in \mathcal{K}, \ q \in \mathcal{S} \tag{5.22}$$

$$\sum_{i\in\mathcal{N}} z_i = p \tag{5.23}$$

$$v^q_{ijk}, \ z_i \in \{0,1\} \qquad i,j \in \mathcal{N}\cup o\cup d, \ k \in \mathcal{K}, \ q \in \mathcal{S} \tag{5.24}$$

The objective function minimizes sum of all flow costs in the extended network. Since the c_{ojk} and c_{idk} values are zero, they can be dropped from the summations in the objective function. The coverage constraints (5.19) and budget constraint (5.20) are as the same as the ones of the previous models. Constraint set (5.21) requires further explanation. There must be a unit outflow from the origin node o for each commodity. Since, the locations of the sensors are not known, it is also unknown which of the arcs v^q_{ojk} connecting the origin node o to the destination points $j \in \mathcal{N}$ in the network will be set to one. As a consequence, we should use inequality constraints to ensure that there is an outflow from the origin node o, only if there is a sensor at the destination point. Similarly, there is an inflow to the destination point d only if there is a sink at the originating point $j \in \mathcal{N}$. Coverage constraints (5.19) and budget constraint (5.20) will force the necessary number of arc variables to be set to one. Constraint set (5.22)

prohibits the opening of destination arcs if there is no sink at the corresponding points and (5.23) ensures locating exactly p sinks in the network. Finally constraint set (5.24) is the binary restriction on the variables.

5.1.4. An Assignment Formulation

We also formulate ECLRP in such a way that each type k sensor at point i is assigned to a sink at point j. We use binary variables x_{ijk} to represent such assignments. To determine the routing cost of these assignments we need to know which arcs are used in each of these assignments. Thus we introduce binary variable u_{lm}^{ijk}, which is equal to one when arc (l, m) is used in the assignment of a type k sensor at point i to a sink at point j. Similar to the previous formulation, c_{lmk} represents the flow cost between points l and m, when there is a type k sensor at point l. In case the sensor locations are given this formulation directly reduces to PELRP, which is a very efficient SLRP formulation. Hence, we call this integrated model as PECLRP, which is given as follows.

PECLRP:

$$\min \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}\sum_{l\in\mathcal{N}}\sum_{m\in\mathcal{N}}\sum_{k\in\mathcal{K}} c_{lmk}u_{lm}^{ijk} \tag{5.25}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}}\sum_{k\in\mathcal{K}} a_{jik}s_{jk} \geq b_i \qquad i \in \mathcal{N} \tag{5.26}$$

$$\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{N}} h_k s_{jk} \leq B \tag{5.27}$$

$$\sum_{j\in\mathcal{N}} x_{ijk} = s_{ik} \qquad i \in \mathcal{N},\ k \in \mathcal{K} \tag{5.28}$$

$$x_{ijk} \leq z_j \qquad i, j \in \mathcal{N},\ k \in \mathcal{K} \tag{5.29}$$

$$\sum_{j\in\mathcal{N}} z_j = p \qquad i, j \in \mathcal{N},\ k \in \mathcal{K} \tag{5.30}$$

$$\sum_{m\in\mathcal{N}} u_{lm}^{ijk} - \sum_{m\in\mathcal{N}} u_{ml}^{ijk} = \left\{ \begin{array}{l} x_{ijk},\ l = i \\ 0,\ l \in \mathcal{N}\backslash\{i,j\} \\ -x_{ijk},\ l = j \end{array} \right\}$$

$$i,j,l \in \mathcal{N}, i \neq j,\ k \in \mathcal{K} \tag{5.31}$$

$$\sum_{m\in\mathcal{N}} u_{lm}^{ijk} \leq s_{lk} \qquad i,j,l \in \mathcal{N}, i \neq j,\ k \in \mathcal{K} \tag{5.32}$$

$$u_{lm}^{ijk}, x_{ijk} \geq 0 \qquad i,j,l,m \in \mathcal{N},\ k \in \mathcal{K} \tag{5.33}$$

$$s_{ik},\ z_i \in \{0,1\} \qquad i \in \mathcal{N},\ k \in \mathcal{K} \tag{5.34}$$

The objective function (5.25) minimizes the sum of the routing costs over all arcs. Coverage constraints (5.26) and budget constraint (5.27) are the same as the previous formulations. Assignment constraints (5.28) ensure that each sensor is assigned to only one sink. Feasibility constraints (5.29) prohibit any assignment if there is no sink at point j. Here we prefer the strong version of this constraint set, which can be replaced by $\sum_{i\in\mathcal{N}} \sum_{k\in\mathcal{K}} x_{ijk} \leq NKz_j$ to obtain an equivalent formulation. This version has fewer constraints, but it yields a weaker LP relaxation. Constraint (5.30) sets the number of sinks to be installed in the sensor field to p. Constraint set (5.31) is the flow balance constraints. If there is a sensor at point i and a sink at point j, then there is an assignment x_{ijk}. Hence, there must be a unit outflow from the source (type k sensor at point i) and a unit inflow to the destination (sink at point j), and for all other intermediate points the sum of the total inflow and outflow should be zero. Observe that the cost of using arc u_{lm}^{ijk} is c_{lmk}. Hence, the objective function and these constraints force each sensor to use the shortest path from itself to the sink it is assigned to. Arc-assignment feasibility constraints (5.32) are necessary to restrict the use of an arc (l,m) if there is no sensor at the origin point l of arc (l,m). The final set of constraints (5.33) and (5.34) are again non-negativity and binary restrictions on the variables. Although we defined x_{ijk} and u_{lm}^{ijk} as binary, these requirements can be relaxed to non-negativity requirements, because at the optimum solution these variables will be either zero or one. When the sensor locations are given this problem can be reduced to the classical *p-median* problem. In this formulation the number of variables is $O(N^4K)$ and the

number of constraints is $O(N^3K)$. Although, it seems more difficult, this formulation becomes very useful for computing approximate solutions of ECLRP as explained in the next section.

5.2. Solution Methods

All four problems presented in this chapter are NP-hard, so no polynomial-time algorithm is available for solving them. Although SECLRP is the most efficient formulation among the four formulations, due to the smaller number of variables and constraints, the computation time and memory requirements increase exponentially in the problem size. Therefore, using commercial solvers, one can find optimal solutions only for small networks and none of these models are appropriate to solve ECLRP for medium and large networks. Thus, alternative techniques should be developed to solve ECLRP efficiently.

5.2.1. A Nested Solution Approach

We propose a two-level nested solution approach to solve the ECLRP. In the first level, we place sensors such that the coverage and budget restrictions are satisfied. In the second level, given the sensor locations, we solve the remaining sink location and routing problem (SLRP).

Given a feasible sensor location set $\mathcal{I} \subseteq \mathcal{N}$ (i.e., s_{ik} or s^q_{ik} are fixed), with $|\mathcal{I}| = I$ sensors located in the sensor field, all formulations can be re-written in simpler forms. For fixed sensor locations that satisfy coverage and budget requirements, we can drop variables s_{ik} or s^q_{ik}. Also constraints (5.2) and (5.3) can be dropped from SECLRP. The same is true for constraints (5.10) and (5.11) in MECLRP, and constraints (5.26) and (5.27) in PECLRP. After these reduction steps SECLRP, MECLRP and PECLRP reduce to SELRP, MELRP and PELRP, respectively and have been addressed in the previous chapter. We do not give the reduced versions of the two flow formulations SECLRP and MECLRP, because the reduced formulation of PECLRP, which we call PECLRP$'$, is more efficient. Hence, PECLRP$'$ is used in our nested solution approach,

which is given below.

PECLRP′:

$$\min \sum_{i\in\mathcal{I}} \sum_{k\in\mathcal{S}_i} \sum_{j\in\mathcal{N}} g_{ijk} x_{ijk} \tag{5.35}$$

$$\text{s.t.} \sum_{j\in\mathcal{N}} x_{ijk} = 1 \qquad i \in \mathcal{I},\ k \in S_i \tag{5.36}$$

$$x_{ijk} \leq z_j \qquad i \in \mathcal{I},\ k \in S_i,\ j \in \mathcal{N} \tag{5.37}$$

$$\sum_{j\in\mathcal{N}} z_j = p \tag{5.38}$$

$$x_{ijk},\ y_j \in \{0,1\} \qquad i \in \mathcal{I},\ k \in S_i,\ j \in \mathcal{N} \tag{5.39}$$

To obtain PECLRP′, we first apply a many-to-many shortest path algorithm with the fixed sensor locations to determine the objective function cost parameters g_{ijk}. Next, we formulate the remaining sink location and routing problem as a p-median problem. Observe that the reduced problem PECLRP′ is actually the same as PELRP. Hence, even though PECLRP is the least efficient formulation due its excessive number of variables and constraints when tackled as a pure MILP, it becomes the most efficient formulation in this setting because of its favorable structure after the reduction obtained by fixing the sensor locations.

5.2.2. Metaheuristic Solution Procedures to Determine Sensor Locations

Optimum sensor locations satisfying coverage and budget constraints do not necessarily lead to an optimum solution for PECLRP. Our numerical observations show that optimum sensor locations for coverage and budget produce non-optimal solutions, whose objective values deviate too much from the optimum objective value. Hence, a systematic methodology to obtain favorable sensor locations that produce good solutions for PECLRP is desired. This can be achieved by using metaheuristics to determine sensor locations. We consider two well-known metaheuristic techniques for this purpose: Variable Neighbourhood Search (VNS) and Tabu Search (TS).

5.2.2.1. Initial Sensor Locations. We apply two initialization strategies to start VNS and TS algorithms. Our first strategy is to apply the greedy heuristic we developed in Chapter 2 to obtain an initial feasible sensor set. Our second strategy is just placing the sensors randomly until feasibility in terms of coverage is obtained without violating the budget constraint. We compare the performances of these two deployment strategies in the next chapter, where we present computational results.

5.2.2.2. Variable Neighbourhood Search. VNS is a popular technique to solve combinatorial optimization problems [102]. The basic idea of VNS is a systematic change of neighbourhood within a local search. In our case, we use VNS to obtain good sensor locations that also provide good (hopefully optimal) solutions to ECLRP, when PECLRP$'$ is solved with these sensor locations. The basic steps of VNS is described below in Figure 5.1:

---

1. Initialize. Select neighborhood structures N_k for
   $k = 1, ..., k_{\max}$. Also find an initial solution x;
2. Repeat the following until a stopping condition is satisfied:
   - 2.1. Set $k = 1$;
   - 2.2. Repeat the following steps until $k = k_{\max}$;
     - 2.2.1. Shaking: Generate a point x' at random from the k^{th} neighborhood of x
     - 2.2.2. Local Search: Apply some local search method with x' as initial solution, denote with x'' the local optimum
     - 2.2.3. Move or Not: If this local optimum is better than the incumbent, move there and continue the search with N_1, otherwise set $k := k + 1$;

Figure 5.1. Algorithm VNS_COVER

In our case, every VNS solution is a sensor set and the local optimum is the sensor set with the minimum PECLRP$'$ objective value. The core of algorithm VNS_COVER

is in Step 2.2. While $k \leq k_{\max}$, we repeat the local search steps, where we look for good sensor locations. In the *Shaking* step a feasible random sensor set is generated from the initial solution and in the *Local Search* step r neighboring sets are generated from this set by using various operations, which are explained in the following sections. Next, all these neighbors are evaulated by solving the corresponding PECLRP′. The stopping criteria is a maximum number of iterations without any improvement in the best solution of VNS_COVER.

5.2.2.3. Tabu Search. TS is a metaheuristic algorithm that guides the local search to prevent it from being trapped in premature local optima or cycling [103, 104]. This is achieved by prohibiting the moves that cause to return to previously visited solutions during a certain number of iterations. The basic steps of TS_COVER algorithm are given in Figure 5.2.

In the TS_COVER algorithm *num_iter* and *num_nonimp_iter* are counters to keep the total number of iterations and number of iterations without improvement in objective function value of the heuristic. The parameters *max_iter* and *max_nonimp_iter* are the limits for *num_iter* and *num_nonimp_iter*, respectively. *Obj* is the objective function value of the current sensor set and it is obtained by solving PECLRP′ for this set. *Obj_Best_Neigh* is the best solution among all the neighboring solutions in every tabu iteration and Obj^* is the best solution obtained throughout the algorithm. Finally, Max_Tabu_Tenure is a parameter that sets the number of iterations a solution is kept in the tabu list.

Similar to VNS_COVER, TS_COVER algorithm begins with a feasible initial sensor set. Until the stopping criteria is violated, the basic steps of the algorithm is carried out. At each iteration of TS_COVER, r neighbors are generated by using various operations on the current best feasible sensor set. However, while generating new neighbors, the operations in the tabu list are not allowed to be used for certain number of iterations, unless the objective value of the PECLRP′ corresponding to the disallowed neighbor is better than the best solution so far.

5.2.2.4. Neighborhood Structures. At each step of VNS and TS we create candidate neighborhood sets by using SWAP, ADD and DROP operations. The neighborhood size parameter, $N_k = 1, 2$ for VNS. So in each move one or two sensors are swapped, added or dropped. There is no such parameter N_k in TS, but we define 1-SWAP, 2-SWAP; 1-ADD, 2-ADD and 1-DROP, 2-DROP operations for TS.

In the SWAP moves we randomly drop one sensor and then add one new sensor so that the total number of sensors is unchanged. We randomly add (drop) a sensor in the ADD (DROP) move. When $N_k = 2$ for VNS or when we are executing 2-operators in TS, then we swap, drop or add two sensors simultaneously. If the new sensor set is feasible, its associated PECLRP$'$ is constructed and the corresponding objective value is calculated by solving PECLRP$'$ using the techniques we described in the previous chapter. In a heterogeneous sensor network, with different sensing ranges and sensor costs, all of these moves may cause infeasibility by violating the coverage or budget constraints, respectively. When such infeasibilities are observed, PECLRP$'$ is not solved.

The total number of neighbors generated by each of these moves is a parameter, which we call r. In our experiments we use different r values to enrich the local search space, where the neighboring solutions are generated by each move.

In the TS_COVER algorithm, the tabu list contains the inverse of the operations that generate the best neighboring solution, instead of keeping the whole sensor list of a solution. For instance, suppose a best neighboring solution is obtained by a 1-ADD operation so that a type k sensor is added at point i. Then the corresponding 1-DROP operation is recorded in the tabu list so that in the next *Tabu_Tenure* iterations, all neighbours generated by removing this type k sensor at point i are rejected, unless the best objective value is improved, which is called the aspiration criteria.

5.2.2.5. Determining a Local Minimum. When we are given the feasible sensor locations, we can easily reduce PECLRP to PECLRP$'$. The evaluation criterion of a feasible

1. Initialize. Initialize TS parameters, find an initial solution;
2. While ($num\_iter < max\_iter$) AND ($num\_nonimp\_iter < max\_nonimp\_iter$)
3. do
4. Set Obj_Best_Neigh:=0.
5. For each move type i :
6. Calculate $size\_neigh_i$ and *set* num_neigh:=0.
7. While *(*$num\_neigh < size\_neigh_i$*)*
8. Generate a new neighbor by executing move i on the current solution.
9. Calculate Obj for this new solution.
10. Record this new solution so it is not re-generated in current iteration.
11. If $Obj > Obj\_Best\_Neigh$ then do
12. Check if the new solution is tabu-active.
13. If it is not tabu-active or if $Obj > Obj^*$ then do
14. Set $Obj\_Best\_Neigh := Obj$
15. If $Obj > Obj^*$ then do
16. Set this neighbor as the incumbent, update $Obj^* := Obj$
Set num_nonimp_iter:=0.
17. End If
18. End If
19. End If
20. $num\_neigh := num\_neigh + 1$.
21. End While
22. End For
23. Add the best neighboring solution to tabu list and set it as the current solution.
Generate a random integer number in the interval [1, Max_Tabu_Tenure] and set it as the current solution's tabu tenure.
Record num_iter as the start time of the current solution's tabu status.
24. $num\_nonimp\_iter := num\_nonimp\_iter + 1$. $num\_iter := num\_iter + 1$.
25. End While

Figure 5.2. Algorithm TS_COVER

sensor set is the objective value obtained by solving the corresponding PECLRP′. Both VNS and TS start with a feasible initial sensor set and at each step of both algorithms a local search around this set is performed. We generate r neighbors (sensor sets) by using the operations described above and all the feasible sensor sets among these r neighbors are evaluated by calculating the objective function of the corresponding PECLRP′. If we find a better solution, we update our best feasible sensor set. Hence, at every iteration of our metaheuristic algorithm, first we generate r neighbors. Next, if a generated neighbor (sensor set) is feasible with respect to coverage and budget constraints, then we do a preprocessing step with Floyd-Warshall shortest-path algorithm to determine the shortest path costs g_{ijk} and then solve the sink location - data routing problem (PECLRP′). This process continues until the maximum number of total iterations or maximum number of iterations without any improvement in the objective value is exceeded.

While solving PECLRP′, we test two methods, which are the Lagrangean Heuristic and the Nested-Dual Heuristic. Notice that these are the two most efficient methods to solve PECLRP′, which are explained in the previous chapter. As NDH is a faster algorithm compared to LH, it can solve more instances of PECLRP′ within the same time interval. As a consequence, we increase the neighborhood size parameter r to $2r$ for NDH. Experimental analysis show that increasing the neighborhood size considerably improves the quality of the heuristic solution.

6. COMPUTATIONAL RESULTS

In this section, we present the computational results obtained by solving various different test instances of the problems introduced in the previous chapters. First, we will analyze the computational results on the Coverage Problems. Then we will continue with the Sink Location and Routing Problems and finally conclude this chapter with the results of the Integrated Problems.

6.1. Coverage Problems

The performance of various solution techniques for Effective Coverage Problem (ECP) is analyzed detailly in this section. We report results only for perfect and uncertain detection and exclude the results for imperfect detection. The structure of the coefficient matrix in case of imperfect detection is very much like the one of uncertain detection and one should expect similar behavior in the solution methods. This can also be observed from the definition of their gain factors used in GREEDY_HEURISTIC.

6.1.1. Test Environment

We consider three types of sensors in our test problems. The specifications of these three sensors are given in Table 6.1 in terms of unit costs ($) assumed to be independent of the location (i.e. $c_{jk} = c_k$, $(j,k) \in J$), sensing ranges (grid points) and exponential decay parameters, which are necessary for calculating the sensing probabilities for uncertain detection. We have paid special attention in order to have an increasing relation between the price and quality of a sensor, as done in [12], while selecting these values. As can be observed, as the unit cost of the sensor increases, its sensing range increases and its decay parameter decreases. We actually add the third sensor type to the data used in [12]. In our experiments, we consider the case of uniform coverage.

We assume that the sensor field is in the form of a two-dimensional grid and con-

Table 6.1. Sensor specifications used in the Coverage Problems

Type (k)	Unit cost (c_k)	Sensing range (r_k)	Decay parameter (α_k)
1	100	1	0.60
2	150	2	0.48
3	500	4	0.40

sider 14 different cases with $N = n^2$ grid points where $n = 5$–$15, 20, 30, 40$. This makes a total of 28 instances (perfect and uncertain detection) with $N \times N \times 3$ binary variables where the largest instances contains $1600 \times 1600 \times 3 = 7,680,000$ variables. We first solve these 28 problems optimally by using CPLEX 9.1 [105] to obtain optimal objective values z_{IP}. We then run greedy and Lagrangean heuristics over these instances, and solve the LP relaxations. We let z_{RR}, z_{GA}, z_{GH}, and z_{LH} denote respectively the objective values computed by RELAX_AND_ROUND, GREEDY_APPROXIMATION, GREEDY_ HEURISTIC and LAGRANGEAN_ HEURISTIC, while z_{LP} denotes the lower bound calculated by solving the LP relaxation. We report the percent deviations in Table 6.2 and Table 6.4. They are calculated according to the formula:

$$100 \times \frac{z - z_{IP}}{z_{IP}} \tag{6.1}$$

where $z \in \{z_{LP}, z_{GH}, z_{LH}, z_{RR}, z_{GA}\}$. The first column includes the number of points in the sensor field. The second column includes the optimal objective values for the corresponding field size. The values in the last row are obtained by taking the column averages. Notice that since they are equal to the bounds computed by solving LP relaxations as explained before, no lower bound is reported by solving the Lagrangean dual. z_{LP} is always less than optimal objective value z_{IP}, which explains the negative percent deviations reported in the third columns of Table 6.2 and Table 6.4. CPU times are given in Table 6.3 and Table 6.5. Their columns are labelled with the indices of the elements of z. On some instances CPLEX 9.1 runtimes become prohibitive. We therefore put 10,000 seconds as a limit on the execution time. This limit is determined depending on the time consumed by LAGRANGEAN_HEURISTIC to compute z_{LH}. It is set to 10,000 for all instances since the computational time of the heuristics

are below this value. The only exception is the last instance of uncertain detection with $N = 1600$; LAGRANGEAN_HEURISTIC consumes between 10,000 and 15,000 seconds, thus we set the limit to 15,000 for this instance. In short, every reported z_{IP} value is not necessarily optimal. Marked ones are the best values CPLEX 9.1 computes within the listed time bounds. They are marked with an "*" in both tables.

We perform the experiments on a PC with 1 GByte RAM and 1.6 GHz Centrino processor operating under Windows 2000 Server. We run CPLEX 9.1 [105] with default options while solving ECP instances.

6.1.2. Perfect Detection

We set $b = 2$, meaning that every point must be covered by at least two sensors. The percent deviations calculated according to formula (6.1) are reported in Table 6.2. We can easily observe that the upper bounds obtained with LAGRANGEAN _HEURISTIC are the tightest on the average: It is 11.6%. The second and third best are GREEDY_ APPROXIMATION with 17.1%, and GREEDY_HEURISTIC with 21.7%. RELAX_AND_ROUND has the lowest accuracy with 395.5%.

The CPU performances can be found in Table 6.3. Observe the increase in the efficiency with increasing grid size. CPLEX 9.1 consumes 22157 seconds on the average for 14 instances. These values become 5.4, 961.9, 46.23 and 12.12 seconds for GREEDY_HEURISTIC, LAGRANGEAN_HEURISTIC, RELAX_AND_ROUND and GREEDY_APPROXIMATION.

6.1.3. Uncertain Detection

We set $b = 0.99$, which gives an upper bound of $T = 0.01$ on the miss probability. The percent deviations calculated according to formula (6.1) are reported in Table 6.4. It has the same format as Table 6.2. As can be observed, heuristics become more accurate for uncertain detection. The improvements in the upper bounds are more prominent. GREEDY_HEURISTIC results in an average percent deviation

Table 6.2. Percent deviations for perfect detection

N	z_{IP}	z_{LP}	z_{GH}	z_{LH}	z_{RR}	z_{GA}
25	1000	- 4.0	15.0	5.0	150.0	15.0
36	1200	- 2.0	16.7	16.7	283.3	12.5
49	1550	- 4.5	25.8	25.8	216.1	19.4
64	2050	- 4.4	19.5	17.1	212.2	12.2
81	2450	- 3.7	26.5	18.4	230.6	20.4
100	2900	- 2.4	20.7	13.8	244.8	25.9
121	3500	- 5.7	21.4	5.7	245.7	14.3
144	4000	- 2.4	32.5	0.0	260.0	22.5
169	4550	- 3.7	24.2	6.6	271.4	25.3
196	5200	- 3.8	26.0	8.7	276.9	23.1
225	5950	- 3.6	28.6	10.1	278.2	21.8
400	10400*	- 7.7	25.0	13.5	284.6	13.9
900	23600*	- 12.5	16.7	10.0	281.4	11.7
1600	43350*	- 17.9	9.2	7.2	269.1	5.3
Average		- 5.6	22.0	11.3	250.3	17.4
Standard Dev.		4.4	6.2	6.7	38.0	6.1

of 7.0% which is outperformed by the Lagrangean heuristics with an average value of 3.7%. In short, the heuristics make the real difference for uncertain detection. However, there is a considerable decrease in the accuracy of the approximation algorithms RELAX_AND_ROUND and GREEDY_APPROXIMATION. The new average deviations are 220.8% and 36.5%. There is also a decrease in the efficiency as expected. According to the values of Table 6.5, CPLEX 9.1 consumes 8981.45 seconds on the average. These values become 4.76 and 1245.01 seconds for GREEDY_HEURISTIC and LAGRANGEAN_HEURISTIC, respectively. The two approximation algorithms are also quite efficient: RELAX_AND_ROUND and GREEDY_APPROXIMATION require respectively 85.71 and 14.47 seconds.

Table 6.3. CPU times for perfect detection in seconds

N	$CPLEX$	GH	LH	RR	GA
25	0.04	0.01	1.2	0.01	0.05
36	0.04	0.01	5.6	0.02	0.09
49	0.11	0.01	4.4	0.03	0.10
64	0.23	0.01	6.7	0.05	0.10
81	0.43	0.02	9.6	0.07	0.10
100	1.01	0.06	17.8	0.10	0.10
121	2.22	0.08	25.8	0.12	0.11
144	1.12	0.12	35.9	1.14	0.12
169	2.82	0.13	58.3	0.17	0.17
196	9.56	0.21	84.3	0.31	0.46
225	14.59	0.23	91.4	0.46	0.74
400	10000	1.03	334	1.30	1.1
900	10000	6.8	1921	9.3	8.1
1600	10000	62.0	6029	39.9	53.3
Average	2145.1	5.4	616.1	3.78	4.62
Standard Dev.	4256.7	16.5	1637.5	10.67	14.17

6.2. Sink Location and Routing Problems

In this section, we report our experimental results on the energy-aware location and routing problems. Energy aware models are more realistic than cost aware models since saving in total energy is a more important issue than saving in total cost in most of the real life applications. Besides, both formulations have similar characteristics. They only differ in the way they restrict the number of sinks. In the energy-aware models there is an explicit constraint involving the number of sinks, which is replaced by the fixed-cost penalty term added to the objective function in the cost-aware models. Therefore, we have preferred to conduct our computational tests for only energy-aware models.

Table 6.4. Percent deviations for uncertain detection

N	z_{IP}	z_{LP}	z_{GH}	z_{LH}	z_{RR}	z_{GA}
25	1500	- 47.9	13.3	6.7	66.7	33.3
36	1950	- 46.6	7.7	2.6	84.6	23.1
49	2400	- 44.3	12.5	4.2	104.2	33.3
64	2950*	- 43.5	8.5	1.7	116.9	25.4
81	3500*	- 42.0	8.6	2.9	131.4	25.7
100	4150*	- 41.4	8.4	3.6	141.0	22.9
121	4800*	- 40.3	12.5	4.2	152.1	22.9
144	5500*	- 39.4	7.3	5.5	161.8	25.5
169	6250*	- 38.6	10.4	5.6	170.4	23.2
196	7150*	- 38.8	6.3	6.3	174.1	21.7
225	8000*	- 28.2	8.8	3.8	181.3	22.5
400	13350*	- 37.6	7.5	3.4	199.6	16.1
900	27750*	- 38.7	2.7	1.6	224.3	11.7
1600	47300*	- 35.4	2.1	0.3	238.3	9.3
Average		- 40.9	8.3	3.7	153.3	22.6
Standard Dev.		3.5	3.2	1.8	48.2	6.5

6.2.1. Test Environment

Test problems are generated for $N = n \times n$ square sensor fields with $n \in \{5 - 15,$ 20, 30, 40, 50, 60, 70, 80, 90, 100$\}$. This means that there is a total of $N = n^2$ points in the grid. The sensor locations on the sensor field are given a priori and it is assumed that the sensors are located in such a way that connectivity and coverage requirements are guaranteed. There are two types of sensors with different ranges and costs. The sensor locations and types are determined by solving the effective coverage problem. This also gives the number of sensors in the network. We classify a test problem *small or medium* if $n \in \{5, \ldots, 50\}$ and *large* if $n \in \{60, \ldots, 100\}$. The smallest test instance considers a grid of size 25 with already located 11 sensors. The grid size becomes 10000 and the number of sensors becomes 2500 for the largest instance. The experiments are

Table 6.5. CPU times for uncertain detection in seconds

N	$CPLEX$	GH	LH	RR	GA
25	0.39	0.03	2.0	0.02	0.1
36	31.4	0.04	4.1	0.03	0.11
49	263	0.05	5.3	0.03	0.12
64	10000	0.12	13.6	0.06	0.14
81	10000	0.12	15.9	0.06	0.17
100	10000	0.15	22.1	0.11	0.20
121	10000	0.18	46.8	0.15	0.26
144	10000	0.22	62.0	0.20	0.41
169	10000	0.34	94.2	0.31	0.54
196	10000	0.39	108	0.44	0.74
225	10000	0.54	214	0.64	0.96
400	10000	1.7	581	2.65	2.1
900	10000	9.3	1483	23.7	22.6
1600	15000	41.4	11040	119	104.3
Average	8235.4	3.90	978.1	10.52	9.51
Standard Dev.	4436.7	10.66	2817.3	30.66	27.01

repeated for $p = 1$, $p = 2$ and $p = 3$. Namely, we conduct our experiments with $20 \times 3 = 60$ test instances.

Parameter γ_k, which represents the cost per unit energy spent, is set equal to 10nAh (nano Ampere hour), and parameter α which is the path loss factor is set to 2. The cost associated with the unit data flow between points i and j denoted by c_{ij} is computed as $c_{ij} = \gamma d_{ij}^2$. All experiments are carried out on a Dell PowerEdge 2400 computer with two 64-bit, 2.66-GHz Xeon 5355 Quad Core processors and 16GB SDRam memory, operating within Windows 2003 Server environment. We run the commercial solver CPLEX 11.0 [105] and Concert technology of CPLEX with the default options. We report the best solution provided by CPLEX 11.0 within $10,000$ seconds.

Table 6.6. Key features of the models and methods

	SELRP	MELRP	PELRP	LH	NDH	CR
Smallest (N/S)	25/11	25/11	25/11	25/11	25/11	25/11
Largest (N/S)	400/86	225/52	2500/465	10000/2500	6400/1634	2500/465
Provides LB	yes	yes	yes	yes	yes	yes
Provides UB	yes	yes	yes	yes	yes	no
Solution Approach	Exact MILP	Exact MILP	Exact MILP	Lagr. Relax.	DUALOC	Reduced LP

Computational experiments are organized in two parts. In the first part, we test the performance of the single and multi-commodity formulations (SELRP and MELRP) and the p-median formulation (PELRP) for the energy-aware location routing problem (ELRP). As will be noticed, PELRP turns out to be the most efficient among them. Therefore, we concentrate on PELRP and evaluate the proposed approximate solution methods, LAGRANGEAN_HEURISTIC (LH), NESTED_DUAL_HEURISTIC (NDH) and CANONICAL_REDUCTION (CR). Flow based formulations suffer from memory insufficiency and we can not solve the MILP formulations SELRP and MELRP for $N > 400$ and $N > 225$. However, PELRP is a more efficient formulation and we can solve problems for $N \leq 2500$. Essentially, the bound found by CR is based on the solution of an LP which we can not solve for instances with $N > 2500$ due to the memory insufficiency. The performances of the heuristics NDH and LH are better: we obtained results for $N \leq 6400$ by NDH and for $N \leq 10000$ by LH. Table 6.6 summarizes the key features of the models and methods. The first and second rows display the smallest and the largest instances we could solve by each method. The third and the fourth rows indicate that all methods provide lower and upper bounds except CR which produces only a lower bound. The fifth row lists the solution approach we used.

6.2.2. Quality of the Formulations

We use two criteria for assessing the quality of the formulations. The first one is the strength, which is measured using the lower bounds that can be obtained by solving their LP relaxations. A stronger formulation has an LP relaxation lower bound closer to the optimal objective value: for a minimization problem the larger the bound, the stronger is the formulation. Our second criterion is the efficiency. We collect CPU statistics for both the solution of the IP models and their LP relaxations using CPLEX 11.0 to evaluate the efficiency of the formulations.

Results on the strength of the formulations are summarized in Table 6.7, Table 6.8 and Table 6.9 for the number of sinks $p = 1$, $p = 2$, and $p = 3$, respectively. Problem sizes are listed in the first two columns according to the grid size N and the number of sensors S. The third column (z^*) includes the optimal objective values for each test instance. There are three groups of columns for each formulation. The first group (z_{LP}), namely columns four, five and six display the quality of the lower bounds, which are obtained by solving the LP relaxations for the formulations SELRP, MELRP and PELRP. They are computed as

$$100 \times \frac{z^* - z_{LP}}{z^*}.$$

Columns seven to nine display the solution times of the LP relaxations while the last three columns provide the solution times of the IP formulations. The largest test instance has $N = 2500$ points and can be solved for PELRP formulation only. Hence, we report the averages and standard deviations for $N \leq 225$, which is the size of the largest instance that can be solved by all formulations within the time limit of 10000 seconds.

We cannot obtain exact optimal solutions for SELRP when $N > 400$ and for MELRP when $N > 225$ because of the excessive number of variables and constraints they have. Thus, compared to SELRP and MELRP, the p-median formulation PELRP is much more efficient. This can be observed by analyzing the numerical results. In

Table 6.7. Comparison of the formulations, $p = 1$

			z_{LP} (%)			CPU_{LP} (sec.)			CPU_{IP} (sec.)		
N	S	z^*	SELRP	MELRP	PELRP	SELRP	MELRP	PELRP	SELRP	MELRP	PELRP
25	11	0.48	79.2	0.0	0.0	0.01	0.05	0.01	0.09	0.06	0.01
36	14	1.52	89.5	0.0	0.0	0.02	0.14	0.01	0.10	0.17	0.01
49	17	2.23	87.4	0.0	0.0	0.03	0.26	0.01	0.14	0.28	0.02
64	20	3.12	91.7	0.0	0.0	0.06	0.30	0.03	0.23	0.34	0.03
81	24	4.68	92.3	0.0	0.0	0.09	0.32	0.04	0.40	0.39	0.04
100	28	6.55	90.2	0.0	0.0	0.10	0.59	0.04	0.39	0.68	0.06
121	32	9.35	93.4	0.0	0.0	0.13	1.10	0.04	0.65	1.23	0.04
144	36	9.78	92.8	0.0	0.0	0.18	1.88	0.12	0.89	1.93	0.12
169	41	12.68	92.7	0.0	0.0	0.22	2.09	0.15	2.15	2.28	0.15
196	47	16.58	91.9	0.0	0.0	0.29	3.89	0.38	3.45	4.15	0.39
225	52	19.00	93.8	0.0	0.0	0.40	4.72	0.41	5.06	5.28	0.42
400	86	38.44	94.1	-	0.0	1.01	-	1.89	35.81	-	1.93
900	180	119.10	-	-	0.0	-	-	30.20	-	-	32.53
1600	297	253.62	-	-	0.0	-	-	179.3	-	-	187.09
2500	465	472.40	-	-	0.0	-	-	1639	-	-	1726
Avg. ($N \leq 225$)			90.7	0.0	0.0	0.14	1.40	0.11	1.23	1.53	0.12
St. dev ($N \leq 225$)			4.1	0.0	0.0	0.12	1.60	0.14	1.61	1.73	0.14

all cases, PELRP is solved faster than the other two formulations. On the average, it takes 0.124, 0.130 and 0.193 seconds to solve PELRP, for $p = 1$, $p = 2$ and $p = 3$, respectively. Multi-commodity formulation MELRP is solved in 1.530, 1.279 and 0.842 seconds on the average. Although the performance of SELRP is better than MELRP for $p = 1$ with an average CPU time of 1.236 seconds, it becomes considerably worse for $p = 2$ and $p = 3$ resulting in the average CPU times of 78.628 and 1865.451 seconds, respectively.

The percent deviation of the lower bounds from the optimal objective value provides a means of quick comparison for the strength of the formulations. The LP relaxation of SERLP is weak; on the average it deviates 90.76%, 93.29% and 90.73% for $p = 1$, $p = 2$ and $p = 3$, respectively. However, the LP relaxations of both MELRP and PELRP are very strong. Indeed, they all yield the optimum integer solutions.

Table 6.8. Comparison of the formulations, $p = 2$

			z_{LP} (%)			CPU_{LP} (sec.)			CPU_{IP} (sec.)		
N	S	z^*	SELRP	MELRP	PELRP	SELRP	MELRP	PELRP	SELRP	MELRP	PELRP
25	11	0.33	78.8	0.0	0.0	0.01	0.06	0.01	0.11	0.07	0.01
36	14	0.70	88.6	0.0	0.0	0.02	0.08	0.01	0.42	0.09	0.01
49	17	1.16	89.7	0.0	0.0	0.04	0.13	0.02	0.78	0.14	0.02
64	20	1.61	94.4	0.0	0.0	0.06	0.16	0.03	2.20	0.17	0.03
81	24	2.42	93.4	0.0	0.0	0.09	0.27	0.04	4.84	0.29	0.05
100	28	3.56	94.7	0.0	0.0	0.11	0.43	0.08	11.23	0.50	0.09
121	32	5.01	95.6	0.0	0.0	0.13	0.73	0.08	21.11	0.78	0.08
144	36	6.26	96.2	0.0	0.0	0.20	1.50	0.15	44.82	2.12	0.16
169	41	8.04	96.5	0.0	0.0	0.25	1.63	0.25	129.9	1.64	0.25
196	47	9.28	96.6	0.0	0.0	0.31	3.19	0.33	229.6	3.34	0.33
225	52	11.50	97.1	0.0	0.0	0.44	4.77	0.39	419.8	4.89	0.40
400	86	27.26	98.0	-	0.0	1.14	-	2.81	4911	-	2.85
900	180	82.60	-	-	0.0	-	-	39.66	-	-	40.31
1600	297	193.2	-	-	0.0	-	-	265.2	-	-	267.9
2500	465	351.5	-	-	0.0	-	-	1881	-	-	1939
Avg. ($N \leq 225$)			93.3	0.0	0.0	0.15	1.18	0.13	78.62	1.27	0.13
St. dev. ($N \leq 225$)			5.4	0.0	0.0	0.13	1.52	0.13	134.3	1.59	0.09

Theoretically, LP relaxations of these formulations do not guarantee integer solutions, so we included the CPU times of the formulations when they are solved as IPs. In all IP instances, optimal integer solutions are obtained in the root node of the branch-and-bound tree: the optimal solution of the first LP relaxation is integer.

This is a consequence of the strength of the MELRP and PELRP formulations. Although we have not been able to show the strength of the MELRP and PELRP formulations theoretically, the past experience points that the multi-commodity flow approach results in very strong formulations for some of the difficult combinatorial optimization problems, such as the asymmetric traveling salesman problem [91, 106] and facility location and transportation design problem [92]. Furthermore p-median formulations have very tight linear programming lower bounds [94, 107].

Table 6.9. Comparison of the formulations, $p = 3$

			z_{LP} (%)			CPU_{LP} (sec.)			CPU_{IP} (sec.)		
N	S	z^*	SELRP	MELRP	PELRP	SELRP	MELRP	PELRP	SELRP	MELRP	PELRP
25	11	0.24	70.8	0.0	0.0	0.01	0.05	0.01	0.20	0.06	0.01
36	14	0.48	83.3	0.0	0.0	0.02	0.07	0.01	0.64	0.07	0.03
49	17	0.83	85.5	0.0	0.0	0.04	0.09	0.02	2.92	0.09	0.03
64	20	1.11	91.9	0.0	0.0	0.06	0.13	0.04	13.31	0.14	0.04
81	24	1.86	90.4	0.0	0.0	0.09	0.21	0.05	33.13	0.21	0.05
100	28	2.46	92.3	0.0	0.0	0.11	0.33	0.06	99.87	0.34	0.06
121	32	3.45	93.6	0.0	0.0	0.14	0.53	0.12	291.1	0.54	0.12
144	36	4.36	94.5	0.0	0.0	0.23	0.91	0.13	853.1	0.92	0.13
169	41	5.86	95.4	0.0	0.0	0.30	1.41	0.34	3536	1.45	0.35
196	47	6.55	95.1	0.0	0.0	0.34	2.10	0.52	6841	2.18	0.53
225	52	8.56	98.4	0.0	0.0	0.45	3.16	0.76	8847	3.21	0.77
400	86	20.18	97.5	-	0.0	1.30	-	3.18	9999*	-	3.24
900	180	62.25	-	-	0.0	-	-	75.40	-	-	78.20
1600	297	147.9	-	-	0.0	-	-	729.8	-	-	734.1
2500	465	273.6	-	-	0.0	-	-	2558	-	-	2600
Avg. ($N \leq 225$)			90.7	0.0	0.0	0.18	0.83	0.14	1865	0.84	0.19
St. dev. ($N \leq 225$)			7.68	0.00	0.00	0.14	1.01	0.25	3164	1.04	0.25

Also one can notice that as p increases, the optimal value indicating the energy spent in the network, decreases. This is expected, because when we have more sinks, the average distance over which each sensor transmits its data decreases, which in turn results in energy savings.

6.2.3. Performance of the Solution Methods

In this section, the performance of the new heuristics and lower bounding procedures are evaluated. The performance of a computational method depends on both its accuracy and efficiency. According to our numerical analysis, both LAGRANGEAN_HEURISTIC and NESTED_DUAL_HEURISTIC can find exact solutions for small and medium size instances and provide good feasible solutions for very large instances that cannot be solved exactly due to resource limitations.

Table 6.10. Comparison of the upper bounds: % deviation from the optimum value

		$p = 1$		$p = 2$		$p = 3$	
N	S	LH	NDH	LH	NDH	LH	NDH
25	11	0.0	0.0	0.0	0.0	50.0	4.2
36	14	0.0	0.0	0.0	0.0	81.3	31.3
49	17	0.0	0.0	0.0	0.0	9.6	9.6
64	20	0.0	0.0	0.0	16.8	0.0	18.0
81	24	0.0	0.0	0.0	24.0	12.0	27.7
100	28	0.0	0.0	0.0	3.4	0.0	8.5
121	32	0.0	0.0	0.0	0.4	0.0	0.0
144	36	0.0	0.0	0.0	56.2	0.0	1.4
169	41	0.0	0.0	0.0	57.7	0.0	0.3
196	47	0.0	0.0	4.5	78.7	69.6	26.9
225	52	0.0	0.0	0.0	65.2	0.0	11.4
400	86	0.0	0.0	0.0	41.0	31.6	6.4
900	180	0.0	0.0	20.2	1.8	0.0	11.2
1 600	297	0.0	0.0	19.8	26.0	63.9	23.1
2 500	465	0.0	0.0	3.7	34.9	48.9	72.6
Average		0.00	0.00	3.21	27.08	24.46	16.84
Std dev.		0.00	0.00	6.97	27.24	30.10	18.57

6.2.3.1. Small and Medium Size Instances. We can solve every instance in this group using all three methods, which are LAGRANGEAN_HEURISTIC (LH), NESTED_DUAL_HEURISTIC (NDH) and CANONICAL_REDUCTION (CR). The first two provide both lower and upper bounds, while CANONICAL_REDUCTION gives only lower bounds (i.e., it is a dual based LP method). So we compare the upper bounds generated by LH and NDH, and the lower bounds as well as CPU times of all three methods.

In Table 6.10, we report the percent deviation of the upper bounds obtained by LH and NDH from the optimal objective values. The lower the percent deviation, the better is the performance of the algorithm. For $p = 1$ both methods yield the optimum integer solution for all instances. When $p = 2$, LH outperforms NDH with an average percent deviation of 3.21% against 27.08%. However, for $p = 3$, NDH outperforms LH with an average percent deviation of 16.84% compared to 24.46%.

Table 6.11. Comparison of the lower bounds: % deviation from the optimum value

		$p=1$			$p=2$			$p=3$		
N	S	CR	LH	NDH	CR	LH	NDH	CR	LH	NDH
25	11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.2
36	14	0.0	0.0	0.0	0.0	0.0	10.0	0.0	0.0	21.9
49	17	0.0	0.0	0.0	0.0	0.1	21.6	0.0	0.0	34.1
64	20	0.0	0.0	0.0	0.0	0.0	8.1	0.0	0.0	2.4
81	24	0.0	0.0	0.0	0.0	0.0	10.7	0.0	0.0	2.2
100	28	0.0	0.0	0.0	0.0	0.0	20.6	0.0	0.0	7.9
121	32	0.0	0.0	0.0	0.0	0.0	5.8	0.0	0.0	1.0
144	36	0.0	0.0	0.0	0.0	0.0	5.9	0.0	0.0	8.1
169	41	0.0	0.0	0.0	0.0	0.0	3.6	0.0	0.0	5.9
196	47	0.0	0.0	0.0	0.0	0.0	3.7	0.0	0.0	11.3
225	52	0.0	0.0	0.0	0.0	0.0	7.1	0.0	0.0	13.2
400	86	0.0	0.0	0.0	0.0	0.0	2.3	0.0	0.1	8.1
900	180	0.0	0.0	0.0	0.0	0.1	3.4	0.0	0.0	4.5
1 600	297	0.0	0.0	0.0	0.0	0.2	13.8	0.0	0.0	0.6
2 500	465	0.0	0.0	0.0	0.0	0.0	22.4	0.0	0.0	15.6
Average		0.00	0.00	0.00	0.00	0.03	9.27	0.00	0.01	9.39
Std. dev.		0.00	0.00	0.00	0.00	0.06	7.26	0.00	0.03	9.02

Table 6.11 compares the lower bounds of the algorithms in terms of percent deviations. An optimum integer solution is obtained when the lower bound is equal to the upper bound; so the higher the lower bound, the better is its quality. Here CR and LH are the best performers; in all instances they provide an optimal integer solution. Due to the integrality property, LH is expected to give eventually the solution of the LP relaxation of PELRP. In some instances, very small deviations are observed. This is a consequence of the final precision used for stopping the subgradient optimization; a higher precision results in smaller deviations. The lower bounds obtained with NDH deviate from the optimal by 9.27% and 9.39% for $p = 2$ and $p = 3$, respectively.

Finally, in Table 6.12, the CPU times are reported. Notice that for small sized instances CR and NDH perform the best. As the problem size increases, the performance of CR deteriorates considerably. On the average, NDH outperforms the other

Table 6.12. Comparison of the CPU times

		$p = 1$			$p = 2$			$p = 3$		
N	S	CR	LH	NDH	CR	LH	NDH	CR	LH	NDH
25	11	0.01	0.07	0.01	0.01	0.09	0.01	0.01	0.04	0.01
36	14	0.01	0.04	0.01	0.01	0.04	0.01	0.01	0.09	0.01
49	17	0.01	0.04	0.01	0.01	0.06	0.01	0.01	0.16	0.02
64	20	0.01	0.07	0.02	0.03	0.94	0.03	0.03	0.12	0.02
81	24	0.02	0.09	0.03	0.03	0.14	0.03	0.03	0.18	0.03
100	28	0.02	0.14	0.03	0.05	0.19	0.05	0.05	0.34	0.03
121	32	0.02	0.15	0.03	0.09	0.25	0.09	0.09	0.42	0.10
144	36	0.02	0.21	0.06	0.11	0.42	0.11	0.09	0.45	0.08
169	41	0.04	0.31	0.09	0.20	0.55	0.20	0.19	0.75	0.16
196	47	0.05	0.40	0.14	0.25	0.69	0.25	0.28	0.92	0.18
225	52	0.09	0.48	0.18	0.35	0.85	0.35	0.37	1.45	0.24
400	86	0.51	1.67	0.56	1.64	2.61	1.64	2.77	3.60	1.00
900	180	3.76	9.94	13.96	8.39	15.29	8.39	18.38	18.34	14.82
1600	297	17.81	32.39	26.13	33.18	62.79	33.18	69.99	78.27	73.96
2500	465	414.84	89.52	89.96	123.32	160.38	123.32	1256.76	241.01	125.60
Average		29.14	9.04	8.75	77.93	16.29	11.18	89.93	23.08	14.42
Std. dev.		107.79	23.81	23.64	275.81	43.03	32.19	323.30	63.57	36.20

two methods with an average CPU time of 8.8, 11.2 and 14.4 seconds for $p = 1$, $p = 2$ and $p = 3$, respectively. The computational effort required by LH is less than that of CR with 9.04, 16.3 and 23.1 seconds. Because of the bad performance on larger instances, CR yields an average of 29.2, 77.9 and 89.9 seconds.

When we compare the CPU performances of these three methods with the solution times required for exactly solving the MILP formulations SELRP, MELRP and PELRP, we can easily see that these methods are very efficient. Although it is not displayed in the previous section, even the most accurate formulation PELRP can only be solved within an average of 133.5, 146.8 and 227.9 seconds for $p = 1$, $p = 2$, and $p = 3$ respectively. This clearly shows the efficiency of the heuristics.

6.2.3.2. Large Instances. In real life applications, there are thousands of candidate nodes and sensors in a WSN. Obtaining an optimal solution for such problems is almost impossible with even very efficient formulations. Hence, heuristic methods are inevitable to obtain good solutions within a reasonable computation time. Therefore we extend our numerical experiments for large size networks and consider two methods, namely LAGRANGEAN_HEURISTIC and NESTED_DUAL_HEURISTIC. In Table 6.13, the lower and upper bounds as well as CPU times are given. Since none of the formulations can be solved exactly, we cannot calculate the percent deviations from the optimal objective value. As a result, we report the best values computed by the heuristics. We can only be sure that a solution is optimal when the lower and upper bounds are equal. LAGRANGEAN_HEURISTIC can solve instances with $N \leq 10000$ and NESTED_DUAL_HEURISTIC can solve instances with $N \leq 6400$. As a result, the averages and standard deviations are reported for $3600 \leq N \leq 6400$. For $p = 1$, both methods compute the optimum solutions since lower and upper bounds are all equal. However, LAGRANGEAN_HEURISTIC is faster than NESTED_DUAL_HEURISTIC with an average CPU time of 626.92 seconds. None of the methods can guarantee an optimal solution in any of the instances with $p = 2$ and $p = 3$. LAGRANGEAN_HEURISTIC is better in all aspects: gives higher lower bounds and smaller upper bounds in shorter CPU times.

If we summarize our observations, LH is a viable method to deal with SRLP for very large wireless sensor networks. For a network with 10000 nodes and 2500 sensors, one can determine a good solution within an hour of computation time.

6.3. Integrated Coverage, Sink Location and Routing Problems

In this section, we report the accuracy and efficiency of the solutions generated by the nested solution approach to solve the Coverage, Sink Location and Routing Problem (ECLRP). In this approach, we first seek for the best sensor locations using a metaheuristic procedure (VNS or TS), then by fixing the sensor locations found by VNS or TS, we solve the sink location and routing problem.

Table 6.13. Comparison of the heuristics for large instances

			Lower bound		Upper bound		CPU	
	N	S	LH	NDH	LH	NDH	LH	NDH
$p = 1$	3600	659	756.82	756.82	756.82	756.82	187.89	231.38
	4900	786	1,270.41	1,270.41	1,270.41	1,270.41	366.58	799.77
	6400	1634	1,402.55	1,402.55	1,402.55	1,402.55	1,326.29	3,523.98
	8100	2041	1,864.31	-	1,864.31	-	1,095.57	-
	10000	2500	2,587.66	-	2,587.66	-	1,726.69	-
	Average		1,143.26	1,143.26	1,143.26	1,143.260	626.92	1,518.38
	Std. dev.		341.12	341.12	341.12	341.12	612.23	1,760.00
			Lower bound		Upper bound		CPU	
	N	S	LH	NDH	LH	NDH	LH	NDH
$p = 2$	3600	659	595.02	355.50	670.25	726.31	323.83	427.41
	4900	786	944.65	702.89	944.66	1,143.22	681.43	790.43
	6400	1634	1,018.86	938.70	1,672.98	1,631.32	2,500.08	8,946.05
	8100	2041	1,420.66	-	1,925,18	-	3,512.65	-
	10000	2500	1,924.60	-	3,492.76	-	5,790.79	-
	Average		852.84	665.69	1,095.96	1,166.95	1,168.45	3,387.97
	Std. dev.		226.34	293.36	518.20	452.97	1,167.01	4,816.86
			Lower bound		Upper bound		CPU	
	N	S	LH	NDH	LH	NDH	LH	NDH
$p = 3$	3600	659	478.20	458.01	580.99	616.24	447.65	434.24
	4900	786	739.53	635.37	1,191.70	1,224.15	779.24	798.52
	6400	1634	805.53	805.01	1,250.87	1,119.80	2,454.74	7,366.74
	8100	2041	1,139.36	-	1,800.51	-	3,859.34	-
	10000	2500	1,585.53	-	2,471.64	-	6,838.56	-
	Average		674.42	632.79	1,007.85	986.73	1,227.21	2,866.51
	Std. dev.		173.10	173.51	370.85	325.06	1,075.93	3,901.58

6.3.1. Test Environment

We propose two different metaheuristics to optimize the locations of the sensors: Variable neighborhood Search (VNS) and Tabu Search (TS). We compare two types of initial sensor deployment strategies: random and greedy. We use Lagrangean Heuristic (LH) and Nested-Dual Heuristic (NDH) in solving PECLRP′. To summarize, we have total eight different combinations.

We also solve ECLRP as a stand-alone integer program using Cplex 11.0. The best performing version of ECLRP is the single-commodity formulation version (SE-CLRP). Hence, the best results obtained by solving SECLRP with CPLEX 11.0 are presented. We denote the solutions obtained for the nested solution approach with z_{LH} and z_{NDH} to represent LH and NDH, respectively and the solution of SECLRP obtained by using Cplex 11.0 is represented by z_{IP}. We measure the accuracy of the objective values using percent deviations calculated according to formula:

$$100 \times \frac{X - z_{IP}}{z_{IP}}. \tag{6.2}$$

Here, $X \in \{z_{LR}, z_{ND}\}$. If the result of this formula is positive then the objective value of the solution of the nested approach is worse than the one obtained using Cplex 11.0 and if it is negative the vice versa is true. We also compare the average CPU times of the nested solution approach with the CPU times of IP solution.

Our testbed consists of 12 test problems which are generated for $n \times n$ square grid sensor fields with $n \in \{5\text{–}15, 20\}$. This means that there is a total of $N = n^2$ points in the grid. We consider an open area without many obstacles to test our formulations, by choosing a path loss factor of $\theta = 2$. In addition, we assume that d_{uv}, which is the distance between points u and v of the sensor field, is the Euclidean distance. All experiments are carried out on Dell PowerEdge 2400 with two 64-bit, 2.66-GHz Xeon 5355 Quad Core processors and 16GB SDRam memory, operating within Windows 2003 Server environment. We run CPLEX 11.0 with the default options. We set a limit of 3 hours (10800 seconds) on the running time.

Table 6.14. Properties of Sensors

Type (k)	Cost (h_k)	Coverage (α_k)	Energy Consumption (γ_k)
1	10	0.5	10
2	20	0.4	20

We only have routing costs in the objective function and they are directly related to the energy spent per unit of data flow. A sensor typically spends energy for sensing, processing and transmitting. We consider only the transmission energy, since the other two are neglible. A typical AA type of battery contains 2200 mAh of energy and to transmit one packet of data 10 nAh energy is spent for unit distance [108]. Therefore, we assume the values of the connection weights are simply energy spent per unit data flow. We consider two different types of sensors with different costs h_k, coverage parameters α_k and energy consumption parameters γ_k, as displayed in Table 6.14.

There is no fixed cost associated with sink installation and the number of sinks is a parameter as mentioned before. We consider the cases with one sink ($p = 1$) and two sinks ($p = 2$). We solve each test problem five times with different seeds and report the best solution.

The neighborhood size of the local search algorithms, namely VNS and TS, is an important parameter. It dramatically affects the solution quality. At every iteration of VNS or TS, we make a local search and here we solve PECLRP$'$ using either Lagrangean Heuristic or Nested Dual Heuristic. So the size of the neighborhood is dependent on how fast we can solve LH or NDH. Since NDH can be solved much faster, we set the neighborhood size parameter $r = 100$ for NDH and $r = 50$ for LH. The 50 neighbors are obtained by 20 *SWAP*, 20 *ADD* and 10 *DROP* moves for LH. These values are doubled for NDH since it is computationally cheapter than LH.

6.3.2. Experimental Results for Variable Neighborhood Search

The results for the nested solution approach when VNS is used for the search of the sensor locations is analyzed in this secion. The objective values and relative percent deviations are displayed in Table 6.15 and Table 6.16 for $p = 1$ and $p = 2$, respectively. The first column includes the number of points in the sensor field for each test problem. The second column shows the amount of energy spent in the network computed by Cplex 11.0 in no more than 10,800 seconds. Notice that optimum integer solutions are obtained within this time limit only for problems with $N \leq 81$. Solutions for $N \geq 100$ are the best feasible solutions obtained by Cplex within the time limit and they are marked with "$*$". Columns three and four are the results of VNS with greedy initial deployment, whereas, columns five and six are the results by random initial deployment. Last four columns show the percent deviations from the optimum or best IP solution. On the average, $21.2\% - 22.1\%$ more energy for $p = 1$ and $9.4\% - 20.8\%$ more energy for $p = 2$ is spent for data routing when we construct the WSN using VNS based solution approach. For both p values, solving the sink location and routing problem with NDH performs better. This is expected since NDH explores a neighborhood, whose size is twice as the size of the neighborhood used in LH. In Table 6.17 the CPU values are compared. As easily observed, VNS perform is much faster than IP. Also when we compare the effects of initial deployment strategies (greedy or random), we see that the overall performance is better with greedy initialization, especially in terms of CPU performance.

6.3.3. Experimental Results for Tabu Search

The results obtained by using TS Heuristic are presented in this section. The objective values and percent deviations from the best IP solution are displayed in Table 6.18 and Table 6.19 for $p = 1$ and $p = 2$, respectively. For $p = 1$, on the average the best solution deviates from the IP solution between $6.6\% - 14.6\%$, which becomes $3.8\% - 15.2\%$ for $p = 2$. As can be easily seen, TS outperforms VNS in terms of solution quality. The price for that is the increase in CPU times. Although TS is faster than IP, it is slower than VNS as observed from Table 6.20. Under TS, NDH performs much

Table 6.15. Results for VNS, p=1

	Objective (nAh)					% deviation			
		greedy_init		random_init		greedy_init		random_init	
N	z_{IP}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}
9	6	6	6	6	6	0.0	0.0	0.0	0.0
16	14	14	14	14	14	0.0	0.0	0.0	0.0
25	20	20	20	20	20	0.0	0.0	0.0	0.0
36	37	37	37	37	37	0.0	0.0	0.0	0.0
49	50	65	59	57	58	30.0	18.0	14.0	16.0
64	76	98	94	88	86	28.9	23.7	15.8	13.2
81	102	139	130	129	128	36.3	27.5	26.5	25.5
100*	144	195	192	179	160	35.4	33.3	24.3	11.1
121*	191	264	249	241	271	38.2	30.4	26.2	41.9
144*	252	321	323	316	357	27.4	28.2	25.4	41.7
169*	315	424	450	474	466	34.6	42.9	50.5	47.9
196*	402	523	518	603	561	30.1	28.9	50.0	39.6
225*	482	618	673	617	643	28.2	39.6	28.0	33.4
400*	1221	1467	1513	1703	1543	20.1	23.9	39.5	26.4
Avg.	236.6	299.4	305.6	320.3	310.7	22.1	21.2	21.4	21.2

better than LH in terms of solution quality. In terms of the CPU times NDH is faster than LH for $p = 2$, but slower for $p = 1$. Similar to the case in VNS, greedy initial deployment strategy results in better objective values and shorter CPU times when compared to the random initial deployment.

Table 6.16. Results for VNS, p=2

	Objective (nAh)					% deviation			
		greedy_init		random_init		greedy_init		random_init	
N	z_{IP}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}
9	3	3	3	3	3	0.0	0.0	0.0	0.0
16	8	8	8	8	8	0.0	0.0	0.0	0.0
25	13	13	13	13	13	0.0	0.0	0.0	0.0
36	23	24	23	23	23	4.3	0.0	0.0	0.0
49	33	43	33	37	33	30.3	0.0	12.1	0.0
64	50	58	50	57	50	16.0	0.0	14.0	0.0
81	67	89	86	84	76	32.8	28.4	25.4	13.4
100*	96	109	119	136	102	13.5	24.0	41.7	6.3
121*	126	170	150	183	169	34.9	19.0	45.2	34.1
144*	170	218	198	224	209	28.2	16.5	31.8	22.9
169*	213	293	245	293	281	37.6	15.0	37.6	31.9
196*	272	348	336	406	352	27.9	23.5	49.3	29.4
225*	336	436	400	455	429	29.8	19.0	35.4	27.7
400*	1120	1031	968	1110	1044	-7.9	-13.6	-0.9	-6.8
Avg.	180.7	203.1	188.0	216.6	199.4	17.7	9.4	20.8	11.4

Table 6.17. Efficiency of VNS

	CPU (sec.) for p=1					CPU (sec.) for p=2				
		greedy_init		random_init			greedy_init		random_init	
n	IP	LH	NDH	LH	NDH	IP	LH	NDH	LH	NDH
9	0.2	0.6	0.1	1.1	0.2	0.1	1.4	0.1	2.8	0.3
16	1.0	1.8	0.4	3.6	0.7	0.8	3.5	0.3	7.0	0.7
25	2.2	16.9	9.2	33.7	18.4	13.2	23.6	8.6	47.2	17.2
36	7.8	27.3	17.9	54.7	35.8	168.8	43.9	18.2	87.9	36.4
49	31.5	51.0	36.7	102.0	73.4	407.8	75.2	27.9	150.4	55.9
64	341.2	111.4	88.1	222.7	176.3	1443.5	164.7	52.8	329.4	105.6
81	344.4	180.7	196.0	361.4	392.0	3694.7	223.8	110.5	447.6	221.0
100*	10800	295.7	276.3	591.4	552.7	10800	425.9	229.8	851.9	459.6
121*	10800	457.1	701.4	914.2	1402.8	10800	577.6	361.2	1155.1	722.4
144*	10800	866.8	783.9	1733.5	1567.7	10800	825.0	556.8	1650.0	1113.6
169*	10800	858.0	2137.9	1716.0	4275.8	10800	883.5	730.0	1767.0	1460.0
196*	10800	1083.6	2505.1	2167.2	5010.2	10800	1453.1	857.9	2906.2	1715.8
225*	10800	1811.6	4389.8	3623.2	8779.6	10800	1737.5	1432.7	3475.0	2865.4
400*	10800	9975.6	10800	10800	10800	10800	7427.8	10372.6	10800	10800
Avg.	5452.0	1124.1	1567.3	1594.6	2363.3	5809.2	990.5	1054.3	1691.2	1398.1

Table 6.18. Results for Tabu Search, p=1

	Objective (nAh)					% deviation			
		greedy_init		random_init		greedy_init		random_init	
N	z_{IP}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}
9	6	6	6	6	6	0.0	0.0	0.0	0.0
16	14	14	14	14	14	0.0	0.0	0.0	0.0
25	20	20	20	20	20	0.0	0.0	0.0	0.0
36	37	37	37	37	37	0.0	0.0	0.0	0.0
49	50	61	57	57	58	22.0	14.0	14.0	16.0
64	76	89	81	88	86	17.1	6.6	15.8	13.2
81	102	111	111	129	128	8.8	8.8	26.5	25.5
100*	144	166	159	179	160	15.3	10.4	24.3	11.1
121*	191	205	207	220	241	7.3	8.4	15.2	26.2
144*	252	292	264	324	283	15.9	4.8	28.6	12.3
169*	315	352	334	417	411	11.7	6.0	32.4	30.5
196*	402	458	464	463	451	13.9	15.4	15.2	12.2
225*	482	572	564	559	569	18.7	17.0	16.0	18.0
400*	1221	1332	1236	1421	1399	9.1	1.2	16.4	14.6
Avg.	236.6	265.4	253.9	281.0	275.9	10.0	6.6	14.6	12.8

Table 6.19. Results for Tabu Search, p=2

	Objective (nAh)					% deviation			
		greedy_init		random_init		greedy_init		random_init	
N	z_{IP}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}	z_{LH}	z_{NDH}
9	3	3	3	3	3	0.0	0.0	0.0	0.0
16	8	8	8	8	8	0.0	0.0	0.0	0.0
25	13	13	13	13	13	0.0	0.0	0.0	0.0
36	23	24	23	23	23	4.3	0.0	0.0	0.0
49	33	38	33	37	33	15.2	0.0	12.1	0.0
64	50	56	50	57	50	12.0	0.0	14.0	0.0
81	67	87	76	84	76	29.9	13.4	25.4	13.4
100*	96	112	105	136	102	16.7	9.4	41.7	6.3
121*	126	152	145	154	140	20.6	15.1	22.2	11.1
144*	170	209	188	209	192	22.9	10.6	22.9	12.9
169*	213	258	225	266	266	21.1	5.6	24.9	24.9
196*	272	332	281	361	343	22.1	3.3	32.7	26.1
225*	336	412	365	407	391	22.6	8.6	21.1	16.4
400*	1120	1028	972	1072	1015	-8.2	-13.2	-4.3	-9.4
Avg.	180.7	195.1	177.6	202.1	189.6	12.8	3.8	15.2	7.3

Table 6.20. Efficieny of Tabu Search

	CPU (sec.) for p=1					CPU (sec.) for p=2				
		greedy_init		random_init			greedy_init		random_init	
N	IP	LH	NDH	LH	NDH	IP	LH	NDH	LH	NDH
9	0.2	1.5	0.3	1.9	0.4	0.1	4.0	0.3	4.8	0.3
16	1.0	7.6	1.5	10.0	1.2	0.8	11.0	1.2	15.3	1.1
25	2.2	54.0	21.0	38.4	15.7	13.2	44.7	11.1	56.7	12.4
36	7.8	91.9	46.5	108.4	40.2	168.8	82.7	25.3	103.8	21.4
49	31.5	119.3	88.9	167.4	80.7	407.8	188.2	52.6	222.8	86.1
64	341.2	182.1	161.1	280.2	221.9	1443.5	339.6	131.0	393.2	137.8
81	344.4	391.0	323.2	534.1	429.7	3694.7	450.1	262.5	608.6	398.8
100*	10800	979.2	874.0	1032.9	825.8	10800	673.4	547.6	885.0	648.1
121*	10800	1330.6	1736.7	1279.4	2056.4	10800	1252.5	741.5	2570.2	1053.2
144*	10800	1783.4	3667.5	2738.0	5009.7	10800	1326.6	1263.6	2411.3	1584.8
169*	10800	3994.9	8556.2	4180.5	10800	10800	2369.7	2463.0	4294.6	3044.3
196*	10800	5520.2	10800	8164.4	10800	10800	3730.0	3653.7	6331.9	4342.9
225*	10800	7756.8	10800	10800	10800	10800	4503.8	4546.0	10068	10800
400*	10800	10800	10800	10800	10800	10800	10800	10800	10800	10800
Avg.	5452.0	2358.0	3419.8	2866.8	4816.3	5809.2	1841.2	1750.0	2769.0	2352.2

7. CONCLUSIONS

Wireless Sensor Networks are one of the new paradigms in the wireless communications and digital electronics world, which provide lots of challenging real life problems for the researchers. In this thesis, we tackled the main optimization problems faced in the design of WSNs.

First, we considered the differentiated point coverage problem. Coverage is one of the most important tasks of a WSN and it is accepted as a measure of quality of service of the sensing function. Assuming that a sensor field represented by a set of finite points with heterogeneous coverage requirements, we developed a general binary integer programming model to determine the minimum cost sensor placement strategy for perfect, imperfect and uncertain detections. The new binary integer programming model is general enough to include many of the previous uniform coverage and/or homogeneous network formulations as special cases. We proposed two approximation algorithms with proven bounds, and two heuristics. The first heuristic is based on a greedy approach and the second one is based on Lagrangean relaxation combined with the subgradient optimization.

The new methods are tested in terms of both their solution quality and computation time on random instances with varying sizes. It is interesting to observe that for all instances irrespective of the sensing type, the Lagrangean heuristic has the highest accuracy.

In most real-life situations, it is difficult to place sensors at predetermined grid points. We can give battlefield surveillance as a sample application of sensor networks where sensors may be dropped from airplanes. Thus the position of sensors may be uncertain and there can be a sensor placed on a point in the sensor field with a certain probability. As a future work, an optimal placement strategy for this case can be investigated. Also, as we mentioned in the literature survey, a huge amount of research is carried out in the line of random deployment of redundant WSNs, where optimal

sensor scheduling procedures are investigated. Our models can be extended to cover these requirements.

Another important issue in WSN design is to determine optimum places to locate sinks and the data routes of sensors to send their data to these sinks. Sink location problem and routing problem can be studied separately, but if investigated jointly better results are obtained in terms of energy consumption. We proposed two main lines of IP models. The first one is the energy-aware sink location and routing problem (ELRP), where the total routing energy is considered in the objective function and the number of sinks to be installed is given as a parameter. The second one is the cost-aware sink location and routing problem (CLRP), where both routing and sink installation costs are minimized in the objective function. Our focus is mainly on the ELRP, where three IP formulations are provided. Single-commodity Flow (SELRP) and Multi-commodity Flow (MELRP) formulations are inefficient. However, the p-median formulation PELRP is very efficient.

For SELRP and MELRP we investigated Lagrangean Relaxation based solution techniques, together with directly solving it as an IP. We observed that both methods are inefficient. We proposed four solution approaches for PELRP. The first one is solving it directly as an IP. Since IP's are intractable for large instances, we devised three other methods for solving large instances heuristically or solving small and medium instances quickly. Two of these methods are LAGRANGEAN_HEURISTIC and NESTED_DUAL_HEURISTIC which can provide solutions for large PELRP instances. The third method, CANONICAL_REDUCTION is a technique to obtain a reformulation of PELRP with a more efficient lower bound.

We tested the new formulations, heuristic solution procedures and lower bounding methods for small, medium, and large sized WSNs and compared their accuracy and efficiency. We found that for small sized problems, NESTED_DUAL_HEURISTIC and CANONICAL_REDUCTION were very accurate and efficient. As the problem size increases, LAGRANGEAN_HEURISTIC performed better and became the best among the heuristics. The solution times required by the IP formulations were prohibitive for

large instances. As a result, we can say that LAGRANGEAN_HEURISTIC is the most suitable method for large real-life applications of WSNs.

All of the models introduced in this work are centralized. We believe they provide insights for the development of distributed algorithms through Lagrangean duality and variable splitting. Although they only consider data routing, the paper by Mandan and Hall [55] for the SLRP is an encouraging example. They use Lagrangean duality and subgradient optimization to develop a partially distributed algorithm; Lagrangean duality, subgradient optimization and add variable splitting to develop a fully distributed algorithm.

Last but not the least, another possible future research direction is the development of a mathematical programming model for joint optimal routing of multiple sinks through a given set of candidate sink locations and optimal data routing to the points on the optimal sink routes in heterogeneous WSNs.

Recall our assumption that sensors have already been deployed and their locations are known. We believe that the development and solution of an integrated model for optimally locating both the sensors and sinks, and determining an optimal sensor-to-sink data routes will be a remarkable contribution to the state-of-the-art. Our final analysis focused on this paradigm. Here we studied the joint optimization of point coverage, sink location and data routing problem (ECLRP).

Four new mixed integer linear programming formulations to represent this problem is proposed. Our first theee formulations (SECLRP, MECLRP and CECLRP) are based on network design models where the data routing variables are defined on arcs. However, in our fourth formulation (PECLRP), we prefer a location-allocation based model where the data routing variables are based on assignments. Because of their complexity, solving these formulations using commercial solvers is inefficient, even for small-sized problems. Hence, a nested solution procedure is proposed. The best sensor locations are sought using two metaheuristics, namely, variable neighborhood search and tabu search. We test two types of initial sensor deployment strategies, which are

random deployment and greedy deployment. Given the sensor locations, ECLRP is reduced to a simpler formulation which is the sink location and data routing problem (SLRP). As we have given in Chapter 4, we obtained the p-median formulation (PELRP) by using a shortest-path algorithm. PELRP can be effeciently and accurately solved using a Lagrangean Heuristic or Nested-Dual Heuristic. Experimental results indicate that, the nested solution approach can generate close-to-optimal solutions in the proposed computation times.

As a future work on the integrated models, mobility can be one of the interesting line of research, where sensor and/or sinks are mobile. Also distributed solution techniques, where the sensors have only local information (neighboring sensors within their few-hop communication ranges) can be investigated.

REFERENCES

1. Yick, J., B. Mukherjee and D. Ghosal, "Wireless sensor network survey", C*omputer Networks: The International Journal of Computer and Telecommunications Networking*, Vol.52, pp. 2292-2330, 2008.

2. Callaway, E. H. Jr., *Wireless Sensor Networks: Architectures and Protocols*, Auerbach Publications, Boca Raton, 2004.

3. Cardei, M. and J. Wu, *Handbook of Sensor Networks, Coverage in Wireless Sensor Networks*, CRC Press, 2004.

4. Younis, M. and K. Akkaya, "Strategies and Techniques For Node Placement In Wireless Sensor Networks: A Survey", *Ad Hoc Networks*, Vol. 6, Is. 4, pp. 621-655, 2008.

5. Akyildiz, I. F., W. Su, Y. Sankarasubramaniam and E. Çayırcı, "A Survey on Sensor Networks", *IEEE Communications Magazine*, Vol. 40, pp. 102–114, August 2002.

6. Altınel, I. K., E. Güney, N. Aras and C. Ersoy, "Effective Coverage in Sensor Networks: Binary Integer Programming Formulations and Heuristics", *Proceedings of the IEEE International Conference on Communications (ICC '06)*, Vol. 9, pp. 4014-4019, 2006.

7. Meguerdichian, S., F. Koushanfar, M. Potkonjak and M. B. Srivastava, "Coverage Problems in Wireless Ad-hoc Sensor Networks", *Proc. of INFOCOM 2001, Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 3, pp. 1380-1387, 2001.

8. Bogdanov, A., E. Maneva, and S. Riesenfeld, "Power-aware base station positioning for sensor networks", *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Conference (INFOCOM 2004)*, pp. 575–585, March

2004.

9. Güney, E., I. K. Altınel, N. Aras and C. Ersoy, "Efficient integer programming formulations for optimum sink location and routing in wireless sensor networks", *Proc. Of the 23rd of the International Symposium on Computer and Information Sciences (ISCIS 2008)*, Istanbul, 2008.

10. Ghosh, A., S. K. Das, "Coverage and Connectivity Issues In Wireless Sensor Networks: A Survey", *Pervasive and Mobile Computing*, Vol. 4, pp. 303-334, 2008.

11. Ab Aziz, N. A., K. Ab Aziz and W. Z. Wan Ismail, "Coverage Strategies for Wireless Sensor", *Networks, Engineering and Techonology*, Vol. 38, pp. 145-150, 2009.

12. Chakrabarty, K., S. S. Iyengar, H. Qi, and E. Cho, "Grid Coverage for Surveillance and Target Location in Distributed Sensor Networks", *IEEE Transactions on Computers*, Vol. 51, pp. 1448-1453, 2002.

13. Meguerdichian S. and M. Potkonjak, "Low Power 0/1 Coverage and Scheduling Techniques in Sensor Networks", *UCLA Technical Reports*, 030001, January 2003.

14. Dhillon, S. S., K. Chakrabarty, and S. S. Iyengar, "Sensor Placement for Grid Coverage under Imprecise Detections", *Proc. International Conference on Information Fusion*, pp. 1581-1587, 2002.

15. Dhillon, S. S. and K. Chakrabarty, "Sensor Placement for Effective Coverage and Surveillance in Distributed Sensor Networks", *Proc. IEEE Wireless Communications and Networking Conference*, pp. 1609-1614, 2003.

16. Zou, Y. and K. Chakrabarty, "Uncertainty-Aware and Coverage-Oriented Deployment for Sensor Networks", *Journal of Distributed and Parallel Computing*, Vol. 64, pp. 788-798, 2004.

17. Zou, Y. and K. Chakrabarty, "Sensor Deployment and Target Localization Based

on Virtual Forces", *Proc. IEEE Infocom Conference*, Vol. 2, pp. 1293-1303, 2003.

18. Jourdan, D. B. and O. L. Weck, "Layout optimization for a wireless sensor network using a multi-objective genetic algorithm", *IEEE 59th Vehicular Technology Conference*, Vol. 5, pp. 2466-2470, 2004.

19. Alba, E. and G. Molina, "Optimal Wireless Sensor Network Layout with Metaheuristics: Solving a Large Scale Instance", *Large-Scale Scientific Computing: 6th International Conference, LSSC 2007*, Sozopol, Bulgaria, 2007.

20. Zhang M., X. Du, and K. Nygard, "Improving Coverage Performance in Sensor Networks by Using Mobile Sensors", *Proceedings of the Military Communications Conference (MILCOM 2005)*, Vol. 5, pp. 3335-3341, 2005.

21. Seo, J. H., Y. H. Kim, H. B. Ryou, S. H. Cha and M. Jo, "Optimal Sensor Deployment for Wireless Surveillance Sensor Networks by a Hybrid Steady-State Genetic Algorithm", *IEICE Transaction Communications*, Vol. E91, pp. 3534-3543, 2008.

22. Wang J. and N. Zhong, "Efficient Point Coverage in Wireless Sensor Networks", *Journal of Combinatorial Optimization*, Vol. 11, pp. 291-304, 2006.

23. Zhang J., T. Yan, and S. H. Son, "Deployment Strategies for Differentiated Detection in Wireless Sensor Networks", *Proceedings of the 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks (SECON '06)*, Vol. 1, pp. 316-325, 2006.

24. Qishi, W., N. S. V. Rao, X. Du, S. S. Iyengar and V. K. Vaishnavi, "On Efficient Deployment of Sensors on Planar Grid", *Computer Communications*, Vol. 30, pp. 2721-2734, 2007.

25. Cavalier, T. M., W. A. Conner, E.D . Castillo and S. I. Brown, "A Heuristic Algorithm For Minimax Sensor Location In The Plane", *European Journal of Operational Research*, Vol. 183, pp. 42-55, 2007.

26. Lazos, L., R. Poovendran and J. A. Ritcey, "On the Deployment of Heterogeneous Sensor Networks for Detection of Mobile Target", *5th International Symposium on Ad Hoc and Wireless - WiOpt*, pp. 1-10, 2007.

27. O'Rourke, J., *Art Gallery Theorems and Algorithms*, Oxford University Press, Oxford, 1987.

28. Gonzalez-Banos, H. H., "A Randomized Art-gallery Algorithm For Sensor Placement", *Symposium on Computational Geometry*, pp. 232-240, 2001.

29. Biagioni, E. and G. Sasaki, "Wireless Sensor Placement For Reliable and Efficient Data Collection", *Proc. Hawaii Int. Conf. System Sciences*, pp. 127-136, 2003.

30. Kar, K. and S. Banerjee, "Node Placement for Connected Coverage in Sensor Networks", *Proc. of Workshop on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt'03)*, France, 2003.

31. Toumpis, S., "Optimal Design and Opeartion of Massively Dense Wireless Networks", *Proc. of Workshop on Interdisc. Sys. Approach in Performance Evaluation and Design of Computer and Communications Sytems*, Italy, 2006.

32. Slijepcevic, S. and M. Potkonjak, "Power Efficient Organization of Wireless Sensor Networks", *IEEE International Conference on Communications*, Helsinki, Finland, Vol. 2, pp. 472-476, 2001.

33. Li, X. Y., P. J. Wan and O. Frieder, "Coverage in Wireless Ad Hoc Sensor Networks", *Transactions on Computers*, Vol. 52, pp. 1-11, 2003.

34. Wang, X., G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks", *Proceedings of the 1st Int. Conference on Embedded Networked Sensor Systems (SenSys'03)*, pp. 28-39. ACM Press, 2003.

35. Zhang, H. and J. C. Hou, "Maintaining Sensing Coverage and Connectivity in

Large Sensor Networks", *Technical Report*, UIUCDCS-R-2003-2351, 2003.

36. Türkoğulları, Y. B., *Private Communications*, 2009.

37. Tian, D. and N. D. Georganas, "Connectivity Maintenance and Coverage Preservation in Wireless Sensor Networks", *Ad Hoc Networks*, Vol. 3, pp. 744-761, 2005.

38. Carle, J. and D. Simplot-Ryl, "Energy-Efficient Area Monitoring for Sensor Networks", *Computer*, Vol. 37, pp. 40-46, 2004.

39. Ishizuka, M. and M. Aida, "Performance Study of Node Placement in Sensor Networks", *Proc. of the 24th Int. Conf. on Distributed Computing Systems Workshops*, Vol. 7, pp. 598-603, 2004.

40. Lin, F. Y. S. and P. L. Chiu, "A Simulated Annealing Algorithm for Energy-Efficient Sensor Network Design", *Proceedings of the Third International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, pp. 183-189, 2005.

41. Onur, E., C. Ersoy, and H. Deliç, "How Many Sensors For An Acceptable Breach Detection Probability?", *Computer Communications*, Vol. 29, pp. 173-182, 2005.

42. Cardei, M., M. T. Thai, Y. Li, and W. Wu, "Energy-efficient Target Coverage in Wireless Sensor Networks", *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communication Societies (INFOCOM)*, Vol. 3, pp. 1976-1984, 2005.

43. Zou, Y. and K. Chakrabarty, "A Distributed Coverage- and Connectivity-Centric Technique for Selecting Active Nodes in Wireless Sensor Networks", *IEEE Transactions on Computers*, Vol. 54, pp. 978-991, 2005.

44. Yang, S., F. Dai and M. Cardei, J. Wu, and F. Patterson, "On Connected Multiple Point Coverage in Wireless Sensor Networks", *International Journal of Wireless Information Networks*, Vol. 13,pp. 289-301, 2006.

45. Cao, M., L. T. Yang, X. Chen and N. Xiong, "Node Placement of Linear Wireless Multimedia Sensor Networks for Maximum Lifetime", *Advances in Grid and Pervasive Computing*, CRC Press, Berlin, pp. 373-383, 2008.

46. Al-Karaki, J. N., and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey", *IEEE Wireless Communications*, Vol. 11, pp. 6-28, 2004.

47. Akkaya, K. and M. Younis, "A Survey on Routing Protocols For Wireless Sensor Networks", *Ad Hoc Networks*, Vol. 3, pp. 325-349, 2005.

48. Younis, M., M. Youssef, and K. Arisha, "Energy-aware Routing in Cluster-based Sensor Networks", *Proceedings of the 10th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS2002)*, pp. 129-136, 2002.

49. Sankar, A. and Z. Liu, "Maximum Lifetime Routing in Wireless and Ad-hoc Networks", *IEEE Infocom Conference*, pp. 1089-1097, 2004.

50. Kim, H., Y. Seok, N. Choi, Y. Choi, and T.Kwon, "Optimal Multi-sink Positioning and Energy-Efficient Routing in Wireless Sensor Networks", *ICOIN 2005*, LCNS 3391, pp 264-275, 2005.

51. Hou, Y. T., Y. Shi, H. D. Sherali, and S. F. Midkiff, "On Energy Provisioning and Relay Node Placement For Wireless Sensor Networks", *IEEE Transactions on Wireless Communication*, Vol. 4, pp. 2579-2590, 2005.

52. Chang, J. H., and L. Tassiulas, "Maximum Lifetime Routing in Wireless Sensor Networks", *IEEE Transactions on Networking*, Vol. 12, pp. 609-619, 2004.

53. Xue, Y., Y. Cui, and K. Nahrstedt, "Maximizing Lifetime For Data Aggregation in Wireless Sensor Networks", *Mobile Networks Applications*, Vol. 10, pp. 853-864, 2005.

54. Hua, C. and T. P. Yum, "Maximum Lifetime Routing and Data Aggregation For

Wireless Sensor Networks", Lect. Notes Comput. Sc., Vol. 3976, pp. 840-855, 2006.

55. Madan, R. and S. Lall, "Distributed Algorithms For Maximum Lifetime Routing in Wireless Sensor Networks", *IEEE Transactions on Wireless Communications*, Vol. 5, pp. 2185-2193, 2006.

56. Ergen, S. C. and P. Varaiya, "Energy Efficient Routing With Delay Guarantee For Sensor Networks", *Wireless Networks*, Vol. 13, pp. 679-690, 2007.

57. Pham, M., D. Kim, T. Kim, and S. Yoo, "Maximize the Coverage Lifetime of Sensor Networks", *Lect. Notes Comput. Sc.*, Vol. 4097, pp. 475-484, 2006.

58. Ciciriello, P., L. Mottola, and G. P. Picco, "Efficient Routing from Multiple Sources to Multiple Sinks in Wireless Sensor Networks", *Lect. Notes Comput. Sc.*, Vol. 4373, pp. 35-40, 2007.

59. Vincze, Z., R. Vida, and A. Vidacs, "Deploying Multiple Sinks in Multi-hop Wireless Sensor Networks", *Proceeding of IEEE International Conference on Pervasive Services (ICPS 2007)*, pp. 55-63, 2007.

60. L. Cooper, "Location-allocation Problems", *Operations Research*, Vol. 11, pp. 37-53, 1963.

61. L. Cooper, "Heuristic Methods for Location-allocation Problems", *SIAM Review*, Vol. 6, pp. 37-42, 1963.

62. Hartigan, J. A. and M. A. Wong, "A k-means Clustering Algorithm", *Applied Statistics*, Vol. 8, pp.100-108, 1979.

63. E.I. Oyman, and C. Ersoy, "Multiple Sink Network Design Problem in Large Scale Wireless Sensor Networks", *Proc. of the International Conference on Communications (ICC 2004)*, pp. 3663-3667, 2004.

64. Gandam S. R., M. Dawande, R. Prakash, and S. Venkatesan, "Energy Efficient

Schemes For Wireless Sensor Networks With Multiple Mobile Base Stations", *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'03)*, pp. 377-381, 2003.

65. Luo, J. and J. P. Hubaux, "Joint Mobility and Routing For Lifetime Elongation in Wireless Sensor Networks", *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communication Societies (INFOCOM 2005)*, Vol. 3, pp. 1735-1744, 2005.

66. Efrat, A., S. Har-Peled, and J. Mitchell, "Approximation Algorithms For Location Problems in Sensor Networks", *Proceedings of the IEEE/Creat-Net BROADNETS–Wireless Networking Symposium*, pp. 767-776, 2005.

67. Shi, Y. and Y. T. Hou, "Approximation Algorithm For Base Station Placement in Wireless Sensor Networks", *Proceedings of the 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '07)*, pp. 512-519, 2007.

68. Shi, Y., Y. T. Hou, and A. Efrat, "Algorithm Design For a Class of Base Station Location Problems in Sensor Networks", *Wireless Networks*, Vol. 15, pp. 21-28, 2009.

69. Wang, Z. M., S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting Sink Mobility For Maximizing Sensor Networks Lifetime", *Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS 2005)*, pp. 1-9, 2005.

70. Basagni, S., A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang, "A New MILP Formulation and Distributed Protocols For Wireless Sensor Networks Lifetime Maximization", *Proceedings of the IEEE International Conference on Communications*, Vol. 8, pp. 3517-3424, 2006.

71. Basagni, S., A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang, "Controlled Sink Mobility For Prolonging Wireless Sensor Networks Lifetime", *Wireless*

Networks, Vol. 14, pp. 831-858, 2008.

72. Basagni, S., A. Carosi, C. Petrioli, and C. A. Philips, "Moving Multiple Sinks Through Wireless Sensor Networks For Lifetime Maximization", *Proceedings of the 5th IEEE International Conference on Mobile Ad Hoc Sensor Systems (MASS 2008)*, pp. 523-526, 2008.

73. Papadimitriou, I. and L. Georgiadis, "Energy-aware Routing to Maximize Lifetime in Wireless Sensor Networks With Mobile Sink", *Journal of Communications Software and Systems*, Vol. 2, pp. 141-151, 2006.

74. Gatzianas, M. and L. Georgiadis, "A Distributed Algorithm For Maximum Lifetime Routing in Sensor Networks With Mobile Sink", *IEEE Transactions on Wireless Communications*, Vol. 7, pp. 984-994, 2008.

75. Garey, M. R. and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, NY, 1979.

76. Cardei, M., J. Wu, M. Lu, and M. O. Pervaiz, "Maximum Network Lifetime in Wireless Sensor Networks with Adjustable Sensing Ranges", *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications 2005, (WiMob'2005)*, Vol. 3, pp. 438-445, 2005.

77. Zhang, H. and J. Hou, "Maintaining Sensing Coverage and Connectivity in Large Sensor Networks", *Wireless Ad Hoc and Sensor Networks: An International Journal*, Vol. 1, pp. 89-123, 2005.

78. Caprara, A. P., Toth, and M. Fischetti, "Algorithms for the Set Covering Problem", *Annals of Operations Research*, Vol. 98, pp. 353-371, 2000.

79. Brooks, R. R. and S. S. Iyengar, *Multi-Sensor Fusion: Fundamentals and Applications With Software*, Prentice Hall PTR, Upper Saddle River, NJ, 1998.

80. Orçun, S., T. Cebe, İ. K. Altınel and Ö. Hortaçsu, "Optimal Planning and Schedul-

ing of Batch Plants with Operational Uncertainties: An Industrial Application to Baker's Yeast Production", *Computers and Chemical Engineering*, Vol. Supp. pp. S183-S186, 1999.

81. Onur, E., C. Ersoy, H. Deliç, and L. Akarun, "Coverage in Sensor Networks When Obstacles Are Present", *Proc. of the IEEE International Conference on Communications (ICC '06)*, Vol. 9, pp.4077-4082, 2006.

82. Lund, C. and M. Yannakakis, "On the Hardness of Approximating Minimization Problems", *Journal of the ACM*, Vol. 41, pp. 960-981, 1994.

83. Karmarkar, N., "A New Polynomial-time Algorithm for Linear Programming", *Combinatorica*, Vol. 4, pp. 373-395, 1984.

84. Bertsimas, D. and R. V. Vohra, "Rounding Algorithms for Covering Problems", *Mathematical Programing*, Vol. 80, pp. 63-89, 1998.

85. Dobson, G., "Worst-case Analysis of Greedy Heuristics for Integer Programming With Non-negative Data", *Math. Oper. Res.*, Vol. 7, pp. 515-531, 1982.

86. Rajagopalan, S. and V. V. Vazirani, "Primal-Dual RNC Approximation Algorithms for Set Cover and Covering Integer Programs", *SIAM J. Comput.*, Vol. 28, pp. 525-540, 1998.

87. Chvátàl, V., "A Greedy Heuristic for the Set Covering Problem", *Mathematics of Operations Research*, Vol. 4, pp. 233-235, 1979.

88. Fisher, M. L., "The Lagrangean Relaxation Method for Solving Integer Programming Problems", Management Science, Vol. 27, pp. 1-18, 1981.

89. Held, M. H., P. Wolfe and H. D. Crowder, "Validation of the Subgradient Optimization", *Mathematical Programming*, Vol. 6, pp. 62-88, 1974.

90. Beasley, J. E., "A Lagrangean Heuristic for the Set Covering Problem", *European*

Journal of Operational Research, Vol. 31, pp. 85-93, 1987.

91. Wong, R. T., "Integer Programming Formulations of the Travelling Salesman Problem", *Proceedings of the IEEE Conference on Circuits and Computers*, pp.149-152, 1980.

92. Melkote, S. and M. S. Daskin, "An Integrated Model of Facility Location and Transportation Network Design", *Transportation Research Part A*, Vol. 35, pp. 515-538, 2001.

93. Floyd, R.W., "Algorithm 97 : Shortest Path", *Communication of the ACM*, Vol. 6, pp. 345, 1962.

94. Mirchandani, P. B. and R. L. Francis, Discrete Location Theory, *Wiley-Interscience Publications*, New York, 2000.

95. Geoffrion, A. M., "Lagrangean Relaxation for Integer Programming", *Mathematical Programming Study*, Vol. 2, pp. 82-114, 1974.

96. Beasley, J. E., "Lagrangean Heuristics for Location Problems", *European Journal of Operation Research*, Vol. 65, pp. 383-399, 1993.

97. Erlenkotter, D., "A Dual-based Procedure for Uncapacitated Facility Location", *Operations Research*, Vol. 26, pp. 992-1009, 1978.

98. Cornuéjols, G., G. L. Nemhauser, and L. A. Wolsey, "A Canonical Representation of Simple Plant Location Problems and Its Applications", *SIAM J. Algebraic and Discrete Methods*, Vol. 1, pp. 262-272, 1980.

99. Simao, H. P. and J. M. Thizy, "A Dual Simplex Algorithm for the Canonical Representation of the Uncapacitated Facility Location Problem", *Operation Research Letters*, Vol. 8, pp 279-286, 1989.

100. Altınel, İ. K., N. Aras, E. Güney and C. Ersoy, "Binary Integer Programming

Formulations and Heuristics for Differentiated Coverage in Heterogeneous Sensor Networks", *Computer Networks*, Vol. 52, pp. 2419-2431, 2008.

101. Bazaraa, M. S., J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*, 3rd edition, Wiley-Intescience, New York, 2005.

102. Hansen, P. and N. Mladenovic, "Variable Neighborhood Search", *Computers & Operations Research*, Vol. 24, pp. 1097-1100, 1997.

103. Glover, F. and M. Laguna, *Tabu Search*, Kluwer Academic Publishers, Dordrecht, 1997.

104. Aras, N. and D. Aksen, "Locating Collection Centers for Distance-and Incentive-Dependent Returns", *Int. Journal of Production Economics*, Vol. 111, pp. 316-333, 2008.

105. CPLEX 11.0, *User's Manual*, ILOG, 2008.

106. Öncan, T., İ. K. Altınel, and G. Laporte, "A Comparative Analysis of Several Asymmetric Traveling Salesman Problem Formulations", *Computers and Operations Research*, Vol. 36, pp. 637-654, 2009.

107. Rosing, K. E., C. S. ReVelle, and H. Rosing-Vogelaar, "The p-median and Its Linear Programming Relaxation: An Approach to Large Problems", *Journal of the Operational Research Society*, Vol. 30, pp. 815-823, 1979.

108. Mainwaring, A., J. Polastre, R. Szewczyk, S. Culler and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring", *WSNA'02*, Atlanta, Georgia, September 2002.