LOCATION-ALLOCATION PROBLEMS WITH MULTI-COMMODITY FLOWS: EXACT AND APPROXIMATE SOLUTION METHODS

by

Mehmet Hakan Akyüz

B.S., Industrial Engineering, İstanbul Technical University, 2002

M.S., Industrial Engineering, Galatasaray University, 2005

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 Industrial Engineering

Boğaziçi University

2011

LOCATION-ALLOCATION PROBLEMS WITH MULTI-COMMODITY FLOWS: EXACT AND APPROXIMATE SOLUTION METHODS

APPROVED BY:

Prof. İ. Kuban Altınel (Thesis Supervisor)	.....
Assoc. Prof. Temel Öncan (Thesis Co-supervisor)	.....
Assoc. Prof. Necati Aras	.....
Assist. Prof. Tınaz Ekim Aşıcı	.....
Assoc. Prof. Orhan Feyzioğlu	.....

DATE OF APPROVAL: 4.11.2011

"Whatever you do or dream you can, begin it!
Boldness has genius, power and magic in it.
Begin it now."
Goethe.

ACKNOWLEDGEMENTS

First and foremost, I owe my deepest gratitudes to my thesis advisors Professor İ. Kuban Altınel and Associate Professor Temel Öncan for their invaluable patience, guidance and encouragement. Definitely, this dissertation would not have been possible without them. Their critics, suggestions and corrections have always motivated me to better myself and to improve the quality of our works. Their broad knowledge, work and research disciplines have been very important to my accomplishment. I have seen them as my role models in my scientific and social life. I feel very lucky to have a chance to benefit from their experiences. It is an honor for me to be their student and I thank them for all their generously bestowed helps. I hope to continue learning from both of my advisors on both the personal and scientific levels throughout my life.

I would like to express my special thanks to Assistant Professor Tınaz Ekim Aşıcı who has been in my thesis evaluation committee. She has always been positive and made encouraging and constructive suggestions since April 2008. A tremendous debt of gratitude is owned to Associate Professor Orhan Feyzioğlu not only for his inspiring suggestions that he made in my thesis evaluation committee but also for his helps and guidance in the Computer Integrated Manufacturing Laboratory at the Galatasaray University. He has always shown an utmost care to create a comfortable and silent working environment in the laboratory during my thesis. I am grateful to the other member of my dissertation committee particularly to Associate Professor Necati Aras for his interest and patience.

This work is dedicated to my father Naci and my mother Zahide. This dissertation would have never been possible without their endless love, sacrifices and patient support. They have always been the mainstay of my life and accomplishments. I am absolutely indebted to them for what I am today. My special gratitude extends to my brother Hadi and my lovely nephews Efe and Ekin. They never give up hoping my future visits over the past few years.

I owe special thanks to Cem Doğu Kocatürk for his devoted friendship which has been and will always be invaluable to me. Special thanks also go to my friend Serkan who has always cheered me up in difficult times. I would like to thank a lot my friends Ahu, Ayberk,

Hande, Jbid, Sunay, Tolga, and my benign cousins Olcay and Tolga for their friendship, trust and encouragements over the past few years.

This research is partially supported by the Turkish Scientific and Technological Research Council (TÜBİTAK) Research Grant No: 107M462, and Galatasaray University Scientific Research Projects Grant No: 07.402.014, 10.402.001 and 10.402.019. The author acknowledges the partial support of National Graduate Scholarship Program for PhD Students awarded by TÜBİTAK.

ABSTRACT

LOCATION-ALLOCATION PROBLEMS WITH MULTI-COMMODITY FLOWS: EXACT AND APPROXIMATE SOLUTION METHODS

A multi-commodity and capacitated extension of the Multi-facility Location-Allocation Problem, namely the Multi-commodity Capacitated Multi-facility Weber Problem (MCMWP) is considered, and exact and approximate solution methods are proposed. The MCMWP is new in the literature and aims to locate new facilities on the plane in order to meet the demand of customers for multiple types of products. The objective is to minimize total transportation costs, which are proportional to the distances measured with ℓ_r-norm for $1 \leq r < \infty$ between the facilities and customers, while satisfying the demand and capacity restrictions. The MCMWP has a non-convex objective function and it is difficult to solve. In the first part of this work, approximate solution methods are suggested. The first one of them is based on the Cooper's alternate location-allocation (ALA) heuristic and both continuous and discrete variants of the ALA heuristics are developed for the MCMWP. The second approximate solution method employs Discrete Approximation (DA) strategies. When the location of facilities are selected from a finite set of candidate sites it is possible to obtain approximate solutions of the MCMWP. The proposed DA strategies enable to obtain not only upper bounds but also lower bounds on the MCMWP. The third approximate solution method employs a Lagrangean Relaxation (LR) scheme. The MCMWP is relaxed such that the LR subproblems are variants of Multi-facility Weber Problems (MWP) with no capacity restrictions. The last approximate solution method produces confidence intervals for the optimal solution of the MCMWP using the Fisher-Tippett's theorem. In the second part of this work, exact solution methods are developed for both the Capacitated MWP (CMWP) and MCMWP. In particular, two different branch-and-bound (BB) algorithms, which partition allocation and location spaces, are suggested to exactly solve the CMWP and MCMWP. Lastly, a beam search heuristic using the location space based BB algorithm is implemented.

ÖZET

ÇOK MALLI YERLEŞİM-DAĞITIM PROBLEMLERİ: KESİN VE YAKLAŞIK ÇÖZÜM YÖNTEMLERİ

Tesis yer seçimi, ya da kısaca yerleşim problemleri fabrika, depo, bakkal, alışveriş merkezi gibi tesisleri eniyi yerlere yerleştirmeyi amaçlar. Bir yandan eniyi yerleri bulmaya çalışırken diğer yandan da müşterilerden elde edilecek kazancı enbüyüklemek veya tesis açmaktan ve müşterilere hizmet etmekten kaynaklanan giderleri enküçüklemek gibi iki temel amacı gerçekleştirmeye çalışır. Dağıtım problemleri genel olarak, tesislerin yerleri yerine ürettikleri malların tüketicilere dağıtılmasıyla ilgilenirler. Amaçları alıcı istemlerini doyuran, tesis sığalarını aşmayan ve dağıtım ağından kaynaklı kısıtları sağlayan enküçük toplam giderli dağıtım planını belirlemektir. Bu ailenin bir bireyi olan taşıma probleminde tesisler ile tüketiciler, tesislerden tüketicilere doğru yönlenen oklarla bağlı yönlü çift kümeli ağlar üzerindedir ve tesislerde tek bir ürün türünün üretildiğini varsayarlar. Problemin çok mallı sürümü ise biraz daha geneldir ve bu varsayımı gevşetir: tesislerin üretimi değişik ürün türlerini içerir ve taşıma ağındaki akış çok mallıdır. Bu durum tek mallı problemin olağan kısıtlarına demetleme kısıtlarını ekleyerek matematiksel eniyileme gösterimine yansıtılır ve çok mallı taşıma problemi gösterimi elde edilir. Çalışmada asıl olarak bu iki problemi bütünleyen Sınırlı sığalı Çok mallı Çok tesisli Weber Problemi (SÇÇWP) ele alınmakta, yaklaşık ve kesin çözüm algoritmaları geliştirilmektedir. Yapılanları iki ana kümede özetlemek olanaklıdır. İlk küme SÇÇWP için yaklaşık çözüm yöntemleri içermektedir. Bu amaçla etkinlik ve doğruluk başarımları yüksek sezgiseller önerilmiştir. Ayrıca, Fisher-Tippet teoreminden yararlanılarak eniyi amaç değeri üzerinde güven aralıkları üretilmiştir. İkinci kümede hem Sınırlı Sığalı ÇWP (SÇWP), hem de SÇÇWP için kesin çözüm yöntemleri önerilmektedir. Özel olarak SÇWP ve SÇÇWP'yi kesin olarak çözen birisi taşıma diğeri tesis yeri değişkenleri uzayını parçalayan iki değişik dal-sınır (DS) dizgi işlemi geliştirilmektedir. Değişik dallanma kuralları denenmekte ve tesis yeri değişkenleri uzayını parçalayan DS dizgi işlemini kullanan bir ışın arama sezgiseli de önerilmektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathbf{a}$	Location vector of a customer or candidate point
A	Weibull location parameter
$\mathscr{b}$	The fundamental direction of a block norm
B	Weibull scale parameter
$\mathscr{B}$	The sets for the branches of the branch-and-price procedure
$\mathcal{B}$	Closed and bounded sets defining a norm
c	The constant denoting transportation cost coefficients
C	Transportation cost function
$\mathcal{C}$	Facility-region combination
$d(\mathbf{x}, \mathbf{a})$	The function measuring distance between vectors $\mathbf{x}$ and $\mathbf{a}$
d_α	A constant used in reformulation-linearization technique based formulation
D	Weibull shape parameter
g	The index denoting candidate points
G	The number of candidate points
$\mathscr{g}$	The index denoting fundamental directions for a block norm
$\mathcal{G}$	The number of fundamental directions of a block norm
i	The index denoting facility i
I	The number facilities to be opened
$\mathcal{I}$	The set of facilities
j	The index denoting customer j
J	The number of customers
$\mathcal{J}$	The set of customers
$\mathcal{J}_p$	The set of customer-commodity pairs in column p
k	The index denoting commodity k
K	The number of commodities
l	Lower bounds on the flows between facilities and customers
m	Index denoting the samples
M	The number of samples
n	The index denoting dimension n
N	The number of dimensions of the real vector space

o	The binary variable denoting a column or subset
p	The index denoting a column p
P	The number of columns
$\mathscr{P}$	The number of facets of the convex hull of customers
Pr	Probability of an event
r	ℓ_r-norm parameter
$\mathbb{R}$	The real vector space
R	Region or rectangle in location space based branch-and-bound algorithms
s	Facility capacity quantity
SL	First maximum slack value associated with the capacity constraints
SU	Second maximum slack value associated with the capacity constraints
t	Node t of the branch-and-bound tree
$\mathcal{T}$	Branch-and-bound tree
q	Customer demand quantity
QL	First maximum slack value associated with the demand constraints
QU	Second maximum slack value associated with the demand constraints
u	Upper bound on the flows between facilities and customers
U	Upper bound on a flow
UL	Maximum slack value associated with the bundle constraints
v	The binary variables denoting if a facility is located on a candidate point
w	Assignments or allocation quantities between facilities and customers
W	Beam width
$\mathcal{W}$	The set of arcs representing flows between facilities and customers
$\mathbf{x}$	Location vector of a facility
y	The variables denoting the flows between facilities located at some candidate points and customers
z	Objective value with given allocations
Z	Objective value

α	Decision variable in the reformulation-linearization technique based formulation measuring the distance between customers and facilities
$\boldsymbol{\beta}$	Lagrangean multiplier associated with the demand constraints
γ	Decision variable in the reformulation-linearization technique based formulation
Γ	Circle radius adjustment factor for the dynamic region rejection heuristic MDRR
$\boldsymbol{\delta}$	Lagrangean multipliers associated with the second discrete approximation MDAP2
Δ	Parameter used in region rejection heuristics MRR and MDRR
ζ	Level of confidence
η	Parameter used in region rejection heuristics MRR and MDRR
θ	Decision variable in the reformulation-linearization technique based formulation
ϑ	Rotation angle around origin
$\boldsymbol{\mu}$	Lagrangean multiplier associated with the bundle constraints
ϖ	Denotes a nonnegative number
ρ	Constant positive numbers used in the limiting probability distribution approach
σ	A positive number
ς	Quantiles used in Weibull parameter estimation
τ	The constant denoting circle radius
Φ	A real number
$\boldsymbol{\varphi}$	Lagrangean multiplier associated with the facility capacity constraints
χ^2	Goodness-of-fit measure
Ψ	Parameter used in the beam search's evaluation function
ϱ	The index denoting facets of the convex hull of customers
$\digamma$	Constants used in the limiting probability distribution approach

LIST OF ACCRONYMS/ABBREVIATIONS

ABB	Allocation space based Branch-and-Bound
A-D	Anderson-Darling
ALA	Alternate Location-Allocation
BB	Branch-and-Bound
BFS	Best-First Search
BP	Branch-and-Price
BrFS	Breadth-First Search
BrS	Branching variable selection Strategy
BSSS	Big Square Small Square
BS	Beam Search
BTST	Big Triangle Small Triangle
CALA	Capacitated Alternate Location-Allocation
CG	Column Generation
CI	Confidence Interval
CKP	Continuous Knapsack Problem
C-MCALA	Continuous Multi-commodity Capacitated Alternate Location-Allocation heuristic
CMLAP	Capacitated Multi-facility Location-Allocation Problem
CMWP	Capacitated Multi-facility Weber Problem
CL-MDA1	Discrete Approximation heuristic using customer locations and the first approximating formulation MDAP1
CL-MDA2	Discrete Approximation heuristic using customer locations and the second approximating formulation MDAP2
CL-RMDA1	Discrete Approximation heuristic using customer locations and the Lagrangean relaxation of the first approximating formulation MDAP1
CL-RMDA2	Discrete approximation heuristic using customer locations and the Lagrangean relaxation of the second approximating formulation MDAP2
C-MDRR	Continuous Multi-commodity Dynamic Region Rejection heuristic
C-MRR	Continuous Multi-commodity Region Rejection heuristic

COP	Combinatorial Optimization Problem
DA	Discrete Approximation
DFS	Depth-First Search
D.C.	Difference of Convex
DLAP	Discrete Location-Allocation Problem
D-MCALA	Discrete Multi-commodity Capacitated Alternate Location-Allocation heuristic
D-MDRR	Discrete Multi-commodity Dynamic Region Rejection heuristic
D-MRR	Discrete Multi-commodity Region Rejection heuristic
ECMWP	Capacitated Multi-facility Weber Problem with Euclidean distance
EVT	Extreme Value Theory
FBS	Filtered Beam Search
GA	Genetic Algorithm
GCP	Gradual Covering Problem
GFA	Goodness-of-Fit Approach
GMNFP	Generalized Multi-commodity Network Flow Problem
GRASP	Greedy Randomized Adaptive Search Procedure
K-S	Kolmogorov-Smirnov
LAP	Location-Allocation Problem
LADP	Location with Acceleration-deceleration Distance Problems
LBB	Location space based Branch-and-Bound
LBBCE	Location space based Branch-and-Bound with Complete Enumeration
LCMWP	Capacitated Multi-facility Weber Problem with ℓ_r distance
LLA	Los and Lardinois' Approach
LP	Linear Programming
LPDA	Limiting Probability Distribution Approach
LPR	Linear Programming Relaxation
LR	Lagrangean Relaxation
ℓ_r-MDA1	Discrete Approximation heuristic using ℓ_r distances and the first approximating formulation MDAP1
ℓ_r-MDA2	Discrete Approximation heuristic using ℓ_r distances and the second approximating formulation MDAP2

ℓ_r-RMDA1	Discrete Approximation heuristic using ℓ_r distances and the Lagrangean relaxation of the first approximating formulation MDAP1
ℓ_r-RMDA2	Discrete Approximation heuristic using ℓ_r distances and the Lagrangean relaxation of the second approximating formulation MDAP2
LSE	Least Square Estimators
MABB	Multi-commodity Allocation space based Branch-and-Bound
MBP	Minimum-Bias Percentile
MBrS	Multi-commodity Branching variable selection Strategy
MCALA	Multi-commodity Capacitated Alternate Location-Allocation
MCMWP	Multi-commodity Capacitated Multi-facility Weber Problem
MCMWP1	First Multi-commodity Capacitated Multi-facility Weber Problem formulation
MCMWP2	Second Multi-commodity Capacitated Multi-facility Weber Problem formulation
MDA	Multi-commodity Discrete Approximation
MDAP	Multi-commodity Discrete Approximation Problem
MDAP1	First Multi-commodity Discrete Approximation Problem formulation
MDAP2	Second Multi-commodity Discrete Approximation Problem formulation
MDAP3	Third Multi-commodity Discrete Approximation Problem formulation for allocation space based branch-and-bound subproblems
MDRR	Multi-commodity Region Rejection with dynamic circle radius
MILP	Mixed Integer Linear Programming
MLAP	Multi-facility Location-Allocation Problem
MLBB	Multi-commodity Location space based Branch-and-Bound
MLBBCE	Multi-commodity Location space based Branch-and-Bound with Complete Enumeration
MLE	Maximum Likelihood Estimator
MRA	McRoberts' Approach
MRR	Multi-commodity Region Rejection with fixed circle radius
MS	Modified Subgradient

MTP	Multi-commodity Transportation Problem
MWP	Multi-facility Weber Problem
OFP	Obnoxious Facility Problem
PCMWP	Capacitated Multi-facility Weber Problem with probabilistic customer locations
PMCMWP	Multi-commodity Capacitated Multi-facility Weber Problem with probabilistic customer locations
PS	Pricing Subproblem
PSL	Partial Solution List
RBS	Recovering Beam Search
RCMWP	Capacitated Multi-facility Weber Problem with Rectilinear distance
RLT	Reformulation-Linearization Technique
RMDAP	Lagrangean Relaxation of the Multi-commodity Discrete Approximation Problem
RMDAP1	Lagrangean Relaxation of the first approximating formulation MDAP1
RMDAP2	Lagrangean Relaxation of the second approximating formulation MDAP2
RR	Region Rejection
SA	Simulated Annealing
SABB	Single-commodity Allocation space based Branch-and-Bound
SC	Set Covering
SCLP	Linear Programming relaxation of Set Covering
SECMWP	Capacitated Multi-facility Weber Problem with Squared Euclidean distance
SLBB	Single-commodity Location space based Branch-and-Bound
SLBBCE	Single-commodity Location space based Branch-and-Bound with Complete Enumeration
SPE	Simple Point Estimator
TA	Threshold Accepting
TP	Transportation Problem
TS	Tabu Search
TSP	Traveling Salesman Problem
UDAH	Uncapacitated Discrete Approximation Heuristic

UDAP	Uncapacitated Discrete Approximation Problem
UFL	Uncapacitated Facility Location
VNS	Variable Neighborhood Search
WAR	Weber problem with Attraction and Repulsion
WP	Weber Problem
WPLD	Weber Problem with Limited Distances

1. INTRODUCTION

Location-allocation problems (LAPs) address locating new facilities (service or production centers i.e., warehouses, supermarkets, distribution centers, factories, etc.) to serve existing facilities (customers) in order to meet their demand for services or products. The objective is to minimize total transportation costs which are proportional to the distances between new and existing facilities while satisfying the demand requirements. The foundations of the LAP dates back to early 17^{th} century. (Kuhn, 1967) attributes the following problem to Pierre de Fermat (1601-1665) "*let he who does not approve of my method attempt the solution of the following problem: given three points in the plane, find a fourth point such that the sum of its distances to the three given points is a minimum*". Although there is no agreement in the literature that who has first proposed the Fermat's problem, its solution is on the intersection of three circles circumscribing three equilateral triangles which are drawn on the sides of the triangle constructed by those "*three given points*". The solution to Fermat's problem is often called as "*Toricelli point*" crediting to Evangelista Toricelli (1609-1647). Toricelli point makes a 120^o angle with the outer points of the equilateral triangles constructed to find it. The solution of the problem becomes one of the "three given points" (the point at the largest angle corner of the triangle) when the triangle formed by "three given points" has an angle greater than 120^o. In 1909, Alfred Weber has described figures for the theory of locations of industries and considered the Fermat's problem with unequal weights on the three points. Actually, Weber has made a distinction between the consumption (or demand) centers and raw material (supply) centers which feed the production facility to be located. However, the consumption and supply centers can be interpreted as existing facilities. Weber has also addressed the general case for more than three market locations. The name *Weber Problem* (WP) and many others are then used to describe the *single facility* location problems in which a new facility is located so as to minimize the sum of transportation costs of serving several existing facilities (customers). More discussions on the history of Fermat's problem, as well as the WP and its synonyms used in different domains of scientific areas can be found in (Wesolowsky, 1993) and (Drezner *et al.*, 2002). (Weiszfeld, 1937)'s iterative method is the first solution procedure for the WP when $J > 3$ with unequal weights where J is the number of customers. Several independent discoveries of the Weiszfeld's algorithm are made two decades later by (Miehle, 1958), (Kuhn and Kuenne, 1962) and (Cooper, 1963).

When there are more than one facility to be opened, the *multi-facility* location problem arises. In this case, there are two decisions to be made: finding the locations of facilities and deciding on the amount of shipment (allocation quantities) from a facility to a customer. Multi-facility LAP (MLAP) is first introduced by (Cooper, 1963, 1964) and its objective function is shown to be neither convex nor concave. For the MLAP, facilities are assumed to have an unlimited capacity and the customers are served by the supplier with the smallest transportation cost. However, when the facilities have capacity restrictions a customer can be supplied by more than one facility (Cooper, 1972) which results in Capacitated MLAP (CMLAP). CMLAP is often called as the Capacitated Multi-facility Weber Problem (CMWP). Actually, WP is the single facility location problem when the distances between facilities and customers are measured using Euclidean distances on the plane. Without loss of generality, in the sequel we refer to the location problem in which the sum of the transportation costs is minimized as the WP regardless of the distance measure used. Generally speaking, we distinguish the problems according to the particular distance measure used. Most of the distance measures satisfy the following norm properties:

$$\text{(i)} \quad d(\mathbf{x},\mathbf{0}) \geq 0, \forall \mathbf{x} \in \mathbb{R}^N \qquad \text{(positivity)} \qquad (1.1)$$

$$\text{(ii)} \quad d(\mathbf{x},\mathbf{0}) = 0 \Leftrightarrow \mathbf{x} = \mathbf{0} \qquad \text{(definiteness)} \qquad (1.2)$$

$$\text{(iii)} \quad d(c\mathbf{x},\mathbf{0}) = cd(\mathbf{x},\mathbf{0}) \forall \mathbf{x} \in \mathbb{R}^N, c \geq 0 \qquad \text{(homogeneity)} \qquad (1.3)$$

$$\text{(iv)} \quad d(\mathbf{x}+\mathbf{a},\mathbf{0}) \leq d(\mathbf{x},\mathbf{0}) + d(\mathbf{a},\mathbf{0}), \forall \mathbf{x},\mathbf{a} \in \mathbb{R}^N \qquad \text{(triangle inequality)} \qquad (1.4)$$

$$\text{(v)} \quad d(\mathbf{x},\mathbf{0}) = d(-\mathbf{x},\mathbf{0}), \forall \mathbf{x} \in \mathbb{R}^N \qquad \text{(symmetry)} \qquad (1.5)$$

where $\mathbf{0}$ is N dimensional zero vector and $d(.,.)$ is the distance function measuring the distance between two given points (i.e., $\mathbf{x}$ and $\mathbf{a}$). Symmetry property stated by Equation 1.5 need not be satisfied by all distance measures. However, its lack harms the norm property and the distance measures are generally taken as (symmetric) norms in location problems. For instance, ℓ_r-norm in the plane, is defined as $d(\mathbf{x},\mathbf{0}) = \left(\sum_{n=1}^{2} |x_n|^r\right)^{1/r}$, and its special cases rectilinear (for $r = 1$) and Euclidean (for $r = 2$) distances, squared Euclidean (i.e., $(\ell_2)^2$) distances are mostly used for location problems. Several methods and functions to model the distances have been analyzed in the literature. (Brimberg and Love, 1995), (Alpaydın *et al.*, 1996) and (Brimberg *et al.*, 2007) are examples which also include concise surveys on distance measures.

When the facilities are to be located on a finite set of given points, *discrete* version of LAPs, namely DLAPs arise. On the other hand, the LAPs can be named as *continuous* LAPs when the location space is continuous. The location space is generally taken as the two-dimensional space (i.e., Euclidean space). However, there are several studies considering the case of more than two-dimensions (Plastria, 1995). DLAPs can be modeled using Mixed Integer Linear Programming (MILP) formulations and facilities can have fixed opening costs for some particular cases. Nevertheless, continuous LAPs consist of nonlinear terms (i.e., objective function and/or constraints) which result in non-convex optimization problems. Therefore, it is frequently assumed that the number of facilities to be opened is known and fixed costs are embedded in transportation costs for continuous LAPs.

In this thesis, we consider a multi-commodity (multiple types of products or services) and capacitated extension of the continuous LAP with multiple facilities. Namely, we deal with the "*Multi-commodity Capacitated Multi-facility Weber Problem*" (MCMWP). Given the locations of J customers and their demands, the MCMWP is concerned with locating I capacitated facilities in the Euclidean space in order to satisfy the demands of J customers for K types of commodities so that the total transportation cost is minimized. Customer locations, demands and capacities for each commodity, and bundle restrictions are known a priori and thus, deterministic. The transportation costs, which are proportional to the distance between customers and facilities, depend on the commodity type. We consider ℓ_r-norm with $1 \leq r < \infty$ or its weighted form obtained by multiplying with a nonnegative coefficient as the distance measure used within this work.

When there is only single commodity type ($K = 1$) and there is no bundle restrictions, limiting the total flow quantity between each facility and customer, the MCMWP reduces to the CMWP which is shown to be NP-hard even if the customer locations constitute a straight line (Sherali and Nordai, 1988). The MCMWP is more realistic than the CMWP since the number of product types produced in a facility is generally more than one. There exist a few studies considering MLAPs with multiple commodities (Pirkul and Jayaraman, 1998; Gendron *et al.*, 2003) where all of these are examples of DLAPs. For all we know, there does not exist a continuous MLAP which consider multiple commodities. One of the motivations of this work is to fill this gap in the literature by introducing heuristic algorithms and exact solution procedures for the MCMWP. The outline of the thesis can be summarized as follows. In the next chapter, we introduce two equivalent mathematical formulations for

the MCMWP. The first formulation is the original MCMWP formulation. The second one is equivalent to the first formulation but it includes redundant constraints which are useful for relaxation techniques and decomposition. Chapter 3 presents a critical literature survey on WP, multi-facility WP (MWP) and CMWP. Moreover, several extensions of the LAPs and multi-commodity DLAPs are mentioned. This chapter also describes the motivation of our research direction in the light of the existing techniques developed for the MWP and CMWP.

Chapter 4 is dedicated to Alternate Location-Allocation (ALA) heuristics implemented for the MCMWP. The MCMWP reduces to a multi-commodity transportation problem (MTP) when the location of the facilities are known. Furthermore, when a feasible allocation plan is given, the MCMWP decomposes into pure single facility location problems, namely the WPs, which can be easily solved. ALA heuristics, first introduced by (Cooper, 1964), sequentially solve the location and allocation problems by fixing locations or allocations each time. We present both continuous and discrete variants of the ALA heuristics for the MCMWP.

In Chapter 5, we present some discretization strategies resulting in heuristics which are also capable to produce lower bounds for the MCMWP. Discretization strategies reduce the continuous location space, which consists of the convex hull of customer locations, into a finite set of candidate facility locations. Discretization results in a DLAP which can be modeled as a MILP problem. We implement two Discrete Approximation (DA) formulations to obtain approximate solutions of the MCMWP. These DA formulations are also combined with a lower bounding norm function to produce lower bounds. Using the DA formulations we also propose several DA heuristics. The efficiency of the DA heuristics are improved with Lagrangean Relaxation (LR) schemes.

In Chapter 6, we give a LR scheme whose subproblems are variants of the MWP and propose a Modified Subgradient (MS) algorithm. The Lagrangean subproblems are formulated as equivalent Set Covering (SC) problems and a Column Generation (CG) procedure is applied to solve them. Several strategies are combined with the MS algorithm to improve the efficiency of the suggested LR scheme.

Point and interval estimates on the objective value of the MCMWP are presented in Chapter 7. The calculation of these estimates are based on using the famous (Fisher and Tippett, 1928)'s theorem for the distribution of the extreme (i.e., minimum and maximum) values of samples. Some of the heuristics implemented in Chapter 4 and Chapter 5 are devised. A brief literature survey on the estimation methods suggested is also presented.

We implement two branch-and-bound (BB) exact solution algorithms and a Beam Search (BS) heuristic for the CMWP and MCMWP. One of the BB algorithm considers partitioning of the allocation space while the other one considers partitioning of the location space. In Chapter 8, we present allocation space based BB (ABB) algorithms for both CMWP and MCMWP. Several lower bounding procedures are embedded within the ABB algorithm and various branching variable selection strategies are tested. In Chapter 9, a brand new BB algorithm, namely location space based BB (LBB) algorithm, is proposed for both CMWP and MCMWP. We devise two lower bounding schemes for LBB algorithm together with a specially tailored branching strategy. The LBB algorithm is later used to develop a BS heuristic for both problems. Our test bed and computational results are given in Chapter 10. Finally, we conclude with Chapter 11 where future research directions are discussed.

2. PROBLEM FORMULATION

Given fixed customer demands and their locations, the MCMWP consists of locating I capacitated facilities in the Euclidean plane to satisfy the demand of J customers for K types of commodities with minimum transportation cost under capacity restrictions on the total quantity of commodities shipped from facilities to customers. The objective is to determine the locations of the facilities and to determine how to allocate the capacity of facilities to customers while minimizing the sum of total transportation costs. This chapter is devoted to present two equivalent mathematical formulations for the MCMWP. The first formulation is directly based on the given definition of the MCMWP. The second formulation uses a different point of view than the first one and replicates each facility for each commodity. These two formulations are presented in the following.

2.1. First Formulation

Let I, J and K denote respectively the number of facilities, the number of customers and the number of commodities that each facility can ship. $\mathbf{a}_j = (a_{j1}\ a_{j2})^T$ and q_{jk} represent the coordinates of customer j and its demand for commodity k. The capacity of facility i for commodity k is given by s_{ik}. $\mathbf{x}_i = (x_{i1}\ x_{i2})^T$ and w_{ijk} are the unknown coordinates of facility i and the unknown amount of commodity k shipped from facility i to customer j with the unit shipment cost c_{ijk} per unit distance, respectively. The distance between facility i and customer j is measured by ℓ_r-norm with $1 \leq r < \infty$ given as follows.

$$d(\mathbf{x}_i, \mathbf{a}_j) = (|x_{i1} - a_{j1}|^r + |x_{i2} - a_{j2}|^r)^{1/r} \tag{2.1}$$

where $d(\mathbf{x}_i, \mathbf{a}_j)$ denotes the function measuring the distance. Then, the first formulation of the MCMWP can be stated as follows.

MCMWP1:

$$\min \quad Z = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} w_{ijk} c_{ijk} d(\mathbf{x}_i, \mathbf{a}_j) \tag{2.2}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} w_{ijk} = s_{ik} \qquad i = 1, \ldots, I; k = 1, \ldots, K, \tag{2.3}$$

$$\sum_{i=1}^{I} w_{ijk} = q_{jk} \qquad j = 1, \ldots, J; k = 1, \ldots, K, \tag{2.4}$$

$$\sum_{k=1}^{K} w_{ijk} \leq u_{ij} \qquad i = 1, \ldots, I; j = 1, \ldots, J, \tag{2.5}$$

$$w_{ijk} \geq 0 \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K. \tag{2.6}$$

The objective function given by Equation 2.2 is the sum of total transportation costs. Constraints given by Equation 2.3 make sure that the total amount of commodity k produced by facility i should be exactly shipped. Constraints given by Equation 2.4 state that the total amount of commodity k required by customer j should be exactly satisfied. We assume that, according to regulations, total amount of allocations on a road connecting facility i with customer j should not be larger than the given upper bound u_{ij}. These regulations may be legislative stipulations such as narrow straits which have limitations on the size of ships that can pass, the restrictions on the total amount of hazardous materials that can be shipped, etc. For instance, there may be capacity and cost restrictions on the transportation fleet of companies. What is more, business contracts between suppliers and customers may also impose some upper bounds. These situations can be formulated with the *flow capacity* constraints or the *bundle* constraints given by Equation 2.5. In this formulation we assume that the problem is balanced, i.e., $\sum_{j=1}^{J} q_{jk} = \sum_{i=1}^{I} s_{ik}$ holds for $k = 1, \ldots, K$, and the equalities in Equation 2.3 and 2.4 can be replaced with "$\leq$" and "$\geq$", respectively without changing the optimal solution. In anyway, in case the problem is not balanced, say the total capacity of facilities is larger than or equal to the demand of customers then the problem can be transformed into a balanced form by using dummy customers with zero transportation costs. Moreover, note also that, whenever bundle constraints given by Equation 2.5 are relaxed the remaining problem can not be decomposed into K subproblems because the facility location variables $\mathbf{x}_i$ are common for the subproblems.

It can be observed that, in the MCMWP formulation, the transportation costs are directly proportional to the quantity of commodities sent from facilities to customers and the distance between facilities and customers. The MCMWP has always an optimum solution which occurs at one of the extreme points of the polyhedron defined by multi-commodity transportation constraints given by Equation 2.3 – 2.6. This is guaranteed as long as the transportation cost is a function of only location variables without additional conditions on its structure such as convexity of the distance function $d(\mathbf{x}_i, \mathbf{a}_j)$.

2.2. Second Formulation

In developing the second formulation a different strategy has been followed. The first formulation assumes that each facility i produce all its K commodities at the same location. On the other hand, it is possible that facility i produces each commodity at different production centers somewhere in the plane. The second formulation initially relaxes this restriction and then enforces all K production centers of a facility i to be opened at the same location. Hence, each production center producing commodity k for facility i has its own location variable but they are restricted to be equal for each facility i. Let $\mathbf{x}_i^k = (x_{i1}^k \; x_{i2}^k)^T$ denote the unknown location of the production center of a facility i producing commodity k. Then the second formulation as follows.

MCMWP2:

$$\min \quad Z = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} w_{ijk} c_{ijk} d(\mathbf{x}_i^k, \mathbf{a}_j) \tag{2.7}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} w_{ijk} = s_{ik} \qquad i = 1, \ldots, I; k = 1, \ldots, K, \tag{2.8}$$

$$\sum_{i=1}^{I} w_{ijk} = q_{jk} \qquad j = 1, \ldots, J; k = 1, \ldots, K, \tag{2.9}$$

$$\sum_{k=1}^{K} w_{ijk} \leq u_{ij} \qquad i = 1, \ldots, I; j = 1, \ldots, J, \tag{2.10}$$

$$\begin{cases} \mathbf{x}_i^1 - \mathbf{x}_i^K = 0 & i = 1, \ldots, I \\ \mathbf{x}_i^k - \mathbf{x}_i^{k-1} = 0 & i = 1, \ldots, I; k = 1, \ldots, K \end{cases} \tag{2.11}$$

$$w_{ijk} \geq 0 \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K. \tag{2.12}$$

Constraints given by Equation 2.11 guarantee that each production center k of facility i is opened at the same location and the rest of the MCMWP2 formulation is the same as MCMWP1. In other words, each facility has K copies that must be located at the same point. Actually, constraints given by Equation 2.11 are redundant for MCMWP1 and add additional location variables to MCMWP2. In particular, we can directly discard these constraints when $K = 1$. However, their relaxation enables the decomposition of the MCMWP2 into smaller subproblems for each commodity k, which is not the case for the MCMWP1. An alternative representation of the MCMWP2 for the discrete LAP case is given in Chapter 5 and we explain details on how the problem is decomposed into subproblems within the DA context.

3. LITERATURE SURVEY

3.1. The Weber Problem

The Weber Problem (WP) tackles with locating single facility in the plane to satisfy demand of J customers with known locations at a minimum total transportation cost. Let $\mathbf{x} = (x_1\ x_2)^T$ denotes the unknown location of a facility to be located, then the WP can be stated as

WP:

$$\min_{\mathbf{x}} Z_{WP} = \sum_{j=1}^{J} c'_j d(\mathbf{x}, \mathbf{a}_j), \tag{3.1}$$

where $c'_j = c_j q_j$ and q_j is the known demand of customer j with a unit transportation cost of c_j. The WP is a pure location problem and one may think that the optimum facility location occurs in the weighted average of the customer locations at the first glance. However, this intuitive argument is valid for the case where the distances are measured with the squared Euclidean norm and can be misleading with other norms. As a reminder, the WP is the problem where the distances are measured with the Euclidean norm. Yet, we use the name WP with any distance function and make a distinction of the WP according to the particular distance measure (i.e., norms) used.

The solution of the WP when $J > 3$ with unequal weights is accomplished by (Weiszfeld, 1937). The Weiszfeld's algorithm is an iterative procedure which employs the derivative of the distance function. In the late 50's and early 60's, three studies by (Miehle, 1958), (Kuhn and Kuenne, 1962) and (Cooper, 1963) independently rediscover the Weiszfeld's algorithm. The iterative method updates the facility location using its previous iteration value (namely, an iterate) within the first derivative of the distance function. However, the derivative is not defined on the customer coordinates and the algorithm may fail when such a case occurs. In particular, (Wesolowsky, 1993) states that the convergence of the algorithm is slow when the facility location falls in the close vicinity of the customer locations at an iteration.

Initial convergence results for the Weiszfeld's algorithm are given in the studies by (Kuhn, 1973) and (Ostresh, 1977). (Brimberg and Love, 1993) show the global convergence of the algorithm under the assumption that an iterate does not coincide with the coordinates of customer locations. Their proof holds for ℓ_r distances with $1 \leq r \leq 2$ while the global convergence is not guaranteed when $r > 2$. These results are all generalizations of the (Kuhn, 1973)'s results for the Euclidean distance (i.e., $r = 2$). (Rosen and Xue, 1993) argue that a hyperboloid approximation procedure is always convergent for the Euclidean distance and their approach eliminates the problem with the derivatives which are not defined on customer locations. (Frenk *et al.*, 1994) extend the results by (Brimberg and Love, 1993). (Frenk *et al.*, 1994) also use a hyperbolic approximation of the WP and work on a perturbed problem to show the convergence of the Weiszfeld's algorithm again for $1 \leq r \leq 2$. (Brimberg *et al.*, 1998) propose a scheme to accelerate the convergence of the Weiszfeld's algorithm with ℓ_r distance by using a step size factor. When $r > 2$ the WP can be solved by classical unconstrained minimization techniques (Bazaraa *et al.*, 1993). (Love *et al.*, 1988) assert the necessary and sufficient optimality conditions of the WP when optimal facility location is on any customer locations. Briefly, the WP can be solved without much difficulty using the convexity of the distance measure. A more detailed survey on the WP can be found in (Wesolowsky, 1993) and (Drezner *et al.*, 2002).

3.2. The Multi-facility Weber Problem

The Multi-facility Weber Problem (MWP) is concerned with locating I uncapacitated facilities in the plane and allocating them to J customers with known locations in order to satisfy their demand. Let $w'_{ij} \in \{0, 1\}$ for $i = 1, \ldots, I; j = 1, \ldots, J$ be the binary decision variables indicating whether customer j is served from facility i or not with an associated unit transportation cost c_{ij}. Then, the MWP formulation which is introduced by (Cooper, 1963, 1964) is as follows.

MWP:

$$\min \quad Z_{MWP} = \sum_{i=1}^{I} \sum_{j=1}^{J} w'_{ij} q_j c_{ij} d(\mathbf{x}_i, \mathbf{a}_j) \tag{3.2}$$

$$\text{s.t.} \quad \sum_{i=1}^{I} w'_{ij} = 1 \qquad j = 1, \ldots, J, \tag{3.3}$$

$$w'_{ij} \in \{0, 1\} \qquad i = 1, \ldots, I; j = 1, \ldots, J, \tag{3.4}$$

where the rest of the notation is the same as in the WP and MCMWP. The objective function of the MWP is shown to be neither convex nor concave by (Cooper, 1972). An equivalent formulation of the MWP can be obtained by substituting binary variables $w'_{ij} = \frac{w_{ij}}{q_j}$ where $w_{ij} \geq 0$ and $q_j > 0$ hold, which makes the constraint set linear. Nevertheless, this transformed formulation does not make the solution of the MWP easier. Thus, we prefer to represent the MWP by Equation 3.2 – 3.4. Each customer is served from a single facility which has the smallest weighted transportation cost in the MWP. Hence, in an optimal solution of the MWP, the demand of each customer is satisfied by exactly one facility. This is also indicated by constraints given by Equation 3.3. Actually, a feasible solution of the MWP can be obtained by partitioning the set of customers into I distinct subsets and finding optimal facility locations for the resulting allocations of customers to facilities. Once all such I distinct subsets of customers are generated, an optimal solution of the MWP can be found by selecting the minimum cost solution among them. However, the number of such subsets is given by the Stirling number of second type as

$$\frac{1}{I!}\sum_{i=0}^{I}\binom{I}{i}(-1)^i(I-i)^J, \tag{3.5}$$

which makes the solution of the MWP difficult even for small numbers of facilities and customers. Indeed, the MWP is shown to be NP-hard by (Megiddo and Supowit, 1984) and heuristics take place an important part of the research to on the MWP. ALA heuristic is still one of the most used algorithm to produce feasible solutions for the MWP (Cooper, 1964). It consists of two phases: allocation and location phases. In the allocation phase, facility locations are assumed given and it remains to allocate each customer to the least weighted transportation cost facility. Note that when all transportation costs between each customer and facility are equal, the assignment of a customer is made to the nearest facility. In the location phase, it is assumed that customers are already allocated to facilities and optimal facility locations are calculated. In this case, the problem decomposes into pure location problems as given in Equation 3.1. Each one of them is a single facility problem and can be solved by one of the methods developed for the WP.

(Cooper, 1963) suggests a heuristic for small problems. This heuristic is later enhanced by (Cooper, 1964) where its randomized version is also proposed. These heuristics consider generating subsets of customers on which the facilities is to be opened. However, the number

of combinations grow rapidly and the optimum facility locations are not guaranteed to be on customer locations, which deteriorates the performance of these heuristics. (Eilon *et al.*, 1971) perform a series of experiments on the ALA heuristic and observed that a multi-start ALA can yield 40.9% deviation on a 50 customers 5 facilities instance. (Brandeau and Chiu, 1993) apply Fisher and Tippett's theorem and made a worst case performance analysis on the ALA heuristic. The authors observe a maximum deviation varying from 8.4% to 73% corresponding to the best and worst performance of the ALA heuristic on the instances having facilities from 3 to 7 and customers from 50 to 75.

(Love and Juel, 1982) propose five heuristics based on ALA heuristic where they define a neighborhood structure. The neighborhood structure starts from a solution and exchanges the assignment of customers to facilities accordingly until no further improvement is possible. Their algorithm moves from one solution to its adjacent solutions by exchanging only single assignment of a customer to a facility. The authors also test exchanging the pair of customers assigned to a facility in order to increase the chance of finding an improved solution. (Chen, 1983) employs an everywhere differentiable approximation of the objective function given by Equation 3.2 in order to apply a quasi-Newton method. The author states that this algorithm produces slightly better results than the ones reported by (Eilon *et al.*, 1971) who apply a multi-start ALA heuristic. (Bongartz *et al.*, 1994) use a projection method after relaxing the binary restrictions on the allocation variables. They employ the second-order information of location and first-order information of allocation variables to determine the descent direction and, propose an iterative algorithm to find a local minima. Their results are better than the ones multi-start ALA heuristic finds, but the method is slower than the multi-start ALA heuristic.

(Hansen *et al.*, 1998) benefit from the idea of selecting facility locations on customer locations introduced previously by (Cooper, 1963, 1964). First of all, the authors offer to solve a p-median problem which uses the customer locations as candidate facility sites. Then, an improvement step namely an ALA heuristic, which is initialized with the locations obtained from the solution of the p-median problem, is applied. The p-median problem based heuristic is shown to have an outstanding performance on their test instances.

(Gamal and Salhi, 2001) put forward two constructive heuristics where they apply Cooper's ALA heuristic whose initial facility locations are constructed by preprocessing procedures rather than a random initialization. The first constructive procedure tries to locate

facilities as far as possible on the customer locations. The second constructive procedure finds a feasible solution for the p-median problem by a heuristic procedure to initialize the facility locations. Note that in the second constructive heuristic the p-median problem is not solved exactly as it is done in (Hansen *et al.*, 1998). These heuristics are efficient in particular on large MWP instances. Later, (Gamal and Salhi, 2003) suggest a cellular heuristic in which the customer plane is divided into grids and a random multi-start ALA heuristic is run. The cells containing facility locations, which come from multi-start ALA heuristic, are merged or divided such that their number equals to the number of facilities. Then, the center of facility locations in each cell is found to initialize an ALA heuristic. The results obtained by cellular heuristic are not better than the ones of the constructive heuristic by (Gamal and Salhi, 2001).

The performance of the p-median problem based heuristic by (Hansen *et al.*, 1998) deteriorates on large instances in spite of its excellent accuracy. (Brimberg *et al.*, 2000) report improvements on the existing metaheuristic approaches which includes Genetic Algorithm (GA), Tabu Search (TS) and Variable Neighborhood Search (VNS) for the MWP where they outperform the p-median problem based heuristic. (Salhi and Gamal, 2003) explain a GA based heuristic which performs better than GA approach but worse than the TS and VNS approaches by (Brimberg *et al.*, 2000). Recently, (Brimberg *et al.*, 2006) associate a decomposition strategy with the VNS approach for large-scale problems, which outperforms the results by (Brimberg *et al.*, 2000) on a set of instances with 1060 customers.

Exact methods for the MWP are considered in several studies. The first exact method is a BB algorithm developed on the allocation space by (Kuenne and Soland, 1972). This BB algorithm is limited to only small instances and halts quickly. The size of instances changes between ($I = 2$, $J = 20$) and ($I = 4$, $J = 15$), which are solved within 0.5% of the optimal value.

(Ostresh, 1975) and (Drezner, 1984) consider the special case where there are only two facilities with unit transportation costs. The optimal partitioning of customers can be done by a straight line when $I = 2$ since each customer is served from the nearest facility and both studies employ this argument. Then, they generate all possible lines separating the customers into two distinct sets where there are at most $J(J-1)/2$ such partitions. (Drezner, 1984) analyzes instances up to 100 customers (i.e., $J = 100$).

(Rosing, 1992) develops a modified SC problem formulation to exactly solve the MWP. This method first generates all distinct nonintersecting convex hulls of customers and then each convex hull, which is interpreted as a column, is added to the SC problem formulation. However, this approach halts quickly on medium instances since the number of convex hulls grows exponentially. (Rosing, 1992) reports optimal partitions for instances of sizes up to ($I = 5$, $J = 30$) and ($I = 6$, $J = 25$).

(Chen *et al.*, 1998) express the MWP as the Difference of Convex (D.C.) functions. The minimization of D.C. functions can be transformed into a concave minimization problem and then this problem can be solved by an outer approximation procedure (Horst and Tuy, 1996). However, this method is not applicable for instances with $I > 3$. (Chen *et al.*, 1998) obtain very efficient solutions on instances with two facilities and 1000 customers. The D.C. programming based approach can not solve problems having larger than 3 facilities and 30 customers in reasonable CPU times.

(Krau, 1997) considers the SC problem formulation and focuses on a CG procedure combined with a Branch-and-Price (BP) algorithm to exactly solve the MWP. This method is based on generating customer subsets rather than their convex hulls as suggested by (Rosing, 1992). Krau's approach increased the size of instances that can be solved up to 50 facilities and 287 customers.

(Righini and Zaniboni, 2007) prefer to replace the solution approach used by (Krau, 1997) for the Pricing Subproblem (PS) with a polynomial time algorithm developed by (Drezner *et al.*, 1991) for the solution of the facility location problem with limited distances. The authors solve instances with hundreds of facilities (i.e., $I = 800$) and thousands of customers (i.e., $J = 2000$). (Righini and Zaniboni, 2007) claim that their approach performs better than the Krau's approach in particular when I/J ratio is relatively high. As a recent and brief survey on the MWP including exact, heuristic and metaheuristic methods, we refer to the work of (Brimberg *et al.*, 2008). The Lagrangean heuristic, which will be presented in Chapter 6, benefits from the strengths of both exact methods by (Krau, 1997) and (Righini and Zaniboni, 2007) on the MWP.

3.3. The Capacitated Multi-facility Weber Problem

The CMWP deals with locating I capacitated facilities in the plane to satisfy the demand of J customers with the minimum total transportation cost of a single commodity. The CMWP is introduced by (Cooper, 1972) and also known as the transportation-location problem. Let the decision variable w_{ij} stands for the unknown allocation quantity sent from facility i to customer j and consider the decision variable $\mathbf{x}_i$ denoting the unknown location of facility i with a given capacity of s_i for $i = 1, \ldots, I$. Then, the mathematical formulation of the CMWP can be stated as

CMWP:

$$\min \quad Z_{CMWP} = \sum_{i=1}^{I}\sum_{j=1}^{J} w_{ij} c_{ij} d(\mathbf{x}_i, \mathbf{a}_j) \tag{3.6}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} w_{ij} = s_i \quad i = 1, \ldots, I, \tag{3.7}$$

$$\sum_{i=1}^{I} w_{ij} = q_j \quad j = 1, \ldots, J, \tag{3.8}$$

$$w_{ij} \geq 0 \qquad i = 1, \ldots, I; j = 1, \ldots, J. \tag{3.9}$$

Notice that the rest of the notation is as defined for the MWP. This formulation assumes that the CMWP is balanced, i.e., the total demand and total supply are equal, namely $\sum_{j=1}^{J} q_j = \sum_{i=1}^{I} s_i$ holds. When the total supply is larger than the total capacity the problem is unbalanced and it can be transformed into a balanced one by adding artificial customers to the formulation. However, the problem is infeasible when the total demand is larger than the total capacity. The objective function given by Equation 3.6 is the total transportation cost. Constraints given by Equation 3.7 ensure that the total amount produced by facility i should be completely shipped. Constraints given by Equation 3.8 enforce that the demand of customer j should be met.

The transportation cost is usually assumed to be proportional to both the amount shipped and the distance between the facilities and customers. In the CMWP customers can be served from more than one facility and these facilities need not be the closest ones

as in the MWP case. Moreover, the allocation quantities w_{ij} can take a value within the interval $[0, q_j]$. For the MWP w'_{ij} values are equal to either 0 or q_j. The CMWP with unit transportation costs (i.e., $c_{ij} = 1$ for all facility and customer pairs) is defined by (Cooper, 1967) as a preliminary formulation given by (Cooper, 1972). The CMWP is shown to be NP-hard by (Sherali and Nordai, 1988) even if the customers are located on a straight line.

(Cooper, 1967) implements a heuristic which produces solutions for the CMWP with unit transportation costs. The heuristic algorithm locates facilities on a selected subset of customer locations and initially treats the CMWP as a MWP to determine the allocation quantities by assigning each customer to its closest facility. Then, the initial solution (feasible for the MWP but probably infeasible for the CMWP) is improved by reallocating surplus or deficit in facility capacities among customers. This may occur since the allocations are made according to the nearest facility regardless of the capacity of the facilities. (Cooper, 1972) generalizes this algorithm and earlier ALA heuristic by replacing the usual allocation phase with the solution of the Transportation Problem (TP) with constraints given by Equation 3.7 – 3.9 of the CMWP. We call this version of the ALA heuristic as the Capacitated ALA (CALA) heuristic in the sequel.

(Cooper, 1976) introduces a neighboring structure defined on the allocation quantity assignments of the TP. The author initializes a feasible allocation plan with the classical northwest corner rule for the TP and find a local optimum with the CALA heuristic. A local optimum is a feasible solution at which neither facility locations nor allocation quantities change when one of them is fixed within the allocation and location phases, respectively. Then, the local optimum solution is subjected to one, two and three-variable exchanges respectively until no further improvement is possible in each case. The variable exchanges starts from scratch and a CALA heuristic is applied to find another local optimum as long as an improvement is obtained. This work produces superior results than the CALA heuristic with multi-start initializations. In addition, this algorithm can be regarded as an early VNS procedure presented by (Hansen and Mladenović, 2001).

(Zainuddin and Salhi, 2007) modify Cooper's heuristic by combining with a perturbation-based procedure. The modified Cooper's heuristic alternates the transportation-location-allo- cation-location phases, respectively. The location and the transportation phases are the same with the CALA heuristic. However, in the additional allocation phase the cus-

tomers are assigned to their nearest facilities as in the MWP. This phase may play a role for shaking the facility locations towards other local optima and may increase the chance to find better solutions. Once the alternating steps are finished, current solution is improved by a perturbation-based procedure. The perturbation procedure enforces some customers to be served from their nearest facility in case they are (fully or partially) served from their second nearest facility. Lastly, a reduced neighborhood is also scanned to obtain improved solutions when possible. The reduced neighborhood exchanges allocation of customers among a smaller subset of selected facilities.

(Aras *et al.*, 2007) present a MILP formulation which approximates the CMWP. The authors consider a finite set of candidate facility locations which transform the CMWP into a DLAP to obtain approximate solutions. Their work is concerned with the ℓ_r-norm CMWP (LCMWP) with $1 < r \leq 2$ and they offer three heuristic procedures. First heuristic employs a LR scheme on the approximating MILP formulation which uses a candidate location set defined on the intersection points of a predefined grid structure by dividing the customer plane into uniform rectangular areas. The second heuristic uses the p-median problem based heuristic idea of (Hansen *et al.*, 1998), which employs the customer locations as candidate sites. The third heuristic is an adaptation of the cellular heuristic by (Gamal and Salhi, 2003) to determine the candidate facility sites. The second and third heuristics solve the MILP with suggested candidate facility locations and apply the CALA heuristic to obtain better feasible solutions. (Aras *et al.*, 2007) observe that the second heuristic, which uses the customer locations, produce results with excellent accuracy. Their second heuristic finds almost all best known solution values on standard test instances.

Rectilinear distance (ℓ_1-norm) CMWP (RCMWP) is addressed in the study by (Aras *et al.*, 2008) which is an earlier version of the DA heuristics designed by (Aras *et al.*, 2007). Actually, the publication years of these two studies may mislead the reader that the work by (Aras *et al.*, 2007) is prior to (Aras *et al.*, 2008). The MILP formulation proposed by (Aras *et al.*, 2008) gives the exact solution of the RCMWP where the optimum facility locations occur on a finite set of points defined by the intersection points of the horizontal and vertical lines drawn on customer locations (Wendell and Hurter, 1973). (Aras *et al.*, 2008) provide also similar DA heuristics to the ones presented in (Aras *et al.*, 2007) but in this case, the suggested DA heuristics are for the more general ℓ_r distance CMWP (i.e., for $1 \leq r \leq \infty$).

In their recent work, (Luis *et al.*, 2009) come up with Region-Rejection (RR) heuristics for the CMWP. Their heuristics gradually initialize facility locations within the smallest rectangle covering customer locations. Once a set of facilities are randomly located, the regions that are close to these already located facilities are forbidden for a new facility to be located. Actually, a new facility can not be located within the circles centered at previously located facilities. Furthermore, the radius of these circles are iteratively adjusted until all facilities are initially located. With this initialization of facilities a CALA heuristic is followed to produce heuristic solutions. RR heuristic is similar to the first constructive heuristic by (Gamal and Salhi, 2001) for the MWP. The computational results indicate that the constructive heuristic is more efficient but more inaccurate than the DA heuristic of (Aras *et al.*, 2007) which uses customer locations as candidate facility sites.

(Aras *et al.*, 2006) put forward several metaheuristic approaches including Simulated Annealing (SA), Threshold Accepting (TA) and GA on the LCMWP. One and two-variable exchange neighborhood structure is devised for the SA and TA methods. The SA method with two-variable exchange outperforms the other metaheuristics in their study. However, these results are also slightly inferior to the ones obtained in the study by (Aras *et al.*, 2007).

Recently, (Luis *et al.*, 2011) examine a Greedy Randomized Adaptive Search Procedure (GRASP), which is a two-phase metaheuristic method using a randomized multi-start local search technique, for both MWP and CMWP. In the first phase a feasible initial solution is constructed from a candidate list of the facility locations defined over a selected subset of customer locations. The second phase consists of the CALA heuristic as the local search procedure. These two phases are repeated several times and the best feasible solution found is reported as the final outcome. This metaheuristic approach does not perform better than the DA heuristic results by (Aras *et al.*, 2007) on standard test instances. A comparative analysis is not reported on additional data sets which are larger than the standard instances in (Luis *et al.*, 2011). However, the standard MWP instance results indicate that GRASP has lower accuracy than the heuristic by (Brimberg *et al.*, 2006).

Since its introduction by (Cooper, 1972), several researchers address exact solution procedures on the CMWP. The first exact algorithm attempt to solve the Euclidean distance CMWP (ECMWP) is performed by (Cooper, 1972) where the author proposes a complete enumeration strategy of generating all extreme points of the transportation polyhedron. An-

other exact solution procedure is proposed in the unpublished dissertation by (Selim, 1979) in which a biconvex programming cutting plane procedure has been devised. Unfortunately, both of these exact procedures can only solve ECMWP instances with few facilities and customers. (Cooper, 1972) solves instances with sizes up to $(I,J) = (2,4)$ while (Selim, 1979) considers instances with sizes up to $(I,J) = (5,5)$.

For different types of distance functions used for the CMWP, different BB algorithms are developed. For the squared-Euclidean distance CMWP (SECMWP), (Sherali and Tunçbilek, 1992) propose a BB algorithm which employs (Sherali and Adams, 1999)'s Reformulation-Linearization Technique (RLT). The authors transform the SECMWP into an equivalent convex maximization problem. Then, they develop three closed form upper bounding functions of allocation variables that require the solution of several Linear Programming (LP) problems. These three upper bounds and the RLT bound is employed together within the BB algorithm. (Sherali and Tunçbilek, 1992) can find solutions of the SECMWP test instances with sizes up to $(I,J) = (4,20)$ – $(6,14)$ within 1% to optimality. Another exact solution algorithm is suggested by (Sherali *et al.*, 1994) for the RCMWP. The authors apply the RLT to an equivalent RCMWP formulation with bilinear objective function. They implement a BB algorithm which works in a partial location space consisting of the customer coordinates in each of the x-axis and the y-axis. (Sherali *et al.*, 1994) can solve test instances with sizes up to $(I,J) = (4,20)$ – $(5,12)$ within 1% to optimality. In their excellent study, (Sherali *et al.*, 2002) propose a BB algorithm for the LCMWP with $1 \leq r < \infty$. This BB algorithm employs the RLT based lower bounds and the authors have noted that they can solve instances with sizes up to $(I,J) = (5,10)$ within 0.1% to optimality.

3.4. Other Variants and Extensions

In a variant of the multi-facility location problem, there are interactions among the new facilities to be opened without making the allocation decisions (Miehle, 1958). This problem is an unconstrained convex minimization problem, which is not everywhere differentiable, and several solution techniques exist for it (see (Rosen and Xue, 1993; Al-Loughani, 1997)). There are also several extensions of the location problems in which the location space is continuous but restricted to lie within some specified regions and/or barriers limiting the passage (Aneja and Parlar, 1994; Fliege and Nickel, 2000). The WP transforms to the Obnoxious Facility Problem (OFP) when the facility to be opened is desired to be as far

as possible by the customers (Hansen *et al.*, 1981). Similar to the OFP, in some cases customer weights may be negative which implies that the facility is unwanted resulting in Weber problem with Attraction and Repulsion (WAR) (Chen *et al.*, 1992; Plastria, 1995). When the objective is to minimize the maximum distance between facilities and customers, the problem is called a minimax problem introduced by (Rawls, 1971) in a totally different context: theory of justice in a social framework (Plastria, 1995). Another variant of the WP may arise when the customer locations are not known exactly i.e., when customer locations are scattered on the Euclidean space according to a probability density function (Altınel *et al.*, 2009). Surveys and more details on WP and LAPs can be found in (Drezner *et al.*, 2002), (Plastria, 1995), (Wesolowsky, 1993), (Love *et al.*, 1988) and (Francis *et al.*, 1992). Moreover, (Hamacher and Nickel, 1998) present a 5-position classification scheme by which all location problems can be described. (Nickel and Puerto, 2005) implement a unifying approach to construct a standard framework for the location theory.

3.5. The Multi-commodity Location-Allocation Problems

For all we know, the multi-commodity LAPs are limited to DLAPs having fixed costs. In DLAPs, facilities can be located on the points which are selected from a predefined candidate location set resulting in MILP formulations. Then, the problem reduces to finding the minimal cost location for each facility within the candidate location set and to determining the allocations. The evolution of the multi-commodity location problems is parallel to their continuous counterparts, MLAPs. Initial models consider facilities without capacity limitations (Neebe and Khumawala, 1981; Karkazis and Boffey, 1981) while later formulations integrate facility capacities, multiple production stages (i.e., multiple types of facilities) and periods into the model (Pirkul and Jayaraman, 1998; Canel *et al.*, 2001).

We can site the early studies by (Neebe and Khumawala, 1981) and (Karkazis and Boffey, 1981) as two examples for the uncapacitated muti-commodity LAPs. Both studies address the multi-commodity extension of the classical Uncapacitated Facility Location (UFL) problem (Wolsey, 1998) with fixed costs. In both studies BB algorithms are developed. (Neebe and Khumawala, 1981) modify the first UFL formulation by (Wolsey, 1998) and use three lower bounding schemes for their BB algorithm. (Karkazis and Boffey, 1981) adapt the alternative (stronger) UFL formulation by (Wolsey, 1998). They suggest a dual-based approach and Lagrangean dual-based approach with hill-climbing.

(Pirkul and Jayaraman, 1998) consider a two-stage distribution network with multiple commodities subject to capacity limitations of both production plants and warehouses which are to be located with fixed opening costs. They use an efficient LR scheme to solve the resulting DLAP. (Canel *et al.*, 2001) take into account not only multi-stage (three level) distribution networks but also multiple production periods. Their sophisticated algorithm consists of two parts including BB and Dynamic Programming algorithms. (Gendron *et al.*, 2003) formulate a multi-commodity capacitated DLAP with balancing requirements which incorporates the conservation of flow constraints in their formulation. The authors implement a parallel heuristic in which an iterative and a neighborhood heuristic are simultaneously applied such that each heuristic exchanges their outcomes during the run of the proposed method.

4. ALTERNATE LOCATION-ALLOCATION HEURISTICS

LAPs aim to determine the optimal locations of a set of facilities and optimal allocations of customer demands to the facilities subject to the capacity and demand restrictions at minimum total transportation cost. Clearly, any LAP becomes a pure multi-facility location problem when an allocation scheme is given. On the other hand, a LAP becomes a pure allocation problem when facility locations are known. First, (Cooper, 1964) observes this property and proposes ALA heuristic for the MWP, which simply consists of the solution of the location and allocation problems alternately, starting with an initial set of facility locations until no further improvement is possible. ALA heuristic ends up with a local optima. Namely, no better locations can be found given the current allocations and no better allocations can be found given the current locations. The author also implements the capacitated version of ALA (i.e., CALA) heuristic in his subsequent work on the CMWP (Cooper, 1972).

In this chapter[1] we suggest several ALA heuristics for the MCMWP. We first present the multi-commodity extension of the ALA heuristic. Namely, we introduce the Multi-commodity Capacitated Alternate Location-Allocation (MCALA) heuristic. Then, we focus on multi-commodity extensions of two Region Rejection (RR) heuristics which are originally devised for the CMWP by (Luis *et al.*, 2009). RR heuristics are extensions of the CALA heuristic with sophisticated initialization procedures. We should note that there are several ALA heuristics that can be adapted for the MCMWP. However, RR heuristics are more efficient and more accurate than other ALA heuristics such as the cellular heuristics by (Gamal and Salhi, 2003) and (Aras *et al.*, 2007) for the MWP and CMWP, respectively. (Luis *et al.*, 2009) also reports that RR heuristics are superior than the perturbation based heuristic by (Zainuddin and Salhi, 2007) for the CMWP. Encouraged by their superior performance, we adapt the RR heuristics for the MCMWP. Lastly, we propose discrete enhancements of the MCALA and RR heuristics resulting in totally six ALA heuristics for the MCMWP.

[1]The article by (Akyüz *et al.*, 2012), and the technical report (Akyüz *et al.*, 2010a) are partially based on this chapter.

4.1. Multi-commodity Capacitated Alternate Location Allocation Heuristic

The MCMWP is also a LAP, and given a feasible transportation plan, it reduces to solving I WPs which can be given as

$$\min_{\mathbf{x}_i} Z_{WP} = \sum_{j=1}^{J} c'_{ij} d(\mathbf{x}_i, \mathbf{a}_j), \tag{4.1}$$

where c'_{ij} is defined as $c'_{ij} = \sum_{k=1}^{K} w_{ijk} c_{ijk}$ for each facility $i = 1, \ldots, I$. Note that Equation 4.1 is the same as Equation 3.1, which can be solved by Weiszfeld's algorithm (Weiszfeld, 1937) and one of its generalizations (Brimberg and Love, 1993; Frenk *et al.*, 1994; Brimberg *et al.*, 1998). Although the summation is taken over all customers, it only considers $|\mathcal{I}_i|$ of them, which is the size of the set $\mathcal{I}_i = \{(j,k) : w_{ijk} > 0\}$. Clearly, $\sum_{i=1}^{I} |I_i| \geq J \times K$ holds since a customer can be served by more than one facility. In short, when a feasible assignment of w_{ijk} variables is given, the problem reduces to the determination of the optimal locations of single facilities with respect to $|I_i|$ customers.

The MWP, CMWP and MCMWP consist of similar location components. However, their allocation problems differ. For the MWP, the allocation problem consists of the assignment of each customer to the least weighted cost facility. The allocation problem becomes the solution of an ordinary transportation problem for the CMWP and the solution of the Multi-commodity Transportation Problem (MTP) for the MCMWP which is as follows.

MTP:

$$\min \quad Z_{MTP} = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \widehat{c}_{ijk} w_{ijk} \tag{4.2}$$

$$\text{s.t.} \qquad \text{Equation } 2.3 - 2.6. \tag{4.3}$$

Here, given the current facility locations, unit transportation costs are defined as $\widehat{c}_{ijk} = c_{ijk} d(\mathbf{x}_i, \mathbf{a}_j)$. In short, the MCALA heuristic uses the alternate solutions of the I WPs given by Equation 4.1 and a MTP given by Equation 4.2 – 4.3 until a local optima is obtained. A formal outline of the MCALA heuristic is given in Figure 4.1.

In their work on the dominance and convexity in location theory, (Hansen *et al.*, 1980) prove that for ℓ_r-norm with $r \geq 1$ the optimal facility locations of a multi-facility location

1. Locate the facilities at arbitrarily selected points $\mathbf{x}_i = (x_{i1}\ x_{i2})^T$ $i = 1, \ldots, I$.
2. For each facility i and customer j calculate the distance $d(\mathbf{x}_i, \mathbf{a}_j)$ between them, and set the new unit transportation cost as $\widehat{c}_{ijk} = c_{ijk} d(\mathbf{x}_i, \mathbf{a}_j)$.
3. Determine feasible allocations w_{ijk} by solving the MTP with costs $\widehat{c}_{ijk}$.
4. Solve I Weber problems given by Equation 4.1 to relocate I facilities.
5. Repeat Step 2– Step 4 until either facility locations $\mathbf{x}_i = (x_{i1}\ x_{i2})^T$ for $i = 1, \ldots, I$ or allocations w_{ijk} for $i = 1, \ldots, I$, $j = 1, \ldots, J$, and $k = 1, \ldots, K$ remain unchanged.

Figure 4.1. The MCALA heuristic.

problem lie within the convex hull of customers. Taking into consideration this result, the MCALA heuristic is initialized by randomly choosing the facility locations within the convex hull of customer locations. We refer to this version of the MCALA as the *continuous* MCALA (C-MCALA) where in Step 1 of MCALA heuristic presented in Figure 4.1, the initial facility locations are randomly selected within the convex hull of customers.

4.2. Region Rejection Heuristics

In a recent work (Luis *et al.*, 2009) claim that if the initial facility locations are well separated the accuracy of both Cooper's ALA and CALA increase. They propose the basic RR heuristic which accomplishes this task by randomly locating facilities one by one. The RR heuristic checks each time whether or not there is an already placed facility that remains within the circle of a given radius centered at the newly initialized facility. In case there is at least one circle, a new random location is selected within the convex hull of customer locations for the new facility. The initialization becomes complete when all the facilities are placed. Then a solution is computed by running the usual alternating location and allocation steps, i.e., CALA. According to their results a variant of RR performs better. In this one the radius is not fixed but dynamically calculated according to the capacity of the facility and the demands of the customers around the newly initialized facility. More precisely, in this Dynamic radius enhancement (DRR) the rejection circle of a facility contains the set of customers whose total demand is approximately equal to the facility capacity at which it is centered. As can be noticed it is not difficult to adapt these two initialization strategies

for the MCMWP. We call these multi-commodity extensions of RR and DRR, as MRR and MDRR, respectively.

4.2.1. The Multi-commodity Region Rejection Heuristic

MRR heuristic gradually initializes facility locations within the convex hull of customer locations. Once a set of facilities are randomly located, the regions that are close to those already opened facilities are forbidden for a new facility to be located. Actually, a new facility can not be located within the circles centered at the previously located facilities. In addition, the radius of these circles are iteratively adjusted until all facilities are located. The heuristic initially selects a predefined radius length which is the same for all facilities. If it is not possible to generate locations (i.e., existing circles cover the entire convex hull of the customers) for uninitialized facilities, all radii of the existing circles are shrunk with an adjustment factor until all facilities are initialized. Then given these facility locations a MCALA heuristic is run in order to produce the final solution. The outline of the fixed radius MRR heuristic algorithm is given in Figure 4.2.

1. Choose a random point within the convex hull of customers and set $\Delta_1 = \left(\max_{j\in\mathcal{J}}\{a_{j1}\} - \min_{j\in\mathcal{J}}\{a_{j1}\}\right)$, $\Delta_2 = \left(\max_{j\in\mathcal{J}}\{a_{j2}\} - \min_{j\in\mathcal{J}}\{a_{j2}\}\right)$. Draw a circle with radius $\tau = \min(\eta\frac{\Delta_1}{I}, \eta\frac{\Delta_2}{I})$.
2. Choose another random point. If the point is outside of existing circle(s) fix this point and draw a circle with radius t. Otherwise repeat Step 2 until a fixed number of times.
3. In case no point found outside the existing circle(s) and the iteration limit is exceeded, decrease the radii of all circles (e.g., by 10%) and start from the beginning.
4. Run the MCALA heuristic with the facilities located on the centers of the circles.

Figure 4.2. The MRR heuristic (fixed circle radius).

In the original version of the MRR heuristic, η is a parameter which is chosen from the interval $(0, 1)$ and used to control the radius $\tau = min(\eta\frac{\Delta_1}{I}, \eta\frac{\Delta_2}{I})$. (Luis *et al.*, 2009) state that for the CMWP setting $\eta = 0.5$ seems to be a reasonable choice. We keep their setting and use the suggested parameters for the MCMWP as well. In Step 1, the selection

of facility locations are random within the convex hull of customers and we refer to this continuous version as C-MRR heuristic.

4.2.2. The Multi-commodity Dynamic Region Rejection Heuristic

(Luis *et al.*, 2009) put forward two versions of the RR. In the first one circle radius is fixed, and in the second one circle radius is adjusted dynamically. The authors observe that the dynamic radius version of the algorithm performs better than the fixed radius version for the CMWP case. Now, we present the MDRR heuristic (RR with dynamic circle radius adjustment) for the MCMWP. Recall that in Step 3 and Step 4 of MRR heuristic, when the iteration limit is exceeded and there are still facilities to be initialized, the MRR heuristic adjusts the radii of all circles together with an adjustment factor. On the other hand, adjusting all circles together may be disadvantageous for some facilities. In order to overcome this issue the MDRR heuristic takes into account the ratio of a facility's capacity to meet the demand of customers within its forbidden circle. Then the MDRR dynamically adjusts only the radius of the corresponding facility while keeping others as they are. The formal outline of the MDRR heuristic is presented in Figure 4.3.

We use the same parameter setting suggested by (Luis *et al.*, 2009) for the CMWP. Notice that the dynamic adjustment of circle radii may end up with a covering of the convex hull of the customers. Then, there is no room to locate uninitialized facilities. In this case, the MDRR heuristic pursues a random initialization for the remaining facilities and stops adjusting circle radii of already located facilities. As the facility locations are initialized within a continuous space defined by the convex hull of the customer locations, we refer to this version as C-MDRR heuristic.

4.3. Discrete Enhancements of the Alternate Location-Allocation Heuristics

ALA, CALA and MCALA are all simple and efficient; but the quality of the final solution they compute depends very much on the initial solution. One quick remedy is to benefit from its efficiency and to repeat this heuristic many times starting at random locations. We also apply a multi-start strategy for C-MCALA, C-MRR and C-MDRR heuristics and select random initialization within the convex hull of customers. Although a rigorous theoretical result does not exist, it is observed that optimal facility locations are usually either on customer

1. Let cnt_1 and cnt_2 denote the number of iterations and the number of trials for radius adjustments, respectively. Set $\Delta_1 = \left(\max_{j\in\mathcal{J}}\{a_{j1}\} - \min_{j\in\mathcal{J}}\{a_{j1}\}\right)$, $\Delta_2 = \left(\max_{j\in\mathcal{J}}\{a_{j2}\} - \min_{j\in\mathcal{J}}\{a_{j2}\}\right)$, $cnt_1 = 0$ and $cnt_2 = 0$.
2. Choose a random point within the convex hull of customers and draw a circle with a radius $\tau = \min(\eta\frac{\Delta_1}{I}, \eta\frac{\Delta_2}{I})$. Increase cnt_1 by one.
3. **if** cnt_1 does not exceed J, **then** set $cnt_2 = 0$ and go to Step 4.
   else go to Step 6.
4. In case the total demand of the customers within the corresponding circle is between 70% and 100% of the total capacity of the corresponding facility fix this facility's location and go to Step 5. The ratio used is $\Gamma = \frac{\text{Total demand within the corresponding circle}}{\text{Total capacity of the corresponding facility}}$. For the other case go to Step 2 and consider the next facility. **if** there are no more facilities, **then** go to Step 7.
5. Adjust the radius with an adjustment factor of $\sqrt{\Gamma}$, increase cnt_2 by one. **if** cnt_2 does not exceed $\left\lceil\frac{J}{I}\right\rceil$, **then** go to Step 4. **else** apply the bisection method between the previous and current circles in order to adjust their radii.
6. **if** there are still facilities not yet randomly located, **then** open them within the convex hull of customers.
7. Apply the MCALA heuristic initialized with the current facility locations.

Figure 4.3. The MDRR heuristic (dynamic adjustment of circle radii).

locations or very close to them. This is exploited first by (Hansen *et al.*, 1998) for the MWP, then by (Aras *et al.*, 2007) for the CMWP to propose very accurate Discrete Approximation (DA) heuristics. We also exploit this observation and modify location-allocation heuristics described above in order to obtain their discrete versions, Discrete MCALA (D-MCALA), Discrete MRR (D-MRR) and Discrete MDRR (D-MDRR) which work over customer locations instead of their entire convex hull. Actually, the idea to initialize the ALA heuristic randomly over customer locations and to work on such a discrete location space is first tested by (Cooper, 1964) in his random destination algorithm for which the author reports more accurate results than the classical ALA algorithm does. This result has also motivated us to implement a discrete version of our ALA heuristics.

In the discrete enhancements of ALA heuristics, only a countable number of candidate points are chosen within the convex hull. We locate I facilities on the minimum cost customer locations instead of running Weiszfeld's algorithm. Therefore, for each facility i we first sort the costs of customer locations and then we assign the location of each facility i to the minimum cost customer location. In other words, the location phase of the ALA heuristics replaces the WP given by Equation 4.1 with the following *1*-median problem

$$\underset{j'=1,\ldots,J}{\arg\min} \left\{ \sum_{j=1}^{J} c'_{j'j} d(\mathbf{a}_{j'}, \mathbf{a}_j) \right\} \tag{4.4}$$

Here $c'_{j'j} = \sum_{k=1}^{K} c_{ijk} w_{ijk} d(\mathbf{a}_{j'}, \mathbf{a}_j)$ for $i = 1, \ldots, I$ with given allocation quantities w_{ijk}. Besides, once a facility is opened on a candidate point, the rest of the facilities are randomly located one by one on the customer locations that are not covered by the circles centered at the previously located facilities, during the initialization of the RR heuristics MRR and MDRR. As a remark, the discretization need not be limited to the customer locations. The candidate point set can be selected in a different way but we prefer to use customer locations as explained. We should also point out that both continuous and discrete versions of the MCALA heuristic are employed within the LR schemes implemented for the DA heuristics which are explained in the next chapter.

5. DISCRETE APPROXIMATION HEURISTICS

(Hansen *et al.*, 1980) generalize the dominance properties obtained for the rectilinear distance WP by (Wendell and Hurter, 1973) to show that optimal solutions of the two-dimensional location problems with the rectilinear distance function (i.e., the multiplication of the rectilinear norm of the difference of customer and facility location vectors by a non-negative coefficient) always occur within the convex hull of the customer locations and at the intersection of the vertical and horizontal lines drawn through them. In the same work (Hansen *et al.*, 1980) also show that optimal location of the two-dimensional location problems remain within the convex hull of customer locations for all distance functions that are ℓ_r-norms. (Aras *et al.*, 2008) take advantage of these properties and reformulated the RCMWP equivalently as a MILP problem, by restricting optimal facility locations to belong to a candidate location set. This set is constructed by intersecting horizontal and vertical lines drawn through the customer locations and considering the intersection points remaining within their convex hull. They also suggest to use these properties to approximate the CMWP with a norm distance function by a MILP problem formulated over a set of candidate points selected within the convex hull of the customer locations (Aras *et al.*, 2007). Thus, it becomes also possible to formulate approximating MILP problems over the points selected from the convex hull of customer locations for the MCMWP when more general norms are used as distance functions.

This chapter[2] is devoted to multi-commodity extensions of the Discrete Approximation (DA) heuristics and our improvements on them to produce both lower and upper bounds. We first present two approximating MILP formulations for the MCMWP. The first formulation is the discrete version of the original MCMWP formulation given by Equation 2.2 – 2.6. The second MILP formulation is the discrete counterpart of the second MCMWP formulation given by Equation 2.7 – 2.12. Then, we propose LR schemes for both of these MILP formulations. Afterwards, we put forward two discretization strategies together with the selection of candidate points. For that purpose, we first introduce the block norms which have the potential to produce both lower and upper bounding candidate point sets for the MILPs. The second discretization strategy takes into account only the customer locations

[2]The article by (Akyüz *et al.*, 2012), the technical report by (Akyüz *et al.*, 2010a) and the conference proceeding by (Akyüz *et al.*, 2009a) are partially based on this chapter.

as the candidate points set. Last part is dedicated to the formal description of the Multi-commodity Discrete Approximation (MDA) heuristics which are capable to produce both tight lower and upper bounds for the MCMWP. Our approach can also be extended for both MWP and CMWP cases.

5.1. Approximating Mixed Integer Linear Programming Formulations

Let $g = 1, \ldots, G$ label the given set of candidate facility locations (approximating points) remaining within the convex hull of customer locations and define the variables y_{ijkg} as the amount of commodity k shipped from facility i located at candidate point g to customer j. Binary variables v_{ig} are set to 1 if facility i is located at point g and 0 otherwise. c_{ijkg} is the cost of transporting one unit of commodity k from facility i located at candidate point g with known coordinates $\widehat{\mathbf{a}}_g = (\widehat{a}_{g1}\ \widehat{a}_{g2})^T$ to customer j. It is obtained by multiplying the unit shipment cost of commodity k per unit distance from facility i to customer j, namely c_{ijk}, with the distance $d(\widehat{\mathbf{a}}_g, \mathbf{a}_j)$ between point g and customer j. In other words $c_{ijkg} = c_{ijk} d(\widehat{\mathbf{a}}_g, \mathbf{a}_j)$ where $d(\widehat{\mathbf{a}}_g, \mathbf{a}_j) = [|\widehat{a}_{g1} - a_{j1}|^r + |\widehat{a}_{g2} - a_{j2}|^r]^{1/r}$ with $1 \leq r < \infty$. Then, the first formulation of the Multi-commodity Discrete Approximation Problem (MDAP) can be given as

MDAP1:

$$\min \quad Z_{MDAP1} = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G} c_{ijkg} y_{ijkg} \tag{5.1}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} y_{ijkg} = s_{ik} v_{ig} \qquad i = 1, \ldots, I; k = 1, \ldots, K; g = 1, \ldots, G, \tag{5.2}$$

$$\sum_{i=1}^{I}\sum_{g=1}^{G} y_{ijkg} = q_{jk} \qquad j = 1, \ldots, J; k = 1, \ldots, K, \tag{5.3}$$

$$\sum_{g=1}^{G} v_{ig} = 1 \qquad i = 1, \ldots, I, \tag{5.4}$$

$$\sum_{k=1}^{K}\sum_{g=1}^{G} y_{ijkg} \leq u_{ij} \qquad i = 1, \ldots, I; j = 1, \ldots, J, \tag{5.5}$$

$$y_{ijkg} \geq 0 \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K; g = 1, \ldots, G, \tag{5.6}$$

$$v_{ig} \in \{0, 1\} \qquad i = 1, \ldots, I; g = 1, \ldots, G. \tag{5.7}$$

Here, constraints given by Equation 5.2 – 5.5 and 5.6 are the discrete equivalent of the multi-commodity transportation constraints given by Equation 2.3 – 2.6. Constraints given by Equation 5.2 ensure that the total amount of commodity k shipped from facility i located at point g is equal to its capacity. Binary variables v_{ig} guarantee that whenever a facility i is opened at a candidate point g^* then y_{ijkg} are set to 0 for $j = 1, \ldots, J; k = 1, \ldots, K$ and $g \in \{g' : g' = 1, \ldots, G \text{ and } g' \neq g^*\}$. Clearly, when there is no open facility at point g then no shipment can originate from there. Constraints given by Equation 5.3 state that the demand of each customer j for each commodity type k is satisfied. Constraints given by Equation 5.4 enforce that each facility i is located at exactly one of the candidate points $g = 1, \ldots, G$.

Indeed, constraints given by Equation 5.4 become redundant when MDAP1 is balanced. To see this, aggregate constraints given by Equation 5.2 and 5.3 which result in

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G} y_{ijkg} = \sum_{i=1}^{I}\sum_{k=1}^{K} s_{ik} \sum_{g=1}^{G} v_{ig} \tag{5.8}$$

and

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G} y_{ijkg} = \sum_{j=1}^{J}\sum_{k=1}^{K} q_{jk}, \tag{5.9}$$

respectively. Since the left-hand sides of both equalities in Equation 5.8 and 5.9 are the same, it follows that $\sum_{i=1}^{I}\sum_{k=1}^{K} s_{ik} \sum_{g=1}^{G} v_{ig} = \sum_{k=1}^{K}\sum_{j=1}^{J} q_{jk} = \sum_{k=1}^{K}\sum_{i=1}^{I} s_{ik}$ because MDAP1 is balanced. The last equality is clearly satisfied if and only if $\sum_{g=1}^{G} v_{ig} = 1$ holds for $i = 1, ..., I$. Without loss of generality MDAP1 can be transformed into a balanced form by using dummy facilities or customers with zero transportation cost. On the other hand, constraints given by Equation 5.4 are valid equalities for MDAP1 and according to our observations their existence considerably improves the LP relaxation bounds. Therefore, we keep constraints given by Equation 5.4 which makes the MDAP1 formulation stronger.

It is also possible to model a discrete approximation as a bilinear programming problem by using the approach proposed for the RCMWP by (Sherali *et al.*, 1994). This will require an additional linearization effort resulting in an increase in the number of variables and constraints. Although MDAP1 can be optimally solved by a general-purpose MILP solver, the required CPU time exponentially increases with the increasing instance size. When we

consider a problem with I facilities, J customers, K commodities and G candidate points, this results in a MILP formulation with $I \times G$ binary variables v_{ig} and $I \times J \times K \times G$ continuous variables y_{ijkg}.

The formulation of MDAP1 allows locating more than one facility at a candidate point g. Moreover, the unit shipment cost does not only depend on the location of a facility, but also on the facility itself. This means that transporting the same amount of commodity k to customer j from two different facilities located at point g may incur different costs. The formulation becomes more compact when these facilities are uniform and the unit shipment cost depends on both location and commodity. In this case, we redefine flow variables y_{jkg} as the amount of commodity k shipped to customer j from point g, and cost coefficients c_{jkg} as the unit shipment cost of commodity k from point g to customer j. The latter is obtained by multiplying the unit shipment cost per unit distance of commodity k, that is independent of facility i, with the distance between point g and customer j. In fact, the second formulation can be directly obtained from MDAP1 by setting $c_{ijkg} = c_{jkg}$ for all $i = 1, \ldots, I$ and using the aggregated flow variables $y_{jkg} = \sum_{i=1}^{I} y_{ijkg}$ in the objective function as well as constraints given by Equation 5.2 and 5.3.

In the second MDAP formulation, namely in MDAP2, we split location variable v_{ig} over the commodities using K binary variables (one for each commodity to represent a production center of facility i producing commodity k) $\widehat{v}_{ikg}$. In other words, v_{ig} is split to $\widehat{v}_{ikg}$ which is set to 1 when production center k of facility i is located at candidate point g, and 0 otherwise. We define new constraints to force production centers behave unanimously. When $\widehat{v}_{ikg} = 1$ ($\widehat{v}_{ikg} = 0$) holds for one of the production centers of facility i, say center k^*, then $\widehat{v}_{ikg} = 1$ ($\widehat{v}_{ikg} = 0$) also holds for all production centers, for $k \neq k^*$ and $k = 1, \ldots, K$, of facility i. This is an "all or none" type behavior.

MDAP2:

$$\min \quad Z_{MDAP2} = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G} c_{ijkg} y_{ijkg} \tag{5.10}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} y_{ijkg} = s_{ik}\widehat{v}_{ikg} \qquad i = 1, \ldots, I; k = 1, \ldots, K; g = 1, \ldots, G, \tag{5.11}$$

$$\sum_{l=1}^{G} \widehat{v}_{ikg} = 1 \qquad i = 1, \ldots, I; k = 1, \ldots, K, \tag{5.12}$$

$$\sum_{i=1}^{I}\sum_{g=1}^{G} y_{ijkg} = q_{jk} \quad j = 1, \ldots, J; k = 1, \ldots, K, \tag{5.13}$$

$$\sum_{k=1}^{K}\sum_{g=1}^{G} y_{ijkp} \leq u_{ij} \quad i = 1, \ldots, I; j = 1, \ldots, J, \tag{5.14}$$

$$\widehat{v}_{ikg} \leq v_{ig} \qquad i = 1, \ldots, I; k = 1, \ldots, K; g = 1, \ldots, G, \tag{5.15}$$

$$\sum_{k=1}^{K} \widehat{v}_{ikg} \geq K v_{ig} \qquad i = 1, \ldots, I; g = 1, \ldots, G, \tag{5.16}$$

$$\sum_{g=1}^{G} v_{ig} = 1 \qquad i = 1, \ldots, I, \tag{5.17}$$

$$v_{ig} \in \{0, 1\} \qquad i = 1, \ldots, I; g = 1, \ldots, G, \tag{5.18}$$

$$\widehat{v}_{ikg} \in \{0, 1\} \qquad i = 1, \ldots, I; k = 1, \ldots, K; g = 1, \ldots, G, \tag{5.19}$$

$$y_{ijkg} \geq 0 \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K; g = 1, \ldots, G. \tag{5.20}$$

In this formulation, constraints given by Equation 5.15 ensure that none of the K production centers of facility i can be opened at candidate point g unless the original facility i is located there. Furthermore, when facility i is not located on candidate point g then no production center of that facility is allowed to be opened on candidate point g. Constraints given by Equation 5.16 together with Equation 5.15 enforce that whenever facility i is opened at candidate point g then all production centers of facility i must also be located on candidate point g. Additionally, constraints given by Equation 5.17 guarantee that a facility must be located at exactly one candidate point.

Also, it is possible to replace constraints given by Equation 5.15 and 5.16 with the equalities

$$\sum_{k=1}^{K} \widehat{v}_{ikg} = K v_{ig} \qquad i = 1, \ldots, I; g = 1, \ldots, G \tag{5.21}$$

to obtain a more compact formulation. Unfortunately, it is weaker and produces a loser LP relaxation lower bound, which makes MDAP2's exact solution computationally more challenging.

5.2. Lagrangean Relaxation and Discrete Approximations

For some instances, the sizes of the MDAP1 and MDAP2 formulations can be quite huge and their exact solution can be computationally intractable. Hence, it may sound wiser to solve them approximately. Since both MDAP1 and MDAP2 belong to the family of discrete location problems, LR approach can successfully be applied for their solution. We devise a LR scheme and Subgradient Optimization (SO) to compute lower bounds and good feasible solutions for both MDAP1 and MDAP2.

5.2.1. Lagrangean Relaxation for the First Approximation

We can relax demand constraints given by Equation 5.3 and bundle constraints given by Equation 5.5 with Lagrangean multipliers β^1_{jk} and μ^1_{ij} respectively and we obtain the Lagrangean subproblem

RMDAP1($\boldsymbol{\beta}^1$, $\boldsymbol{\mu}^1$):

$$\min\ Z_{LR1}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G}(c_{ijkg} - \beta^1_{jk} + \mu^1_{ij})y_{ijkg} + \sum_{j=1}^{J}\sum_{k=1}^{K}\beta^1_{jk}q_{jk} - \sum_{i=1}^{I}\sum_{j=1}^{J}\mu^1_{ij}u_{ij} \tag{5.22}$$

$$\text{s.t. Equation 5.2, 5.4, 5.6 and 5.7,} \tag{5.23}$$

$$y_{ijkg} \leq \widehat{u}_{ijk} \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K; g = 1, \ldots, G. \tag{5.24}$$

Notice that simple upper bounds given by Equation 5.24 are added to the formulation, where $\widehat{u}_{ijk} = \min\{s_{ik}, q_{jk}, u_{ij}\}$. Although they are redundant for the original problem MDAP1, they improve optimal value $Z^*_{LR1}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1)$ of the relaxed problem. The last two terms in the objective function given by Equation 5.22 are constants and RMDAP1($\boldsymbol{\beta}^1$, $\boldsymbol{\mu}^1$) decomposes over the facilities. Although it seems possible to decompose also with respect to the commodities at the first look, this is not possible because of constraints given by Equation 5.4. The solution of RMDAP1($\boldsymbol{\beta}^1$, $\boldsymbol{\mu}^1$) becomes equivalent to the solution of the following I subproblems

RMDAP1$_i(\boldsymbol{\beta}^1,\, \boldsymbol{\mu}^1)$:

$$\min\ Z_{LR1^i}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1) = \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G}\overline{c}_{ijkg}y_{ijkg} \tag{5.25}$$

$$\text{s.t.} \sum_{j=1}^{J} y_{ijkg} = s_{ik}v_{ig} \qquad k = 1,\ldots,K; g = 1,\ldots,G, \tag{5.26}$$

$$\sum_{g=1}^{G} v_{ig} = 1 \qquad i = 1,\ldots,I, \tag{5.27}$$

$$y_{ijkg} \leq \widehat{u}_{ijk} \qquad j = 1,\ldots,J; k = 1,\ldots,K; g = 1,\ldots,G, \tag{5.28}$$

$$y_{ijkg} \geq 0 \qquad j = 1,\ldots,J; k = 1,\ldots,K; g = 1,\ldots,G, \tag{5.29}$$

$$v_{ig} \in \{0,1\} \qquad g = 1,\ldots,G. \tag{5.30}$$

Here $\overline{c}_{ijkg} = (c_{ijkg} - \beta^1_{jk} + \mu^1_{ij})$ is the new unit cost obtained for the given multiplier vectors $\boldsymbol{\beta}^1$ and $\boldsymbol{\mu}^1$. The solution of subproblem RMDAP1$_i(\boldsymbol{\beta}^1,\, \boldsymbol{\mu}^1)$ is not difficult. We can use a "greedy" inspection procedure where, for each point g, we determine those customers that are supplied from facility i when located at point g so that the shipment cost $Z_{LR1^{ig}}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1) = \sum_{j=1}^{J}\sum_{k=1}^{K}\overline{c}_{ijkg}y_{ijkg}$ is minimized subject to $\sum_{j=1}^{J} y_{ijkg} = s_{ik}v_{ig}$, $k = 1,\ldots,K$ with $v_{ig} = 1$ and $0 \leq y_{ijkg} \leq u_{ijk}$, $j = 1,\ldots,J$; $k = 1,\ldots,K$. Observe that this LP problem decomposes further over the commodities resulting in K bounded Continuous Knapsack Problem (CKP) each of which can be solved in $\mathcal{O}(J)$ times in the worst case (Martello and Toth, 1990). The solution approach we adopt in this work is conceptually simpler, but computationally less efficient. It can be found in any standard textbook on the LP (Bazaraa *et al.* 2010). A sorting effort is required at the beginning, which results in $\mathcal{O}(J \log J)$ time complexity.

When we repeat this procedure K times for each candidate point $g = 1,\ldots,G$ a good optimal location for facility i can be determined; this is the point where $Z_{LR1^i}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1)$ is minimized. As a result, the optimal value $Z^*_{LR1^i}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1)$ is determined by setting $Z^*_{LR1^i}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1) = \min_g \left\{Z^*_{LR1^{ig}}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1)\right\}$ and $v_{ig} = 1$ for only one of the candidate points, and $v_{ig} = 0$ for the others satisfying constraints given by Equation 5.27. Clearly, $Z^*_{LR1^{ig}}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1) = \sum_{k=1}^{K} Z^*_{LR1^{ikg}}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1)$ holds where $Z^*_{LR1^{ikg}}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1)$ denotes the optimal value of the bounded CKP which is separately solved for each commodity k.

As soon as we solve all subproblems RMDAP1$_i(\boldsymbol{\beta}^1,\, \boldsymbol{\mu}^1)$, we can calculate the optimal value of the RMDAP1$(\boldsymbol{\beta}^1,\, \boldsymbol{\mu}^1)$ as $Z^*_{LR1}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1) = \sum_{i=1}^{I} Z^*_{LR1^i}(\boldsymbol{\beta}^1, \boldsymbol{\mu}^1) + \sum_{j=1}^{J}\sum_{k=1}^{K}\beta^1_{jk}q_{jk} -$

$\sum_{i=1}^{I}\sum_{j=1}^{J}\mu_{ij}^{1}u_{ij}$ for given multiplier vectors $\boldsymbol{\beta}^1$ and $\boldsymbol{\mu}^1$. $Z^*_{LR1}(\boldsymbol{\beta}^1,\boldsymbol{\mu}^1)$ is a lower bound on the optimal value of MDAP1 for any Lagrange multiplier vectors $\boldsymbol{\beta}^1$ and $\boldsymbol{\mu}^1$. To find the best lower bound, we have to solve the Lagrangean dual problem, i.e., $\max_{\boldsymbol{\beta}^1,\boldsymbol{\mu}^1} Z^*_{LR1}(\boldsymbol{\beta}^1,\boldsymbol{\mu}^1)$, which can be achieved by performing the SO algorithm (Held *et al.*, 1974).

5.2.2. Lagrangean Relaxation for the Second Approximation

It is also possible to apply Lagrangean relaxation on MDAP2. For this purpose we relax constraints given by Equation 5.13, 5.14, 5.15 and 5.16 with multipliers β_{jk}^2, μ_{ij}^2, δ_{ikg} and $\widehat{\delta}_{ig}$ to obtain the subproblem

RMDAP2($\boldsymbol{\beta}^2$,$\boldsymbol{\mu}^2$,$\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$):

$$\begin{aligned}\min \quad Z_{LR2}(\boldsymbol{\beta}^2,\boldsymbol{\mu}^2,\boldsymbol{\delta},\widehat{\boldsymbol{\delta}}) = &\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G}(c_{ijkg}-\beta_{jk}^2+\mu_{ij}^2)y_{ijkg}+\sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{g=1}^{G}(\delta_{ikg}-\widehat{\delta}_{ig})\widehat{v}_{ikg}\\ &+\sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{g=1}^{G}(\widehat{\delta}_{ig}-\delta_{ikg})v_{ig}+\sum_{j=1}^{J}\sum_{k=1}^{K}\beta_{jk}q_{jk}-\sum_{i=1}^{I}\sum_{j=1}^{J}\mu_{ij}^2u_{ij} \qquad (5.31)\end{aligned}$$

$$\text{s.t.} \quad \text{Equation 5.11, 5.12, 5.17 – 5.20 and 5.24.} \qquad (5.32)$$

The last two terms of the relaxed objective function given by Equation 5.31 are constant and can be disregarded during the solution. Again, we add simple redundant upper bounds given by Equation 5.24 to strengthen the Lagrangean lower bound. Observe that RMDAP2($\boldsymbol{\beta}^2$,$\boldsymbol{\mu}^2$,$\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$) can be separated into two subproblems one with both variables y_{ijkg} and $\widehat{v}_{ikg}$, and the other with variables v_{ig} variables. In addition, the former problem decomposes further over the customers and facilities. In other words RMDAP2($\boldsymbol{\beta}^2$,$\boldsymbol{\mu}^2$,$\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$) can be solved by solving the following $I \times K + 1$ subproblems

RMDAP2$_{ik}$($\boldsymbol{\beta}^2$,$\boldsymbol{\mu}^2$,$\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$):

$$\min \quad Z_{LR2^{ik}}(\boldsymbol{\beta}^2,\boldsymbol{\mu}^2,\boldsymbol{\delta},\widehat{\boldsymbol{\delta}}) = \sum_{j=1}^{J}\sum_{g=1}^{G}\Big[(c_{ijkg}-\beta_{jk}^2+\mu_{ij}^2)y_{ijkg}+(\delta_{ikg}-\widehat{\delta}_{ig})\widehat{v}_{ikg}\Big] \qquad (5.33)$$

$$\text{s.t.} \quad \sum_{j=1}^{J} y_{ijkg} = s_{ik}\widehat{v}_{ikg} \qquad g=1,\ldots,G \qquad (5.34)$$

$$\sum_{g=1}^{G} \widehat{v}_{ikg} = 1 \qquad i = 1, \ldots, I; k = 1, \ldots, K, \tag{5.35}$$

$$y_{ijkg} \leq \widehat{u}_{ijk} \qquad j = 1, \ldots, J; g = 1, \ldots, G, \tag{5.36}$$

$$y_{ijkg} \geq 0 \qquad j = 1, \ldots, J; g = 1, \ldots, G, \tag{5.37}$$

$$\widehat{v}_{ikg} \in \{0, 1\} \qquad i = 1, \ldots, I; k = 1, \ldots, K; g = 1, \ldots, G. \tag{5.38}$$

and

RMDAP2$_v$($\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$):

$$\min Z_{LR2^v}(\boldsymbol{\delta}, \widehat{\boldsymbol{\delta}}) = \sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{g=1}^{G}(\widehat{\delta}_{ig} - \delta_{ikg})v_{ig} \tag{5.39}$$

$$\text{s.t.} \qquad \text{Equation 5.17 and 5.18.}$$

Notice that for $\widehat{v}_{ikg} = 1$, RMDAP2$_{ik}$($\boldsymbol{\beta}^2$,$\boldsymbol{\mu}^2$,$\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$) can be solved using the optimal solutions of G CKPs in y_{ijkg}. Each of them takes $\mathcal{O}(J \log J)$ times if the ordering procedure mentioned in Section 5.2.1 is used. This gives the optimal value of $Z^*_{LR2^{ik}}(\boldsymbol{\beta}^2, \boldsymbol{\mu}, \boldsymbol{\delta}, \widehat{\boldsymbol{\delta}}) = \min_g \left\{ Z^*_{LR2^{ikg}}(\boldsymbol{\beta}^2, \boldsymbol{\mu}^2, \boldsymbol{\delta}, \widehat{\boldsymbol{\delta}}) \right\}$ where $Z^*_{LR2^{ikg}}(\boldsymbol{\beta}^2, \boldsymbol{\mu}, \boldsymbol{\delta}, \widehat{\boldsymbol{\delta}})$ is the optimal value of the CKPs. Then, the optimal value of the Lagrangean subproblem RMDAP2($\boldsymbol{\beta}^2$,$\boldsymbol{\mu}^2$,$\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$) can be calculated as $Z^*_{LR2}(\boldsymbol{\beta}^2, \boldsymbol{\mu}^2, \boldsymbol{\delta}, \widehat{\boldsymbol{\delta}}) = \sum_{i=1}^{I}\sum_{k=1}^{K} Z^*_{LR2^{ik}}(\boldsymbol{\beta}^2, \boldsymbol{\mu}^2, \boldsymbol{\delta}, \widehat{\boldsymbol{\delta}}) + Z^*_{LR2^v}(\boldsymbol{\delta}, \widehat{\boldsymbol{\delta}}) + \sum_{j=1}^{J}\sum_{k=1}^{K} \beta^2_{jk} q_{jk} - \sum_{i=1}^{I}\sum_{j=1}^{J} \mu^2_{ij} u_{ij}$. Here, $Z^*_{LR2^v}(\boldsymbol{\delta}, \widehat{\boldsymbol{\delta}})$ is the optimal value of RMDAP2$_v$($\boldsymbol{\delta}$,$\widehat{\boldsymbol{\delta}}$), which can be easily obtained, since the subproblem decomposes over all facilities into I subproblems each of which can be solved by inspection. In order to find the best lower bound, we solve the Lagrangean dual problem $\max_{\boldsymbol{\delta},\mu,\delta,\widehat{\delta}} Z^*_{LR2}(\boldsymbol{\beta}^2, \boldsymbol{\mu}^2, \boldsymbol{\delta}, \widehat{\boldsymbol{\delta}})$ by employing the SO algorithm (Held *et al.*, 1974).

5.3. The Determination of Candidate Locations

The solutions of the approximating MILP formulations MDAP1 and MDAP2 do not guarantee an optimal solution for the MCMWP unless the set of candidate points includes the set of optimal locations as a subset. Actually, we try to solve a continuous nonconvex problem approximately by restricting optimal locations to be within a set of points rather

than within the entire convex hull of customer locations. As the number of candidate points increases, we expect that the objective value of MDAP becomes closer to the optimal value of the original MCMWP. Hence, we experience a trade-off between the quality of the solutions provided by MDAP1 and MDAP2 and the required computational effort. Even a small MILP problem may yield high-quality or optimal solutions by selecting a small but promising set of points as candidate facility locations. Now, we give two discretization strategies for the selection of candidate facility locations. The first strategy uses the block norms and the second one chooses customer locations for that purpose.

5.3.1. Discrete Approximation Using Block Norms

Theorem 15.2 of (Rockafellar, 1970) states that there is a one-to-one correspondence between the set of norms and the set of closed, bounded and convex sets which are symmetric and their interior contains the origin. Let $\mathcal{B}$ be such a set and then its corresponding norm $\|.\|_{\mathcal{B}}$ is defined as $\|\mathbf{x}\|_{\mathcal{B}} = \inf\{\varpi : \mathbf{x} \in \varpi\mathcal{B}, \varpi \geq 0\}$ with $\varpi\mathcal{B} \equiv \{\hat{\varpi}\hat{\mathbf{x}} : \forall \hat{\mathbf{x}} \in \mathcal{B}, \forall \hat{\varpi} \leq \varpi\}$. $\mathcal{B}$ is called the unit ball when $\mathcal{B} = \{\mathbf{x} : \|\mathbf{x}\|_{\mathcal{B}} \leq 1\}$. (Witzgall, 1964) defines a family of norms with polyhedral contours, called polyhedral norms, that generalize ℓ_1-norm (rectilinear norm), and recognizes their potential capability to model a wide variety of road travel distances. (Ward and Wendell, 1980) follow this research avenue and investigate a family of norms, called one-infinity norms, generalizing again the ℓ_1-norm and yielding distance approximations comparable to ℓ_r-norm. A one-infinity norm ($\ell_{1\infty}$-norm) is obtained by taking a nonnegative weighted sum of the ℓ_1 and ℓ_∞ (Tchebycheff) norms. In their succeeding work (Ward and Wendell, 1985) discuss the class of block norms and emphasize their generalization of the properties of the ℓ_1-norm. They define a block norm as a norm whose unit ball $\mathcal{B}$ is a polytope. In fact, any block norm can be represented by a symmetric polyhedral cone pointed at the origin whose extreme directions are the so-called fundamental directions of the block norm $\|.\|_{\mathcal{B}}$. Alternatively, the block norm can be viewed as a union of cones generated by the facets of $\mathcal{B}$ and the origin (Durier and Michelot, 1994). This interpretation includes the use of polar set $\mathcal{B}^0$ of $\mathcal{B}$ which is defined by $\mathcal{B}^0 = \left\{\mathbf{x}^0 : \forall \mathbf{x} \in \mathcal{B}, \mathbf{x}^T\mathbf{x}^0 \leq 1\right\}$. For more details on polar sets and their norms we refer to (Rockafellar, 1970), (Ward and Wendell, 1985) and (Durier and Michelot, 1994). ℓ_1 and ℓ_∞-norms are two well known examples of the block norms. They include the $\ell_{1\infty}$-norm and provide distance approximations almost as good as the round norms, which have round and smooth contours like the ℓ_r-norm for $1 < r < \infty$.

(Ward and Wendell, 1985) give a characterization of the block norms as the minimum distance to a point along the prescribed fundamental directions and it is possible to observe that this is equivalent to the characterization of the polyhedral norms by (Witzgall, 1964). Mathematically, the block norms can be characterized as

$$\|\mathbf{x}\|_{\mathcal{B}} = \min\left\{\sum_{g=1}^{\mathcal{G}}\varpi_g : \mathbf{x} = \sum_{g=1}^{\mathcal{G}}\varpi_g \boldsymbol{b}_g, \varpi_g \geq 0\right\} \tag{5.40}$$

where $\boldsymbol{b}_g$ are the fundamental directions of $\|.\|_{\mathcal{B}}$. The extreme points of the block norms also define a unit travel length in that direction.

Furthermore, (Ward and Wendell, 1985) also claim that block norms are dense in the set of all norms that every norm is either a block norm or a sequence of block norms can be found where their limit converges to that norm. It is shown by (Thisse *et al.*, 1984) that the block norms are linear over the cones generated by each facet of $\mathcal{B}$ and the origin. Any block norm can be represented by a symmetric polyhedral cone pointed at the origin whose extreme directions and their negatives are the so-called fundamental directions of the block norm. For example, the contours of the ℓ_1-norm are 45^o rotated squares centered at the origin and the four fundamental directions are the unit vectors and their negatives. The extreme rays can be represented by horizontal and vertical lines (i.e., x and y-axis) intersecting at the origin. The ℓ_∞-norm has also four fundamental directions but they are 45^o rotated because the contours are regular squares centered at the origin and the two extreme rays overlaps with the diagonals of the squares. As a consequence the fundamental directions are the vectors $\begin{pmatrix}1\\1\end{pmatrix}$, $\begin{pmatrix}-1\\1\end{pmatrix}$, $\begin{pmatrix}1\\-1\end{pmatrix}$ and $\begin{pmatrix}-1\\-1\end{pmatrix}$; the 45^o rotated unit vectors and their negatives. As for the contours of the $\ell_{1\infty}$-norm, they are octagons centered at the origin. There are four extreme rays which can be represented by four lines making respectively 0^o, 45^o, 90^o and 135^o angles with the x-axis and intersecting at the origin. They can be obtained by superposing the ones of the ℓ_1 and ℓ_∞-norms. Hence, there are eight fundamental directions, each defined by one of the extreme points of the polyhedral contours. Unit contours of ℓ_1, ℓ_∞, $\ell_{1\infty}$ and ℓ_r-norms including the Euclidean norm (i.e., ℓ_2-norm) are illustrated with Figure 5.1.

Theorem 6 of (Thisse *et al.*, 1984) implies that for a block norm $\|.\|_{\mathcal{B}}$ an optimal solution to the planar WP

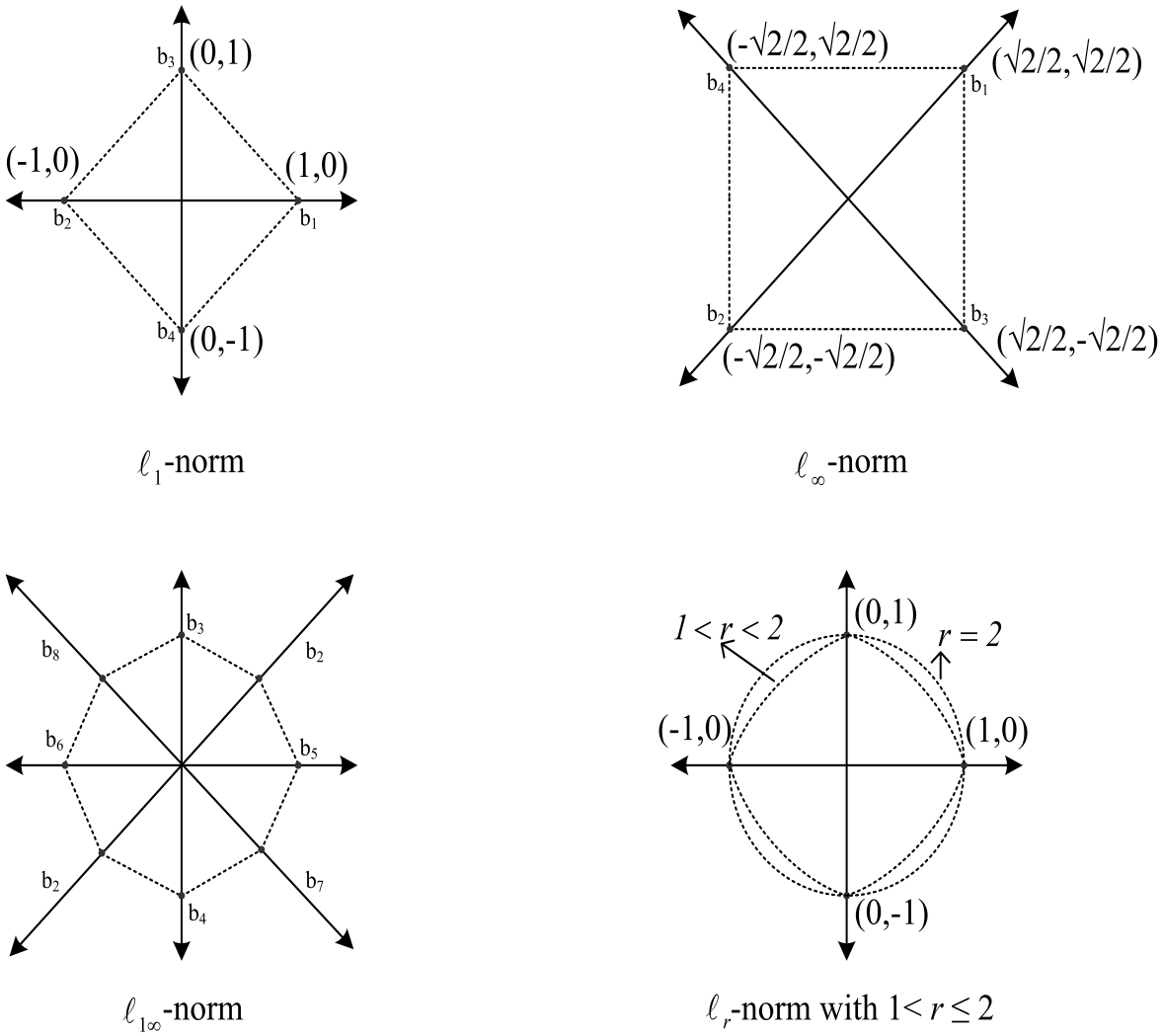

Figure 5.1. Unit balls of several norms.

$$\min \sum_{j=1}^{J} C_j \left(\|\mathbf{x} - \mathbf{a}_j\|_{\mathcal{B}} \right) \tag{5.41}$$

occurs at one of the intersection points of the extreme rays of the block norm $\|\cdot\|_{\mathcal{B}}$ passing through the customer locations and remaining within the convex hull of these locations. Here the functions $C_j(\cdot)$ are nondecreasing and concave. Stronger results on the local optima of Equation 5.41 are also derived in the more recent work of (Idrissi *et al.*, 1988). They are all generalizations of (Wendell and Hurter, 1973)'s early dominance results on the planar WP with ℓ_1-norm. According to this result, the set of points that include an optimal solution consists of the intersection points of the vertical and horizontal lines drawn through the customer locations and remaining within their convex hull. For the ℓ_∞-norm

these are the intersection points of the lines making 45^o and 135^o angles with the x-axis and passing through the customer locations where the extreme directions are $(\sqrt{2}/2, \sqrt{2}/2)$, $(\sqrt{2}/2, -\sqrt{2}/2)$ and their negatives. Similar results can be obtained by rotating the unit ball corresponding to the ℓ_1-norm around the origin by an angle $\vartheta \leq 45^o$ (for $\vartheta > 45^o$ one starts to obtain the same balls as in the case $\vartheta \leq 45^o$). The corresponding block norm distance equals $\ell_1/(\cos\vartheta + \sin\vartheta)$ on the ϑ rotated axis. The candidate location sets for the weighted ℓ_1 and ℓ_∞-norms are illustrated with Figure 5.2. The lines originating from customer locations (i.e., bold points with square frames) are the extreme rays of the ℓ_1 and ℓ_∞-norms. One can obtain the candidate location set for the $\ell_{1\infty}$-norm by simply superposing these two figures. As a verdict, it is possible to say that the higher is the number of fundamental vectors (which means a higher number of extreme points of the polyhedral contours and a higher number of extreme rays and directions), the larger is the set of candidate locations.

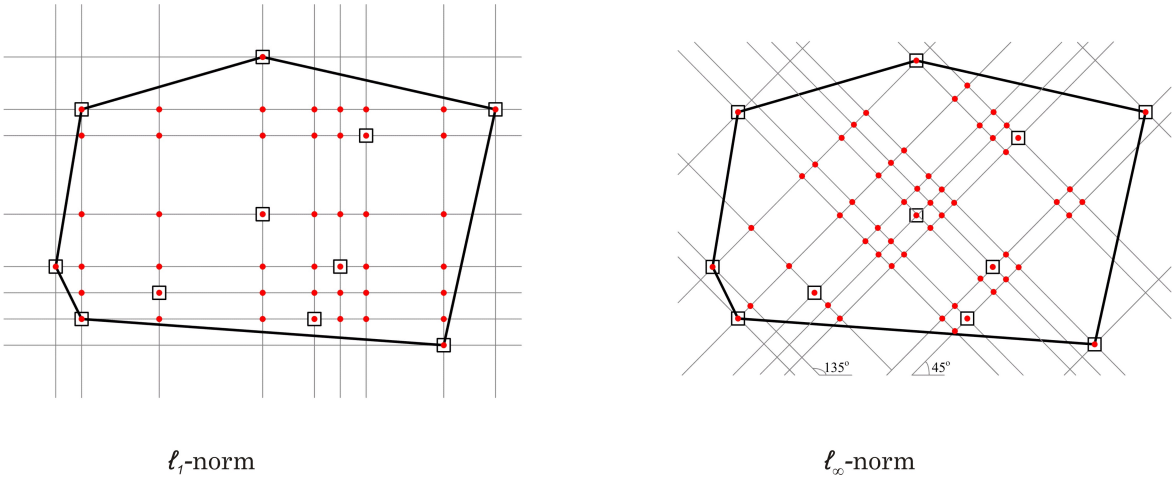

Figure 5.2. The candidate location sets for the WP with ℓ_1 and ℓ_∞-norms.

Observe that given a feasible set of allocations the MCMWP decomposes into I WPs with $C_j(\|\mathbf{x} - \mathbf{a}_j\|)$ where, given allocation quantities w_{ijk} for facility i, $C_j = \sum_{k=1}^{K} c_{ijk} w_{ijk}$ is a nonnegative constant. In particular, this observation enables the application of (Thisse *et al.*, 1984)'s results also for the MCMWP. In short, when the distance function is obtained by multiplying a block norm with a nonnegative weight, solving the approximations MDAP1 and MDAP2 over the intersection of the extreme rays passing through the customer locations and remaining within their convex hull, becomes equivalent to solving the MCMWP with that weighted block norm exactly. What is more, by the solution of MDAP1 and MDAP2, it is also possible to obtain approximate solutions of the MCMWP when the distances are measured with a weighted ℓ_r-norm, since it can be approximated by a block norm. Notice

that better approximations of the ℓ_r-norm require the use of block norms with more extreme rays, which increase the number of intersection points in the candidate set, and thus the size of MDAP1 and MDAP2. It should be pointed out that the approximating problems MDAP1 and MDAP2 enable also the computation of weaker lower bounds using their LP and Lagrangean relaxations.

As an additional property of the ℓ_1, ℓ_∞ and $\ell_{1\infty}$-norms we can list the inequalities

$$\ell_\infty \leq \ell_r, \tag{5.42}$$

$$2^{\frac{1-r}{r}}\ell_1 \leq \ell_r, \tag{5.43}$$

$$\frac{1}{2^{\frac{r-1}{r}}\varpi_1 + \varpi_2\sqrt{2}}\ell_{1\infty} \leq \ell_r, \tag{5.44}$$

for $1 \leq r < \infty$, and $\ell_{1\infty} = \varpi_1\ell_1 + \sqrt{2}\varpi_2\ell_\infty$ with nonnegative constants ϖ_1 and ϖ_2. The first one of them follows directly from the definition of ℓ_r and ℓ_∞-norms. The second one is a consequence of the well known mathematical inequality

$$\left(\sum_{n=1}^{N}\sigma_n\right)^r \leq N^{r-1}\left(\sum_{n=1}^{N}\sigma_n^r\right) \tag{5.45}$$

where σ_n is a positive constant and $r \geq 1$, which particularly implies

$$(|x_1| + |x_2|)^r \leq 2^{r-1}\left(|x_1|^r + |x_2|^r\right) \tag{5.46}$$

for $N = 2$ and $r \geq 1$ (Korovkin, 1986). Finally, the last inequality can be directly obtained using the first two inequalities (i.e., Equation 5.42 and 5.43) and the definition of the $\ell_{1\infty}$-norm.

5.3.2. Discrete Approximation Using Customer Locations

(Hansen *et al.*, 1998) take into account the observation that optimal facility locations are usually either on customer locations or very close to them in developing their accurate p-median heuristic for the MWP. Benefiting from this observation, namely by choosing the set of candidate facility locations as the set of customer locations, (Aras *et al.*, 2007) have designed a very accurate heuristic for the CMWP. Consequently, we are encouraged to use

the discrete approximations MDAP1 and MDAP2 over the customer locations in order to compute good upper bounds on the optimal value of the MCMWP.

5.4. Heuristics Using Discrete Approximations

Optimal values of an approximating MILP problem is clearly an upper bound on the optimal value of the MCMWP since it is a restriction of the original problem. The MDA heuristic is developed based on the desire of improving further this upper bound using an improvement heuristic. In short, MDA consists of the exact solution of an approximating MILP problem (i.e., MDAP1 and MDAP2) and a run of an improvement heuristic initialized at one of the optimal or good feasible solutions obtained with the approximating MILP problem. As the improvement heuristic, we prefer to use MCALA starting at the facility locations obtained with the optimal solution of MDAP1 or MDAP2.

The implication of the inequalities given by Equation 5.42 – 5.44 on the optimal value of the MCMWP can be summarized by the inequalities

$$Z_{\infty}^{*} \leq Z_{r}^{*}, \tag{5.47}$$

$$2^{\frac{1-r}{r}} Z_{1}^{*} \leq Z_{r}^{*}, \tag{5.48}$$

$$\frac{1}{2^{\frac{r-1}{r}} \varpi_1 + \varpi_2 \sqrt{2}} Z_{1\infty}^{*} \leq Z_{r}^{*}, \tag{5.49}$$

where Z_r^* is the optimal value of the ℓ_r-norm MCMWP for $1 \leq r < \infty$, Z_1^*, Z_∞^* and $Z_{1\infty}^*$ stand for the optimal value of the ℓ_1, ℓ_∞ and $\ell_{1\infty}$-norm MCMWPs, respectively. Since MDAP1 and MDAP2 are equivalent to MCMWP when distance $d(\mathbf{x}_i, \mathbf{a}_j)$ is modeled by the ℓ_1, ℓ_∞ and $\ell_{1\infty}$-norms, and the candidate locations are the set of the intersection points determined as explained in Section 5.3.1, the optimal value can be obtained by solving MDAP1 or MDAP2 formulations. Hence, for these three particular norms, the optimal values of MDAP1 and MDAP2 can also be used to obtain lower bounds on Z_r^* for $1 \leq r < \infty$ using inequalities given by Equation 5.47 – 5.49. These lower bounds become weaker if LP and Lagrangean relaxations of MDAP1 or MDAP2 are solved and the relaxed optimal value is used to determine the lower bounds. However, we can not repeat the same property for the situation where the customer locations are used as the candidate locations. The lower bounds computed by a relaxation strategy on the optimal value of MDAP1 or MDAP2

can not be used to determine a lower bound on Z_r^*, since there is no Equation 5.47 – 5.49 type obvious relations between the optimal values of MDAP1, MDAP2, and Z_r^*, when the customer locations are selected as the candidate facility sites.

The way we use MCALA in the relaxed version of the MDA heuristic is different than the one of its use in MDA. At every step of the SO algorithm, prior to the multiplier updates, C-MCALA is run as a feasibility heuristic to obtain a good feasible solution of the MCMWP (not the approximating MILP problem), in order to update the upper bound on the optimal value of MCMWP: the upper bound is set to the minimum of the new and existing ones. The upper bound used in the SO algorithm for updating the Lagrange multipliers (i.e., the upper bound on the optimal value of the approximating MILP problem) is computed by means of the D-MCALA: the single facility location problems are simply 1-median problems solved over the candidate points.

We should remind that the Lagrangean lower bound SO algorithm computes, is a lower bound on the optimal value of the approximating MILP problem, but not necessarily a lower bound on the optimal value of the MCMWP. This is only true for the approximating MILP formulations which are equivalent to the MCMWP. Therefore, discrete approximation heuristics using MDAP1 and MDAP2 based on ℓ_1, ℓ_∞ and $\ell_{1\infty}$-norms (ℓ_1-MDA1, ℓ_∞-MDA1, $\ell_{1\infty}$-MDA1, ℓ_1-MDA2, ℓ_∞-MDA2 and $\ell_{1\infty}$-MDA2), and their relaxed versions obtained through the Lagrangean relaxations of MDAP1 and MDAP2 (ℓ_1-RMDA1, ℓ_∞-RMDA1, $\ell_{1\infty}$-RMDA1, ℓ_1-RMDA2, ℓ_∞-RMDA2 and $\ell_{1\infty}$-RMDA2) can be used to compute both lower and upper bounds on Z_r^* for $1 \leq r < \infty$. However, this is not true for the DA heuristics based on customer locations (CL-MDA1, CL-MDA2) and their relaxed versions (CL-RMDA1, CL-RMDA2). They can only be used to compute upper bounds.

Finally, we should point out that the MILP formulations used in a MDA heuristic affects only its efficiency. In other words, two versions of the approximating heuristics using the exact solutions of MDAP1 and MDAP2 have the same accuracy since both of them are equivalent formulations. However, this is not necessarily true for the relaxed versions of the DA heuristics since MDAP1 and MDAP2 have different Lagrangean subproblems resulting in different Lagrangean lower bounds and different initial facility locations for the MCALA heuristic.

6. USING LAGRANGEAN RELAXATION AND A MODIFIED SUBGRADIENT ALGORITHM

In this chapter[3] a Lagrangean Relaxation (LR) scheme is proposed for the MCMWP together with novel acceleration strategies which can also be applied to many other optimization problems with intractable Lagrangean subproblems. For that purpose, we relax both constraints given by Equation 2.3 and 2.5 which results in a variant of the MWP with multiple commodities. In the MWP, each customer is served only from the least weighted cost facility. Thus, the set of customers can be separated into I distinct nonempty subsets each of which is assigned to a single facility. Indeed, it is possible to find the optimal solution by generating all such partitions of the customer set. As discussed in Chapter 3, (Rosing, 1992) considers the MWP with unit transportation costs and developed a modified Set Covering (SC) problem formulation to solve it. The method suggested by (Rosing, 1992) consists of generating all customer subsets with non-intersecting convex hulls and then each convex hull, which is interpreted as a column, is added to the SC problem formulation. Although Rosing's method halts quickly, (Krau, 1997) uses this SC formulation and develops a CG procedure combined with a Branch-and-Price (BP) algorithm to solve the MWP exactly. The method by (Krau, 1997) generates customer subsets instead of their convex hulls as the method by (Rosing, 1992) does. (Krau, 1997) solves a Concave Minimization (CM) problem for the Pricing Subproblem (PS). Later, (Righini and Zaniboni, 2007) replace the solution method of the PS by a polynomial time algorithm developed in the study by (Drezner *et al.*, 1991) for the solution of WP with limited distances (WPLD). Both approaches by (Krau, 1997) and (Righini and Zaniboni, 2007) are efficient methods for the exact solution of the MWP. In short, existing exact methods for the MWP can be adapted for the MWP variant which arises when constraints given by Equation 2.3 and 2.5 of the MCMWP are relaxed. We should note that it may seem interesting to relax only constraints given by Equation 2.5 from the MCMWP with the purpose to obtain K CMWP variants for each commodity. However, the facility location variables $\mathbf{x}_i$ in $d(\mathbf{x}_i, \mathbf{a}_j)$, are common for all commodities. Therefore, relaxing only constraints given by Equation 2.5 does not yield a decomposition over the commodities and the resulting LR subproblem is not easier to solve than the original MCMWP. Let φ_{ik} and μ^3_{ij} be the Lagrangean multipliers associated with constraint sets

[3]The article by (Akyüz *et al.*, 2011) and the conference proceeding by (Akyüz *et al.*, 2010b) are partially based on this chapter.

given by Equation 2.3 and 2.5, respectively. Then we obtain the following LR subproblem:

LR3($\boldsymbol{\varphi}, \boldsymbol{\mu}^3$):

$$\min \quad Z_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} w_{ijk}\left(c_{ijk}d(\mathbf{x}_i, \mathbf{a}_j) + \varphi_{ik} + \mu_{ij}^3\right) - \xi \tag{6.1}$$

$$\text{s.t.} \qquad \text{Equation 2.4 and 2.6} \tag{6.2}$$

where $\xi = \sum_{i=1}^{I}\sum_{k=1}^{K}\varphi_{ik}s_{ik} + \sum_{i=1}^{I}\sum_{j=1}^{J}\mu_{ij}^3 u_{ij}$ is a constant term which can be ignored from the LR subproblem and added to the optimal value afterwards. Let the binary variable w'_{ijk} be equal to 1 if and only if the demand of customer j for commodity k is met by facility i. When we substitute w_{ijk} with $w'_{ijk}q_{jk}$ and define $d_k(\mathbf{x}_i, \mathbf{a}_j) = (c_{ijk}d(\mathbf{x}_i, \mathbf{a}_j) + \varphi_{ik} \ + \mu_{ij}^3)$, the LR subproblem given by Equation 6.1 – 6.2 reduces to

LR3($\boldsymbol{\varphi}, \boldsymbol{\mu}^3$):

$$\min \quad Z_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} w'_{ijk}q_{jk}d_k(\mathbf{x}_i, \mathbf{a}_j) - \xi \tag{6.3}$$

$$\text{s.t.} \quad \sum_{i=1}^{I} w'_{ijk} = 1 \qquad j = 1, \ldots, J; k = 1, \ldots, K, \tag{6.4}$$

$$w'_{ijk} \in \{0, 1\} \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K. \tag{6.5}$$

Notice that LR3($\boldsymbol{\varphi}, \boldsymbol{\mu}^3$) given by Equation 6.3 – 6.5 is a variant of the MWP where the distance between facility i and customer j is calculated with a particular distance function $d_k(\mathbf{x}_i, \mathbf{a}_j)$ for each commodity type k. Best lower bounds can be found by solving $LR3(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ multiple times with different Lagrange multiplier values which are adjusted within a SO algorithm (Held *et al.*, 1974). In fact, we suggest a Modified Subgradient (MS) algorithm which differs from the classical SO algorithm in the computation of lower bounds and multiplier updates. We discuss the MS algorithm in detail in the following sections including its convergence properties. Now, we give a brief summary for it. Its formal outline is presented at the end of this chapter. In the initialization step of the MS algorithm the Lagrangean multipliers φ_{ik} and μ_{ij}^3 are set to 0. At the beginning, the best lower bound value Z_{LB}^{best} is set to $-\infty$ and the best upper bound value Z_{UB}^{best} is set to ∞. Then a lower bound Z_{LB} is found

on the optimum value Z^*, by solving the LR subproblem with current Lagrange multipliers φ_{ik} and μ^3_{ij}. In case $Z_{LB} > Z^{best}_{LB}$ holds we set $Z^{best}_{LB} = Z_{LB}$. Next, an upper bound Z_{UB} on Z^* is found. When $Z_{UB} < Z^{best}_{UB}$ holds we set $Z^{best}_{UB} = Z_{UB}$. Given the current lower and upper bound values we update Lagrange multipliers φ_{ik} and μ^3_{ij}. The MS algorithm stops when a predetermined number of iterations is performed or the step size parameter π becomes sufficiently small. Finally, the MS algorithm outputs Z^{best}_{LB} and Z^{best}_{UB}. We should also point out that a similar MS algorithm is previously suggested by (Boyacı, 2009) for the CMWP. The upper bounds are determined by the MCALA heuristic. In order to produce upper bounds on the optimal value of MCMWP, we first modify the initialization of the MCALA. That is to say, we solve I WPs to obtain initial facility location values $\mathbf{x}_i$ by using the optimal assignment obtained from the solution of the subproblem $LR3(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$. Each initial $\mathbf{x}_i$ is computed with respect to customer set assigned to facility i obtained from the optimal assignment $LR3(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ determines. Then, the MCALA heuristic solves alternately the allocation (i.e., the MTP) and location (i.e., I WPs) problems until no improvement is possible. It is clear that the final locations and allocations form a feasible solution for the MCMWP and thus their value is an upper bound on the optimal value of the MCMWP.

Recall that the lower bound value Z_{LB} computed during the run of the MS algorithm can be either the optimal value of the LR subproblem $Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ or a lower bound satisfying $Z_{LB} \leq Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ for any choice of multiplier vectors $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$. In order to solve the Lagrangean subproblem given by Equation 6.3 – 6.5, we first give its equivalent SC problem formulation which is obtained by partitioning the customer set into subsets for each facility. Then, the LP relaxation of the SC formulation (namely, SCLP) is solved by a CG procedure for this SC problem formulation. We also sketch a BP approach that can be used to solve the SC formulation exactly. Note that the solution of the SCLP yields a lower bound on $Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ and we suggest two different solution procedures for solving the PSs to generate the columns. We also devise an approximating MILP formulation for the LR subproblem and use two more lower bounds which employ block norm based approximations. These are ℓ_1 and ℓ_∞-norms to determine lower bounds as discussed in Chapter 5 for the DA heuristics. The only difference is the solution of an approximating MILP formulation for the MWP variant which substitutes the solution of the MDAP. In short, there are basically four lower bounding approaches for the Lagrangean subproblem. All of them require excessive CPU times and they are executed only once at the end or when necessary. This is accomplished by using heuristic solutions, which are treated as if they are lower bounds on the LR sub-

problem within the MS algorithm in order to increase the efficiency of the overall algorithm. Actually, heuristic solution of the LR subproblem is only an upper bound on it. As a result, our algorithm turns into a modified subgradient heuristic search procedure in the dual space of the Lagrangean multipliers. When the MS algorithm converges, the best Lagrange multipliers are used to compute a lower bound on the MCMWP. Furthermore, the efficiency of the MS algorithm increases by accelerating the solution procedures of the PS.

6.1. Equivalent Set Covering Formulation for the Lagrangean Subproblem

In an optimal solution of the MWP variant given by Equation 6.3 – 6.5, each customer of a commodity is served by exactly one facility which has the minimum weighted cost. Note that as there are no capacity restrictions left on the facilities, each commodity k of a customer j can be treated as one of the K distinct customers each having identical locations. We call them as customer-commodity pairs (j,k) to indicate customer j of commodity k. Hence, it is required to divide the customer-commodity pair set into I subsets such that each subset is covered by exactly one facility. Keeping this in mind, we define the following binary decision variables. Let o_p be equal to 1 if and only if subset p is selected. Let $\mathbf{b}^p = \left\{b^p_{11}, b^p_{12}, \ldots, b^p_{1K}, b^p_{21}, b^p_{22}, \ldots, b^p_{2K}, \ldots b^p_{J1}, b^p_{J2}, \ldots, b^p_{JK}, b^p_{JK+1}, b^p_{JK+2}, \quad \ldots, b^p_{JK+I}\right\}^T$ denote column p where each entry takes 0 or 1 values. The first $J \times K$ entries of $\mathbf{b}^p$ denote the customer-commodity pairs and the last I entries stand for the corresponding facility of column p. If a customer j of commodity k is served from a subset denoted by $\mathcal{J}_p$ then the corresponding element b^p_{jk} is set to 1, otherwise it is set to 0. Similarly, when $\mathcal{J}_p$ is a subset of facility i^*, then $b^p_{JK+i^*} = 1$ and $b^p_{JK+i} = 0$ hold for $i = 1, \ldots, I$ with $i \neq i^*$. An equivalent SC problem formulation of the MWP variant given by Equation 6.3 – 6.5 is as follows.

SC($\boldsymbol{\varphi}, \boldsymbol{\mu}^3$):

$$\min \quad Z_{SC}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) = \sum_{p=1}^{P} c_p o_p - \xi \tag{6.6}$$

$$\text{s.t.} \quad \sum_{p=1}^{P} b^p_{jk} o_p \geq 1 \qquad j = 1, ..., J; k = 1, ..., K, \tag{6.7}$$

$$\sum_{p=1}^{P} b^p_{JK+i} o_p \leq 1 \qquad i = 1, ..., I, \tag{6.8}$$

$$o_p \in \{0, 1\} \qquad p = 1, ..., P, \tag{6.9}$$

where $P = I \times (2^{J\times K} - 1)$ denotes the number of all possible subsets. The cost coefficient $c_p = \min_{\mathbf{x}_i} \sum_{j=}^{J} \sum_{k=1}^{K} b_{jk}^{p} q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j)$ of each customer subset $\mathcal{J}_p$ for a given facility i is computed by solving a WP via Weiszfeld's algorithm or its generalizations. We should note that the applicability of the Weiszfeld's algorithm for the ℓ_r distance WP with $1 \leq r \leq 2$ is shown by (Brimberg and Love, 1993). Fortunately, the modified distance function $d_k(\mathbf{x}_i, \mathbf{a}_j) = \left(c_{ijk} d(\mathbf{x}_i, \mathbf{a}_j) + \varphi_{ik} \ + \mu_{ij}^{3}\right)$ does not change the derivative information of the ℓ_r distance. As a result, the constant Lagrange multiplier terms do not effect the formula on which the Weiszfeld's algorithm is based.

Constraints given by Equation 6.7 guarantee that each customer j of commodity k is covered by (served from) at least one subset (facility). Constraints given by Equation 6.8 state that the number of subsets served from a facility i is at most 1. This implies that each facility can be opened at most once. Hence, the total number of customer-commodity pair subsets are enforced to be at most the total number of facilities I. Note that an equivalent set partitioning problem formulation can also be proposed by replacing inequalities in Equation 6.7 and 6.8 with equalities. However, inequalities in Equation 6.7 and 6.8 do not harm the optimality since each customer-commodity pair should be served by exactly one facility and exactly I facilities should be opened at optimality when $J \geq I$ holds which follows as a generalization of a result by (Drezner, 1984).

6.1.1. Column Generation Procedure

We consider the SCLP, which is obtained by replacing integrality constraints given by Equation 6.9 with $o_p \geq 0$ for $p = 1, \ldots, P$. The CG procedure employs the dual variable information to generate necessary columns which solves the PS. At each step of the CG procedure, an optimal dual solution is employed to solve a PS and to generate additional columns with negative reduced cost. Initialized with a feasible set of columns, the relaxed problem is iteratively solved until it is not possible to find a column with negative reduced cost. Considering the MWP variant given by Equation 6.3 – 6.5, let λ_{jk} and ω_i be the dual variables associated with constraints given by Equation 6.7 and 6.8, respectively. Then, the dual problem DP of the SCLP can be given as the following.

DP:

$$\max \quad Z_{DP} = \sum_{j=1}^{J}\sum_{k=1}^{K}\lambda_{jk} - \sum_{i=1}^{I}\omega_i \tag{6.10}$$

$$\text{s.t.} \quad \sum_{j=1}^{J}\sum_{k=1}^{K}b_{jk}^{p}\lambda_{jk} - \sum_{i=1}^{I}b_{JK+i}^{p}\omega_i \leq c_p, \quad p = 1, \ldots, P, \tag{6.11}$$

$$\lambda_{jk} \geq 0, \qquad j = 1, \ldots, J; k = 1, \ldots, K, \tag{6.12}$$

$$\omega_i \geq 0, \qquad i = 1, \ldots, I. \tag{6.13}$$

We should mention that the initial column set is constructed by generating all columns where a facility serves only a single customer-commodity pair for each facility. This makes a total of $I \times J \times K$ columns which are added to the SCLP and the corresponding dual variables are calculated. Let $\widehat{c}^{*}_{p(i)}$ denote the minimum reduced cost which corresponds to facility i. Then, the PS is as follows.

PS:

$$\widehat{c}^{*}_{p(i)} = \min_{\mathbf{x}_i, \mathbf{b}^p} Z^{i}_{PS}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) = \sum_{j=1}^{J}\sum_{k=1}^{K} b_{jk}^{p}\left\{q_{jk}d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk}\right\} + \omega_i \tag{6.14}$$

$$\text{s.t.} \qquad b_{jk}^{p} \in \{0, 1\} \qquad j = 1, ..., J; k = 1, ..., K. \tag{6.15}$$

At each step of the CG procedure, we solve the PS for each facility i and select the columns which yield the negative reduced cost. All columns generated by using the solution of the PS are added to the column set of SCLP. We now present two alternative approaches for the solution of the PS.

6.1.1.1. Pricing by D.C. Programming. The reduced cost expression given by Equation 6.14 – 6.15 can be transformed into a D.C. (i.e., difference of two convex functions) form as follows.

$$\begin{aligned}\widehat{c}^{*}_{p(i)} &= \min_{\mathbf{x}_i, \mathbf{b}^p}\left\{\sum_{j=1}^{J}\sum_{k=1}^{K} b_{jk}^{p}\left\{q_{jk}d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk}\right\} + \omega_i : b_{jk}^{p} \in \{0, 1\}, j = 1, ..., J; k = 1, ..., K\right\} \\ &= \min_{\mathbf{x}_i}\left\{\sum_{j=1}^{J}\sum_{k=1}^{K} \min\left\{q_{jk}d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk}, 0\right\} + \omega_i\right\}\end{aligned}$$

$$= \min_{\mathbf{x}_i} \left\{ \left\{ \sum_{j=1}^{J} \sum_{k=1}^{K} (q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk}) - \sum_{j=1}^{J} \sum_{k=1}^{K} \max\{q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk}, 0\} \right\} + \omega_i \right\} \quad (6.16)$$

Then, the problem given by Equation 6.16 can be reduced to a CM problem by using the additional variable h as

CM:

$$\min_{\mathbf{x}_i, h} \; Z^i_{CM}(\mathbf{x}_i, h) = h - \sum_{j=1}^{J} \sum_{k=1}^{K} \max\{q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk}, 0\} - \sum_{j=1}^{J} \sum_{k=1}^{K} \lambda_{jk} + \omega_i \quad (6.17)$$

$$\text{s.t.} \quad \sum_{j=1}^{J} \sum_{k=1}^{K} q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) \leq h. \quad (6.18)$$

The CM problem given by Equation 6.17 – 6.18 can be solved by an Outer Approximation (OA) algorithm (Chen *et al.*, 1998), which is run as long as it outputs columns with negative reduced cost. A similar approach is also proposed by (Krau, 1997) to solve the PS arising for the MWP case.

6.1.1.2. Pricing by Solving the Weber Problem with Limited Distances. The PS given by Equation 6.14 – 6.15 can be considered as WPLD. In this variant of the WP, facilities are located within a threshold distance from the customers. Solving the WPLD is equivalent to producing all columns explicitly. Column p of facility i is generated by setting $b^p_{jk} = 1$ when $q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk} = q_{jk} \left(c_{ijk} d(\mathbf{x}_i, \mathbf{a}_j) + \varphi_{ik} + \mu^3_{ij} \right) - \lambda_{jk} < 0$ holds, otherwise we set $b^p_{jk} = 0$ for $j = 1, \ldots, J; k = 1, \ldots, K$. Also, we set $b^p_{JK+i^*} = 1$ by which the corresponding facility of column p is determined and $b^p_{JK+i} = 0$ for $i = 1, \ldots, I$ and $i \neq i^*$. Note that the PS can be geometrically interpreted as finding the location of facility i, which lies on the intersection of circles whose centers are customer locations with a radius of $\tau_{ijk} = \max\left\{ (\frac{\lambda_{jk}}{q_{jk}} - \varphi_{ik} - \mu^3_{ij})/c_{ijk}, 0 \right\}$ for $j = 1, ..., J; k = 1, ..., K$. When the location of facility i is in the intersection of the circles, we can get a solution with negative reduced cost value for the PS. Therefore, it suffices to consider only the intersection points of the circles to find a negative reduced cost column which corresponds to an intersecting region.

(Drezner *et al.*, 1991) show that the number of distinct intersecting regions are bounded by $2JK(JK-1)$ for the WPLD and propose a polynomial time algorithm to optimally solve it. This polynomial time algorithm, which we term as the Feasible Subset Generation (FSG) algorithm, is employed by (Righini and Zaniboni, 2007) within a CG procedure developed for the MWP. We also use the FSG algorithm as a subroutine to solve the PS given by Equation 6.14 – 6.15.

The FSG algorithm is run for each facility i and outputs a feasible subset (column) list $\mathcal{P}_i$ for $i = 1, ..., I$. It is initialized with parameters $\lambda_{jk}, q_{jk}, \varphi_{ik}, \mu^3_{ij}$ and c_{ijk}. As the first step, radii τ_{ijk} drawn on the customer locations are calculated. Then, the FSG algorithm performs the following steps for all customer pairs j' and j'' and commodity pairs k' and k'' for $j' = 1, \ldots, J-1$; $j'' = j'+1, \ldots, J$; $k', k'' = 1, \ldots, K$. The FSG algorithm fixes a point $\overline{\mathbf{a}}$ which lies on the intersection of two circles whose centers are located on the customers j' and j'' for commodities k' and k'', respectively. Without loss of generality, we can denote the centers of these circles as (j', k') and (j'', k''). All circles covering the intersection point $\overline{\mathbf{a}}$ other than the circles whose center are located on (j', k') and (j'', k'') are used to construct a subset $\mathcal{J}$. In other words, the subset $\mathcal{J}$ is constructed of all circles covering $\overline{\mathbf{a}}$ whose centers are located on the point (j''', k''') such that $(j''', k''') \neq (j', k')$ and $(j''', k''') \neq (j'', k'')$ for $j' = 1, \ldots, J-1$; $j'' = j'+1, \ldots, J$; $j''' = 1, \ldots, J$; $k', k'', k''' = 1 \ldots K$. Given the feasible subset $\mathcal{J}$, additional subsets $\mathcal{J}_1 = \{\mathcal{J} \cup (j', k')\}$, $\mathcal{J}_2 = \{\mathcal{J} \cup (j'', k'')\}$ and $\mathcal{J}_3 = \{\mathcal{J} \cup (j', k') \cup (j'', k'')\}$ are generated as well. Then, subsets $\mathcal{J}$, $\mathcal{J}_1$, $\mathcal{J}_2$ and $\mathcal{J}_3$ are added to the feasible subset list $\mathcal{P}_i$. Finally, the FSG algorithm outputs $\mathcal{P}_i$ and stops. For each facility i, the FSG algorithm generates all possible subsets with negative reduced cost. Consequently, we run the FSG algorithm I times and select subsets with negative reduced costs among the elements of $\mathcal{P} = \bigcup_{i=1}^{I} \mathcal{P}_i$ and we add them to the column set of SCLP. The formal outline of the FSG algorithm is given in Figure 6.1.

6.1.2. Branch-and-Price Procedure

The optimum solution of the SCLP can be an integer or a fractional solution. In the former case, we can deduce that the solution is also optimal for the SC problem. Otherwise, a fractional solution to the SCLP implies that an optimal solution of the SC problem has not been reached yet. One can resort to a BP algorithm to obtain the exact solution of the SC

1. Set $\tau_{ijk} := \max\left\{\left(\frac{\lambda_{jk}}{q_{jk}} - \varphi_{ik} - \mu_{ij}^3\right)/c_{ijk}, 0\right\}$, $j = 1, ..., J; k = 1, ..., K$ and $\mathcal{P}_i \leftarrow \emptyset$.
2. For each pair (j', k'), $j' = 1, \ldots, J-1$; $k' = 1, \ldots, K$; and (j'', k''), $j'' = j'+1, \ldots, J; k'' = 1, \ldots, K$, set $\mathcal{J} \leftarrow \emptyset$,
   if $d(\mathbf{a}_{j'}, \mathbf{a}_{j''}) < \tau_{ij'k'} + \tau_{ij''k''}$, **then** construct a set $\mathcal{F}$ of intersection points $\overline{\mathbf{a}}$, i.e., $\mathcal{F} = \{\overline{\mathbf{a}}_1, \overline{\mathbf{a}}_2\}$ by using $\mathbf{a}_{j'}, \tau_{ij'k'}, \mathbf{a}_{j''}$ and $\tau_{ij''k''}$.
   For each point $\overline{\mathbf{a}} \in \mathcal{F}$ and each pair (j''', k'''), $j''' = 1, \ldots, J$; $k''' = 1, \ldots, K$,
   if $(j', k') \neq (j'', k'') \neq (j''', k''')$ and $d(\overline{\mathbf{a}}, \mathbf{a}_{j'''}) < \tau_{ij'''k'''}$, **then** set $\mathcal{J} \leftarrow \mathcal{J} \cup (j''', k''')$.
   Set $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup \mathcal{J} \cup \{\mathcal{J} \cup (j', k')\} \cup \{\mathcal{J} \cup (j'', k'')\}$.
3. Report feasible subset (column) list $\mathcal{P}_i$ for facility i.

Figure 6.1. The FSG algorithm.

problem by using a fractional solution of the SCLP. For that purpose, a branching scheme developed by (Ryan and Foster, 1981) for the crew scheduling problem can be applied within the BP algorithm for the MWP variant.

Let j' and j'' be two customers and k' and k'' be two commodities. The branching scheme devised by (Ryan and Foster, 1981) is based on partitioning the customer subsets into two branches which are denoted by $\mathscr{B}^0$ and $\mathscr{B}^1$. In $\mathscr{B}^0$, two customer-commodity pairs $\{(j', k'), (j'', k'')\}$ are either included in a subset or none of them are included in this subset, i.e., the customer-commodity pairs which satisfy $b^p_{j'k'} = b^p_{j''k''} \in \{0, 1\}$. In $\mathscr{B}^1$, only one or none of the customer-commodity pair $\{(j', k'), (j'', k'')\}$ is included in a subset, i.e., the customer-commodity pairs which satisfy $b^p_{j'k'} + b^p_{j''k''} \leq 1$. The fractional columns can have several customer-commodity pairs either in common or not. The customer-commodity pairs which are not covered by all fractional columns are subject to branching.

In order to select the branching customer-commodity pair, the set of fractional columns is divided into two subsets. In one subset, each column covers both or none of the customer-commodity pairs. In the other subset, each column covers exactly one of the customer-commodity pairs. The branching customer-commodity pair can then be selected by considering the smallest difference between the sum of these two subsets of fractional solution

values for all pairs as mentioned in (Righini and Zaniboni, 2007).

During the run of the BP algorithm we have to solve a modified PS considering the customer-commodity pairs within the subsets. The FSG algorithm can be easily adapted for the BP. However, the reduction of the PS into a CM problem requires additional efforts. In practice, one should also penalize the columns which do not satisfy current branchings of the customer-commodity pairs. The BP algorithm that we have sketched out requires excessive CPU times. Even for the CMWP, namely for the single commodity case, the BP algorithm by (Boyacı, 2009) is unable to solve large instances. We skip the details of its implementation here. In order to produce a lower bound on the MCMWP, the exact solution of the LR subproblem is not absolutely required. Indeed, any lower bound Z_{LB} which satisfies $Z_{LB} \leq Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ can also be employed within the SO algorithm. Therefore, we prefer to solve the SCLP since the lower bound obtained with its solution, say $Z^*_{SCLP}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$, satisfies $Z^*_{SCLP}(\boldsymbol{\varphi}, \boldsymbol{\mu}) \leq Z^*_{SC}(\boldsymbol{\varphi}, \boldsymbol{\mu}) = Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^*$.

6.2. Using Discrete Approximations for the Lagrangean Subproblem

The LR subproblem given by Equation 6.3 – 6.5 can also be approximated by a MILP formulation, namely the UDAP (i.e., the Uncapacitated DAP) which can be solved to optimality with the purpose of producing lower bounds on the objective value of the LR subproblem. This can be achieved when a lower bounding norm function is used instead of $d(\widehat{\mathbf{a}}_g, \mathbf{a}_j)$ where $\widehat{\mathbf{a}}_g$ is the two dimensional location vector of a given candidate point g. The idea is the same as discussed in Chapter 5 where DA lower bounds are given on the MCMWP. Here, the MDAP is replaced with UDAP; it is the approximation of the LR subproblem that is a MWP variant. These lower bounds are also lower bounds on the optimal value of the MCMWP. Let the binary variable $\widetilde{y}_{ijkg}$ be equal to 1 if and only if the demand of customer j for commodity type k is met by facility i located at candidate location g and let the binary variable $\widetilde{v}_{ig}$ be equal to 1 if and only if a facility i is opened at candidate location g. Then, the UDAP formulation which can be used to produce lower bound on the LR subproblem can be stated as follows.

UDAP:

$$\min \quad Z_{UDAP}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G} q_{jk}\widetilde{c}_{ijkg}\widetilde{y}_{ijkg} - \xi \tag{6.19}$$

$$\text{s.t.} \quad \sum_{i=1}^{I}\sum_{g=1}^{G}\widetilde{y}_{ijkg} = 1 \qquad j = 1, ..., J; k = 1, ..., K, \tag{6.20}$$

$$\sum_{g=1}^{G}\widetilde{v}_{ig} = 1 \qquad i = 1, ..., I, \tag{6.21}$$

$$\widetilde{y}_{ijkg} \leq \widetilde{v}_{ig} \qquad i = 1, ..., I; j = 1, ..., J; k = 1, ..., K; g = 1, ..., G, \tag{6.22}$$

$$\widetilde{y}_{ijkg} \in \{0, 1\} \qquad i = 1, ..., I; j = 1, ..., J; k = 1, ..., K; g = 1, ..., G, \tag{6.23}$$

$$\widetilde{v}_{ig} \in \{0, 1\} \qquad i = 1, ..., I; g = 1, ..., G, \tag{6.24}$$

where $\widetilde{c}_{ijkg} = c_{ijk}d(\widehat{\mathbf{a}}_g, \mathbf{a}_j) + \varphi_{ik} + \mu^3_{ij}$. Besides, constraints given by Equation 6.21 ensure that a facility i is opened only on one candidate point g and constraints given by Equation 6.22 enforce that facility i located on a candidate point g could serve a customer only if it is opened on the corresponding candidate point.

Notice that the rectilinear distance MWP can be solved to optimality by UDAP when the candidate facility locations are selected as the intersection points of vertical and horizontal lines drawn on customer locations. Other block norms can also be employed within the objective function of the UDAP in order to obtain an approximating solution of the MWP with the ℓ_r distance function for $1 \leq r < \infty$. However, a block norm $\|.\|_{\mathcal{B}}$ can only guarantee a lower bound on the MWP when $\|\mathbf{x}\|_{\mathcal{B}} \leq \|\mathbf{x}\|_r$ is satisfied for any vector $\mathbf{x}$ with $1 \leq r < \infty$ for some r. Given multiplier vectors $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$, let $Z^*_{UDAP_{\mathcal{B}}}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ be the optimal value of the UDAP when a block norm $\mathcal{B}$ is used and let $Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ be the optimum value of the MWP with ℓ_r-norm for $1 \leq r < \infty$. When $\|.\|_{\mathcal{B}} \leq \|.\|_r$ is satisfied for some r with $1 \leq r < \infty$, then $Z^*_{UDAP_{\mathcal{B}}}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ also holds. Consequently, we can use the ℓ_∞-norm and ℓ_1-norm within the objective function of the UDAP formulation. The inequalities given by Equation 5.42 and 5.43 also hold for the UDAP formulation. Then, $Z^*_{UDAP_\infty}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ and $2^{\frac{1-r}{r}} Z^*_{UDAP_1}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^*_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ are also satisfied. In addition to the CG based lower bounds, we use the ℓ_1 and ℓ_∞-norms to produce two other lower bounds on the LR subproblem within the MS algorithm.

6.3. Increasing the Efficiency of the Lagrangean Relaxation Scheme

We present two acceleration strategies which increase the efficiency of the LR scheme. The first strategy is the use of heuristic bounds within the MS algorithm. The second one addresses the acceleration of the CG procedure which requires the solution of the PS. For that purpose, we show two different schemes one of which can be used with the DC optimization approach and the other one can be employed with the FSG algorithm when we generate the columns.

6.3.1. Using Heuristic Upper Bounds within the Modified Subgradient Algorithm

At each step of the MS algorithm, for fixed $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$, we solve the LR subproblem and obtain the allocation quantities. These allocation values constitute the input of the MCALA heuristic which gives valid Z_{UB} values. In case Z_{UB} value is better than Z_{UB}^{best} the MS algorithm updates $Z_{UB}^{best} = Z_{UB}$. Notice that during the run of the MS algorithm it is guaranteed that the relation $Z^* \leq Z_{UB}^{best}$ holds. On the other hand, the MS algorithm also computes Z_{LB} values and updates Z_{LB}^{best}, until the stopping condition is satisfied. The relation $Z_{LB}^{best} \leq Z^* \leq Z_{UB}^{best}$ also holds at each iteration of the MS algorithm.

The most painstaking part of the MS algorithm is the excessive CPU time requirement for the solution of the lower bounding problem (i.e., LR subproblem). One approach to produce lower bounds is to solve the equivalent SC problem given by Equation 6.6 – 6.9 via a BP algorithm and obtain the $Z_{SC}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ values as the Z_{LB} values. Note that $Z_{SC}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) = Z_{LR3}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ holds. However, solving neither the Lagrangean subproblem given by Equation 6.3 – 6.5 nor its equivalent SC problem given by Equation 6.6 – 6.9 by running a BP algorithm is tractable at each step of MS algorithm due to the excessive CPU time requirement. To alleviate this drawback, one strategy may be to resort solving the SCLP via CG procedure and to produce $Z_{SCLP}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ values which can be used as Z_{LB} in MS algorithm rather than solving the SC problem given by Equation 6.6 – 6.9 to optimality. Note that $Z_{SCLP}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z_{LR3}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^*$ holds during the run of MS algorithm. However, to obtain $Z_{SCLP}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$, at each step of the CG procedure, either a DC algorithm or the FSG algorithm is run as a subprocedure, which may also require drastic CPU times especially for large instances.

As the above discussion attests, in practice we can make use of neither the BP algorithm nor the CG procedure within the MS algorithm. Therefore, we suggest running a heuristic algorithm to efficiently solve the LR subproblem, with the purpose to update faster the Lagrangean multiplier vectors $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$. Although heuristic solution values $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$, need not satisfy the relation $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^* \leq Z_{UB}^{best}$, we can actually use $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ as Z_{LB} within the MS algorithm as long as $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z_{UB}^{best}$ is satisfied. When $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) > Z_{UB}^{best}$ holds at some step of the MS algorithm, we re-adjust the latest Lagrangean multiplier vectors $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$. For that purpose, we solve the SCLP via CG procedure and re-update the Lagrangean multiplier vectors $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$ such that the relation $Z_{LB} = Z_{SCLP}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^* \leq z_{UB}^{best}$ is maintained. Then, we continue to run the heuristic algorithm as long as $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z_{UB}^{best}$ holds (otherwise we obtain $Z_{SCLP}^*(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ to adjust $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$). When the MS algorithm converges or the stopping condition is satisfied, we solve either the SCLP via CG procedure or block norm based lower bounding MILP as the final step in order to make sure that $Z_{LB}^{best} \leq Z^*$ is satisfied. Notice that as a result of the foregoing discussion, the use of a heuristic solution as Z_{LB}, does not contradict with the theorems given by (Polyak, 1967, 1969) on the convergence of the MS algorithm. The convergence of the classical SO algorithm is guaranteed as long as the sequence of step sizes and their summation converge to 0 and ∞, respectively (Polyak, 1967, 1969). Indeed, the MS algorithm also has a step size sequence satisfying these properties. Moreover, an upper bound Z_{UB}^{best} satisfying $Z_{UB}^{best} \geq Z^*$ and current lower bound Z_{LB} satisfying $Z_{LB} \leq Z_{UB}^{best}$ are used within the update step of the MS algorithm. Here, Z_{LB} always satisfy $Z_{LB} \leq Z_{UB}^{best}$ (i.e., when $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \geq Z_{UB}^{best}$ holds, then Z_{LB} is calculated by employing a valid lower bounding procedure such that $Z_{LB} \leq Z_{LR3}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z_{UB}^{best}$ is satisfied). Note that this procedure still uses the negative of the current subgradient to determine a descent direction and the norm of the subgradient. Hence, lower and upper bounds on the objective value are used to calculate a step length while updating the multiplier vector at every step. Clearly, the choice of an efficient heuristic algorithm, which will be used within the MS algorithm to compute the $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ value, is a matter of utmost importance. For that purpose, we employ the Uncapacitated Discrete Approximation Heuristic (UDAH) which is a variant of the MDAH. The UDAH also consists of two phases. In the first phase, the UDAP which is the approximation of the MWP variant given by Equation 6.3 – 6.5 is solved. In the second phase, given the optimal locations obtained from the solution of UDAP, an ALA heuristic is run to obtain an improved solution.

In summary, to accelerate the MS algorithm the UDAH is run to solve the Lagrangean

subproblem at each step as long as $Z_{UDAH}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z_{UB}^{best}$ holds. In case $Z_{UDAH}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ exceeds Z_{UB}^{best} at some step of the MS algorithm, a lower bounding subproblem (SCLP via CG procedure or block norm based lower bounding approach) is solved in order to re-update the Lagrangean multipliers $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$. At the final step of the MS algorithm the solution of the SCLP or the approximating MILP is used to ensure that we have a valid lower bound on the MWP variant, i.e., $Z_{LB}^{best} = Z_{SCLP}^{*}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z^{*}$ is satisfied. The use of $Z_{UDAP_\infty}^{*}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ or $2^{\frac{1-r}{r}} Z_{UDAP_1}^{*}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ which are calculated by solving the lower bounding block norm based approximations may also substitute $Z_{SCLP}^{*}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$. The strategy of using heuristic solutions of the LR subproblem in order to accelerate the MS algorithm can also be adapted to other optimization problems with intractable subproblems that can be solved by efficient heuristics.

Improving the Efficiency of Heuristic Upper Bounds: The efficiency of the UDAH can be further improved by a suitable LR scheme. Although such a LR scheme can increase the speed of obtaining heuristic upper bounds, it can also deteriorate the accuracy that will be obtained from the UDAH. In case these upper bounds $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ exceed the best upper bound Z_{UB}^{best}, then the MS algorithm reduces to the classical SO algorithm resulting in a very inefficient solution approach for the MCMWP. We suggest relaxing constraints given by Equation 6.20 of the UDAP formulation associating Lagrangean multipliers $\boldsymbol{\beta}^4$ to obtain the following LR subproblem

RUDAP:

$$\min Z_{LR4}(\boldsymbol{\beta}^4) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{l=1}^{L}(q_{jk}\widetilde{c}_{ijkl} - \beta_{jk}^4)\widetilde{y}_{ijkl} + \sum_{j=1}^{J}\sum_{k=1}^{K}\beta_{jk}^4 - \xi \tag{6.25}$$

$$\text{s.t.} \qquad \text{Equation } 6.21 - 6.24. \tag{6.26}$$

Notice that the Lagrangean subproblem given by Equation 6.25 and 6.26 can be decomposed into I subproblems which can be solved similar to the LR schemes described for MDAP1. Unfortunately, we should state that the accuracy of the RUDAP is not quite satisfactory. Actually, we try to avoid from a time-consuming SO algorithm and the RUDAP becomes the classical SO algorithm when we use it to obtain heuristic upper bounds. In short, we prefer to solve the UDAP exactly to find heuristic upper bounds by UDAH instead of its relaxed version RUDAP.

6.3.2. Improving the Efficiency of Column Generation Procedure within the Modified Subgradient Algorithm

Two different acceleration strategies are used within the CG procedure. The first acceleration strategy is an heuristic algorithm which is devised to generate additional columns with negative reduced cost with D.C. programming approach. (Krau, 1997) examines this algorithm for the MWP. Once the CM problem is solved, its outcome is the corresponding facility location say $\mathbf{x}_i$. Then a column p, which consists of zero-one entries for each customer-commodity pair, is constructed by using $\mathbf{x}_i$. Namely, given $\mathbf{x}_i$, the following formula is used in order to set the values of column elements:

$$b_{jk}^{p} = \begin{cases} 1, & \text{if } \; q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk} < 0 \\ 0, & \text{otherwise} \end{cases} \qquad j = 1, \ldots, J; k = 1, \ldots, K. \tag{6.27}$$

Notice that for the remaining I entries of column p, we set $b_{JK+i}^{p} = 1$ and $b_{JK+i}^{p} = 0$ for $i^* = 1, \ldots, I$ with $i^* \neq i$. The column p is added to the current SCLP formulation and facility i is relocated according to the assignments of column p. Afterwards, a new column p' is constructed with the latest facility location $\mathbf{x}_i$ according to Equation 6.27. p' is added to the current SCLP and we set $p = p'$. All these steps start from scratch until the current p' becomes stable and no changes occur on its elements. Each column p' added to the SCLP has a negative reduced cost less than the ones of the previously added columns. With this heuristic algorithm, the number of negative reduced cost columns to be added after solving a PS increases and the convergence of the CG procedure with D.C. programming is accelerated. Let $\mathcal{J}_p$ be the set of customer-commodity pairs that belong to column p. We provide a formal outline of the heuristic algorithm associated with D.C. programming in Figure 6.2.

The other acceleration strategy, which increases the efficiency of the LR scheme, focuses on the prevention of all possible columns within the CG procedure. Recall that the FSG algorithm generates all possible subsets of customer-commodity pairs which have negative

1. Construct column p using Equation 6.27 and set $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup p$.
2. Find current facility location $\mathbf{x}_i$ using p such that

$$\min_{\mathbf{x}_i} \left\{ \sum_{j=1}^{J} \sum_{k=1}^{K} b_{jk}^{p} q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) \right\}.$$

3. Set $\nu = 0$. For each pair (j,k), $j = 1, \ldots, J; k = 1, \ldots, K$,
   if $q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk} \geq 0$ and $(j,k) \in \mathcal{J}_p$, **then** set $\mathcal{J}_p \leftarrow \mathcal{J}_p \setminus (j,k)$ and $\nu = \nu + 1$,
   else if $q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \lambda_{jk} < 0$, **then** set $\mathcal{J}_p \leftarrow \mathcal{J}_p \cup (j,k)$ and $\nu = \nu + 1$.
4. **if** $\nu > 0$, **then** construct p' by Equation 6.27, set $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup p'$ and $p = p'$, go toStep 2. **else** go to Step 5.
5. Add all columns $p \in \mathcal{P}_i$ to the current SCLP.

Figure 6.2. Heuristic algorithm for the CG with D.C. programming.

reduced costs in polynomial time. All columns corresponding to these subsets are candidates to be added to the SCLP. Nevertheless, the number of columns which enter into the LP model can be restricted by using some lower bounding schemes on a given subset. Let $\mathcal{J}_p$ be a customer-commodity pair subset of facility i for column p, the reduced cost can be restated as $c_{p(i)} = \sum_{(j,k)\in\mathcal{J}_p} q_{jk} d_k(\mathbf{x}_i, \mathbf{a}_j) - \left(\sum_{(j,k)\in\mathcal{J}_p} \lambda_{jk} - \omega_i \right)$. When we substitute the first term of the reduced cost by a lower bound, we obtain a lower bound on the reduced cost value $c_{p(i)}$. In case this lower bound on the reduced cost is negative, we can add the corresponding column to the current SCLP. Otherwise, we neglect this column. (Righini and Zaniboni, 2007) propose two such lower bounding schemes for the MWP case. We adapt them within our CG procedure in order to eliminate some of the subsets generated with the FSG algorithm before considering their addition to the SCLP. These two lower bounding schemes are presented below.

The first one of the lower bounding scheme is originally devised by (Drezner, 1984) for the WP. Let $\mathbf{a}_0$ be the center of gravity of a given subset $\mathcal{J}_p$ for a facility i. The following constitutes a valid lower bound on $c_{p(i)}$.

$$LB1_{p(i)} = \min_{x_{i1}} \left\{ \sum_{(j,k)\in\mathcal{J}_p} q_{jk} c_{ijk} \frac{|a_{01} - a_{j1}|\,|x_{i1} - a_{j1}|}{d(\mathbf{a}_0, \mathbf{a}_j)} \right\} + \min_{x_{i2}} \left\{ \sum_{(j,k)\in\mathcal{J}_p} q_{jk} c_{ijk} \frac{|a_{02} - a_{j2}|\,|x_{i2} - a_{j2}|}{d(\mathbf{a}_0, \mathbf{a}_j)} \right\} + \sum_{(j,k)\in\mathcal{J}_p} q_{jk} \left(\varphi_{ik} + \mu_{ij}\right) - \left(\sum_{(j,k)\in\mathcal{J}_p} \lambda_{jk} - \omega_i \right) \leq c_{p(i)} \tag{6.28}$$

In $LB1_{p(i)}$ a sorting algorithm is run as a subroutine in order to solve one dimensional WP with the rectilinear distance function and hence to optimally locate each facility i. The other lower bounding scheme is used by (Righini and Zaniboni, 2007) for the MWP case and we adapt it to our case as follows:

$$LB2_{p(i)} = \frac{1}{J \cdot K - 1} \left\{ \sum_{(j,k),(j',k')\in\mathcal{J}_p} \min \left\{ q_{jk} \left(c_{ijk} d(\mathbf{a}_j, \mathbf{a}_{j'}) + \varphi_{ik} + \mu^3_{ij} \right), q_{j'k'} \left(c_{ij'k'} d(\mathbf{a}_j, \mathbf{a}_{j'}) + \varphi_{ik'} + \mu_{ij'} \right) \right\} \right\} - \left(\sum_{(j,k)\in\mathcal{J}_p} \lambda_{jk} - \omega_i \right) \leq c_{p(i)} \tag{6.29}$$

6.4. The Modified Subgradient Algorithm

We present a formal outline of the MS algorithm in Figure 6.3 and explain some technical details on the computation of the feasible solutions for the MCMWP. The upper bounds are determined by the MCALA heuristic which is initialized with the allocation values $w_{ijk} = w'_{ijk} q_{jk}$ obtained from the solution of the LR subproblem. The CG procedure ends up with a solution in which some columns are assigned a positive weight. These are denoted by o_p which can take values between 0 and 1, i.e., $0 \leq o_p \leq 1$, since we consider only the SCLP. For each $o_p > 0$, the corresponding column p is evaluated. It is already known that column p belongs to facility i. This is denoted by the last I entries of each column and facility i has value 1 in its i^{th} entry and 0 for the rest. Furthermore, the first $J \times K$ elements of the column indicate whether customer j of commodity k is served by this column and consequently by the corresponding facility i. Clearly, if customer j of commodity k is served

or covered, a value of 1 appears on the corresponding entry of the column and 0 otherwise. Now, we can set the allocation values of the current solution by using these columns and their o_p values. We set $w'_{ijk} = o_p b^p_{jk} b^p_{JK+i}$ and thus the corresponding allocation values can be determined by $w_{ijk} = w'_{ijk} q_{jk}$. Note that in case the b^p_{jk} values are fractional, then w_{ijk} values lie in the interval $0 < w_{ijk} < q_{jk}$ instead of taking either 0 or q_{jk} values, as in the case of the MWP. The corresponding allocation values need not be feasible for the MCMWP; it is very likely that they violate both of the constraints given by Equation 2.3 and 2.5, which have already been relaxed. The constraints that are not satisfied by this solution vector (allocation vector) constitute the subgradients, which are used to update the multipliers $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$. Then the facility locations can be calculated by the Weiszfeld's algorithm with distance values multiplied by w_{ijk}'s. Running MCALA initialized at these facility locations gives an upper bound on the MCMWP, which is used to update the best upper bound Z^{best}_{UB} within the MS algorithm. In addition, the initial Z^{best}_{UB} values need not be selected as ∞. Actually, these are the reasons why we have employed the CL-RMDA1 in our calculations.

1. (Initialization) Set $\varphi_{ik} = 0$,$\mu^3_{ij} = 0$ for all i, j and k, $\pi = 2$, $Z^{best}_{LB} = -\infty$, $Z^{best}_{UB} = \infty$.
2. Repeat Step 3 to Step 6 until the algorithm converges.
3. Find heuristic bound $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ with Lagrange multipliers φ_{ik} and μ^3_{ij}. **if** $Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) > Z^{best}_{UB}$, **then** find a valid lower bound Z_{LB} with φ_{ik} and μ^3_{ij} on the optimum Z^* and update Z^{best}_{LB} if necessary (i.e., $Z^{best}_{LB} = \max\{Z_{LB}, Z^{best}_{LB}\}$). Otherwise, set $Z_{LB} = Z_H(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$.
4. Find an upper bound Z_{UB} on Z^* **if** $Z_{UB} < Z^{best}_{UB}$, **then** set $Z^{best}_{UB} = Z_{UB}$,
5. Update multipliers by setting $\varphi_{ik} = \varphi_{ik} + T_1(\sum_{j=1}^{J} w'_{ijk} q_{jk} - s_{ik})$ and
   $\mu^3_{ij} = \max\{0, \mu^3_{ij} + T_2(\sum_{k=1}^{K} w'_{ijk} q_{jk} - u_{ij})\}$ where
   $T_1 = \pi(Z^{best}_{UB} - Z_{LB}) / \sum_{i=1}^{I}\sum_{k=1}^{K}(\sum_{j=1}^{J} w'_{ijk} q_{jk} - s_{ik})^2$ and
   $T_2 = \pi(Z^{best}_{UB} - Z_{LB}) / \sum_{i=1}^{I}\sum_{j=1}^{J}(\sum_{k=1}^{K} w'_{ijk} q_{jk} - u_{ij})^2$.
6. **if** Z_{LB} did not improve within the last 30 iterations, **then** set $\pi = \pi/2$.
7. Find a final valid lower bound Z_{LB} with the best φ_{ik} and μ^3_{ij} on Z^*, update Z^{best}_{LB} as necessary and output Z^{best}_{LB} and Z^{best}_{UB}.

Figure 6.3. The Modified Subgradient (MS) algorithm.

7. ESTIMATING STATISTICAL BOUNDS ON THE OPTIMAL OBJECTIVE VALUE

Heuristics are systematic procedures which look for feasible solutions of optimization problems at reasonable computational times. Hence, it becomes possible to generate a random sample of objective values by running a randomly initialized heuristic. Presumably, objective values are independent of each others and distributed according to the same probability distribution. Then, an immediate question is how to take advantage of this random sample of objective values to estimate the optimal value Z^* of the problem. (Brandeau and Chiu, 1993) experimentally study the worst case behavior of the Cooper's ALA heuristic and employ it to create a random sample which is then used to statistically estimate Confidence Intervals (CIs) for the optimal value of MWP. In this chapter[4] we follow this line of research and present statistical procedures to estimate the optimal value of the MCMWP as well as other Combinatorial Optimization Problems (COPs). We use heuristic solution procedures presented in Chapter 4 and Chapter 5 which can be randomized through their initial conditions for the MCMWP. These are C-MCALA, C-MDRR and MDA1 heuristics which are randomly initialized within the convex hull of the customer locations.

7.1. Point and Interval Estimators

One possibility is to use the Extreme Value Theory (EVT) and benefit from Fisher and Tippett's theorem (Fisher and Tippett, 1928). EVT deals with the asymptotic behavior of extreme values (i.e., the minimum or maximum values) in samples and tries to fit probability distributions to extreme values. The well-known Fisher and Tippett's theorem is as follows. Consider M independent samples, each of size $\overline{M}$, obtained from the same continuous distribution bounded from below (above) by A. Let Z_m denote the minimum (maximum) value of sample m, then for $\overline{M}$ large enough, Z_m for $m = 1, \ldots, M$ are Weibull distributed with location parameter A. Recall that the probability density and probability

[4]The article by (Akyüz *et al.*, 2010c), the technical report by (Akyüz *et al.*, 2008) and the conference proceeding by (Akyüz *et al.*, 2009b) are partially based on this chapter.

distribution functions of the Weibull distribution are respectively

$$f(Z) = \left(\frac{D}{B^D}\right)(Z-A)^{D-1}\, e^{\left[-\left(\frac{Z-A}{B}\right)^D\right]}, \qquad Z \geq A > 0,\ B > 0,\ D > 0 \tag{7.1}$$

and

$$F(Z) = 1 - e^{\left[-\left(\frac{Z-A}{B}\right)^D\right]}, \tag{7.2}$$

where A, B and D denote the location, scale and shape parameters, respectively.

Note that the location parameter of the Weibull distribution gives the minimum value of the distribution. The Fisher and Tippett's theorem is valid for any continuous distribution from which the sampling is performed. As a result, it is possible to treat an objective value obtained by a randomly initialized run of the heuristic as the minimum of a large random sample and claim that the distribution of the objective values calculated by the heuristic is approximately Weibull. Then, any point estimate of the location parameter of the Weibull distribution estimated using these heuristic objective values yields a point estimate on the minimum objective value. Moreover, the bounds of any interval estimate of the location parameter give a lower bound and an upper bound for the optimal value of the problem with certain confidence level.

Several researchers have employed this result to provide point and interval estimators of Z^* for various difficult COPs. The early study of (McRoberts, 1971) on the facility layout problem is the first attempt to propose a graphical search method for the estimation of the Weibull location parameter A, namely Z^*. (Dannenbring, 1977) employs both graphical search and (Gumbel, 1958)'s method to derive an analytical point estimator of the optimal value of the flow-shop sequencing problem. The first systematic procedure of the point estimation using EVT is for the famous Traveling Salesman Problem (TSP) (Golden, 1977, 1978). This procedure is later improved by (Golden and Alt, 1979) to compute CIs for the optimal value of large COPs. The author has defined $Z_{UB} = \min\{Z_m : 1 \leq m \leq M\}$, where Z_m is the minimum objective value in sample m, and have shown that

$$\Pr\{Z_{UB} - B \leq A \leq Z_{UB}\} = 1 - e^{-M}, \tag{7.3}$$

which in fact means that $[Z_{UB} - B, Z_{UB}]$ is a $100(1 - e^{-M})\%$ CI for the location parameter A, namely Z^*. Notice that the confidence level is almost 1 even for small values of M. (Golden and Stewart, 1985) apply this approach on the TSP and report successful results. (Los and Lardinois, 1982) suggest to use a subset of size $M' \leq M$ local optima with *distinct* values $Z_1, Z_2, \ldots, Z_{M'}$ to fit a Weibull distribution. The reason for this suggestion is that the Fisher and Tippett's theorem assumes the independence of M samples; nevertheless having identical local optima in the set of M samples is equivalent to repeating the same sample several times. They have also indicated that although the sampling is done by using M' distinct local optima, they are gathered by the same heuristic and each solution attempts to reach the same point, resulting in a violation of the independence assumption. It should be noted that (Golden and Alt, 1979) consider a local optimal solution to be the extreme values of a sample when they apply a randomly initialized heuristic and treat the intermediate heuristic step solutions to constitute the corresponding sample. According to (Los and Lardinois, 1982) those samples can have different sizes because the number of intermediate heuristic steps can be different until the convergence of the heuristic. Even if they were equal the independence of samples is again violated since each intermediate step reaches the same local optimum. Therefore, the authors offer to take M' samples each having $\overline{M}$ distinct local optima. They apply Fisher and Tippett's theorem to these M' observations each of which being the minimum of a sample with m distinct values and developed the formula

$$Pr(Z_{UB} - \frac{B}{\Phi} \leq A \leq Z_{UB}) = 1 - exp(-\frac{M'}{\Phi^D}), \tag{7.4}$$

where Φ is any real number. The main advantage of Equation 7.4 over Equation 7.3 is its explicit dependence on the confidence level. In other words, $100(1-\zeta)\%$ confidence interval $\left[Z_{UB} - \frac{B}{\Phi}, Z_{UB}\right]$ for the location parameter A can be achieved by letting

$$\Phi = (-\frac{M'}{\ln \zeta})^{\frac{1}{D}}. \tag{7.5}$$

Notice that the confidence level is fixed to $100(1 - \zeta)\%$ in Equation 7.3, which is not the case for Equation 7.4. However, there is a specific problem with the Los and Lardinois' Equation 7.4: it involves the shape parameter D, which can make the CIs wider or narrower than it should be due to the direct dependence on D. As a remedy, one can consider to take samples of equal size and apply Golden and Alt's procedure in order to avoid from direct dependence of the CI on the Weibull shape parameter.

A critic for the Los and Lardinois' approach is made by (Wilson *et al.*, 2004) who state that two heuristic solutions having the same objective value need not necessarily indicate that they are repetitions of the same sample. That is to say, they may stand for two different feasible solutions. They have also noted that considering only distinct local optima can prevent us from revealing the actual sampling distribution.

In any case, the Fisher and Tippett's theorem requires that the parent distribution is continuous and the samples of equal size drawn from the same population are independent. The independence assumption of the Fisher and Tippett's theorem is verified by the independence tests. The assumption that the parent distribution is continuous does not hold for the discrete optimization problems since the validity of the Fisher and Tippett's theorem has not been shown for the discrete distributions. However, the discrete optimization problems have a huge number of solutions and thus the approximation of a discrete distribution by a continuous distribution does not harm this assumption in practice as claimed by (Los and Lardinois, 1982). On the other hand, we should point out that the MCMWP is a continuous optimization problem which yields a continuous parent distribution. Very encouraging results based on the Fisher and Tippett's theorem have been reported. These results are indicated with EVT on the third column of Table 7.1 where we present applications of statistical bound estimation procedures on several COPs.

7.2. The Limiting Probability Distribution Approach

The Limiting Probability Distribution Approach (LPDA) tries to estimate upper (lower) bounds for independent random observations belonging to the same probability distribution with a confidence level of $100(1-\zeta)\%$ after assuming that the density function is twice differentiable. This method employs limiting probability distributions which are often in the form

$$P(Z_{LB} \leq Z^*) = 1 - \zeta, \tag{7.6}$$

where Z_{LB} is a lower bound on the optimal value Z^*. Given that the observations are in increasing order, namely Z_1 is the smallest, and Z_m is the m^{th} smallest value in the sample, it is possible to set

$$Z_{LB} = Z_1 - F_\zeta(Z_m - Z_1). \tag{7.7}$$

Here, Z_m is frequently selected as the second smallest value, namely Z_2. There are several ways of choosing the constant F_ζ. In their early work on the point estimation and confidence limits for the lower bound of a random variable, (Robson and Whitlock, 1964) suggest

$$F_\zeta^1 = (1-\zeta)/\zeta. \tag{7.8}$$

The authors have used F_ζ^1 in the determination of an approximate $100(1-\zeta)\%$ lower confidence limit of a truncation point $\underline{Z}$ or a lower bound using a sample of size M, from the distribution $F(Z)$ such that $\underline{Z} \leq Z$ and $0 < F(Z) < 1$.

This choice of constant F_ζ is later modified for the distributions having two truncation points at both ends as

$$F_\zeta^2 = \left\{(1-\zeta)^{-\rho} - 1\right\}^{-1} \tag{7.9}$$

by (Cooke, 1979). Here, ρ is a positive number and the author recommends to select its value between 1/5 and 1. Note that for $\rho = 1$, F_ζ^2 equals to F_ζ^1. For $\rho < 1$, the constant F_ζ^2 yields looser lower bounds on Z^* than the ones obtained by using F_ζ^1. (Boender *et al.*, 1982) employ F_ζ^2 for $\rho = 2$, later on. Another suggestion is realized by (Van Der Watt, 1980):

$$F_\zeta^3 = \left\{(1-\zeta^{1/(m-1)})^{-\rho} - 1\right\}^{-1}. \tag{7.10}$$

Again m is a positive integer less than or equal to the sample size standing for the m^{th} smallest observation within the sample. Note that for $m = 2$, F_ζ^3 is equivalent to F_ζ^2. (Van Der Watt, 1980) has pointed out that F_ζ^3 is asymptotically more efficient than the ones proposed by (Robson and Whitlock, 1964) and (Cooke, 1979). Furthermore, the expected length of the CI proposed by (Van Der Watt, 1980) is considerably smaller than the expected length of the one proposed by (Cooke, 1979). According to the experimental results reported by (Monroe, 1982), F_ζ^3 performs better than both F_ζ^1 and F_ζ^2. There are only very few applications of the LPDA in the literature as presented in Table 7.1.

7.3. The Goodness-of-fit Approach

The Goodness-of-fit Approach (GFA) is based on fitting an empirical distribution function to the population distribution. Using the empirical distribution function a lower bound for the population can be obtained with a given confidence level. For that purpose, polynomial functions (e.g., second or higher order) may be employed as approximations to the population distribution. This idea can also be extended to optimization problems when their objective values are considered as our population and it is required to fit a probability distribution to the objective values. However, this task is not trivial since the distribution of the objective values is not known a priori.

For a minimization problem, the GFA starts with an initial valid lower bound Z_{LB} and iteratively progresses by increasing the value of Z_{LB} as long as a predefined goodness-of-fit measure (e.g., χ^2) improves. The value of Z_{LB} corresponding to the best goodness-of-fit value yields an estimate for the lower bound of the optimal value.

The methods adopting the GFA differ primarily in the goodness-of-fit measure, the form of the fitted distribution function, the parameter estimation procedure for the corresponding empirical distribution function and the number of observations used to execute the calculations (Monroe, 1982). For further details on the GFA, we refer to the works by (Hartley and Pfaffenberger, 1969) and (Liau *et al.*, 1973).

7.4. Procedures to Estimate Weibull Parameters

The estimation of the Weibull parameters is a critical issue in the application of the EVT. Basically, three type of estimators are used: Least Square Estimators (LSEs), Simple Point Estimators (SPEs) and Maximum Likelihood Estimators (MLEs).

7.4.1. The Least Squares Error Estimators

(Golden, 1977) adapts the Clarke-Wright saving heuristic (Clarke and Wright, 1964) for the solution of the TSP and employ this heuristic to estimate the global optimal solution of the TSP. In addition, (Golden, 1977) pursues (McRoberts, 1971)'s method and proposes a LSE for the location parameter A. By taking logarithms of the Weibull distribution function

Table 7.1. Applications of the Statistical Bound Estimation Procedures on COPs.

Problem	**Reference**	**Statistical Method**	**Parameter Estimation Procedure**
Facility Layout	(McRoberts, 1971)	EVT	Point Estimator (LSE)
Flow-Shop Sequencing	(Dannenbring, 1977)	EVT	Point Estimator (LSE & Method of Moments)
Traveling Salesman Problem	(Golden, 1977) (Golden and Alt, 1979)	EVT EVT LPDA	Point Estimator (LSE) Golden-Alt Interval (MLE) Robson-Whitlock's Method
Traveling Salesman Problem Quadratic Assignment Problem	(Derigs, 1985)	EVT LPDA	Golden-Alt Interval (SPE) Golden-Alt Interval (MLE) Los-Lardinois Interval (MLE) Robson-Whitlock Method Boender et al. Method Van Der Watt's Method
Hamiltonian Path Problem	(Altınel *et al.*, 2000)	EVT	Golden-Alt Interval (SPE)
Quadratic Assignment Problem	(Bruijs, 1984)	EVT	Point Estimator (SPE)
Set Covering	(Vasko and Wilson, 1984)	EVT	Golden-Alt Interval (LSE)

Table 7.1. Applications of the Statistical Bound Estimation Procedures on COPs cont.

Problem	**Reference**	**Statistical Method**	**Parameter Estimation Procedure**
Multicovering	(Gonsalvez *et al.*, 1987)	EVT	Golden-Alt Interval (MLE)
Set Partitioning	(Sastry and Pi, 1991)	EVT	Golden-Alt Interval (LSE)
Location-Allocation	(Brandeau and Chiu, 1993)	EVT	Point Estimator (LSE)
Job Sequencing	(Ovacık *et al.*, 2000)	EVT LPDA	Point Estimator (SPE) Boender et al's. Method
Flow-Line Scheduling	(Wilson *et al.*, 2004)	EVT	Golden-Alt Interval (LSE with nonlinear fit)
Transportation Network Design	(Los and Lardinois, 1982)	EVT	Los-Lardinois Interval (MLE)
Rail Freight Network Design	(Marin and Salmerón, 1996)	EVT	Los-Lardinois Interval (MLE)
Bin Packing	(Hall *et al.*, 1988)	EVT	Golden-Alt Interval (MLE)
Batch and Semi-Continuous Plant Scheduling	(Kudva *et al.*, 1994)	EVT	Golden-Alt Interval (MLE)
Job-Shop Scheduling	(Ganesan and Sivakumar, 2006)	EVT	Point Estimator (SPE)
Parallel Batch Processing Scheduling	(Mathirajan and Sivakumar, 2006)	EVT	Point Estimator (SPE)

given by Equation 7.2 twice (Golden, 1977) obtains

$$D\ln(Z-A) - D\ln B = \ln\left[-\ln(1-F(Z))\right]. \tag{7.11}$$

Observe that when the location parameter A of the Weibull distribution is fixed then Equation 7.11 can be considered as the equation of a regression line of the form $D_1 Z_A + D_2 = f(Z_A)$ with independent variable $Z_A = \ln(Z-A)$, dependent variable $f(Z_A) = \ln\left[-\ln(1-F(Z))\right]$, slope $D_1 = D$ and intercept $D_2 = -D\ln B$. Their values can be estimated using the least square analysis. Given a fixed value of the location parameter A, it is possible to find D and $D\ln B$, and hence to obtain the scale parameter B and the shape parameter D. By setting different values for the location parameter A, different values of the scale parameter B and the shape parameter D can be obtained. Therefore, this procedure is repeated for different values of location parameter A until the largest correlation coefficient is obtained. (Golden, 1977) notes that when the absolute value of the correlation coefficient is close to 1 then this implies that there is a strong linear relationship between the dependent and independent variables. For the other case, namely when the absolute value of the correlation coefficient is not close to 1, then the null hypothesis, "*the heuristic solutions are Weibull distributed*", can be rejected.

(McRoberts, 1971) refers to the linear regression version of the LSE, which is implemented in several studies (Brandeau and Chiu, 1993; Dannenbring, 1977; Golden, 1977; Sastry and Pi, 1991), as the graphical search. A nonlinear regression version of the LSE is proposed by (Wilson *et al.*, 2004) where the authors employ the Nelder-Mead simplex search (Nelder and Mead, 1965) in order to find the minimum of the corresponding nonlinear least squares error function in terms of Weibull parameters.

7.4.2. Simple Point Estimators

The SPEs or analytical estimators of the Weibull distribution are proposed by several authors. There are mainly four estimators for the location parameter. Let $Z_1 \leq Z_2 \leq \ldots \leq Z_M$ be an ordered sample from a Weibull distribution with unknown location parameter A. The first estimator

$$\widehat{A}_1 = 2Z_1 - Z_2 \tag{7.12}$$

is originally proposed by (Robson and Whitlock, 1964) and used by (Golden and Alt, 1979) and (Dannenbring, 1977). Another estimator is devised by (Dubey, 1967):

$$\widehat{A}_2 = \frac{Z_1 Z_M - Z_2^2}{Z_1 + Z_M - 2Z_2}. \tag{7.13}$$

It is also employed by (Zanakis, 1979) and (Zanakis and Mann, 1982). Note that as Z_2 tends to Z_1, both $\widehat{A}_1$ and $\widehat{A}_2$ approach to Z_1, which makes them biased. We can expect that $\widehat{A}_2$ would be a more accurate estimator than $\widehat{A}_1$, since $\widehat{A}_2$ makes use of 3 observations rather than 2 as $\widehat{A}_1$ does. However, (Muralidhar and Zanakis, 1992) show through Monte Carlo simulation experiments that $\widehat{A}_1$ provides a much closer estimate of the location parameter than $\widehat{A}_2$ in particular when the shape parameter is close to 1. By using the fact that an estimator can be considered to be more effective if it has lower bias and inspired by the second estimator $\widehat{A}_2$, (Muralidhar and Zanakis, 1992) develop a generalization of $\widehat{A}_2$, which is known as the Minimum-Bias Percentile (MBP) estimator:

$$\widehat{A}_3 = \frac{Z_1 Z_M - Z_{m_1}^2}{Z_1 + Z_M - 2Z_{m_1}}. \tag{7.14}$$

Here, Z_{m_1} is selected among M observations such that $\widehat{A}_3$ provides the minimum bias estimator of the location parameter A. The authors suggest that the best estimation can be done with the choice $m_1 = \lceil 0.8829 M^{0.6563} \rceil$. (Muralidhar and Zanakis, 1992) note that in most cases MBP estimator is better than both $\widehat{A}_1$ and $\widehat{A}_2$ since it has smaller mean bias than both $\widehat{A}_1$ and $\widehat{A}_2$. Another estimator for the location parameter is proposed by (Wyckoff *et al.*, 1980) as

$$\widehat{A}_4 = \frac{Z_1 - \frac{\overline{Z}}{M^{1/\Phi_1}}}{1 - \frac{1}{M^{1/\Phi_1}}}, \tag{7.15}$$

where $\overline{Z}$ is the average of observations and $\Phi_1 = \frac{-2.989}{\ln\left[\frac{Z_{\lceil 0.97366M \rceil} - Z_1}{Z_{\lceil 0.16731M \rceil} - Z_1}\right]}$. (Wyckoff *et al.*, 1980) argue that $\widehat{A}_4$ has also good performance in estimating the location parameter.

The scale parameter B is set to

$$\widehat{B}_1 = Z_{\lceil 0.63M \rceil} - \widehat{A} \tag{7.16}$$

in almost all studies in the literature where $\widehat{A}$ is an estimate of the location parameter. (Wyckoff *et al.*, 1980) develop

$$\widehat{B}_2 = \exp\left[0.5772\left(\widehat{D}_4\right)^{-1} + \frac{1}{M}\sum_{m=1}^{M}\ln(Z_m - \widehat{A}_4)\right] \tag{7.17}$$

as another estimator of the scale parameter where $\widehat{D}_4$ is the shape parameter which will be presented later in this section.

The shape parameter D, is the most important parameter which also affects the estimation of other parameters and in case it is miscalculated the CIs produced can be inefficient. (Golden and Alt, 1979) propose to estimate the shape parameter with

$$\widehat{D}_1 = \frac{\ln\left[-\ln(0.5)\right]}{\ln(Z_{\widehat{m}} - \widehat{A}_1) - \ln\widehat{B}_1} \tag{7.18}$$

where $Z_{\widehat{m}}$ stands for the median of the selected sample. Another analytic estimator for the shape parameter is devised by (Zanakis, 1979):

$$\widehat{D}_2 = \frac{\ln\left[\ln(1-\varsigma_{\widehat{m}_2})/\ln(1-\varsigma_{m_2})\right]}{\ln\left[(Z_{\widehat{m}_2} - \widehat{A}_2)/(Z_{m_2} - \widehat{A}_2)\right]} \simeq \frac{2.989}{\ln\left[(Z_{\widehat{m}_2} - \widehat{A}_2)/(Z_{m_2} - \widehat{A}_2)\right]} \tag{7.19}$$

where $\varsigma_{m_2} = 0.16731$ and $\varsigma_{\widehat{m}_2} = 0.97366$ are quantiles minimizing the asymptotic variance of the shape parameter when $\widehat{A}_2$ is known. Here Z_{m_2} and $Z_{\widehat{m}_2}$ are the observations with $m_2 = \lceil 0.16731M \rceil$ and $\widehat{m}_2 = \lceil 0.97366M \rceil$, respectively.

Also the estimator

$$\widehat{D}_3 = \frac{0.5\ln\left[\frac{\ln(1-\varsigma_{\widehat{m}_3})}{\ln(1-\varsigma_{m_3})}\right]}{\ln\left[\frac{Z_{\widehat{m}_3} - Z_{\widetilde{m}_3}}{Z_{\widetilde{m}_3} - Z_{m_3}}\right]} \simeq \frac{3.643}{\ln\left[\frac{Z_{\widehat{m}_3} - Z_{\widetilde{m}_3}}{Z_{\widetilde{m}_3} - Z_{m_3}}\right]}, \tag{7.20}$$

where $\varsigma_{m_3} = 0.0033$, $\varsigma_{\widehat{m}_3} = 0.9920$ and $\varsigma_{\widetilde{m}_3} = 0.1187$ with $m_3 = \lceil 0.0033M \rceil$, $\widehat{m}_3 = \lceil 0.9920M \rceil$

and $\widetilde{m}_3 = \lceil 0.1187M \rceil$, is due to (Zanakis and Mann, 1982).

Finally, (Wyckoff *et al.*, 1980) propose the following analytic estimator

$$\widehat{D}_4 = \frac{M\Phi_2}{-\sum_{m=1}^{\lfloor 0.84M \rfloor} \ln(Z_m - \widehat{A}_4) + \frac{\lfloor 0.84M \rfloor}{M - \lfloor 0.84M \rfloor} \sum_{m=\lceil 0.84M \rceil}^{M} \ln(Z_m - \widehat{A}_4)} \tag{7.21}$$

for the shape parameter where Φ_2 is a constant given by (Engelhardt and Bain, 1977). (Wyckoff *et al.*, 1980) indicate that the parameters $\widehat{A}_4, \widehat{A}_2$ and $\widehat{D}_4$ appear to be the best overall analytic estimators in the literature. As the number of order statistic used for an estimator increases the bias and mean square error of the estimators decrease. Thus, $\widehat{D}_4$ produces better estimates than the others. However, (Zanakis and Mann, 1982) note that for small true shape parameter values, $\widehat{D}_2$ and $\widehat{D}_3$ are also as efficient as $\widehat{D}_4$.

All these analytic estimators are frequently used in the initialization of a MLE procedure. However, (Zanakis, 1979) observes that analytic estimators are good approximations of the true Weibull parameters and it may be preferable to use only SPEs when the sample size is small.

7.4.3. The Maximum Likelihood Estimators

Let $Z_1 \leq Z_2 \leq \ldots \leq Z_M$ be M independent observations obtained from a Weibull distribution. The MLE method aims to estimate the best values of the location parameter A, the scale parameter B and the shape parameter D. In other words, given the Weibull likelihood function

$$L(\Omega) = \prod_{m=1}^{M} f(Z_m) = \prod_{m=1}^{M} \frac{D}{B^D}(Z_m - A)^{D-1} \exp\left(-(\frac{Z_m - A}{B})^D\right) \tag{7.22}$$

where $\Omega = (Z_1, Z_2, ..., Z_M, A, B, D)$ is a vector consisting of observations and parameters, the MLE method estimates A, B and D, by minimizing $L(\Omega)$ such that $A \leq Z_1$, $B \geq 0$ and $D \geq 0$. Since the minimization of $L(\Omega)$ is equivalent to the minimization of the log-likelihood $ln(L(\Omega))$, an approach is the solution of the equality system obtained by setting the partial derivative $ln(L(\Omega))$ with respect to A, B and D, to zero as done by (Golden, 1977). In fact these three equalities in A, B and D are the first order necessary optimality conditions.

(Golden, 1977) states that from this system one can derive the value of the scale parameter B as

$$B = \left(\sum_{m=1}^{M} (Z_m - A)^D / M \right)^{1/D}, \tag{7.23}$$

and then approximately solve the remaining two nonlinear equations using numerical methods.

As for example, there are several other approaches for the maximization of the Weibull log-likelihood function. (Golden and Alt, 1979), (Gonsalvez *et al.*, 1987), (Hall *et al.*, 1988) and (Los and Lardinois, 1982) employ (Harter and Moore, 1965)'s gradient search technique. (Derigs, 1985) combines gradient search technique with a Newton method. (Marin and Salmerón, 1996) employ an interval search method to solve MLE equations. (Zanakis, 1977) states that in general parameter estimation by MLE method generate better sample fits than analytic estimators particularly when the shape parameter becomes larger. (Wilson *et al.*, 2004) remark that Harter and Moore's method causes convergence problems in some of their test instances, which is also noted by (Zanakis, 1977) and (Derigs, 1985).

In the literature, analytic estimators are often used as an initialization step of the MLE procedure. Then these estimates are improved by MLE procedure which requires to find the maximum of a non-convex function. Although the MLE procedure is not trivial, the computational results are encouraging since the additional computational effort required is relatively low for the MLE procedure when compared with the computational effort required using only the SPE procedures.

7.5. Independence and Weibull-Fit Tests

The test for independence of the generated local optima is of great importance since the basic assumption of the Fisher and Tippett's theorem requires identically distributed and independent samples. Therefore, before applying the parameter estimation procedure, it is strongly recommended to test the independence of the random observations. The longest run and runs test are used by many authors for this purpose (Beyer, 1974). The longest run test is considered by (Golden and Alt, 1979) and (Hall *et al.*, 1988) to verify the independence of

the heuristic solutions. The runs test is more popular than the longest run test. The runs test is employed to check the independence of sample objective values by many researchers (Gonsalvez *et al.*, 1987; Kudva *et al.*, 1994; Ovacık *et al.*, 2000; Wilson *et al.*, 2004).

Besides the independence tests, it is also important to test that the objective values computed by means of a heuristic are from a Weibull distribution. For this purpose, two well-known goodness-of-fit tests are used: the Kolmogorov-Smirnov (K-S) test (Law and Kelton, 1991) and the Anderson-Darling (A-D) test (Anderson and Darling, 1952). The K-S test is the most popular and extensively applied in almost all studies (Wilson *et al.*, 2004; Ovacık *et al.*, 2000, Marin and Salmerón, 1996). The A-D test is known to be more strict than the K-S test and (Wilson *et al.*, 2004) suggest the use of the A-D coupled with the K-S because the A-D test is more effective in detecting the discrepancies between the fitted and empirical distributions in the tail regions.

8. ALLOCATION SPACE BASED BRANCH-AND-BOUND METHODS

In this chapter[5] allocation space based BB (ABB) algorithms are suggested for both the CMWP and MCMWP. ABB algorithms partition the allocation variable space into smaller subspaces and aim to find the optimal solution by implicitly enumerating the extreme points of the constraint sets which are defined by the allocation variables. In the first section, we address the ABB algorithm which is originally proposed by (Sherali and Tunçbilek, 1992) for the CMWP and we refer to it as the Single-commodity ABB (SABB) algorithm. In the second section, we develop an ABB algorithm for the MCMWP which we call it as the MABB algorithm. For both ABB and MABB algorithms, we test the performance of block norm based lower bounding procedures, which are previously implemented for the DA heuristics in Chapter 5. Additionally, we consider two other lower bounding procedures: Reformulation-Linearization Technique (RLT) based lower bounding procedure and a straightforward lower bound which uses the solution of WPs. The upper bounds are computed with ALA (CALA or MCALA) heuristics. We follow different search and partitioning strategies for the SABB and MABB algorithms and offer several branching variable selection strategies.

8.1. Solution of the Capacitated Multi-facility Weber Problem

An optimum solution of the CMWP always occur at an extreme point of the Transportation Problem (TP) polyhedron given by Equation 3.7 – 3.9, independent of the distance function $d(\mathbf{x}_i, \mathbf{a}_j)$. Once we are given such an extreme point, namely feasible allocation values, the remaining I WPs can be solved to find the corresponding optimum facility locations. In his earlier work, (Cooper, 1972) tries to enumerate all extreme points of the CMWP. This enumeration method finds the corresponding facility locations and then picks up the minimum cost solution to find the optimum. Although this enumeration method halts quickly, the idea of using allocation space and hence the structure of the TP polyhedron is employed in BB algorithms developed by (Sherali and Tunçbilek, 1992) and (Sherali *et al.*, 2002) for the SECMWP and the LCMWP, respectively.

Each basis of the TP polyhedron corresponds to a spanning tree on a bipartite graph.

[5]The conference proceeding (Akyüz et al., 2011) is partly based on this chapter.

Therefore, there could be at most $I+J-1$ positive allocations at an extreme point for which the bipartite graph is constructed with positive flow arcs. This implies that the remaining flows other than the positive flows are exactly zero. These properties enable to partition the allocation space into two distinct sets for each allocation variable w_{ij}. Notice that the commodity index k vanishes for the CMWP i.e., $w_{ijk} = w_{ij}$ when $K = 1$. Clearly, each w_{ij} variable is either set to zero or enforced to be strictly positive at an extreme point. This binary partitioning approach is first introduced by (Sherali and Tunçbilek, 1992) for the SECMWP and then employed by (Sherali *et al.*, 2002) for the LCMWP.

Actually, the idea of using a binary partitioning of the allocation space corresponds to dealing with an extreme point on the leaf nodes of a BB tree. After setting all allocation variables w_{ij} to either a positive value or zero, an extreme point can be obtained. For that purpose, lower and upper bounds $\widehat{l}_{ij}$ and $\widehat{u}_{ij}$ on w_{ij} variables are defined. Initially, these bounds can be taken as $\widehat{l}_{ij} = 0$ and $\widehat{u}_{ij} = \min\{s_i, q_j\}$ for $i = 1, \ldots, I; j = 1, \ldots, J$. For allocation variables with positive values we have $1 \leq \widehat{l}_{ij} \leq w_{ij}$ and for allocation variables with zero values we have $\widehat{l}_{ij} = \widehat{u}_{ij} = w_{ij} = 0$. According to the values assigned to w_{ij} variables three sets are constructed: $\mathcal{W}^+$, $\mathcal{W}^0$ and $\mathcal{W}^F$, called as *positive variables set*, *zero variables set* and *free variables set*, respectively. Namely, $\mathcal{W}^+$ is the set of allocation variables with positive values, $\mathcal{W}^0$ contains the allocation variables with zero values and $\mathcal{W}^F$ consists of the allocation variables which are not assigned any value. During the exploration of the BB tree, these three sets are gradually updated. As the SABB algorithm progresses, a free variable $w_{ij} \in \mathcal{W}^F$ is selected and added to the positive variable set $\mathcal{W}^+$. Also, the arc (i, j) corresponding to variable w_{ij}, is added to the graph of the current partial solution. Given an arc corresponding to a w_{ij} variable with positive value, we try to detect other arcs (i', j') with $w_{i'j'} \in \mathcal{W}^F$ whose existence create a cycle on the graph and the variables $w_{i'j'}$ corresponding to those arcs are added to set $\mathcal{W}^0$. All variables, which are recently added to $\mathcal{W}^+$ or $\mathcal{W}^0$, are removed from the free variables set $\mathcal{W}^F$. An extreme point is reached when $\mathcal{W}^F$ is empty. Note that the bounds $\widehat{l}_{ij}$ and $\widehat{u}_{ij}$ are also updated accordingly by considering whether the allocation variable w_{ij} belongs to $\mathcal{W}^+$ or $\mathcal{W}^0$. At each node of the BB tree a subproblem is defined by these lower and upper bounds on the allocation variables.

The bounds $\widehat{l}_{ij}$ and $\widehat{u}_{ij}$ are improved using the logical test proposed by (Sherali and Tunçbilek, 1992). A detailed explanation of the logical test is given for the MCMWP later in this chapter where the MABB algorithm is presented. We limit ourselves with a summary

of the logical test for the CMWP in order to be concise. The logical test consists of a sequential update mechanism which uses the maximum slack values of the transportation constraints with current bounds $\widehat{l}_{ij}$ and $\widehat{u}_{ij}$ of the allocation variables. Since any change on the bounds of an allocation variable affects the corresponding maximum slack values, the bounds of all neighbor variables will also change. This situation arises because all resources (i.e., supply and demand quantities) are common for the allocation variables. Hence, when a lower or upper bound of a variable is changed, the bounds of its neighbor variable should be updated with the latest bounds accordingly. This procedure continues until all bounds on the variables become stable. In case one or several maximum slack values become negative, this implies that the allocation space is infeasible with current variable bounds. When the logical test enforces the lower bound of a variable to be positive, $\mathcal{W}^+$ should be updated, all the arcs constituting a cycle with this update should be set to zero and, $\mathcal{W}^0$ and $\mathcal{W}^F$ should be changed appropriately. Consequently, the logical test, should also be integrated with a cycle detection and prevention mechanism.

The SABB algorithm performs a *depth-first search* (DFS) strategy along with the described binary partitioning of the allocation space. The records of the allocation variables on the BB tree are kept on a partial solution list (PSL) with the framework proposed by (Geoffrion, 1967). Each PSL element stands for the status of an allocation variable together with the lower bound value associated with it. A zero variable w_{ij} is indicated by $(i,j)^0$ in the PSL. However, there exist two types of records for positive variables: *selected positive variable* and *necessarily positive variable* which are denoted by $(i,j)^{+S}$ and $(i,j)^{+N}$, respectively. A *selected positive variable* $(i,j)^{+S}$ implies that the allocation variable w_{ij} is set to a positive value via a branching mechanism. A *necessarily positive variable* $(i,j)^{+N}$ implies that the allocation variable w_{ij} is fixed to a positive value within the logical test for the sake of feasibility of the TP constraints (i.e., cycles are not allowed). The complementary zero branch (i.e., the branch where $w_{ij} = 0$) of each w_{ij} with selected positive variable $(i,j)^{+S}$ should also be searched. However, this is not required for a *necessarily positive variable*, $(i,j)^{+N}$, since this can already produce an infeasible solution. The current branch of the BB tree reaches an extreme point if and only if $|PSL| = I \times J$ holds.

At each node of the BB tree a lower and upper bound value which are denoted by Z_{LB} and Z_{UB} should be determined, respectively. If its lower bound is larger than the best upper bound value Z_{UB}^{best} found so far, that node is fathomed and other non-fathomed nodes

are explored. Therefore, a fathoming criterion of $Z_{LB} \geq (1-\epsilon)Z_{UB}^{best}$ can be used to avoid excessive computational effort at each node. We take $\epsilon = 0.001$ in our calculations. We give a generic outline of the SABB algorithm in Figure 8.1.

1. (Initialization): Set $PSL \leftarrow \emptyset$, $\mathcal{W}^+ \leftarrow \emptyset$, $\mathcal{W}^0 \leftarrow \emptyset$, $\mathcal{W}^F \leftarrow \{(i,j) : i = 1, \ldots, I; j = 1, \ldots, J\}$. Set $\widehat{l}_{ij} = 0$ and $\widehat{u}_{ij} = \min\{s_i, q_j\}$ for $i = 1, \ldots, I; j = 1, \ldots, J$
2. (Logical Test and Cycle Prevention): Perform the Logical Test of (Sherali and Tunçbilek, 1992). **if** infeasibility is detected, **then** go to Step 5. **else** set $\mathcal{W}^+ \leftarrow \mathcal{W}^+ \cup (i,j)$, $(i,j) \in \mathcal{W}^F$ with $\widehat{l}_{ij} > 0$ and $PSL \leftarrow PSL \cup (i,j)^{+N}$. Also set $\mathcal{W}^0 \leftarrow \mathcal{W}^0 \cup (i,j)$, $(i,j) \in \mathcal{W}^F$ (i.e., for allocation variables with $\widehat{u}_{ij} = 0$) and for all arcs (i,j) creating a cycle with $(i',j') \in \mathcal{W}^+$, set $PSL \leftarrow PSL \cup (i,j)^0$ at any stage of the update mechanism of the Logical Test. After the Logical Test, **if** $|PSL| < I \times J$, **then** proceed to Step 3. **else** an extreme point is at hand, find an upper bound Z_{UB}, set $Z_{UB}^{best} = \min\{Z_{UB}^{best}, Z_{UB}\}$ and go to Step 5.
3. (Bounding) Find a lower bound Z_{LB} and an upper bound Z_{UB} with current $\widehat{l}_{ij}$ and $\widehat{u}_{ij}$ values to the SABB subproblem. Set $Z_{UB}^{best} = \min\{Z_{UB}^{best}, Z_{UB}\}$. **if** $Z_{LB} \geq (1-\epsilon)Z_{UB}^{best}$, **then** go to Step 5.
4. (Branching) Select a branching variable $(i,j) \in \mathcal{W}^F$ using a branching rule, i.e., BrS1, BrS2 and BrS3. Set $\mathcal{W}^F \leftarrow \mathcal{W}^F \backslash (i,j)$, $\mathcal{W}^+ \leftarrow \mathcal{W}^+ \cup (i,j)$, $PSL \leftarrow PSL \cup (i,j)^{+S}$ and $\widehat{l}_{ij} = 1$. Go to Step 2.
5. (Fathoming) Find a $(i,j)^{+S}$ element in the PSL by backtracking where its previous node has a lower bound smaller than $(1-\epsilon)Z_{UB}^{best}$, **if** there is no such an element within the PSL, **then** STOP: Z_{UB}^{best} is within $100\epsilon\%$ of the optimum. **else** change the element by $(i,j)^0$ in the PSL, set $\mathcal{W}^+ \leftarrow \mathcal{W}^+ \backslash (i,j)$, $\mathcal{W}^0 \leftarrow \mathcal{W}^0 \cup (i,j)$ and $\widehat{l}_{ij} = \widehat{u}_{ij} = 0$. Remove all elements added after (i,j) from the PSL, update $\mathcal{W}^+$, $\mathcal{W}^0$ and $\mathcal{W}^F$ accordingly. Set $\widehat{l}_{ij} = 1$, $(i,j) \in \mathcal{W}^+$ and $\widehat{u}_{ij} = \min\{s_i, q_j\}$ for $i = 1, \ldots, I; j = 1, \ldots, J$. Go to Step 2.

Figure 8.1. The SABB algorithm for the CMWP.

8.1.1. Bounding Procedures

The bounding step includes finding both valid lower and upper bounds for the current subproblem. A SABB subproblem is defined by substituting the constraints given by Equation 3.9 with the following set of inequalities in the CMWP formulation:

$$\widehat{l}_{ij} \leq w_{ij} \leq \widehat{u}_{ij} \qquad i = 1, \ldots, I; j = 1, \ldots, J. \tag{8.1}$$

We suggest employing lower bounding block norms in Equation 3.6 which results in solving a lower bounding MILP for the current subproblem. We have also devised two more lower bounding procedures which are originally developed by (Sherali *et al.*, 2002) for the LCMWP. The next part is dedicated to explain lower bounding procedures that are vital and the most time-consuming step of the SABB algorithm. Additionally, an upper bounding procedure which is used at the initialization of the SABB algorithm and an upper bounding algorithm performed at each node of the BB tree, are presented.

8.1.1.1. Block Norm Based Lower Bounding Procedures. The objective function given by Equation 3.6 includes the distance function which is considered to be the ℓ_r-distance with $1 \leq r < \infty$. The SABB subproblem which is stated by Equation 3.6 – 3.8 and 8.1 can be bounded from below if the distance function is substituted with a lower bounding norm function in Equation 3.6. In particular, candidate locations for the optimum facility locations can be reduced from the entire convex hull of customer locations to a finite set of points within it when a block norm is used in Equation 3.6 as discussed in Chapter 5. This gives a chance to formulate a MILP formulation which can produce a lower bound on SABB subproblems. We first present a MILP formulation to compute the lower bounds with the SABB algorithm and then a Lagrangean Relaxation (LR) scheme is proposed to increase the efficiency of the block norm based lower bounding procedures.

Define the variables y_{ijg} as the amount shipped to customer j from facility i located at candidate point g. Binary variables v_{ig} are set to 1 if facility i is located at point g and 0 otherwise. c_{ijg} is the cost of transporting one unit from facility i located at candidate point g with known coordinates $\widehat{\mathbf{a}}_g = (\widehat{a}_{g1}\ \widehat{a}_{g2})^T$ to customer j where it is defined as $c_{ijg} = c_{ij} d(\widehat{\mathbf{a}}_g, \mathbf{a}_j)$. Recall that here $d(\widehat{\mathbf{a}}_g, \mathbf{a}_j)$ is a block norm distance satisfying

$d(\widehat{\mathbf{a}}_g, \mathbf{a}_j) \leq [|\widehat{a}_{g1} - a_{j1}|^r + |\widehat{a}_{g2} - a_{j2}|^r]^{1/r}$ with $1 \leq r < \infty$. Then, the lower bounding MILP formulation i.e., Discrete Approximation Problem (DAP) of the SABB subproblem can be given as

DAP:

$$\min \quad Z_{DAP} = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{g=1}^{G} c_{ijg}(y_{ijg} + \widehat{l}_{ij}v_{ig}) \tag{8.2}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} y_{ijg} = \overline{s}_i v_{ig} \qquad i = 1, \ldots, I; g = 1, \ldots, G, \tag{8.3}$$

$$\sum_{i=1}^{I}\sum_{g=1}^{G} y_{ijg} = \overline{q}_j \qquad j = 1, \ldots, J, \tag{8.4}$$

$$\sum_{g=1}^{G} v_{ig} = 1 \qquad i = 1, \ldots, I, \tag{8.5}$$

$$0 \leq y_{ijg} \leq u'_{ij} \qquad i = 1, \ldots, I; j = 1, \ldots, J; g = 1, \ldots, G, \tag{8.6}$$

$$v_{ig} \in \{0, 1\} \qquad i = 1, \ldots, I; g = 1, \ldots, G, \tag{8.7}$$

where the parameters $\overline{s}_i$, $\overline{q}_j$ and u'_{ij} are updated with the latest bounds on the variables: $\overline{s}_i = s_i - \sum_{j=1}^{J}\sum_{g=1}^{G} \widehat{l}_{ij}v_{ig} = s_i - \sum_{j=1}^{J} \widehat{l}_{ij}$, $\overline{q}_j = q_j - \sum_{i=1}^{I}\sum_{g=1}^{G} \widehat{l}_{ij}v_{ig} = q_j - \sum_{i=1}^{I} \widehat{l}_{ij}$ which follows from the equality $\sum_{g=1}^{G} v_{ig} = 1$ for $i = 1, \ldots, I$ by constraints given by Equation 8.5 and $u'_{ij} = (\widehat{u}_{ij} - \widehat{l}_{ij})$. Constraints given by Equation 8.3, 8.4, and 8.6 play similar role as the bounded TP constraints given by Equation 3.7, 3.8 and 8.1, respectively. Constraints given by Equation 8.5 enforce each facility to be opened at exactly on one of the candidate locations. Note that the DAP reduces to the original formulation proposed by (Aras *et al.*, 2007, 2008) to produce approximate upper bounding solutions for the LCMWP where no bounds on the allocation variables are imposed (when $\widehat{l}_{ij} = 0$ and $\widehat{u}_{ij} = \min\{s_i, q_j\}$ for all w_{ij}). Lower and upper bounds $\widehat{l}_{ij}$ and $\widehat{u}_{ij}$ on variables require additional binary variable terms which vanish from the formulation when lower bounds are zero.

As we reach the leaf nodes of the BB tree, the DAP can be solved easier than the ones solved for the nodes generated at the beginning of the SABB algorithm. Recall that during the run of the SABB algorithm, the variables are gradually fixed. At some step of the SABB algorithm, positive fixed variables enforce some free variables to take a zero value. The

existence of these positive variables on the current bipartite graph constructed by positive flow arcs creates cycles. Certainly, they are dropped from the subproblem formulation and the remaining problem becomes easier to solve. However, solving the DAP at each node of the BB tree, particularly for nodes explored at the beginning of the SABB algorithm, is not a viable choice as the instance size increases. Therefore, we resort to a LR scheme for the DAP formulation.

Any lower bound on the optimum value of the DAP formulation, which uses a lower bounding norm function as the distance measure, is also valid for the SABB subproblem. We use LR and Subgradient Optimization (SO) algorithm to compute lower bounds for the DAP. When demand constraints given by Equation 8.4 are relaxed with Lagrangean multipliers β_j^5, we obtain the Lagrangean subproblem

RDAP($\boldsymbol{\beta}^5$):

$$\min Z_{LR5}(\boldsymbol{\beta}^5) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{g=1}^{G}(c_{ijg} - \beta_j^5)y_{ijg} + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{g=1}^{G}c_{ijg}\widehat{l}_{ij}v_{ig} + \sum_{j=1}^{J}\beta_j^5\overline{q}_j \tag{8.8}$$

$$\text{s.t.} \qquad \text{Equation 8.3, 8.5, 8.6 and 8.7.} \tag{8.9}$$

The last term in the objective function given by Equation 8.8 is constant and RDAP($\boldsymbol{\beta}^5$) decomposes over the facilities. As a result, the solution of RDAP($\boldsymbol{\beta}^5$) becomes equivalent to the solution of the following I subproblems

RDAP$_i$($\boldsymbol{\beta}^5$):

$$\min \; Z_{LR5^i}(\boldsymbol{\beta}^5) = \sum_{j=1}^{J}\sum_{g=1}^{G}(\overline{c}_{ijg}y_{ijg} + c_{ijg}\widehat{l}_{ij}v_{ig}) \tag{8.10}$$

$$\text{s.t.} \quad \sum_{j=1}^{J}y_{ijg} = \overline{s}_i v_{ig} \qquad g = 1,\ldots,G, \tag{8.11}$$

$$\sum_{g=1}^{G}v_{ig} = 1 \tag{8.12}$$

$$0 \le y_{ijg} \le u'_{ij} \qquad j = 1,\ldots,J; g = 1,\ldots,G, \tag{8.13}$$

$$v_{ig} \in \{0,1\} \qquad g = 1,\ldots,G, \tag{8.14}$$

where $\overline{c}_{ijg} = (c_{ijg} - \beta_j^5)$ is the new unit cost obtained with given multiplier vectors $\boldsymbol{\beta}^5$. The solution of subproblem RDAP$_i$ ($\boldsymbol{\beta}^5$) is not difficult. We can use a "*greedy*" inspection procedure where, for each point g, we determine those customers that are supplied from facility i when located at point g so that the shipment cost $Z_{LR^{ig}}(\boldsymbol{\beta}^5) = \sum_{j=1}^{J} \overline{c}_{ijg} y_{ijg} + c_{ijg}\widehat{l}_{ij}$ is minimized subject to $\sum_{j=1}^{J} y_{ijg} = \overline{s}_i v_{ig}$, with $v_{ig} = 1$ and $0 \leq y_{ijg} \leq u'_{ij}$, $j = 1, \ldots, J$. Note that this yields bounded CKPs whose solutions are very similar to the ones presented for DA heuristics in Chapter 5. Recall that the solution of the CKP requires a sorting effort and assignment of facility i to customer j is done starting from the least cost customer until $\overline{s}_i$ is totally allocated. Here, we should additionally check whether $u'_{ij} \leq \overline{s}_i$ holds or not for an assignment of facility i to customer j. In case, $u'_{ij} \leq \overline{s}_i$ holds, the shipment quantity between facility i and customer j is u'_{ij} and we continue to search for the next minimum cost customer to ship the remaining capacity of facility i. Otherwise, the shipment between the minimum cost customer j and facility i equals to $\overline{s}_i$ and the procedure continues until all capacity of facility i is exhausted. Once the optimal solution $Z^*_{LR^{ig}}(\boldsymbol{\beta}^5)$ for each candidate point g is calculated, $Z^*_{LR^i}(\boldsymbol{\beta}^5)$ is obtained by setting $Z^*_{LR^i}(\boldsymbol{\beta}^5) = \min_{g=1,\ldots,G}\{Z^*_{LR^{ig}}(\boldsymbol{\beta}^5)\}$. As soon as we solve all subproblems RDAP$_i(\boldsymbol{\beta}^5)$, we can calculate the optimal value of the RDAP($\boldsymbol{\beta}^5$) as $Z^*_{LR5}(\boldsymbol{\beta}^5) = \sum_{i=1}^{I} Z^*_{LR5^i}(\boldsymbol{\beta}^5) + \sum_{j=1}^{J} \beta_j^5 \overline{q}_j$ for given multiplier vectors $\boldsymbol{\beta}^5$. The rest of the calculations are the same with the ones presented in Chapter 5. $Z^*_{LR5}(\boldsymbol{\beta}^5)$ is a lower bound on the optimal value of the DAP for any Lagrange multiplier vectors $\boldsymbol{\beta}^5$.

8.1.1.2. Reformulation-Linearization Technique Based Lower Bounding Procedure. (Sherali *et al.*, 2002) devise two lower bounding procedures for the LCMWP. One of them is the RLT based lower bounding formulation of the CMWP. RLT first defines new variables to denote distance measures between the facilities and customers and carry its nonlinearity into the constraint set of the CMWP. Then, the lower bounding supports for these nonlinear terms in the constraints are added and the bilinear terms are reformulated to linearize them. Clearly, the resulting formulation is linear and constitutes a lower bound on the optimal solution of the CMWP. We have also employed the RLT formulation given with the so-called constraints "(15a) – (15j)" and "(16b)" in the study by (Sherali *et al.*, 2002) as the second lower bounding procedure for the SABB algorithm. A similar RLT formulation is also developed for the MCMWP in the next section and we omit the original RLT formulation for the CMWP to avoid repetitions.

The other lower bounding procedure is first proposed by (Sherali *et al.*, 2002). We term it as SAS, which is given as follows.

SAS:

$$Z_{SAS} = \min_{\mathbf{x}} \sum_{i=1}^{I} \sum_{j=1}^{J} l_{ij} c_{ij} d(\mathbf{x}_i, \mathbf{a}_j), \tag{8.15}$$

which reduces to solving I WP's. Both block norm and RLT based lower bounding procedures are strengthened by the SAS lower bound.

8.1.1.3. Upper Bounding Procedures. The computed lower bounds are used to check the current BB node for fathoming. The objective value of the best feasible solution is employed to make a decision. Actually, upper bounding feasible solutions are necessary both at the root node (initialization) and at the subsequent nodes of the BB tree. We summarize the initial upper bounding procedure, which is proposed in the work by (Sherali *et al.*, 2002), that we have also employed in this work.

Initial upper bounding procedure: At the root node, two Capacitated ALA (CALA) heuristics, which are explained in the following, are run and the one with the smallest objective value is used as the initial value of Z_{UB}^{best}. For that purpose, the customer locations are enclosed within the tightest rectangle and it is sliced along the x-axis into I equal parts. Then, the demand quantities in each slice are aggregated and sorted in increasing order. Facilities are also sorted in increasing order of their capacities. Lastly, each aggregate demand is assigned to a facility according to their increasing order and the demands of customers are split among facilities starting from left to right, i.e., unsatisfied demand of customers in a slice is merged to the next slice and thus the next facility. Once a valid transportation plan is obtained, the corresponding optimum facility locations are found and the CALA heuristic is run. This process is also repeated for the y-axis and the best of two feasible solutions is used as the initial upper bound value for the SABB algorithm. More details can be found in (Sherali *et al.*, 2002) for the initial upper bounding procedure. Other initial upper bounding procedures can also be implemented for the SABB algorithm but, we prefer to use this procedure to perform a fair comparison between existing methods of (Sherali *et al.*, 2002) and our BB algorithms.

Intermediate upper bounding procedure: We take the advantage of the CALA heuristic to find upper bounds at intermediate SABB nodes. The lower bounding procedure (including the block norm based, the RLT based or the SAS lower bounding) solutions return the optimal facility locations of the SABB subproblem which are then used to initialize the CALA heuristic to obtain a feasible solution on the CMWP in order to update the incumbent solution value Z_{UB}^{best} when necessary. At the leaf nodes of the BB tree an extreme point is reached and Weiszfeld's procedure is run to find an upper bound. We continue to run the CALA heuristic to obtain an improved feasible solution and to update the value of Z_{UB}^{best}.

8.1.2. Branching Variable Selection Strategies

Another feature of the proposed SABB algorithm is the branching step. In this step, we select the allocation variable which will be added to the positive arc set, $\mathcal{W}^+$. The sequence of variables on which we have selected to branch greatly affects the performance of the SABB algorithm. Therefore, this choice should be done with utmost care. For that purpose, we have implemented three different branching variable selection strategies. Two of them are originally introduced by (Sherali *et al.*, 2002) and the third one is a new branching variable selection rule. In the following we present these three branching variable selection strategies (BrSs).

BrS1: Let $\overline{\mathbf{w}}$ be the current allocation vector satisfying the bounding restrictions imposed. Then, the branching variable w_{ij} is selected with the following rule:

BrS1:

$$(i,j) = \underset{(i',j')\in\mathcal{W}^F}{\arg\, lex\max} \left\{ \overline{w}_{i'j'}, c_{i'j'}(\widehat{u}_{i'j'} - \widehat{l}_{i'j'}) \right\}, \tag{8.16}$$

where ties are broken arbitrarily. Note that in Equation 8.16 two vectors are considered and thus, the selection of the branching variable is done by taking lexicographically maximum element of the vectors.

BrS2: The second rule is very similar to the first one.

BrS2:

$$(i,j) = \underset{(i',j')\in\mathcal{W}^F}{\arg\max} \left\{ (\widehat{u}_{i'j'} - \widehat{l}_{i'j'}) \min \left\{ (\widehat{u}_{i'j'} - \overline{w}_{i'j'}), (\overline{w}_{i'j'} - \widehat{l}_{i'j'}) \right\} \right\}. \tag{8.17}$$

BrS3: The constraints given by Equation 3.7 and 3.8 can be rewritten in the form of two inequalities as

$$\sum_{j=1}^{J} w_{ij} \geq s_i \quad \text{and} \quad \sum_{j=1}^{J} w_{ij} \leq s_i \qquad i = 1, \ldots, I, \tag{8.18}$$

$$\sum_{i=1}^{I} w_{ij} \geq q_j \quad \text{and} \quad \sum_{i=1}^{I} w_{ij} \leq q_j \qquad j = 1, \ldots, J. \tag{8.19}$$

Then, the maximum slack values of these constraints can be expressed by

$$SL_i = s_i - \sum_{j=1}^{J} \widehat{l}_{ij} \quad \text{and} \quad SU_i = \sum_{j=1}^{J} \widehat{u}_{ij} - s_i \qquad i = 1, \ldots, I, \tag{8.20}$$

and

$$QL_j = q_j - \sum_{i=1}^{I} \widehat{l}_{ij} \quad \text{and} \quad QU_j = \sum_{i=1}^{I} \widehat{u}_{ij} - q_j \qquad j = 1, \ldots, J. \tag{8.21}$$

Hence we propose the following new branching variable selection strategy.

BrS3:

$$(i,j) = \underset{(i',j')\in\mathcal{W}^F}{\arg\max} \left\{ c_{i'j'}(u_{i'j'} - l_{i'j'}) \max \left\{ \begin{array}{c} \min\{SL_{i'}, SU_{i'}\}, \\ \min\{DL_{j'}, DU_{j'}\} \end{array} \right\} \right\}. \tag{8.22}$$

There are several other strategies used in the study of (Sherali *et al.*, 2002) other than Equation 8.16 and 8.17. However, Equation 8.16 and 8.17 are the most promising ones and thus we have taken them into account for assessing the performance of Equation 8.22.

8.2. Solution of the Multi-commodity Capacitated Multi-facility Weber Problem

(Sherali and Tunçbilek, 1992) employ a binary partitioning strategy and tried to implicitly enumerate all extreme points for the SECMWP. The authors also devise a cycle

prevention mechanism using the tree property of bases: the flows which constitute a cycle with the arcs defined by the positive variables are set to zero. This setting accelerates the enumeration of extreme points. Furthermore, they also propose a logical test which may tighten lower and upper bounds on allocation variables. Consequently, the logical test also helps reaching an extreme point more quickly. (Sherali *et al.*, 2002) adapt these logical tests and cycle prevention mechanism and propose a new BB aproach which performs a continuous partitioning of the allocation space of the LCMWP. Continuous partitioning, which is different than the dichotomization of allocation variables, divides the allocation space into two subspaces.

The cycle prevention mechanism which is originally devised for the CMWP can not be directly adapted for the MCMWP. The reason for this is that a cycle may occur at an extreme point of the MCMWP polyhedron, which is defined on the bipartite graph constructed by positive flows of each commodity. This is caused by the upper bounds u_{ij} imposed on the total quantity of flows between each facility and customer pair. However, for some particular extreme point each commodity may satisfy the tree structure. It is not easy to design a cycle prevention mechanism which works for all extreme points of the MCMWP polyhedron. Positive allocation variables at an extreme point of the CMWP is at most $I + J - 1$. Additionally, there is not a straightforward formula which states the number of nonnegative variables in a basic feasible solution of the MCMWP. In short, a binary partitioning scheme for the MCMWP, similar to the one suggested by (Sherali and Tunçbilek, 1992) for the CMWP, is not applicable. On the other hand, the continuous partitioning strategy developed by (Sherali *et al.*, 2002) for the CMWP can be used for the MCMWP as well. For that purpose, we impose the following bounds on the allocation variables w_{ijk}

$$l_{ijk} \leq w_{ijk} \leq u_{ijk} \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K. \tag{8.23}$$

Initially, these bounds can be set as $l_{ijk} = 0$ and $u_{ijk} = \min\{s_{ik}, q_{jk}, u_{ij}\}$ for $i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K$, respectively. At each step of the MABB algorithm, an active node (a node which is not yet pruned) denoted by t is selected for further exploration. Let $\mathcal{W}^F = \{(i, j, k) : l_{ijk}^{(t)} < u_{ijk}^{(t)}\}$ and $\mathcal{W}^E = \{(i, j, k) : l_{ijk}^{(t)} = u_{ijk}^{(t)}\}$ denote free variable set and equality allocation variable set associated with node t, respectively. Then, an allocation

variable $w_{ijk} \in \mathcal{W}^F$ is selected and the corresponding interval is divided into two intervals such that $[l_{ijk}^{(t)}, \widetilde{w}_{ijk}^1]$ and $[\widetilde{w}_{ijk}^2, u_{ijk}^{(t)}]$ with $l_{ijk} \leq \widetilde{w}_{ijk}^1 \leq \widetilde{w}_{ijk}^2 \leq u_{ijk}^{(t)}$, where $\widetilde{w}_{ijk}^1$ and $\widetilde{w}_{ijk}^2$ are not necessarily equal. All the remaining allocation variables are inherited from the parent node, namely node t, as they are. These two allocation subsets result in two MABB subproblems which are given by Equation 2.2 – 2.5 and 8.23. A lower and an upper bound on these two subproblems are calculated and added to the BB tree $\mathcal{T}$ whenever they are eligible. The selection of node t is done by the "*best-first search*" (BFS) strategy such that node t has the lowest lower bound among all nodes of the BB tree $\mathcal{T}$. The algorithm proceeds until either BB tree $\mathcal{T}$ is empty or the lower bounds are within a predetermined limit from the best upper bound Z_{UB}^{best} found namely, the incumbent solution value. In the following discussion, we present three lower bounding procedures and an upper bounding procedure. Next, we propose several branching variable selection strategies (BrSs) together with the partitioning and search strategy. Finally, a formal outline of the MABB algorithm is given.

8.2.1. Bounding Procedures

8.2.1.1. Block Norm Based Lower Bounding Procedure. A MABB subproblem is stated by Equation 2.2 – 2.5 and 8.23. Consider two distance functions $\underline{d}(\mathbf{x}, \mathbf{0})$ and $d(\mathbf{x}, \mathbf{0})$ such that $\underline{d}(\mathbf{x}, \mathbf{0}) \leq d(\mathbf{x}, \mathbf{0})$ holds for every $\mathbf{x} \in \mathbb{R}^2$. Clearly, the optimum solutions $\underline{Z}^*$ and Z^* of the corresponding MABB subproblems which use $\underline{d}(\mathbf{x}, \mathbf{0})$ and $d(\mathbf{x}, \mathbf{0})$ in Equation 2.2 also satisfy $\underline{Z}^* \leq Z^*$. Therefore, it is possible to use lower bounding distance functions, namely $\underline{d}(\mathbf{x}, \mathbf{0})$, on the original MABB subproblem given by Equation 2.2 – 2.5 and 8.23. In particular, we offer to use block norms as lower bounding distance functions. In the MDAP3 formulation, which is the approximating MILP formulation of the MABB given by Equation 2.2 – 2.5 and 8.23, the allocation variables y_{ijkg} depend on the candidate facility locations g which is not the case for the w_{ijk} in the MCMWP. We can not directly set $l_{ijk} \leq y_{ijkg} \leq u_{ijk}$ for all allocations. Taking the summation over candidate locations g implies that $Gl_{ijk} \leq \sum_{g=1}^{G} y_{ijkg} = w_{ijk} \leq Gu_{ijk}$, which is not feasible for the original MCMWP. Thus, we replace Equation 8.23 by the constraints

$$l_{ijk} v_{ig} \leq y_{ijkg} \leq u_{ijk} v_{ig} \quad i = 1, ..., I; j = 1, ..., J; k = 1, ..., K; g = 1, ..., G \tag{8.24}$$

in order to adapt the lower bounding approach for the BB algorithm. Hence, we substitute $y_{ijkg} = \overline{y}_{ijkg} + l_{ijk} v_{ig}$ for $i = 1, ..., I; j = 1, ..., J; k = 1, ..., K$ and $g = 1, ..., G$. Then, the

MDAP3 formulation of the lower bounding MABB subproblem associated with node t can be given as

MDAP3$^{(t)}$:

$$\min \quad Z_{MDAP3} = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G} c_{ijkg}\overline{y}_{ijkg} + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G} c_{ijkg} l_{ijk}^{(t)} v_{ig} \tag{8.25}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} \overline{y}_{ijkg} = \overline{s}_{ik} v_{ig} \qquad i = 1,\ldots,I; k = 1,\ldots,K; g = 1,\ldots,G, \tag{8.26}$$

$$\sum_{i=1}^{I}\sum_{g=1}^{G} \overline{y}_{ijkg} = \overline{q}_{jk} \qquad j = 1,\ldots,J; k = 1,\ldots,K, \tag{8.27}$$

$$\sum_{g=1}^{G} v_{ig} = 1 \qquad i = 1,\ldots,I, \tag{8.28}$$

$$\sum_{k=1}^{K}\sum_{g=1}^{G} \overline{y}_{ijkg} \leq \overline{u}_{ij} \qquad i = 1,\ldots,I; j = 1,\ldots,J, \tag{8.29}$$

$$0 \leq \overline{y}_{ijkg} \leq \overline{u}_{ijk}^{(t)} \qquad i = 1,\ldots,I; j = 1,\ldots,J; k = 1,\ldots,K; g = 1,\ldots,G, \tag{8.30}$$

$$v_{ig} \in \{0,1\} \qquad i = 1,\ldots,I; g = 1,\ldots,G, \tag{8.31}$$

where the parameters $\overline{s}_{ik}$, $\overline{q}_{jk}$ and $\overline{u}_{ijk}^{(t)}$ are updated with the latest bounds on the variables at node t: $\overline{s}_{ik} = s_{ik} - \sum_{i=1}^{I}\sum_{g=1}^{G} l_{ijk}^{(t)} v_{ig} = s_{ik} - \sum_{i=1}^{I} l_{ijk}^{(t)}$, $\overline{q}_{jk} = q_{jk} - \sum_{j=1}^{J}\sum_{g=1}^{G} l_{ijk}^{(t)} v_{ig} = q_{ik} - \sum_{j=1}^{J} l_{ijk}^{(t)}$ and $\overline{u}_{ij} = u_{ij} - \sum_{k=1}^{K}\sum_{g=1}^{G} l_{ijk}^{(t)} v_{ig} = u_{ij} - \sum_{k=1}^{K} l_{ijk}^{(t)}$ follows by $\sum_{g=1}^{G} v_{ig} = 1$ for $i = 1,\ldots,I$. Lastly, $\overline{u}_{ijk}^{(t)} = u_{ijk}^{(t)} - l_{ijk}^{(t)}$ denotes current upper bounds on the allocation variables. Constraints given by Equation 8.26, 8.27, and 8.29 play similar role as the bounded Multi-commodity Transportation Problem (MTP) constraints given by Equation 2.3, 2.4 and 2.5, respectively. Constraints given by Equation 8.28 enforce each facility to be opened at exactly on one of the candidate locations. Constraints given by Equation 8.30 are valid upper bounds which substitute $0 \leq \overline{y}_{ijkg} \leq (u_{ijk}^{(t)} - l_{ijk}^{(t)}) v_{ig}$. Fortunately, this does not affect the optimum solution of the MABB subproblem. Given a facility i with $v_{ig^*} = 1$ for some candidate location g^*, then $\overline{y}_{ijkg} = 0$ is satisfied by Equation 8.26 and $0 \leq \overline{y}_{ijkg} \leq (u_{ijk}^{(t)} - l_{ijk}^{(t)})$ is redundant for all $j = 1,...,J; k = 1,...,K; g = 1,...,G$ and $g \neq g^*$. On the other hand, constraints given by Equation 8.30 imply valid bounds on allocations quantities. Note that the MDAP3 reduces to the MDAP1 formulation to produce approximate solutions for the MCMWP where no bounds on the allocation variables are imposed, namely at the root node of the BB tree. Lower and upper bounds on variables require additional binary variable terms which vanish from the formulation when lower bounds take zero values.

When we relax constraints given by Equation 8.27 and 8.29 with respectively Lagrangean multipliers β_{jk}^6 and μ_{ij}^6 we obtain the following Lagrangean subproblem:

RMDAP3$^{(t)}$($\boldsymbol{\beta}^6$, $\boldsymbol{\mu}^6$):

$$\min Z_{LR6}(\boldsymbol{\beta}^6, \boldsymbol{\mu}^6) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{g=1}^{G}\left[(c_{ijkg} - \beta_{jk}^6 + \mu_{ij}^6)\overline{y}_{ijkg} + c_{ijkg}l_{ijk}^{(t)}v_{ig}\right] + \sum_{j=1}^{J}\sum_{k=1}^{K}\beta_{jk}^6\overline{q}_{jk} - \sum_{i=1}^{I}\sum_{j=1}^{J}\mu_{ij}^6\overline{u}_{ij} \tag{8.32}$$

$$\text{s.t.} \quad \text{Equation 8.26, 8.28, 8.30 and 8.31.} \tag{8.33}$$

We skip the details of solving the RMDAP3 and using the SO algorithm for the sake of conciseness since the LR scheme is very similar to the LR schemes RMDAP1 in Chapter 5 and RDAP for the CMWP as explained.

8.2.1.2. Reformulation-Linearization Technique Based Lower Bounding Procedure.

The MCMWP formulation given by Equation 2.2 – 2.6 can be rewritten as

$$\min \quad z = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} w_{ijk}c_{ijk}\alpha_{ij} \tag{8.34}$$

$$\text{s.t.} \quad \alpha_{ij} = (|x_{i1} - a_{j1}|^r + |x_{i2} - a_{j2}|^r)^{1/r} \quad i = 1, \ldots, I; j = 1, \ldots, J \tag{8.35}$$

$$\text{Equation } 2.3 - 2.5 \text{ and } 8.23, \tag{8.36}$$

where continuous variables α_{ij} measure the distance between facility i and customer j. The RLT approach is applied in order to get a lower bounding LP formulation on the formulation given by Equation 8.34 – 8.36, which has bilinear objective function and nonlinear constraints. The nonlinear constraint set is approximated by generating some supports on it as suggested by (Sherali *et al.*, 2002) for the CMWP case. Let $\mathscr{P}$ be the number of facets of the convex hull of the customer locations defined by a set of inequalities of the form

$$\psi_{1\varrho}x_{i1} + \psi_{2\varrho}x_{i2} \leq \psi_{0\varrho} \qquad i = 1, \ldots, I; \varrho = 1, \ldots, \mathscr{P}. \tag{8.37}$$

Also, we define the following constraint sets

$$\begin{aligned} &\alpha_{ij} \geq x_{i1} - a_{j1}, \alpha_{ij} \geq x_{i2} - a_{j2}, \\ &\alpha_{ij} \geq a_{j1} - x_{i1}, \alpha_{ij} \geq a_{j2} - x_{i2}, \end{aligned} \qquad i = 1, \ldots, I; j = 1, \ldots, J, \tag{8.38}$$

$$\begin{aligned}
&\alpha_{ij} \geq 2^{(1-r)/r}\left(x_{i1} - a_{j1} + x_{i2} - a_{j2}\right),\\
&\alpha_{ij} \geq 2^{(1-r)/r}\left(x_{i1} - a_{j1} - x_{i2} + a_{j2}\right),\\
&\alpha_{ij} \geq 2^{(1-r)/r}\left(a_{j1} - x_{i1} + x_{i2} - a_{j2}\right),\\
&\alpha_{ij} \geq 2^{(1-r)/r}\left(a_{j1} - x_{i1} - x_{i2} + a_{j2}\right)
\end{aligned} \qquad i = 1,\ldots,I; j = 1,\ldots,J, \tag{8.39}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\alpha_{ij} \geq z_1 \tag{8.40}$$

$$0 \leq \alpha_{ij} \leq d_{\alpha_{ij}} \qquad i = 1,\ldots,I; j = 1,\ldots,J, \tag{8.41}$$

where Equation 8.38 implies that $\alpha_{ij} \geq \max\left\{\left|x_{i1} - a_{j1}\right|, \left|x_{i2} - a_{j2}\right|\right\}$ and Equation 8.39 imposes $\alpha_{ij} \geq 2^{(1-r)/r}\ (|x_{i1} - a_{j1}| + |x_{i2} - a_{j2}|)$. z_1 in Equation 8.40 is calculated by $z_1 = \min_{\mathbf{x}} \sum_{i=1}^{I}\sum_{j=1}^{J}\left(|x_{i1} - a_{j1}|^r + |x_{i2} \quad -a_{j2}|^r\right)^{1/r}$. Lastly, $d_{\alpha_{ij}} = \max_{\bar{j}=1,\ldots,J}\left(\left|a_{\bar{j}1} - a_{j1}\right|^r + \left|a_{\bar{j}2} - a_{j2}\right|^r\right)^{1/r}$ for $i = 1,\ldots,I; j = 1,\ldots,J$ in Equation 8.41.

In order to linearize bilinear terms, which occur after the reformulation of constraints, we define new decision variables

$$\gamma_{ijk} = w_{ijk}\alpha_{ij}, \theta_{ijk} = w_{ijk}x_{i1} \text{ and } \phi_{ijk} = w_{ijk}x_{i2},\ \ i = 1,\ldots,I; j = 1,\ldots,J; k = 1,\ldots,K. \tag{8.42}$$

Reformulation: We multiply each inequality given by Equation 8.41 with each inequality given by Equation 8.23. Then, all inequalities given by Equation 8.38, 8.39 and 8.37 are multiplied by each of the inequalities in Equation 8.23. We generate the following equalities

$$\left(\sum_{j=1}^{J} w_{ijk} - s_{ik}\right)x_{i1} = 0 \text{ and } \left(\sum_{j=1}^{J} w_{ijk} - s_{ik}\right)x_{i2} = 0,\ \ i = 1,\ldots,I; k = 1,\ldots,K. \tag{8.43}$$

Additionally, the RLT based formulation is strengthened with the following two constraints:

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} c_{ijk}\overline{w}_{ijk}\alpha_{ij} \geq \overline{z}, \tag{8.44}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} c_{ijk}\hat{w}_{ijk}\alpha_{ij} \geq \hat{z}, \tag{8.45}$$

where $\overline{w}_{ijk}$ and $\hat{w}_{ijk}$ are the allocation quantities determined in the parent node upper bound solution and the best feasible solution found so far, and $\overline{z}$ and $\hat{z}$ are their corresponding

objective values, respectively. To calculate $\overline{z}$ and $\hat{z}$ a pure location problem is solved with the associated allocation quantities, i.e., $\overline{z} = \min_{\mathbf{x}} \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} c_{ijk}\overline{w}_{ijk} \left(|x_{i1} - a_{j1}|^r + |x_{i2} - a_{j2}|^r\right)^{1/r}$.

The other supporting constraint uses the first order information of the distance function d. Note that $d(\mathbf{x}, \mathbf{a}) \geq d(\mathbf{a}_0, \mathbf{a}) + \nabla d(\mathbf{a}_0, \mathbf{a})(\mathbf{x} - \mathbf{a}_0)$ holds where $\mathbf{a}_0$ and $\mathbf{a}$ are fixed points on the plane. Here, we employ customer locations as fixed points as recommended by (Sherali *et al.*, 2002). Therefore, the following supporting constraint is added to the RLT based formulation:

$$\alpha_{ij} \geq d(\mathbf{a}_{\overline{j}}, \mathbf{a}_j) + \nabla d(\mathbf{a}_{\overline{j}}, \mathbf{a}_j)(\mathbf{x}_i - \mathbf{a}_{\overline{j}}) \quad i = 1, \ldots, I; j = 1, \ldots, J; \overline{j} = 1, \ldots, J \text{ and } \overline{j} \neq j \tag{8.46}$$

Observe that $\mathbf{a}_{\overline{j}} \neq \mathbf{a}_j$ must necessarily hold since the derivative is not defined for those points. It is also possible to obtain a tighter support by multiplying both sides of Equation 8.46 with the allocation variables w_{ijk}. Nevertheless, we have observed that adding these supports deteriorates the performance of the MABB algorithm employing the RLT. As a result, we have limited ourselves with Equation 8.46 for the RLT based lower bounding LP formulation to produce another lower bound for the MABB algorithm. The RLT based lower bounding formulation for the MCMWP at a BB node t is as follows:

RLT$^{(t)}$:

$$\min \; Z_{RLT} = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} c_{ijk}\gamma_{ijk} \tag{8.47}$$

$$\text{s.t. } l_{ijk}^{(t)}\alpha_{ij} \leq \gamma_{ijk} \leq u_{ijk}^{(t)}\alpha_{ij} \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.48}$$

$$d_{\alpha_{ij}} w_{ijk} + u_{ijk}^{(t)}(\alpha_{ij} - d_{\alpha_{ij}}) \leq \gamma_{ijk} \leq d_{\alpha_{ij}} w_{ijk} + l_{ijk}^{(t)}(\alpha_{ij} - d_{\alpha_{ij}})$$
$$i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.49}$$

$$\theta_{ijk} - w_{ijk}a_{j1} + l_{ijk}^{(t)}(\alpha_{ij} - x_{i1} + a_{j1}) \leq \gamma_{ijk} \leq \theta_{ijk} - w_{ijk}a_{j1}$$
$$+ u_{ijk}^{(t)}(\alpha_{ij} - x_{i1} + a_{j1}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.50}$$

$$w_{ijk}a_{j1} - \theta_{ijk} + l_{ijk}^{(t)}(\alpha_{ij} + x_{i1} - a_{j1}) \leq \gamma_{ijk} \leq w_{ijk}a_{j1} - \theta_{ijk}$$
$$+ u_{ijk}^{(t)}(\alpha_{ij} + x_{i1} - a_{j1}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.51}$$

$$\phi_{ijk} - w_{ijk}a_{j2} + l_{ijk}^{(t)}(\alpha_{ij} - x_{i2} + a_{j2}) \leq \gamma_{ijk} \leq \phi_{ijk} - w_{ijk}a_{j2}$$
$$+ u_{ijk}^{(t)}(\alpha_{ij} - x_{i2} + a_{j2}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.52}$$

$$w_{ijk}a_{j2} - \phi_{ijk} + l_{ijk}^{(t)}(\alpha_{ij} + x_{i2} - a_{j2}) \leq \gamma_{ijk} \leq w_{ijk}a_{j2} - \phi_{ijk}$$
$$+ u_{ijk}^{(t)}(\alpha_{ij} + x_{i2} - a_{j2}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.53}$$

$$\theta_{ijk} + \phi_{ijk} - w_{ijk}(a_{j1} + a_{j2}) + l_{ijk}(2^{(r-1)/r}\alpha_{ij} - x_{i1} + a_{j1} - x_{i2} + a_{j2})$$
$$\leq 2^{(r-1)/r}\gamma_{ijk} \leq \theta_{ijk} + \phi_{ijk} - w_{ijk}(a_{j1} + a_{j2}) + u_{ijk}(2^{(r-1)/r}\alpha_{ij} - x_{i1}$$
$$+ a_{j1} - x_{i2} + a_{j2}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.54}$$

$$\theta_{ijk} - \phi_{ijk} + w_{ijk}(a_{j2} - a_{j1}) + l_{ijk}(2^{(r-1)/r}\alpha_{ij} - x_{i1} + a_{j1} + x_{i2} - a_{j2})$$
$$\leq 2^{(r-1)/r}\gamma_{ijk} \leq \theta_{ijk} - \phi_{ijk} + w_{ijk}(a_{j2} - a_{j1}) + u_{ijk}(2^{(r-1)/r}\alpha_{ij} - x_{i1}$$
$$+ a_{j1} + x_{i2} - a_{j2}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.55}$$

$$\phi_{ijk} - \theta_{ijk} + w_{ijk}(a_{j1} - a_{j2}) + l_{ijk}(2^{(r-1)/r}\alpha_{ij} + x_{i1} - a_{j1} - x_{i2} + a_{j2})$$
$$\leq 2^{(r-1)/r}\gamma_{ijk} \leq \phi_{ijk} - \theta_{ijk} + w_{ijk}(a_{j1} - a_{j2}) + u_{ijk}(2^{(r-1)/r}\alpha_{ij} + x_{i1}$$
$$- a_{j1} - x_{i2} + a_{j2}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.56}$$

$$- \phi_{ijk} - \theta_{ijk} + w_{ijk}(a_{j1} + a_{j2}) + l_{ijk}(2^{(r-1)/r}\alpha_{ij} + x_{i1} - a_{j1} + x_{i2} - a_{j2})$$
$$\leq 2^{(r-1)/r}\gamma_{ijk} \leq -\phi_{ijk} - \theta_{ijk} + w_{ijk}(a_{j1} + a_{j2}) + u_{ijk}^{(t)}(2^{(r-1)/r}\alpha_{ij} + x_{i1}$$
$$- a_{j1} + x_{i2} - a_{j2}) \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.57}$$

$$w_{ijk}\psi_{0\varrho} - u_{ijk}^{(t)}(\psi_{0\varrho} - \psi_{1\varrho}x_{i1} - \psi_{2\varrho}x_{i2}) \leq \psi_{1\varrho}\theta_{ijk} + \psi_{2\varrho}\phi_{ijk} \leq w_{ijk}\psi_{0\varrho} - l_{ijk}^{(t)}(\psi_{0\varrho}$$
$$- \psi_{1\varrho}x_{i1} - \psi_{2\varrho}x_{i2}) \quad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K, \varrho = 1, \ldots, \mathscr{P}, \tag{8.58}$$

$$\sum_{j=1}^{J}\theta_{ijk} - s_{ik}x_{i1} = 0 \text{ and } \sum_{j=1}^{J}\phi_{ijk} - s_{ik}x_{i2} = 0 \qquad i = 1, \ldots, I; k = 1, \ldots, K, \tag{8.59}$$

$$\text{Equation } 2.3 - 2.6, 8.40, 8.41 \text{ and } 8.44 - 8.46. \tag{8.60}$$

In addition to MDAP3 (or RMDAP3) and RLT lower bounds, the following pure location problem also constitutes a lower bound on the MABB subproblem at node t:

$$Z_{MSAS} = \min_{\mathbf{x}} \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} l_{ijk}^{(t)} c_{ijk} d(\mathbf{x}_i, \mathbf{a}_j). \tag{8.61}$$

Here Equation 8.61 can be decomposed into I WP's. Once solved, the sum of the optimal values of I WP's give a lower bound. Both block norm based and RLT based lower bounds are tightened with Z_{MSAS} at each lower bounding step. Notice that MSAS is the multi-commodity version of the SAS given by Equation 8.15.

8.2.1.3. Upper Bounding Procedures. The lower bound value computed at a MABB node is used to check the status of that node for pruning. The incumbent solution value Z_{UB}^{best} is used to make such a pruning decision. In fact, upper bounding feasible solutions are necessary both at the root node (initialization) and at intermediate nodes of the BB tree.

Initial upper bounding procedure: At the root node, we run MCALA heuristic twice with different initializations and use the outcome with the smallest objective value as the initial incumbent solution value Z_{UB}^{best}. Two MCALA heuristics are initialized as suggested by (Sherali *et al.*, 2002) for the CMWP where they select initial allocations by a greedy-like procedure before performing a CALA heuristic as discussed in Section 8.1.1.3 for the ABB algorithm. Here, we adapt their method for the MCMWP. The customer locations are enclosed within the smallest rectangle and it is sliced along x-axis into I equal parts. Then, the demand for all commodities of each customer which lies within a slice is aggregated. These total demand quantities of each slice are sorted in increasing order. Note that there are I slices and hence I aggregated demand values. Similarly, facility capacities are accumulated for all commodities such that $s_i^{total} = \sum_{k=1}^{K} s_{ik}$ holds where s_i^{total} is the total facility capacity for $i = 1, \ldots, I$. Total facility capacity values s_i^{total} are also sorted in increasing order. Lastly, each aggregate demand is assigned to a facility with the same order. Then, total facility capacities are split among the customers from left to right. When a customer demand for a commodity is not totally satisfied, its demand is met by the next facility in the order. Once a feasible allocation vector is obtained, a MCALA is run starting with those allocations. The process is also repeated for the y-axis and the minimum of these two solution values is employed as the initial upper bound value.

Intermediate upper bounding procedure: At the intermediate nodes, the MCALA heuristic is used to find feasible solutions. The initial locations (or allocations) of the MCALA heuristic affects its performance significantly. Observe that the bounds given by Equation 8.23 imposed at a node of the MABB algorithm on allocation variables may lead to a feasible allocation vector that is worse than the incumbent solution and, in some cases, to infeasible solutions. As a result, we prefer to employ the location vectors produced by the lower bounding procedure at an intermediate MABB node to initialize a MCALA heuristic. When $\mathcal{W}^F$ is empty, we obtain an extreme point of the MCMWP. In this case, an upper bound can be found by optimally solving I WPs. Lastly, the Z_{UB}^{best} is updated if an improvement is achieved by the upper bound obtained with the MCALA heuristic for the original MCMWP.

8.2.2. Other Features of the Allocation Space Based Branch-and-Bound Algorithm

8.2.2.1. Logical Test. Constraints given by Equation 8.23 impose both lower and upper bounds on allocation variables. Given l_{ijk} and u_{ijk} values, it is possible to obtain improved bounds by a logical test. It consists of a sequential update mechanism which employs the maximum slack values defined on the bounded MTP constraints given by Equation 2.3 – 2.5 and 8.23 to tighten l_{ijk} and u_{ijk} values. Logical tests are first developed by (Sherali and Tunçbilek, 1992) for the CMWP. Here, we implement a similar approach for the MCMWP. The constraints given by Equation 2.3 – 2.5 can be rewritten as follows:

$$\sum_{j=1}^{J} w_{ijk} \geq s_{ik} \quad \text{and} \quad \sum_{j=1}^{J} w_{ijk} \leq s_{ik}, \qquad i = 1, \ldots, I; k = 1, \ldots, K, \tag{8.62}$$

$$\sum_{i=1}^{I} w_{ijk} \geq q_{jk} \quad \text{and} \quad \sum_{i=1}^{I} w_{ijk} \leq q_{jk}, \qquad j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.63}$$

$$\sum_{k=1}^{K} w_{ijk} \leq u_{ij}, \qquad i = 1, \ldots, I; j = 1, \ldots, J. \tag{8.64}$$

The maximum slack values of these constraints are

$$SL_{ik} = s_{ik} - \sum_{j=1}^{J} l_{ijk} \quad \text{and} \quad SU_{ik} = \sum_{j=1}^{J} u_{ijk} - s_{ik} \qquad i = 1, \ldots, I; k = 1, \ldots, K, \tag{8.65}$$

$$QL_{jk} = q_{jk} - \sum_{i=1}^{I} l_{ijk} \quad \text{and} \quad QU_{jk} = \sum_{i=1}^{I} u_{ijk} - q_{jk} \qquad j = 1, \ldots, J; k = 1, \ldots, K, \tag{8.66}$$

$$UL_{ij} = u_{ij} - \sum_{k=1}^{K} l_{ijk} \qquad i = 1, \ldots, I; j = 1, \ldots, J. \tag{8.67}$$

In the next proposition we suggest using these maximum slack values to calculate new lower and upper bounds l_{ijk}^{new} and u_{ijk}^{new} on the allocation variables, respectively. These bounds are updated when one or more bounds are changed.

Proposition 8.1. *The following set of lower and upper bounds are valid implications of Equation 8.65 – 8.67.*

$$\begin{aligned} l_{ijk}^{new} &= \max\{l_{ijk}, u_{ijk} - SU_{ik}, u_{ijk} - QU_{jk}\}, \\ u_{ijk}^{new} &= \min\{u_{ijk}, SL_{ik} + l_{ijk}, QL_{jk} + l_{ijk}, UL_{ij} + l_{ijk}\} \\ & \qquad i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K. \end{aligned} \tag{8.68}$$

Moreover, at any stage of a continuous sequential updating process, the order of computing l_{ijk}^{new} or u_{ijk}^{new} for a particular variable w_{ijk} does not change the outcome.

Proof. $l_{ijk}^{new} \geq l_{ijk}$ holds trivially. Additionally, for commodity k, if all facilities except facility i send flows to customer j at their upper bounds, then facility i should send an amount of at least $q_{jk} - \sum_{i'=1,i'\neq i}^{I} u_{i'jk} = u_{ijk} - \sum_{i'=1}^{I} u_{i'jk} + q_{jk} = u_{ijk} - \left(\sum_{i'=1}^{I} u_{i'jk} - q_{jk}\right) = u_{ijk} - QU_{jk}$. Similarly, for commodity k, if a facility i sends flows to all customers other than customer j at its upper bound, then it should ship all its remaining capacity which is at least $s_{ik} - \sum_{j'=1,j'\neq j}^{J} u_{ij'k} = u_{ijk} - \sum_{j'=1}^{J} u_{ij'k} + s_{ik} = u_{ijk} - \left(\sum_{j'=1}^{J} u_{ij'k} - s_{ik}\right) = u_{ijk} - SU_{ik}$.

Now consider the upper bounds, obviously $u_{ijk}^{new} \leq u_{ijk}$ holds. Given a commodity k, if all facilities except facility i send flows at their lower bounds to meet the demand of customer j, then facility i can not ship an amount more than $q_{jk} - \sum_{i'=1,i'\neq i}^{I} l_{i'jk} = \left(q_{jk} - \sum_{i'=1}^{I} l_{i'jk}\right) + l_{ijk} = QL_{jk} + l_{ijk}$ to customer j. Furthermore, for a commodity k, if a facility i sends flows at its lower bounds to all customers other than customer j, then it can not ship an amount more than $s_{ik} - \sum_{j'=1,j'\neq j}^{J} l_{ij'k} = \left(s_{ik} - \sum_{j'=1}^{J} l_{ij'k}\right) + l_{ijk} = SL_{ik} + l_{ijk}$ to customer j. In addition, if a facility i sends flows to customer j at its lower bounds for all commodities other than commodity k, then it can not ship commodity k with an amount more than $u_{ij} - \sum_{k'=k',k'\neq k}^{K} l_{ijk'} = \left(u_{ij} - \sum_{k'=1}^{K} l_{ijk'}\right) + l_{ijk} = UL_{ij} + l_{ijk}$ to customer j.

Note that Equation 8.68 can be replaced with the following equations:

$$\begin{aligned} l_{ijk}^{new} &= \max\{l_{ijk}, s_{ik} - \textstyle\sum_{j'=1,j'\neq j}^{J} u_{ij'k}, q_{jk} - \sum_{i'=1,i'\neq i}^{I} u_{i'jk}\}, \\ u_{ijk}^{new} &= \min\{u_{ijk}, s_{ik} - \textstyle\sum_{j'=1,j'\neq j}^{J} l_{ij'k}, q_{jk} - \sum_{i'=1,i'\neq i}^{I} l_{i'jk}, u_{ij} - \sum_{k'=1,k'\neq k}^{K} l_{ijk'}\} \\ & \qquad i = 1, ..., I; j = 1, ..., J; k = 1, ..., K. \end{aligned} \tag{8.69}$$

The value of u_{ijk} does not affect l_{ijk}^{new} and the value of l_{ijk} does not affect u_{ijk}^{new}. Hence, the order of updating either l_{ijk}^{new} or u_{ijk}^{new} first does not change the outcome. □

A change in the l_{ijk} or u_{ijk} values should affect the corresponding maximum slack values given by Equation 8.65 – 8.67 because all resources (i.e., supply, demand or flow capacity quantities) are commonly shared by all allocation variables. Therefore, new bounds l_{ijk}^{new} and u_{ijk}^{new} of all neighboring allocation variables should be updated with the latest bounds accord-

ing to Equation 8.68. For that purpose, we keep a list of allocation variables whose bounds or maximum slack values have changed at some previous step. The update mechanism is repeated until the list is empty and all bounds become stable. During the sequential update mechanism, when we encounter with a negative maximum slack value for any constraint, this implies that the current partitioning of the allocation space is infeasible thereafter and thus the current node of the BB tree is pruned. Finally, the logical test is applied on each subproblem within the MABB algorithm either to obtain tighter bounds on allocation variables or to detect infeasibility.

8.2.2.2. Strategies for Partitioning, Tree Search and Branching Variable Selection. The allocation space associated with an active node of the MABB tree is partitioned into two distinct subsets and hence, resulting in two new MABB subproblems. At each partitioning, an allocation variable $w_{ijk} \in \mathcal{W}^F$ is selected and the corresponding bounds are divided into two subsets while the rest of the bounds remains the same as the current node.

The MABB algorithm performs a BFS strategy: the node with the smallest lower bound value among all active nodes is selected as the node for partitioning. For each node t, we keep record of the current objective value, the lower bound value and the feasible allocation vector $\overline{\mathbf{w}}^{(t)}$, $l_{ijk}^{(t)}$ and $u_{ijk}^{(t)}$ values such that $l_{ijk}^{(t)} \leq \overline{w}_{ijk}^{(t)} \leq u_{ijk}^{(t)}$ for $i = 1, \ldots, I; j = 1, \ldots, J$ and $k = 1, \ldots, K$ associated with the MABB subproblem of node t. $\overline{\mathbf{w}}^{(t)}$ is obtained by solving bounded MTP given by Equation 2.3 – 2.5 and 8.23 after fixing the facility locations to the lower bounding subproblem location vector. All these records are also used within multi-commodity branching variable selection strategies (MBrSs), bounding procedures and pruning of the BB tree nodes.

Once a node is selected for further exploration we need to choose a particular allocation variable w_{ijk}. Clearly, the sequence of variables on which we branch greatly affects the performance of the MABB algorithm. Therefore, this choice should be done carefully. For that purpose, we have implemented three different MBrSs associated with a partitioning scheme. MBrS1 and MBrS2 are suggested by inspiring from BrSs proposed by (Sherali *et al.*, 2002) used for the CMWP as presented. MBrS3 is a tailor made partitioning scheme for the MCMWP. In the following we present these MBrSs.

MBrS1:

$$(i', j', k') = \underset{(i,j,k)\in\mathcal{W}^F}{\arg\max} \{c_{ijk}(u_{ijk} - l_{ijk})\} \tag{8.70}$$

The bounds for the two new subproblems on the selected branching variable $w_{i'j'k'}$ is partitioned as $[l_{i'j'k'}, \lfloor (l_{i'j'k'} + u_{i'j'k'})/2 \rfloor]$ and $[\lceil (l_{i'j'k'} + u_{i'j'k'})/2 \rceil, u_{i'j'k'}]$, respectively.

MBrS2:

$$(i', j', k') = \underset{(i,j,k)\in\mathcal{W}^F}{\arg\max} \{(u_{ijk} - l_{ijk}) \min\{\overline{w}_{ijk} - l_{ijk}, u_{ijk} - \overline{w}_{ijk}\}\} \tag{8.71}$$

MBrS3:

$$(i', j', k') = \underset{(i,j,k)\in\mathcal{W}^F}{\arg\max} \left\{ c_{ijk}(u_{ijk} - l_{ijk}) \max \left\{ \begin{array}{c} \min\{SL_{ik}, SU_{ik}\}, \\ \min\{DL_{jk}, DU_{jk}\}, UL_{ij} \end{array} \right\} \right\} \tag{8.72}$$

For the MBrS2 and MBrS3 we partition the flow (i', j', k') into two as follows: $[l_{i'j'k'}, \overline{w}_{i'j'k'}]$ and $[\overline{w}_{i'j'k'} + 1, \ \ u_{i'j'k'}]$. Note that for $\overline{w}_{i'j'k'} = u_{i'j'k'}$, the partitioning can not be done as suggested for feasibility. Moreover, when $\overline{w}_{i'j'k'}$ is close to or equal to their boundaries $l_{i'j'k'}$ or $u_{i'j'k'}$, it is not of much use to partition the intervals as described. (Sherali *et al.*, 2002) emphasize this difficulty for the CMWP as well. In the context of the CMWP, they propose to switch from one BrS to another. In the MCMWP case we follow a similar approach. They apply a test to check whether such a closeness occurs. In case the test is satisfied, they switch from using Equation 8.17 to Equation 8.70 for the CMWP when the commodity index k is dropped from Equation 8.70. Namely, they apply a hybrid approach of backtracking to partition the interval homogeneously as in the MBrS1. We also use a similar approach and test whether the inequality

$$\min\{\overline{w}_{i'j'k'} - l_{i'j'k'}, u_{i'j'k'} - \overline{w}_{i'j'k'}\} \geq 0.1(u_{i'j'k'} - l_{i'j'k'}) \tag{8.73}$$

holds. Here 0.1 is a parameter value recommended by (Sherali *et al.*, 2002). Whenever Equation 8.73 is satisfied we switch from either MBrS2 or MBrS3 to MBrS1. In order to guarantee the feasibility we further impose the condition $u_{i'j'k'} - \overline{w}_{i'j'k'} \geq 1$. When we consider strategies MBrS1, MBrS2 and MBrS3, ties are broken arbitrarily. Whenever a

variable is selected all the remaining bounds l_{ijk} and u_{ijk} for $i = 1, \ldots, I; j = 1, \ldots, J; k = 1, \ldots, K$ and $(i, j, k) \neq (i', j', k')$ are inherited from the parent node.

The ABB algorithm assumes that right hand-side values of constraints given by Equation 2.3 – 2.6, namely supply, demand and flow capacity quantities are all integers. Thus, the flow quantities are integer for all extreme points of the MTP polyhedron. The MBrSs (MBrS1, MBrS2 and MBrS3) take advantage of this assumption and partition the allocation space over integer flows. Otherwise, when the right hand-side values of constraints given by Equation 2.3 – 2.6 are not necessarily integer, the bounds on allocation variables may be fractional which may lead to a huge BB tree. As a result, the performance of the ABB algorithm may significantly reduce without the integrality of flows assumption.

8.2.2.3. Optimality Check. For each MABB subproblem, we compute lower and upper bound values Z_{LB} and Z_{UB}, respectively. To avoid excessive computational effort, whenever the fathoming criterion $Z_{LB} \geq (1 - \epsilon) Z_{UB}^{best}$ holds for a node t, it is pruned from the BB tree. We set $\epsilon = 0.001$ in our calculations.

8.2.3. Summary of the Allocation Space Based Branch-and-Bound Algorithm

At this point we have all ingredients of the MABB algorithm at hand. Its outline can be formally given as in Figure 8.2.

The MABB algorithm is illustrated with a numerical example in Figure 8.3. We consider an instance named as "mc_2_4_2" which implies that the instance has two facilities to serve four customer in order to meet their demand on two types of commodities. The data consisting of the transportation costs, customer coordinates and the right hand sides of the constraints of "mc_2_4_2" is given as "*transportation costs*: $c_{111} = 1$, $c_{112} = 9$, $c_{121} = 3$, $c_{122} = 5$, $c_{131} = 2$, $c_{132} = 8$, $c_{141} = 4$, $c_{142} = 6$, $c_{211} = 3$, $c_{212} = 4$, $c_{221} = 0$, $c_{222} = 6$, $c_{231} = 1$, $c_{232} = 6$, $c_{241} = 8$, $c_{242} = 3$; *customer coordinates*: $\mathbf{a}_1 = (0, 1)^T$, $\mathbf{a}_2 = (0, 0)^T$, $\mathbf{a}_3 = (0, 2)^T$, $\mathbf{a}_4 = (4, 10)^T$; *facility capacities*: $s_{11} = 17$, $s_{12} = 22$, $s_{21} = 17$, $s_{22} = 21$; *customer demands*: $q_{11} = 5$, $q_{12} = 3$, $q_{21} = 18$, $q_{22} = 12$, $q_{31} = 6$, $q_{32} = 17$, $q_{41} = 5$, $q_{42} = 11$; *bundle restrictions*: $u_{11} = 0$, $u_{12} = 39$, $u_{13} = 39$, $u_{14} = 39$, $u_{21} = 38$, $u_{22} = 38$, $u_{23} = 38$, $u_{24} = 38$". Figure 8.3 presents several consecutive steps of the MABB algorithm on the instance "mc_2_4_2" with Euclidean distance. The nodes of the BB tree are denoted by circles associated with a node number. On the branches connected to the nodes, the selected allocation variable

1. (Initialization): Set $\mathcal{W}^E \leftarrow \varnothing$, $\mathcal{W}^F \leftarrow \{(i,j,k) : i = 1,\ldots,I; j = 1,\ldots,J; k = 1,\ldots,K\}$. Set $l_{ijk}^{(0)} = 0$ and $u_{ijk}^{(0)} = \min\{s_{ik}, q_{jk}, u_{ij}\}$ for $i = 1,\ldots,I; j = 1,\ldots,J; k = 1,\ldots,K$, apply the logical test and bounding procedures. Construct the root node t^0 with lower bound $Z_{LB}^{(t^0)}$, upper bound $Z_{UB}^{(t^0)}$. Update $\mathcal{T} \leftarrow \mathcal{T} \cup t^0$.
2. Select an active node $t \in \mathcal{T}$ such that $t = \arg\min_{t' \in \mathcal{T}}\{Z_{LB}^{(t)}\}$ and $Z_{LB}^{(t)} < (1-\epsilon)Z_{UB}^{best}$ is satisfied. **if** there is no such element in $\mathcal{T}$, **then** STOP: the incumbent solution is within $100\epsilon\%$ of the optimum. **else** select a branching variable $w_{i',j',k'}$ and partition the allocation space into two subspaces accordingly using one of MBrSs (i.e., MBrS1, MBrS2 or MBrS3). Set $Z_{LB}^{best} = Z_{LB}^{(t)}$.
3. For subsets $n' = 1, 2$, perform the logical test as described. Compute a Z_{LB} and Z_{UB} with bounding procedures. Update the incumbent solution when necessary, i.e., $Z_{UB}^{best} = \min\{Z_{UB}^{best}, Z_{UB}\}$. **if** $Z_{LB} < (1-\epsilon)Z_{UB}^{best}$ construct active node $t^{n'}$, **then** update $\mathcal{T} \leftarrow \mathcal{T} \cup t^{n'}$. Go to Step 2.

Figure 8.2. The MABB algorithm for the MCMWP.

and its current interval is shown. A node which is underlined indicates that it is fathomed. Each node is associated with a lower and upper bound shown at their (left or right) bottom and up sides. For example $Z_{UB}^4 = 631.9$ and $Z_{LB}^4 = 585$ are the upper bound value and lower bound value associated with node 4, respectively. We employ the branching variable selection strategy MBrS1 given by Equation 8.70 in this example. As the lower bounding procedure, RMDAP3 with ℓ_∞-norm is used together with Z_{MSAS} given by Equation 8.61. Upper bounds are calculated by the MCALA heuristic. At the root node an initial upper bound is obtained as explained in Section 8.2.1.3 and the lower bound is set to zero. The allocation variable w_{132} has $c_{132}(u_{132} - l_{132}) = 136$ which is maximum among all other allocation variables and it is selected for branching with respect to MBrS1. Hence, its solution space is partitioned into two intervals such that $0 \leq w_{132} \leq 8$ and $9 \leq w_{132} \leq 17$ resulting in node 1 and node 2, respectively. After the bounding procedures are applied for each sub-node, the incumbent solution value Z_{UB}^{best} is updated as $Z_{UB}^{best} = 631.9$ which is the optimal value for the instance "mc_2_4_2". Node 1 is fathomed since its lower bound exceeds Z_{UB}^{best} and the partitioning continues with node 2. Notice that further branching is done on node 4 even if both node 3 and node 4 are active since node 4 has the smallest lower bound value among all active nodes. This is also the reason that the MABB algorithm progresses by exploring

node 8 rather than node 3. The MABB algorithm will also explore node 3 once its lower bound becomes the smallest one in the rest of the BB tree. Note that we employ a logical test to update the current bounds on the allocation variables. This may result in different lower and upper bound values on them. As an example, after the branching at node 5 the allocation variable w_{132} is again selected. However, its lower bound, i.e., l_{132}, has increased from 9 to 12 while going down from node 2 to node 7. This is due to the update mechanism of the logical test used after the branching steps.

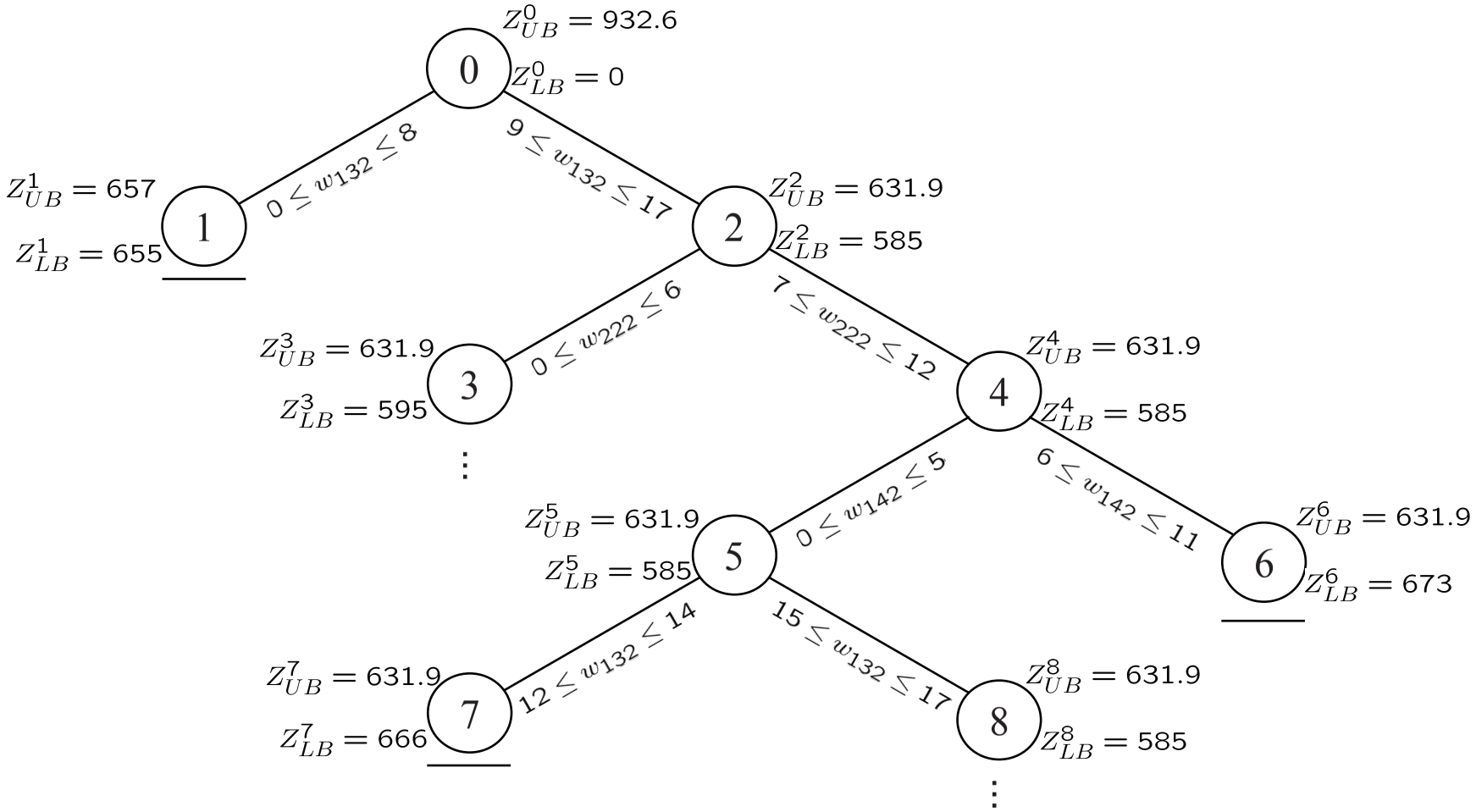

Figure 8.3. A numerical example for the MABB algorithm.

9. LOCATION SPACE BASED BRANCH-AND-BOUND METHODS

(Hansen *et al.*, 1981) devise a location space based BB algorithm named as "Big Square Small Square" (BSSS) for the Obnoxious Facility Problem (OFP). The BSSS technique partitions the plane into smaller squares in which a facility is restricted to be placed. For each square a lower and an upper bound is calculated. Whenever a lower bound value for a square exceeds the incumbent solution value then, it is discarded from further consideration. Otherwise, it is partitioned into four subsquares and the BB search process continues. Actually, the BSSS uses the center point of squares to calculate upper bounds and discards a square if its center is outside the convex hull of customers. (Plastria, 1992) explains that the BSSS technique may fail to find the optimal (or close to optimal) solution in such a case and an upper bound should always be calculated for each square. Hence, the author modifies the BSSS technique to resolve such a case of failure. Unfortunately, the BSSS technique requires an additional control of a square to lie within the convex hull of customers. (Drezner and Suzuki, 2004) employ triangles instead of squares and apply a triangulation method as a preprocessing step. This approach, which is called as "Big Triangle Small Triangle" (BTST) technique, partitions a triangle into four subtriangles by connecting the midpoints of its three sides. Since the triangulation phase eliminates all regions outside the convex hull of customers, the BTST technique does not require an additional check whether a region to be within the convex hull of customers or not. The OFP and Weber problem with attraction and repulsion (WAR) is solved by the BTST technique (Drezner and Suzuki, 2004). (Drezner *et al.*, 2007) present a generic approach to solve many types of location problems with the BTST technique. Among them, the Gradual Covering Problem (GCP) and Location with Acceleration-deceleration Distance Problems (LADP) are handled with the BTST technique in the studies by (Drezner *et al.*, 2004) and (Drezner *et al.*, 2009), respectively.

The basic idea of both BSSS and BTST techniques is to partition the plane into polytopes (i.e., squares, triangles and other types of polytopes) in which a single facility can be placed. To the best of our knowledge, neither the BSSS nor the BTST techniques are used for the Multi-facility LAPs (MLAPs). Several difficulties can be listed in generalizing the BSSS or BTST techniques for the MLAPs. First of all, both of the BSSS and BTST

techniques assume that the allocation values are fixed and known a priori which is not the case for the MLAPs including the CMWP and MCMWP. Furthermore, both techniques are designed to use concave lower bounding functions whose minimums occur at one of the extreme points of the bounded polyhedral regions namely polytopes: squares for the BSSS, triangles for the BTST. These functions may not be concave when in addition to the facility locations, the allocation quantities are also unknown. Therefore, one needs new lower bounding methods for MLAPs. Another difficulty of the BSSS technique is that during the BB tree search some regions can not be directly discarded from consideration for the MLAPs. For instance, assume that a region R of facility i^* is divided into two subregions R^1 and R^2. This results in two new facility-region combinations. Now, other facilities $i \neq i^*$ can also be located in either R^1 or R^2 since R is divided. Hence, there are several facility-region combinations such that other facilities can be restricted to these regions. In some of these facility-region combinations, the lower bound values calculated may or may not exceed the incumbent solution value. As a result, when region R is directly eliminated considering only facility i^*, then we can get rid of some feasible part of the search space. A triangulation method can be used as in the BTST technique developed for single facility location problems, as well. This generates many initial regions (i.e., triangles) that a facility can be located. Unfortunately, in this case, one should generate all possible facility-region (triangle) combinations and calculate lower and upper bounds as part of the LBB algorithm. It is clear that the number of facility-region (triangle) combinations becomes intractable even for small numbers of facilities and regions (triangles). Consequently, we have focused on the location space based BB (LBB) algorithm without such a preprocessing mechanism.

All these difficulties are taken into account to define the LBB algorithm and a continuous binary partitioning of the location space is preferred. For that purpose, we define the following bounds on location variables $\mathbf{x}_i$

$$\underline{a_{i1}} \leq x_{i1} \leq \overline{a_{i1}}, \qquad \underline{a_{i2}} \leq x_{i2} \leq \overline{a_{i2}} \qquad\qquad i = 1, \ldots, I \tag{9.1}$$

Initially these bounds can be defined as $\underline{a_{in}} = \min_{j=1,\ldots,J} \{a_{jn}\}$ and $\overline{a_{in}} = \max_{j=1,\ldots,J} \{a_{jn}\}$ for both axes $n = 1, 2$ and $i = 1, \ldots, I$. This implies that the location space is initially selected as the smallest rectangle covering the customer locations for each facility i. Observe that it is also possible to obtain other types of bounded polyhedral regions than rectangles given by Equation 9.1 by imposing restrictions of type Equation 8.37. At each step of the LBB

algorithm an active node $\bar{t} \in \overline{\mathcal{T}}$ is selected for exploration. Let $\mathcal{C}^{\bar{t}} = \{(\mathbf{x}_1, R_1^{\bar{t}}), \ldots, (\mathbf{x}_I, R_I^{\bar{t}})\}$ denote facility-region (facility-rectangle) combinations at node $\bar{t}$. Then, a region $R_i^{\bar{t}}$ is selected and partitioned into two complementing subregions (sub-rectangles) $R_{i1}^{\bar{t}}$ and $R_{i2}^{\bar{t}}$. Complementing subregions (sub-rectangles) mean that $R_{i1}^{\bar{t}} \cup R_{i2}^{\bar{t}} = R_i^{\bar{t}}$ holds after partitioning $R_i^{\bar{t}}$ and the interiors of $R_{i1}^{\bar{t}}$ and $R_{i2}^{\bar{t}}$ have no intersection (i.e., $int(R_{i1}^{\bar{t}}) \cap int(R_{i2}^{\bar{t}}) = \emptyset$). This results in two subproblems where possible location of facility i is further restricted. All bounds on location variables for the remaining facilities are directly inherited from $\mathcal{C}^{\bar{t}}$. A lower and upper bound is calculated for each subproblem, and the ones with a lower bound smaller than the incumbent solution value are added to the BB tree $\overline{\mathcal{T}}$. The incumbent solution value is updated when a better upper bound is obtained and the algorithm continues until $\overline{\mathcal{T}}$ is empty.

A similar approach is also used by (Sherali *et al.*, 1994) for the RCMWP on a partial location space which consists of the intersection points generated by the fundamental directions of ℓ_1-norm on customer locations. The authors define lower and upper bounds on location variables in both x and y-axes. However, the partitioning is applied on one axis at a time considering the customer locations on the selected axis. As the rectilinear distances are used, there is no need to take into account intermediate points lying between two customers on one axis. Our approach considers both x and y-axes together and can be extended to different polytopes other than rectangles. Moreover, the LBB algorithm can also be used to solve problems having ℓ_r-norm with $1 \leq r < \infty$.

In this chapter[6] we suggest Single-commodity LBB (SLBB) and Multi-commodity LBB (MLBB) algorithms for CMWP and MCMWP, respectively. The next section presents lower bounding procedures for both CMWP and MCMWP. Section 2 gives upper bounding procedures for them. Other features of the LBB algorithms (i.e., SLBB and MLBB algorithms) which include partitioning, tree search and branching strategies together with the optimality check is provided in Section 3. Section 4 establishes a formal outline of the LBB algorithm. Section 5 explains an alternative LBB algorithm in which a complete enumeration strategy is followed. In the last section, we have applied a Beam Search (BS) approach using the LBB algorithm and provide a heuristic procedure that can be successfully used for MLAPs.

[6]The conference proceeding (Akyüz et al., 2011) is partly based on this chapter.

9.1. Lower Bounding Procedures

We first describe two specially tailored lower bounding procedures: they are LP based and block norm based approaches. For each lower bounding procedure, we describe how they can be used within the SLBB and MLBB algorithms developed for the CMWP and MCMWP, respectively.

9.1.1. Linear Programming Based Lower Bounding Procedures

The lower bounds are needed to eliminate unnecessary nodes before adding them to $\overline{\mathcal{T}}$ and to check how close is the incumbent solution value of the LBB algorithm to optimality. Every time we partition a region R, the LBB subproblems of the form given by Equation 2.2 – 2.6 and 9.1 are constructed. In order to find lower bounds for the LBB subproblems, we benefit from the distance function properties. Given facility region combinations $\mathcal{C}^{\bar{t}} = \{(\mathbf{x}_1, R_1^{\bar{t}}), \ldots, (\mathbf{x}_I, R_I^{\bar{t}})\}$, we define distances $d_{ij}^{\bar{t}}$ which stand for the shortest distance between each facility i assigned to a region $R_i^{\bar{t}}$ and customer j. Notice that $d_{ij}^{\bar{t}} = d(\mathbf{a}_{ij}^{\bar{t}}, \mathbf{a}_j) \leq d(\mathbf{x}_i, \mathbf{a}_j)$ holds where $\mathbf{a}_{ij}^{\bar{t}}$ is the closest point of $R_i^{\bar{t}}$ to customer j. In the LBB algorithm the regions are selected as rectangles and $d_{ij}^{\bar{t}}$ can also be easily calculated for various types of regions (e.g., triangles, squares, pentagons, etc.). Lower bounding distances $d_{ij}^{\bar{t}}$ are previously proposed in the study by (Hansen *et al.*, 1985). For example, when the rectangles are considered in the LBB algorithm there are three cases that the closest point $\mathbf{a}_{ij}^{\bar{t}}$ can be situated on $R_i^{\bar{t}}$. These cases are illustrated with Figure 9.1. In the first case, when customer j lies within the rectangle $R_i^{\bar{t}}$ then $\mathbf{a}_{ij}^{\bar{t}} = \mathbf{a}_j$ holds (see Figure 9.1a). In the second case, $\mathbf{a}_{ij}^{\bar{t}}$ is located on one of the borders of $R_i^{\bar{t}}$ (see Figure 9.1b). In the third case (see Figure 9.1c), when customer j is beyond the area constructed by drawing vertical and horizontal lines on the extreme points of the rectangle $R_i^{\bar{t}}$ containing it, $\mathbf{a}_{ij}^{\bar{t}}$ is situated at one of the extreme points of $R_i^{\bar{t}}$ (i.e., $(\underline{a_{i1}}, \underline{a_{i2}}), (\overline{a_{i1}}, \underline{a_{i2}}), (\underline{a_{i1}}, \overline{a_{i2}})$ and $(\overline{a_{i1}}, \overline{a_{i2}})$). In the first case $d_{ij}^{\bar{t}}$ equals to zero. In the second case, $d_{ij}^{\bar{t}}$ is equal to either vertical or horizontal distance from the selected side of $R_i^{\bar{t}}$. In the third case $d_{ij}^{\bar{t}}$ equals to the ℓ_r distance between the selected extreme point of $R_i^{\bar{t}}$ and customer j.

Given the lower bounding distances $d_{ij}^{\bar{t}}$, we solve the usual TP (MTP) within the SLBB (MLBB) algorithm for the CMWP (MCMWP). Observe that $d_{ij}^{\bar{t}}$ is constant and does not depend on the location variable $\mathbf{x}_i$ and $d_{ij}^{\bar{t}} \leq d(\mathbf{x}_i, \mathbf{a}_j)$ holds for $i = 1, \ldots I; j = 1, \ldots, J$.

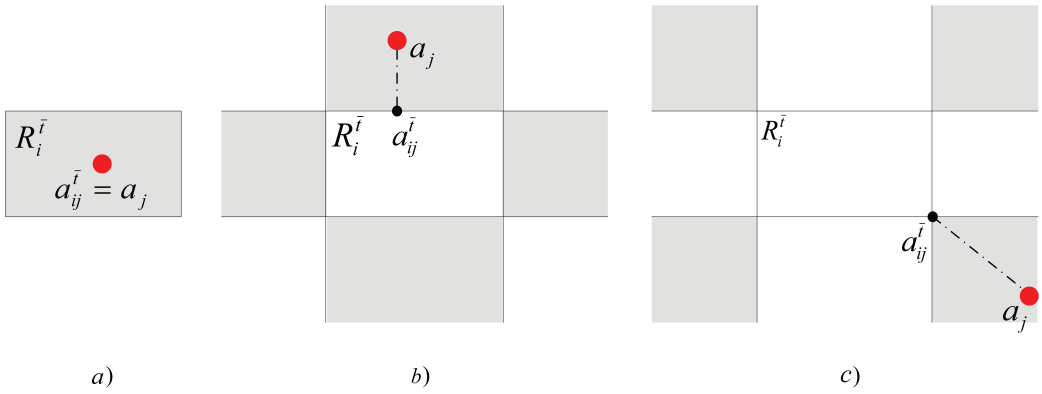

Figure 9.1. Three possible cases for the closest coint $\mathbf{a}_{ij}^{\bar{t}}$ of a rectangle $R_i^{\bar{t}}$.

Clearly, $\mathbf{a}_{ij}^{\bar{t}} \in R_i^{\bar{t}}$ implies that Equation 9.1 is already satisfied and thus, the solution of TP (MTP) constitutes a lower bound for the SLBB and MLBB subproblems given by Equation 3.6 – 3.9 and 9.1, and Equation 2.2 – 2.6 and 9.1, respectively. We provide these lower bounds as follows:

$\mathrm{LP}_{\mathrm{SLBB}}{}^{(\bar{t})}$:

$$\min \; Z_{LP}^{SLBB} = \left\{ \sum_{i=1}^{I} \sum_{j=1}^{J} w_{ij} c_{ij} d_{ij}^{\bar{t}} : \text{Equation } 3.7 - 3.9 \right\} \tag{9.2}$$

for the CMWP and

$\mathrm{LP}_{\mathrm{MLBB}}{}^{(\bar{t})}$:

$$\min \; Z_{LP}^{MLBB} = \left\{ \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} w_{ijk} c_{ijk} d_{ij}^{\bar{t}} : \text{Equation } 2.3 - 2.6 \right\} \tag{9.3}$$

for the MCMWP. Z_{LP}^{SLBB} and Z_{LP}^{MLBB} are the LP based lower bounds for SLBB and MLBB algorithms.

9.1.2. Block Norm Based Lower Bounding Procedures

The block norm based lower bounding idea can be applied to the LBB algorithm as well. Given a facility-region combination $C^{\bar{t}}$, when the block norms are used candidate facility locations can be selected from a finite set of intersection points. That is to say,

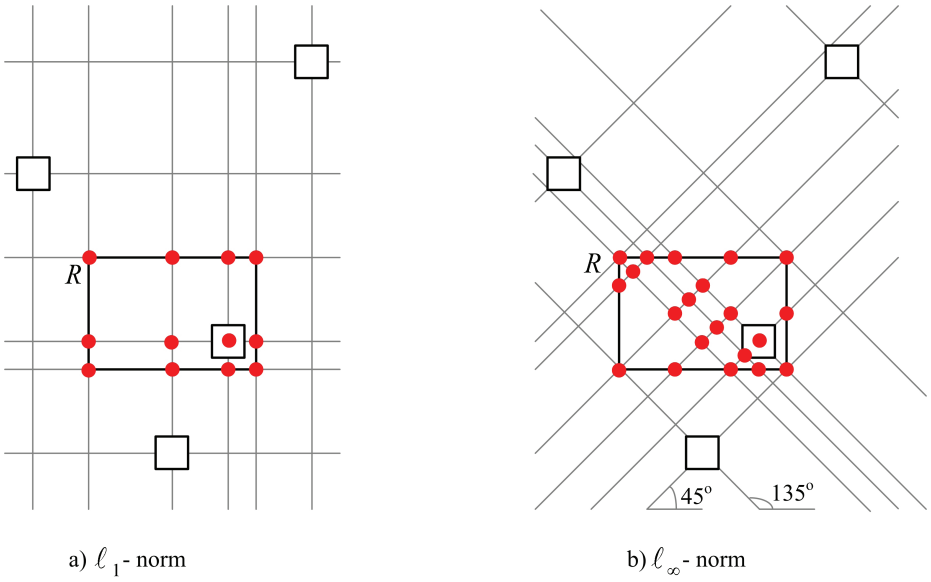

Figure 9.2. Candidate point sets within a rectangle R with ℓ_1 and ℓ_∞-norms.

when a facility is enforced to lie within a rectangle, the set of candidate facility locations are restricted within that rectangle. When there is no region restrictions on facilities, the optimal facility locations lie on the intersection points of the lines drawn on customer locations within their convex hull along the extreme directions of the corresponding block norm. In case we restrict facilities to lie within the rectangles, the extreme points of the rectangles also plays a role to determine the intersection points. In this case, using the results by (Thisse *et al.*, 1984), the candidate locations are the intersection points of the fundamental rays drawn on both the customer locations and extreme points of regions $R_i^{\bar{t}}$ for $i = 1, \ldots, I$ which lie either on their borders or within them. Figure 9.2 illustrates the intersection points to be considered when ℓ_1 and ℓ_∞-norms are used. The rectangle restricting a facility location is denoted by R. Customers are indicated with squares and the intersection points are drawn as filled circles. The fundamental rays are drawn on both customer locations and the extreme points of the rectangle R. The resulting candidate points lie either on the intersection of the borders of the rectangle R and a fundamental ray or on the intersection of two fundamental rays.

A lower bounding MILP formulation similar to DAP and MDAP1 can be proposed to find a block norm based lower bound for LBB subproblems. For the sake of conciseness, we do not explicitly state these MILP formulations. However, we should mention that in DAP or MDAP1 all candidate points are commonly shared by all facilities. This is because there are no restrictions on facility locations for them. For the LBB subproblems, each facility i is

restricted within a rectangle $R_i^{\bar{t}}$ and thus each facility i has its own candidate points within $R_i^{\bar{t}}$. As a consequence, we need an additional index to denote the candidate points of each facility (i.e., $g_i = 1, \ldots, G_i$ with G_i being the number of intersection points in $R_i^{\bar{t}}$ for facility i). On large candidate facility location sets, it is possible to use similar LR schemes for the LBB subproblems to obtain block norm based lower bounds. In the following, we state modifications over DAP (MDAP1) for the CMWP (MCMWP) to find block norm based lower bounds.

In the DAP formulation given by Equation 8.2 – 8.7 for the CMWP, we describe several modifications to adapt it for the SLBB algorithm. As each facility has now its own set of candidate points denoted by g_i, y_{ijg} is replaced with y_{ijg_i} which stands for the amount of flow between facility i located at one of its candidate points g_i to customer j. Similarly, v_{ig} is replaced with v_{ig_i} to denote whether facility i is opened at one of its candidate points g_i or not. $\widehat{l}_{ij}$ and $\widehat{u}_{ij}$ values are set as $\widehat{l}_{ij} = 0$ and $\widehat{u}_{ij} = \min\{s_i, q_j\}$ and hence the additional term $\widehat{l}_{ij} v_{ig}$ in Equation 8.2 is dropped from the formulation. Lastly, all g and G are replaced with g_i and G_i in the DAP formulation given by Equation 8.2 – 8.7, respectively. Clearly, the DAP formulation is exactly the same at the root nodes of both SABB and SLBB algorithms. The rest of the notation is maintained as they are presented in DAP formulation (or RDAP) to find the block norm based lower bound Z_{DAP}^{SLBB} for the SLBB algorithm.

In MDAP1 formulation given by Equation 5.1 – 5.7, we also propose several changes to adapt it for the MLBB algorithm. As each facility has now its own set of candidate points denoted by g_i, y_{ijkg} is replaced with y_{ijkg_i} to show the amount of flow between facility i located at one of its candidate points g_i to customer j of commodity k. v_{ig} is replaced with v_{ig_i} indicate whether facility i is opened at one of its candidate points g_i or not. Lastly, all g and G are replaced with g_i and G_i in the MDAP1 formulation given by Equation 5.1 – 5.7. The rest of the notation is maintained as they are given in MDAP1 formulation (or RMDAP1) to find the block norm based lower bound Z_{MDAP}^{MLBB} within the MLBB algorithm for the MCMWP. Similar modifications can also be done with the MDAP2 formulation.

9.2. Upper Bounding Procedures

A node of the LBB tree can be prunned without further branching when the lower bound of the current node is larger than the best known upper bound value for the problem.

Therefore, it is important to find a good upper bound, Z_{UB}, which is close to optimum, in order to reduce the number of branchings made within the LBB algorithm. The upper bounding procedure is similar to the one described for the SABB and MABB algorithm. Once a lower bound is found for a LBB subproblem, a feasible allocation vector is at hand for the CMWP (or MCMWP). Thus, given this allocation vector, solving the resulting WPs to find the optimum facility locations yields a feasible solution for both the CMWP and MCMWP. The solution can also be enhanced with a CALA (or MCALA) heuristic. We also apply a CALA (MCALA) heuristic and update the incumbent objective value, Z_{UB}^{best} throughout the run of the SLBB (MLBB) algorithm when $Z_{UB} < Z_{UB}^{best}$ holds.

9.3. Other Features of the Location Space Based Branch-and-Bound Algorithms

9.3.1. Partitioning, Search and Branching Strategies

The location space associated with an active node of the LBB tree is partitioned into two complement subsets which results in two new LBB subproblems. Observe that the interior of both complement subsets is empty and the union of the complement subsets is the initial location space before the partitioning. At each partitioning step a rectangle $R_i^{\bar{t}}$ corresponding to facility i is selected and $R_i^{\bar{t}}$ is divided into two complement rectangles separated by a line. All other facility-rectangle pairs are inherited for new subproblems. A rectangle can be partitioned in two ways: vertically and horizontally. For each subproblem, we prefer to partition a rectangle on its longest sides. This implies that if the horizontal (vertical) sides are longer than the vertical (horizontal) sides, then the rectangle is partitioned by connecting the mid-points of two horizontal (vertical) sides. This helps to avoid the width (length) of the rectangles to be too large (small) on their vertical (horizontal) sides. As a result, the rectangles are uniformly partitioned on both of their vertical and horizontal sides.

The LBB algorithm performs a BFS strategy. We select an active node $\bar{t} \in \overline{\mathcal{T}}$ with the smallest lower bound value for partitioning. At every active node, we keep track of facility-rectangle combinations (i.e., $\mathcal{C}^{\bar{t}}$ and $R_i^{\bar{t}}$ for $i = 1, \ldots, I$), and, lower and upper bound values calculated. Each rectangle is also coupled with its defining extreme points, total area and parent rectangle.

The branching is applied over the rectangles and we try to make a balanced partitioning. That is to say, we partition the rectangles such that none of the rectangles of node $\bar{t}$ under consideration has an area greater than twice of the area of the smallest rectangle. Given $\bar{t}$ and its associated facility-region combination $\mathcal{C}^{\bar{t}}$, the following strategy is used to select the branching rectangle $R^{\bar{t}}_{i^*}$:

$$i^* = \underset{i=1,\dots,I}{\arg max} \left\{ \text{ Area of } R^{\bar{t}}_i \in \mathcal{C}^{\bar{t}} \right\} \tag{9.4}$$

Notice that the branching strategy given by Equation 9.4 ensures a homogeneous partitioning of rectangles. Without branching strategy given by Equation 9.4, it is possible to divide a rectangle and its sub-rectangles of the same facility many times which may result in a series of non-improving steps within the LBB algorithm.

9.3.2. Optimality Check

For each LBB subproblem, we calculate lower and upper bound values Z_{LB} and Z_{UB}, respectively. Notice that we partition the location space continuously and this process may not end without a suitable stopping condition. Hence we can say that a stopping condition plays a crucial role on the completion of the LBB algorithm in a finite number of iterations. For that purpose, we impose the condition $Z_{LB} \geq (1-\epsilon)Z^{best}_{UB}$ to prune the nodes satisfying it. We set $\epsilon = 0.001$ in order to avoid excessive computational effort which also ensures the finiteness of our LBB algorithm within $100\epsilon\%$ of the optimal value. Clearly, there is a trade-off between the closeness to optimality and the computational time spent by the LBB algorithms. Furthermore, the number of rectangles is limited to 100000 within the LBB algorithms. In practice, we observe these settings to be useful for the computational times of the LBB algorithms.

9.4. Summary of the Location Space Based Branch-and-Bound Algorithm

Both of the SLBB and MLBB algorithms have more or less the same ingredients. The only difference lies in the bounding procedures. The CMWP requires the solution of a TP for LP based lower bounding, the solution of a DAP (or RDAP) for the block norm based lower bounding and the solution of CALA for upper bounding. On the other

hand, MCMWP requires the solution of MTP and MDAP1 with the suggested modifications for lower bounding procedures and MCALA heuristic for upper bounding procedure. The remaining features are common for both SLBB and MLBB algorithms. The formal outline of the LBB algorithm is given in Figure 9.3. Note that it is generic and it can be used for the implementation of both the SLBB and MLBB algorithms by paying attention to the differences as mentioned.

We should point out that for the LBB algorithm we do not assume the integrality of the right-hand sides of the CMWP and MCMWP constraints. Different than the ABB algorithms, LBB algorithms can be directly used to solve the CMWP (or MCMWP) with fractional variables. As a reminder, SABB and MABB algorithms should be slightly modified when the right-hand sides of the constraint sets are not integral as discussed.

1. (Initialization): Initialize the regions $R_i^0 \leftarrow \{\mathbf{x}_i : \underline{a_{i1}} \leq x_{i1} \leq \overline{a_{i1}}, \underline{a_{i2}} \leq x_{i2} \leq \overline{a_{i2}}\}$ for $i = 1, \ldots, I$. Construct facility-region combinations $\mathcal{C}^0 \leftarrow \{(\mathbf{x}_1, R_1^0), \ldots, (\mathbf{x}_I, R_I^0)\}$, compute Z_{LB}^0 and Z_{UB}^0 values associated with $\mathcal{C}^0$. Create node $\bar{t}^0$ and set $\overline{\mathcal{T}} \leftarrow \overline{\mathcal{T}} \cup \bar{t}^0$. Update Z_{LB}^{best} and Z_{UB}^{best} values accordingly.
2. (Partitioning): Select an active node $\bar{t} \in \mathcal{T}$ such that $\bar{t} = \arg\min_{t^* \in \overline{\mathcal{T}}}\{Z_{LB}^{t^*}\}$ and $Z_{LB}^{\bar{t}} < (1-\epsilon)Z_{UB}^{best}$. **if** there is no such a node, **then** STOP: Z_{UB}^{best} is within $100\epsilon\%$ of the optimum value. **else** select i^* as in Equation 9.4 and divide $R_{i^*}^{\bar{t}}$ into two subsets as described. Set $Z_{LB}^{best} = Z_{LB}^{\bar{t}}$ and $\overline{\mathcal{T}} \leftarrow \overline{\mathcal{T}} \setminus \bar{t}$.
3. (Bounding): For each subset $n' = 1, 2$ compute a lower bound $Z_{LB}^{n'}$ and an upper bound $Z_{UB}^{n'}$ by bounding procedures. Update Z_{UB}^{best} if necessary, i.e., $Z_{UB}^{best} = \min\{Z_{UB}^{best}, Z_{UB}^{n'}\}$. When $Z_{LB}^{n'} < (1-\epsilon)Z_{UB}^{best}$ then, construct an active node $\bar{t}^{n'}$ and set $\overline{\mathcal{T}} \leftarrow \overline{\mathcal{T}} \cup \bar{t}^{n'}$. Go to Step 2.

Figure 9.3. The LBB algorithm.

The LBB algorithm is illustrated with a numerical example in Figure 9.4. We again consider the instance "mc_2_4_2" and Figure 9.4 presents several consecutive steps of the

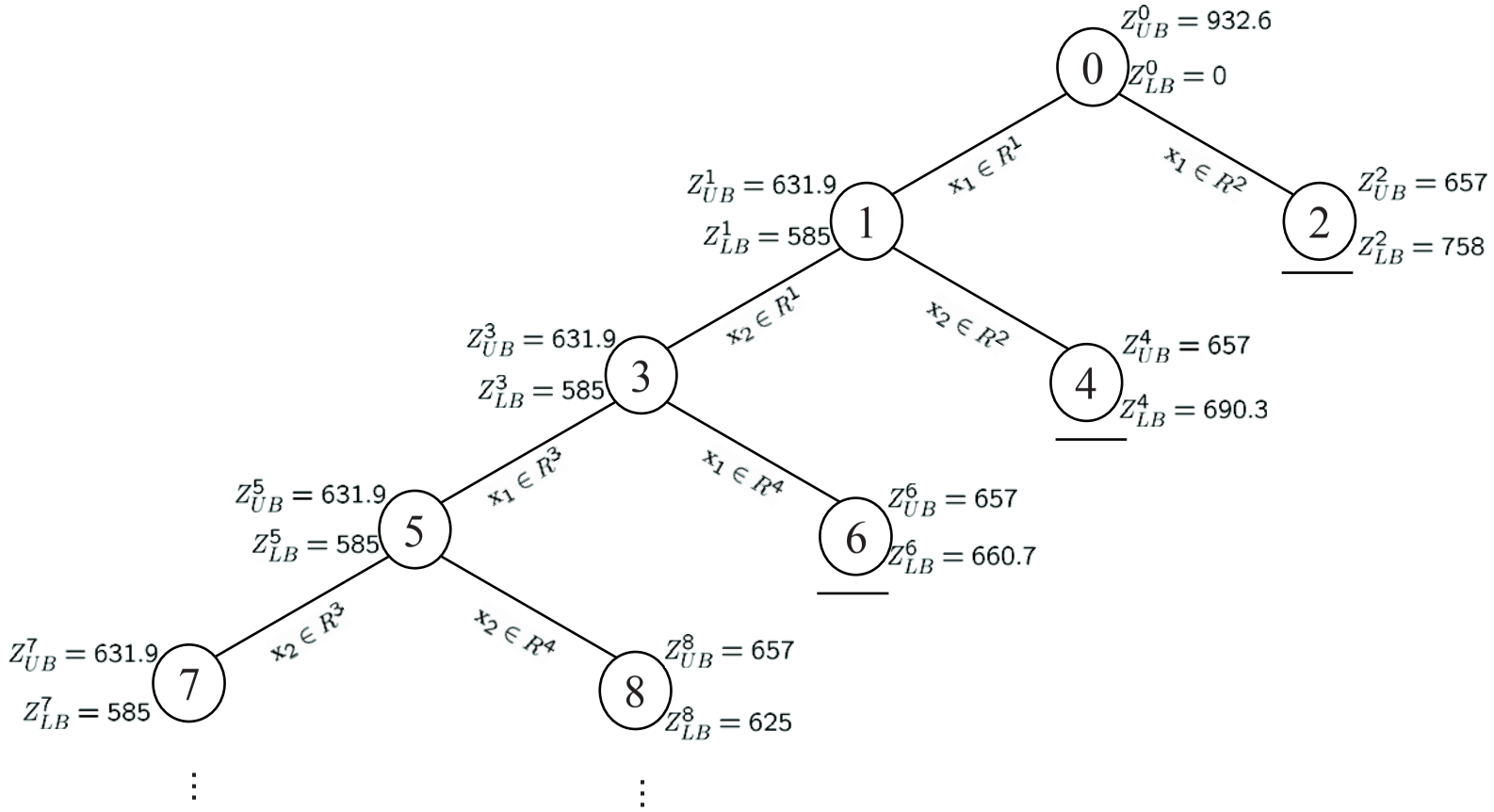

Figure 9.4. A numerical example for the LBB algorithm.

LBB algorithm on it. Each node of the BB tree, which is represented by a circle, is associated with a number. On the branches connected to the nodes, the selected location variable and the rectangle in which it is restricted, is shown. A line is drawn under a node in order to indicate that it is fathomed. Each node is associated with a lower and upper bound shown at their (left or right) bottom and up sides. For example, $Z^3_{UB} = 631.9$ and $Z^3_{LB} = 585$ are the upper and lower bound values associated with node 3, respectively. At each branching the largest rectangle associated with a facility is selected according to Equation 9.4 and then this rectangle is partitioned. Here, both the RMDAP1 with ℓ_∞-norm and the LP based lower bounding procedures are used together as the lower bounding procedure. Upper bounds are calculated by running the MCALA heuristic. At the root node an initial upper bound is obtained as explained in Section 8.2.1.3 and the lower bound is set to zero. Initial rectangle R^0 is defined as $R^0 = \{(x_1, x_2)^T : 0 \leq x_1 \leq 4, 0 \leq x_2 \leq 10\}$ and both facilities are restricted in R^0. R^0 is partitioned into two complementary sub-rectangles R^1 and R^2 for the first facility. This partitioning can also be done for the second facility but we arbitrarily select the first facility. R^1 and R^2 are defined respectively as $R^1 = \{(x_1, x_2)^T : 0 \leq x_1 \leq 4, 0 \leq x_2 \leq 5\}$ and $R^2 = \{(x_1, x_2)^T : 0 \leq x_1 \leq 4, 5 \leq x_2 \leq 10\}$. Notice that the partitioning of R^0 is done on its longer sides where the line combining their midpoints is drawn. This partitioning results in more uniform rectangles than rectangles which are too narrow or too wide on one of their sides. On the left branch, the first facility is restricted in R^1 such that $\mathbf{x}_1 \in R^1$ and we reach node 1. On the right branch, we set $\mathbf{x}_1 \in R^2$. For both node 1 and node 2, $\mathbf{x}_2$ still belongs to R^0. After applying the bounding procedures for each sub-node, the incumbent solution value Z^{best}_{UB} is updated as $Z^{best}_{UB} = 631.9$. Then, node 2 is fathomed and branching continues with node 1. In node 1, the largest rectangle associated with a facility is selected. This time, rectangle R^0 of the second facility is selected since the area of R^0 is greater than the area of R^1. Now, the second facility is restricted within both R^1 and R^2, similar to the previous branchings. For node 3, both the first and second facilities are in R^1. Node 4 is fathomed since its lower bound exceeds Z^{best}_{UB}. At node 3, R^1 is further divided into two complementary rectangles R^3 and R^4 such that $R^3 = \{(x_1, x_2)^T : 0 \leq x_1 \leq 4, 0 \leq x_2 \leq 2.5\}$ and $R^4 = \{(x_1, x_2)^T : 0 \leq x_1 \leq 4, 2.5 \leq x_2 \leq 5\}$. The rest of the steps are the same as in MABB algorithm and the LBB algorithm continues branching on node 7 since it has the smallest lower bound among all active nodes in the BB tree.

9.5. Location Space Based Branch-and-Bound Algorithm with Complete Enumeration

The LBB algorithm selects an active node from the BB tree and partitions the rectangle corresponding to this active node into two complementing rectangles for a facility. This results in two new subproblems to be considered which will probably be added to the BB tree. The partitioning of the selected rectangle is limited to a particular active node $\bar{t}$.

On the other hand, the partitioning of a rectangle R into two sub-rectangles R^1 and R^2 can be executed such that R is replaced with R^1 and R^2 in all active nodes. For that purpose, all active nodes having a facility located within rectangle R is branched further. This branching strategy may generate at most 2^I subproblems for a given active node $\bar{t}$. For example, consider two facilities which is to be located in the plane and we are given an active node $\bar{t}$ associated with a combination $\mathcal{C}^{\bar{t}} = \{(\mathbf{x}_1, R), (\mathbf{x}_2, R)\}$. Here, both of the locations of facilities $\mathbf{x}_1$ and $\mathbf{x}_2$ are restricted within rectangle R which will be partitioned into two complementing sub-rectangles R^1 and R^2. Then, by partitioning R, all possible combinations associated with $\bar{t}$ are generated as $\mathcal{C}^{\bar{t}1}_{new} = \{(\mathbf{x}_1, R^1), (\mathbf{x}_2, R^1)\}$, $\mathcal{C}^{\bar{t}2}_{new} = \{(\mathbf{x}_1, R^1), (\mathbf{x}_2, R^2)\}$, $\mathcal{C}^{\bar{t}3}_{new} = \{(\mathbf{x}_1, R^2), (\mathbf{x}_2, R^1)\}$ and $\mathcal{C}^{\bar{t}4}_{new} = \{(\mathbf{x}_1, R^2), (\mathbf{x}_2, R^2)\}$. The LBB subproblems corresponding to new combinations $\mathcal{C}^{\bar{t}1}_{new}$, $\mathcal{C}^{\bar{t}2}_{new}$, $\mathcal{C}^{\bar{t}3}_{new}$ and $\mathcal{C}^{\bar{t}4}_{new}$ are evaluated and added to the BB tree when they are eligible. Moreover, all remaining active nodes with a combination for which a facility is restricted in R should also be explored similarly by replacing R with R^1 and R^2. Thus, R can be eliminated from the rectangle list since there does not exist an active node containing R anymore in the BB tree. The number of new facility-region combinations doubles for each additional facility which is also assigned to region R for a given combination of $\mathcal{C}^{\bar{t}}$ of an active node $\bar{t} \in \overline{\mathcal{T}}$. In short, $2^{I_{\bar{t}}}$ many new facility-region combinations should be constructed with $I_{\bar{t}} = \left|\left\{i : R(\mathcal{C}^{\bar{t}}_i) = R, i = 1, \ldots, I\right\}\right|$ where $I_{\bar{t}}$ and $R(\mathcal{C}^{\bar{t}}_i)$ denote the number of facilities in the facility-region combination $\mathcal{C}^{\bar{t}}$ whose region is R and the region of the combination $\mathcal{C}^{\bar{t}}$ assigned to facility i, respectively. The enumeration procedure can produce $2^{I_{\bar{t}}}$ subproblems for an active node $\bar{t}$ and all remaining active nodes are subject to this procedure. We call this approach as the LBB with complete enumeration (LBBCE) algorithm which is named as SLBBCE algorithm for the CMWP and MLBBCE algorithm for the MCMWP, respectively. In addition to the LBB algorithm, the performance of the LBBCE algorithm is also tested. We impose an additional stopping condition for LBBCE which limits the size of the BB tree. For that purpose, we allow up to 500000

active nodes in the BB tree considering the number of subproblems which will be produced by active nodes.

9.6. A Beam Search Heuristic

BS is a BB based heuristic search method which dates back to the study by (Lowerre, 1976) on the speech recognition. BS performs a "breadth-first search" (BrFS) strategy on a truncated BB tree. In a complete BB tree search, branching is done such that all possible subproblems are produced and evaluated. This requires a bounding step to calculate lower and upper bounds for each resulting subproblem. On the other hand, BS considers only the most promising W of them which is also called as "beam width" and branches on them. Actually, the beam consists of the active nodes which will be considered for further branching in BS. An active node, which is one of the W nodes in the beam, is further partitioned such that all possible subproblems are generated and the most promising subproblem replaces the active node before the branching. This procedure is repeated for each of the W active nodes which are in the beam. To short, BS applies a BrFS strategy in parallel for all W active nodes in the beam. However, the number of subproblems may be too large to perform a bounding procedure for each of them when the branching width is large, that is, when the number of possible subproblems after a branching step is large. (Ow and Morton, 1988, 1989) modify the BS by adding a filtering step where a cheap and fast evaluation procedure reduces the number of subproblems for which the expensive bounding procedure, is performed. This algorithm is called as Filtered BS (FBS). Clearly, neither BS or FBS does not guarantee an optimal solution for a problem. Indeed, it is probable that one misses the optimal solution at an early pruning step of the BB tree with either BS or FBS. Once a node is pruned, there is no way to backtrack on it for further exploration. (Croce *et al.*, 2004) introduce a recovery step and the Recovering BS (RBS) to overcome this difficulty. The recovery step in RBS looks for an improved feasible solution by interchanging current assignments of variables (i.e., exchanging current values of two zero-one variables) at a given subproblem. Since only the most promising subproblem is selected to replace an active node in the beam, this gives a chance to recover the wrong decision of pruning the branch which leads to an optimal solution.

RBS, which is used for the p-median problem in (Croce *et al.*, 2004), is a heuristic example for a DLAP. For all we know, a BS approach considering a continuous MLAP does

not exist. We develop a BS heuristic with our LBB algorithm which performs a continuous partitioning on the location space. The BS is originally designed for the discrete problems such as scheduling, p-median and many other COPs as well. The LBB algorithm performs a continuous partitioning where the depth of the BB tree is not known a priori. Actually, the finiteness of the LBB algorithm is only guaranteed with the imposed stopping conditions. The LBB algorithm performs a BFS strategy and here we adapt it to produce heuristic solutions using the BS heuristic. At each step of the LBB algorithm an active node having the least lower bound value is selected for further branching and the bounding procedure is performed on the resulting subproblems. The viable nodes are added to the BB search tree.

In our BS approach, only W nodes are allowed to be active in the BB tree. (Croce *et al.*, 2004) state that there is no cheap and fast evaluation procedure (filtering) for the p-median procedure. Hence, we do not expect to find an efficient evaluation procedure for both CMWP and MCMWP. As a result, no filtering is applied to perform a preprocessing step in the BS heuristic. Promising nodes for further branching are selected according to an evaluation function. This is a cost based function of the subproblem associated with the selected active node $\bar{t}$. For that purpose, we employ the evaluation function suggested by (Croce *et al.*, 2004) which uses a convex combination of the lower and upper bound values, namely $Z^{\bar{t}}_{LB}$ and $Z^{\bar{t}}_{UB}$ associated with $\bar{t}$, respectively. The evaluation function is defined as $\widehat{Z}^{\bar{t}} = (1-\Psi)Z^{\bar{t}}_{LB} + \Psi Z^{\bar{t}}_{UB}$ with $0 \leq \Psi \leq 1$. The performance of the BS heuristic is significantly affected by the evaluation function used (i.e., $\widehat{Z}^{\bar{t}}$). At first glance, one may think that the smallest lower bounding (when $\Psi = 0$) solutions are the ones that lead us to an optimal solution. In practice, the least lower bounding nodes may not always produce the best feasible solutions.

We use a BFS strategy and the most promising node is selected for further branching. Namely the node with the smallest of $\widehat{Z}^{\bar{t}}$ such that $\bar{t} \in \overline{\mathcal{T}}$ where $|\overline{\mathcal{T}}| = W$ is chosen. Instead of branching over all W active nodes in the beam at each step as in a BrFS strategy, we perform a single branching over the node having the smallest $\widehat{Z}^{\bar{t}}$ value in the beam. After this single branching step, the most promising (the ones having W least $\widehat{Z}^{\bar{t}}$ values) W nodes of the beam are kept for further exploration. It is also possible to pursue a complete BrFS strategy and branch over W nodes in the beam as in the classical BS. However, we have observed that a BFS strategy yields better outcomes than the BrFS strategy does. Indeed, both BFS and BrFS strategies are the same when $W = 1$. To sum up, instead of branching

over all W active nodes at each step, we perform a single branching and keep the most promising W (W nodes having least $\widehat{Z}^{\bar{t}}$ values) nodes for further exploration in the BB tree. This setting helps us to avoid from expensive bounding procedures, in particular the lower bounding procedures.

10. COMPUTATIONAL RESULTS

The performance of the suggested methods are tested on a set of randomly generated test instances. We first describe our test environment on which our experiments are performed. Then, we summarize the results obtained with the proposed solution procedures.

10.1. Test Environment

Our test bed contains two groups of test instances: instances for the CMWP and instances for the MCMWP. The test library for the CMWP, consists of both existing instances for some of which optimal solutions are known and randomly generated new instances. Our test library for the MCMWP contains of only randomly generated instances since there is no available test problems in the literature.

We report the percent relative deviations of the objective value (Z_M) computed by one of our methods developed and we give a reference value (Z_R) in order to expose the accuracy of the methods. They are calculated according to the formula

$$100 \times \frac{|Z_M - Z_R|}{Z_R}. \tag{10.1}$$

In the ideal situation Z_R is selected as the exact optimum value Z^* while this is not always possible since the exact solution of every test problem is not feasible. Hence, we pursue a pessimistic approach and replace Z_R with a benchmark lower bound when assessing the accuracy of an upper bound and with a benchmark upper bound when assessing the accuracy of a lower bound. In fact, the calculated relative deviations are upper bounds on the true ones, which are definitely smaller. When we report gaps between the final lower bound Z_{LB}^{final} and the final upper bound Z_{UB}^{final} produced by one of our suggested methods we use the formula

$$100 \times \frac{(Z_{UB}^{final} - Z_{LB}^{final})}{Z_R}, \tag{10.2}$$

where Z_R is the reference value used to make a fair comparison among the solution approaches. In case the optimal value is not known, we replace Z_R with a benchmark value. We should make it clear that in our computational experiments the Euclidean distance function (i.e., $r = 2$) is used for calculations. Otherwise, we specify which distance function is used with the CMWP or MCMWP (i.e., RMCMWP stands for Rectilinear MCMWP). We present the details of the generation of the CMWP and MCMWP test beds used for our computational experiments in the following two subsections.

10.1.1. Test Bed for the Capacitated Multi-facility Weber Problem

We performed computational experiments on two classes of test instances for the CMWP. The first class consists of 18 test instances from the literature with their given best known values (Al-Loughani, 1997; Sherali *et al.*, 2002). The second class includes 94 randomly generated test instances with unknown optimal values (Boyacı, 2009).

The instances in the first class are numbered from 1 to 12 and from 15 to 20 as in the original paper (Sherali *et al.*, 2002). We append a prefix "P" in front of their original instance number. For these instances, the number of facilities range from 2 to 10 and the number of customers range from 2 to 30. We consider the total number of allocation variables, namely $I \times J$, as the main criterion for the classification of instances. In other words, the instances satisfying inequalities $I \times J \leq 50$, $50 < I \times J \leq 80$ and $I \times J > 80$ are classified as "*small*", "*medium*" and "*large*" instances for the CMWP, respectively.

The instances in the second class are further grouped into two subgroups as homogeneous (having unit transportation costs i.e., $c_{ij} = 1$) and non-unit (having non-unit transportation costs c_{ij}) instances. Randomly generated test instances have a similar structure as the existing ones (i.e., the test instances from the literature). The number of facilities are selected in the range from 4 to 50 and the number of customers are selected in the range from 8 to 500. The prefixes "hg" and "ht" are used to represent homogeneous and non-unit test instances, respectively. The instance names are followed by the number of facilities and the number of customers for each instance. For example, "hg_4_24" implies that the instance is a unit transportation cost test instance which has 4 facilities and 24 customers. The randomly generated instances in the second class are mostly larger than the ones in the first class. In 90 out of 94 instances of the second class, there are more than 80 allocation variables.

This implies that most of them are *large*. On the other hand, for the first class of instances, only 5 out of 18 are *large*. The instances in the second class are generated in a similar way to the generation of instances in the first class. For the instances in the second class, the customer locations $(a_{j1}\ a_{j2}), j = 1, \ldots, J$ are uniformly selected (for both x and y-axis) from the interval $[0, 25]$ for instances with $I = 4$. These values are selected from the interval $[0, 100]$ for instances with $I > 4$. Unit transportation costs are also uniformly selected from the interval $[0, 25]$ (as in the existing instances) for the non-unit instances while they are set $c_{ij} = 1$ for the homogenous ones. All data entries, other than the cost coefficients c_{ij}, are the same for these two subgroups of instances which have identical sizes. Customer demands are uniformly selected within the interval $[1, 50]$ and facility capacities are split accordingly without harming the balanced structure of the transportation constraints.

Note that the instances in the first class are limited in number. Furthermore, we can say these instances are not *large* and they are all non-unit instances with non-unit transportation costs c_{ij}. Therefore, the instances in the second class are generated keeping in mind these facts. Indeed, new instances are larger than the existing ones and the second class includes also homogenous instances with unit costs. We limit our experiments to BB methods (ABB and LBB algorithms including the BS heuristic) for the CMWP instances. As a final remark, the reference values Z_R are taken as the best known solutions by (Sherali *et al.*, 2002) for the first class of CMWP instances. Similarly, we employ benchmark upper bounds as reference values for the second class of CMWP instances to be consistent with the first class CMWP instance results. The calculation of these benchmark upper bounds are selected as the best solutions among several solution approaches. These solution approaches are the single-commodity variants of CL-MDA and CL-RMDA heuristics, SABB, SLBB and SLBBCE algorithms and BS heuristic results.

10.1.2. Test Bed for the Multi-commodity Capacitated Multi-facility Weber Problem

While generating MCMWP test instances we try to use the available data from the CMWP test instances (Al-Loughani, 1997; Sherali *et al.*, 2002) as much as we can, in order to replicate existing experimental structure. Hence, we have adapted customer coordinates, unit transportation costs, customer demands and facility capacities from the available CMWP instances. Upper bounds on the total amount of multi-commodities shipped from facilities to

customers are randomly generated by GNETGEN [7] which is a modification of the well-known NETGEN generator by (Klingman *et al.*, 1974). GNETGEN is a test instance generator originally designed for the Generalized Multi-commodity Network Flow Problem (GMNFP). Notice that the MCMWP can be considered as a special case of the GMNFP when the number of source nodes is set to the number of facilities I, the number of sink nodes is set to the number of customers J and the number of transhipment nodes is set to 0 when facility locations are known.

We first run GNETGEN K times to generate K single commodity Transportation Problem (TP) instances. Then, we sum up upper bounds on the flow quantities shipped from facilities to customers to obtain bundle restrictions u_{ij} for $i = 1, \ldots, I$ and $j = 1, \ldots, J$. Namely, we set $u_{ij} = \sum_{k=1}^{K} u'_{ijk}$ for $i = 1, \ldots, I$, $j = 1, \ldots, J$ where u'_{ijk} is the upper bound on the flow quantity shipped from facility i to customer j in each of the K TP instances. Note that this setting of u_{ij} does not harm the feasibility of the MCMWP instance since each one of the K single commodity TP instance is guaranteed to be feasible by GNETGEN.

We generated 60 MCMWP test problems of various sizes according to the described procedure. The number of facilities I is selected between 2 and 10 and the number of customers J is selected between 2 and 30 by taking into account the sizes of the available CMWP test instances. The number of commodities K is chosen to be between 2 and 5. The numbers in an instance name stand for the size of the instance where the number of facilities I, the number of customers J and the number of commodities K are added after the prefix "mc", which stands for "multi-commodity". A similar approach is followed to classify the size of the MCMWP instances. Instances with $I \times J \times K \leq 120$, $120 < I \times J \times K \leq 250$ and $I \times J \times K \geq 250$ are qualified as "*small*", "*medium*" and "*large*" instances for the MCMWP, respectively. We call these 60 small to large instances as the first class of the test instances for the MCMWP.

In addition, we generate 18 very large instances which have 10 to 45 facilities and 100 to 150 customers. Clearly, any exact method will fail to solve these instances, in reasonable CPU times and thus, we only performed our heuristic approaches on these instances. These 18 very large instances are called as the second class of the test instances for the MCMWP. Benchmark lower and upper bounds for the MCMWP instances used as reference values Z_R

[7] downloadable from *http://netlib.sandia.gov/lp/ generators/index.html*

in this work are given in Appendix A together with a summary of outcomes of each one of the proposed methods.

10.1.3. Hardware and Software Environment

The experiments are performed on a Dell Server PE2900 with two 3.16 GHz Quad Core Processors and 32 GB RAM operating within Microsoft Windows Server 2003 environment in C^{++}. Cplex 11.0 with default options is used as a subroutine to solve the resulting LPs and MILPs which are part of suggested procedures implemented so far (CPLEX, 2007).

10.2. Computational Experiments

Computational experiments are performed on ALA heuristics, DA heuristics, MS algorithm, CI approach and BB methods (ABB and LBB algorithms and the BS heuristic). We separately present our results for each of the methods with the given order.

10.2.1. Alternate Location-Allocation Heuristics

Alternating location and allocation phases of the MCALA heuristic are executed until the difference between the objective values of two consecutive iterations are less than 0.0001. Recall that the allocation phase requires the solution of the MTP which is easy to solve by a commercial solver (such as Cplex). For the location phase, the Weiszfeld's algorithm is used. Weiszfeld's algorithm is an iterative method which consists of the following steps for the MCMWP.

$$x_{in}^{new} = \frac{\sum_{j=1}^{J}\sum_{k=1}^{K} a_{jn}c_{ijk}w_{ijk}/d(\mathbf{x}_i^{prev}, \mathbf{a}_j)}{\sum_{j=1}^{J}\sum_{k=1}^{K} c_{ijk}w_{ijk}/d(\mathbf{x}_i^{prev}, \mathbf{a}_j)}, i = 1, ..., I; n = 1, 2 \tag{10.3}$$

where the previous location of facility i denoted by $\mathbf{x}_i^{prev}$ is replaced with the new location of facility i indicated with $\mathbf{x}_i^{new}$. The procedure ends when the distance between $\mathbf{x}_i^{prev}$ and $\mathbf{x}_i^{new}$ becomes smaller than a tolerance value which is selected as 10^{-6} in our calculations. The distance term in the denominator may be zero when $\mathbf{x}_i^{prev}$ coincides with a customer location $\mathbf{a}_j$. As a consequence, we use a similar approach with (Frenk *et al.*, 1994) and add a

sufficiently small value which is selected as 10^{-6} to the distance measured by $d(\mathbf{x}_i^{prev}, \mathbf{a}_j)$ at each iteration. Moreover, the convergence of the algorithm is slow when the facility location lies in the close vicinity of a customer even if they do not coincide. Therefore, we limit the number of iterations to be less than 3000 to overcome these particular cases.

The computational results on the performance of the ALA heuristics (C-MCALA, C-MRR, C-MDRR) and their discrete variants (D-MCALA, D-MRR, D-MDRR) are reported in Table 10.1 for the first class of the test instances and Table 10.2 and Table 10.3 for the second class of the test instances. Table 10.1 and Table 10.2 include relative percent deviations of the upper bounds calculated according to Equation 10.1 after replacing Z_M with the best (smallest) values the heuristics compute in κ randomly initialized runs. Here

$$\kappa = \begin{cases} \max\{100, 5 \times I\} & \text{if } J \leq 50 \\ \max\{100, I \times J^{1/3}\} & \text{otherwise} \end{cases} \tag{10.4}$$

as proposed by (Luis *et al.*, 2009). κ is calculated for every instance and given in the second columns of Table 10.1–3 dedicated to location-allocation heuristics. The reference value Z_R to calculate relative percent deviation of the upper bounds "UB (%)" is taken as the benchmark lower bound values presented in Table A.1 and Table A.2. The values under the columns "CPU" in Table 10.1 and Table 10.3 are the total CPU times in seconds of these κ runs. In all these tables the first column indicates the test instance and the last row includes the column averages.

For the first class of MCMCWP instances the discrete variants of the ALA heuristics (D-MCALA, D-MRR, D-MDRR) yield better solutions than their continuous counterparts except for the MRR (with percent deviation less than 1%). However, we can say that the inverse holds true for the second class of the test instances. C-MCALA, C-MRR and C-MDRR perform better than their discrete counterparts. We observe that for the first class of the test instances the most accurate ALA heuristic is D-MCALA and for the second class of instances the most accurate solutions are obtained with C-MRR. At sum, C-MRR and D-MCALA are the most accurate ALA heuristics with 25.59% and 28.75% overall percent deviations in average, respectively. Furthermore, among all heuristics considered in this work, D-MCALA is the fastest one with 124.15 seconds of average total CPU time. The third best accuracy belongs to C-MCALA with 29.67% average percent deviation. Note that

Table 10.1. The performance of the ALA heuristics on the first group of the MCMWP instances.

Instance Name	κ	C-MCALA		D-MCALA		C-MRR		D-MRR		C-MDRR		D-MDRR	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_2_2_2	100	0.00	0.54	0.00	0.69	0.00	0.48	0.00	0.69	0.00	0.59	0.00	0.61
mc_2_2_3	100	0.00	0.89	0.00	0.58	0.00	1.11	0.00	0.70	0.00	1.13	0.00	0.72
mc_2_2_5	100	0.00	0.42	0.00	0.78	0.00	0.44	0.00	0.72	0.00	0.50	0.00	0.67
mc_2_4_2	100	4.08	8.45	0.10	0.72	0.10	4.27	0.10	0.75	0.10	6.08	0.10	0.77
mc_2_4_3	100	47.26	1.33	0.10	0.80	0.10	1.23	0.10	0.80	0.10	1.16	0.10	0.70
mc_2_4_5	100	0.10	2.41	0.10	0.77	0.10	2.00	0.10	0.83	0.10	1.84	0.10	0.88
mc_3_5_2	100	0.10	1.01	0.10	0.89	0.10	1.06	0.10	0.80	0.10	1.11	0.10	0.72
mc_3_5_3	100	0.10	2.69	0.10	0.84	0.10	2.34	0.10	0.86	0.10	2.41	0.10	0.84
mc_3_5_5	100	6.27	1.89	0.10	1.05	2.82	2.25	0.10	1.03	0.10	2.36	0.10	1.08
mc_3_9_2	100	6.33	4.70	1.71	2.56	0.10	7.03	1.71	3.02	0.10	5.80	1.71	2.48
mc_3_9_3	100	0.10	2.52	0.10	1.28	0.10	1.94	0.10	1.38	4.63	2.09	0.10	1.23
mc_3_9_5	100	0.90	10.00	0.90	5.36	0.90	3.34	0.90	5.59	0.90	3.94	0.90	5.30
mc_4_8_2	100	15.76	1.97	0.10	0.95	0.10	1.69	0.10	1.16	0.10	1.89	0.10	1.06
mc_4_8_3	100	0.10	2.42	0.10	1.14	0.10	1.86	0.10	1.22	0.10	3.86	0.10	1.36
mc_4_8_5	100	16.43	3.36	0.10	2.17	0.10	2.89	0.10	1.77	0.10	3.05	0.10	2.03
mc_4_10_2	100	2.47	4.81	0.87	1.36	2.47	2.14	2.47	1.53	2.47	3.53	6.01	1.70
mc_4_10_3	100	12.52	17.09	0.44	1.91	0.44	12.72	0.44	1.77	0.44	12.95	0.44	1.70
mc_4_10_5	100	2.26	2.09	0.10	2.39	0.10	2.41	2.26	3.06	0.10	3.84	0.10	2.98
mc_4_15_2	100	0.10	5.63	0.10	2.00	0.10	5.27	0.10	1.77	0.10	5.33	0.10	1.81
mc_4_15_3	100	7.76	6.64	7.01	5.61	6.91	3.27	7.01	4.88	6.91	3.05	7.01	5.88
mc_4_15_5	100	11.42	4.28	6.33	11.45	6.33	7.81	6.33	14.48	6.33	6.72	6.33	11.31

Table 10.1. The performance of the ALA heuristics on the first group of the MCMWP instances cont.

Instance Name	κ	C-MCALA		D-MCALA		C-MRR		D-MRR		C-MDRR		D-MDRR	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_5_8_2	100	18.59	2.25	7.36	1.16	44.91	1.27	7.36	1.30	18.59	0.88	0.10	1.69
mc_5_8_3	100	31.68	2.61	10.17	1.66	37.53	2.67	9.54	1.39	35.67	2.78	20.02	1.28
mc_5_8_5	100	15.58	3.25	13.51	1.64	13.51	4.53	13.51	1.69	13.51	4.56	6.82	2.44
mc_5_10_2	100	63.06	1.00	0.10	1.42	0.10	3.41	0.10	1.50	20.93	2.59	0.10	1.95
mc_5_10_3	100	133.62	15.27	11.20	3.34	21.91	12.64	14.89	2.80	133.62	13.13	0.99	4.88
mc_5_10_5	100	9.78	4.59	9.11	2.85	9.78	4.63	9.11	3.23	14.09	5.27	9.11	2.63
mc_5_12_2	100	3.40	1.45	3.40	1.45	3.40	1.77	3.40	1.66	3.40	1.59	3.40	1.39
mc_5_12_3	100	15.62	3.97	9.41	2.00	20.31	5.58	10.22	2.09	20.31	4.84	10.22	3.05
mc_5_12_5	100	11.28	6.69	19.85	6.50	19.03	5.73	22.25	4.36	32.51	4.69	18.49	8.61
mc_5_15_2	100	15.44	3.77	6.36	2.25	3.82	2.94	6.36	2.17	3.82	2.86	6.36	2.03
mc_5_15_3	100	10.03	10.78	4.54	9.91	14.23	8.45	14.23	9.27	16.19	7.22	12.25	7.14
mc_5_15_5	100	15.72	5.31	17.42	6.22	10.78	8.17	10.78	6.48	15.72	5.94	16.40	5.92
mc_5_20_2	100	15.60	7.86	12.12	2.69	8.83	8.58	15.64	3.30	8.83	7.84	9.29	4.86
mc_5_20_3	100	17.05	5.66	13.85	6.17	17.05	6.61	12.95	6.31	17.05	7.30	13.85	5.17
mc_5_20_5	100	20.21	6.51	12.38	13.13	11.13	9.00	13.63	10.39	25.18	7.70	14.06	13.25
mc_5_30_2	100	33.59	8.99	18.86	11.75	23.03	10.78	24.42	10.41	22.83	11.92	13.66	13.58
mc_5_30_3	100	11.83	10.63	9.76	15.70	10.02	12.56	11.62	11.47	11.62	17.00	11.62	9.59
mc_5_30_5	100	10.67	39.38	10.70	31.64	11.86	39.22	10.67	34.80	11.86	25.25	10.67	31.89
mc_6_10_2	100	5.18	1.31	5.18	1.16	5.18	1.84	5.18	1.31	18.91	1.91	5.18	1.45
mc_6_10_3	100	27.26	2.67	9.32	1.80	26.57	2.58	9.32	1.66	26.57	2.36	10.62	1.95
mc_6_10_5	100	13.02	6.81	12.76	2.73	12.71	4.00	13.08	2.38	65.48	3.88	12.71	2.44

Table 10.1. The performance of the ALA heuristics on the first group of the MCMWP instances cont.

Instance Name	κ	C-MCALA		D-MCALA		C-MRR		D-MRR		C-MDRR		D-MDRR	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_8_10_2	100	31.14	5.55	29.13	1.88	30.04	2.52	24.16	2.30	41.78	2.89	35.86	2.03
mc_8_10_3	100	70.38	6.73	28.29	1.97	32.65	8.95	26.95	1.97	34.41	5.02	17.26	1.97
mc_8_10_5	100	55.43	5.95	37.36	3.06	14.17	6.41	42.72	3.64	54.28	6.62	42.46	3.17
mc_10_10_2	100	98.85	9.41	68.09	1.88	58.75	6.27	65.13	1.91	72.49	4.89	55.80	1.58
mc_10_10_3	100	31.39	4.52	18.36	2.50	35.19	4.39	44.57	2.58	35.49	4.31	43.79	2.38
mc_10_10_5	100	105.46	5.34	87.94	2.94	117.80	6.64	101.21	2.78	120.32	5.31	124.19	2.80
mc_10_15_2	100	106.34	4.34	66.97	2.67	107.32	5.64	78.40	2.14	106.34	9.11	90.83	2.47
mc_10_15_3	100	34.66	6.11	35.22	2.91	30.52	7.06	46.81	3.16	34.66	6.06	41.24	3.14
mc_10_15_5	100	36.55	8.22	63.93	4.94	44.64	8.98	57.21	4.06	44.82	6.75	57.61	4.20
mc_10_20_2	100	94.68	10.39	53.48	3.28	97.48	8.52	112.78	4.19	114.13	8.75	85.46	3.25
mc_10_20_3	100	71.25	8.02	87.25	4.30	67.69	9.55	93.20	4.03	77.07	8.23	99.28	3.86
mc_10_20_5	100	44.24	10.16	44.95	13.39	43.10	16.41	50.91	8.20	41.44	15.48	53.47	6.89
mc_10_24_2	100	53.54	6.55	59.29	4.11	47.31	7.70	84.96	4.05	35.81	7.50	72.93	5.50
mc_10_24_3	100	59.61	10.48	62.15	6.06	54.09	12.13	41.98	5.06	58.40	11.22	41.48	5.44
mc_10_24_5	100	45.45	16.41	65.37	13.52	58.60	15.77	54.56	17.73	57.54	24.44	70.78	16.02
mc_10_30_2	100	44.51	8.02	47.79	6.11	29.39	9.30	29.51	5.34	49.44	6.69	29.49	5.41
mc_10_30_3	100	32.72	13.16	37.67	17.48	43.28	15.92	37.22	13.39	30.85	15.44	32.12	19.31
mc_10_30_5	100	60.56	37.86	49.58	12.00	46.12	37.17	56.40	21.36	41.60	34.23	38.12	13.91
Average	100	28.48	6.88	19.64	4.62	21.75	6.90	21.82	4.61	26.84	6.63	21.03	4.61

Table 10.2. The accuracy of the ALA heuristics on the second class of the MCMWP instances.

Instance Name	κ	C-MCALA UB(%)	D-MCALA UB(%)	C-MRR UB(%)	D-MRR UB(%)	C-MDRR UB(%)	D-MDRR UB(%)
mc_10_100_2	100	5.56	11.33	8.51	17.45	11.49	14.20
mc_10_100_3	100	8.82	16.99	17.58	26.76	17.09	24.04
mc_10_100_5	100	9.30	12.27	11.47	10.75	9.77	8.10
mc_15_150_2	100	27.58	33.45	36.38	34.84	28.42	36.89
mc_15_150_3	100	20.28	24.18	24.79	29.86	25.72	29.93
mc_15_150_5	100	16.35	16.62	20.14	22.35	21.22	20.93
mc_20_100_2	100	41.98	65.26	58.06	83.45	49.45	78.40
mc_20_100_3	100	61.81	78.60	71.76	117.19	68.05	101.64
mc_20_100_5	100	23.57	41.82	27.15	49.60	22.87	47.73
mc_30_100_2	140	214.73	199.09	193.58	215.05	234.29	255.36
mc_30_100_3	140	73.80	115.09	99.00	160.15	92.74	131.96
mc_30_100_5	140	27.44	55.51	44.91	55.59	39.17	68.80
mc_30_150_2	160	49.18	84.88	75.78	109.25	62.76	91.58
mc_30_150_3	160	45.00	53.98	61.95	69.92	56.24	86.23
mc_30_150_5	160	28.67	60.57	32.89	63.53	32.24	68.57
mc_45_150_2	240	186.58	313.39	153.13	316.72	181.19	362.99
mc_45_150_3	240	78.34	136.33	68.42	145.01	76.66	167.62
mc_45_150_5	240	66.46	113.27	55.84	104.13	70.48	102.15
Average	140	54.75	79.59	58.96	90.65	61.10	94.29

Table 10.3. The efficiency of the ALA heuristics on the second class of the MCMWP instances.

Instance Name	κ	C-MCALA CPU	D-MCALA CPU	C-MRR CPU	D-MRR CPU	C-MDRR CPU	D-MDRR CPU
mc_10_100_2	100	79.20	56.80	75.05	73.47	67.39	69.23
mc_10_100_3	100	95.52	85.56	124.52	87.81	148.20	135.42
mc_10_100_5	100	398.03	276.47	360.34	371.16	372.66	326.13
mc_15_150_2	100	408.23	126.55	291.16	123.30	349.94	167.64
mc_15_150_3	100	396.95	264.83	365.63	242.80	603.19	256.44
mc_15_150_5	100	1010.44	931.86	1002.09	1059.05	1075.70	994.64
mc_20_100_2	100	172.16	38.84	132.08	40.91	125.34	65.97
mc_20_100_3	100	258.05	77.06	190.50	81.95	225.69	72.13
mc_20_100_5	100	791.41	172.11	546.91	266.39	809.39	274.34
mc_30_100_2	140	329.00	102.09	277.48	146.70	317.55	82.77
mc_30_100_3	140	644.70	237.58	550.75	164.48	580.50	129.19
mc_30_100_5	140	1094.23	416.59	1322.11	448.12	1673.02	332.33
mc_30_150_2	160	1132.83	226.72	917.80	286.97	1504.84	237.83
mc_30_150_3	160	1367.34	493.70	1611.19	435.53	2272.35	491.81
mc_30_150_5	160	2505.94	1049.02	3445.20	1225.80	4347.55	986.78
mc_45_150_2	240	1145.63	763.61	1873.42	573.00	1706.77	690.20
mc_45_150_3	240	4379.14	1457.97	3325.48	1002.28	3753.66	1049.95
mc_45_150_5	240	11398.10	2629.06	11706.70	3604.95	8779.23	3476.73
Average	140	1533.72	522.58	1562.13	568.59	1595.16	546.64

the overall average accuracy of C-MCALA is slightly worse than that of D-MCALA. In particular, continuous ALA heuristics perform much better than their discrete variants for the second class of instances (namely, very large instances). Hence, we can say that among all ALA heuristics C-MRR and C-MCALA are the most viable ones. In addition, ALA heuristics play an important role in our work. In particular, C-MCALA and D-MCALA heuristics are incorporated in the proposed DA heuristics, MS algorithm and BB algorithms to calculate efficient upper bounds on the MCMWP.

10.2.2. Discrete Approximation Heuristics

10.2.2.1. Rectilinear Distance. Table 10.4 shows the strength of MDAP1 and MDAP2 formulations on the first class of the RMCMWP. The first column stands for the instance names. The second column presents the optimal values for these instances. As a reminder, the solution of the MDAP1 (or MDAP2) with the candidate facility locations constructed by the intersection points of vertically and horizontally drawn lines on the customers is optimum for the RMCMWP. In columns three to eight we give the LP relaxation (LPR) performance of MDAP1 and MDAP2. In particular, columns five and six stand for the LPR performance of MDAP2 where constraints given by Equation 5.15 and 5.16 are replaced with Equation 5.21. The duality gaps are measured according to Equation 10.2 and given under the columns named "GAP(%)". Z_{LB}^{final} and Z_{UB}^{final} are the final outcomes of the LPRs and Z_R is taken as the optimal value of the RMCMWP.

Table 10.5 shows the performance of the MDA heuristic on the first class of RMCMWP instances. We state only the CPU times of MDAP1 and MDAP2 formulations as their optimum solution is the same as in Table 10.4. The "LB(%)" and "UB(%)" stand for the relative percent deviations from the optimal value, respectively. They are calculated using Equation 10.1 by replacing the reference value Z_R with the optimal value of the RCMWP which is given under the second column of Table 10.4. Z_M is the lower bound and upper bound values of the instances for "LB(%)" and "UB(%)", respectively. The computational times in seconds are reported under "CPU". RMDAP1 and RMDAP2 indicate the results obtained with the LR of the MDAP1 and MDAP2, respectively.

Clearly, MDAP1 is more efficient than the MDAP2 with respect to their CPU times. Some particular instances are very difficult to solve and require drastic CPU times. Although

Table 10.4. The strength of the MDAP formulations on the first group of the RMCMWP instances.

Instance Name	Optimum Value	MDAP1		MDAP2 with Equation 5.21		MDAP2	
		Duality Gap(%)	CPU	Duality Gap(%)	CPU	Duality Gap(%)	CPU
mc_2_2_2	64	100.00	0.00	100.00	0.00	100.00	0.02
mc_2_2_3	464	100.00	0.02	100.00	0.02	100.00	0.00
mc_2_2_5	100	65.17	0.02	100.00	0.00	65.17	0.02
mc_2_4_2	797	99.03	0.02	100.00	0.02	99.03	0.00
mc_2_4_3	585	92.46	0.00	100.00	0.00	92.46	0.00
mc_2_4_5	2209	92.55	0.00	100.00	0.00	92.55	0.02
mc_3_5_2	1438	84.76	0.02	100.00	0.02	84.76	0.00
mc_3_5_3	572	99.97	0.02	100.00	0.02	99.97	0.02
mc_3_5_5	4701	61.71	0.03	100.00	0.02	61.71	0.05
mc_3_9_2	1817	100.00	0.03	100.00	0.05	100.00	0.05
mc_3_9_3	5997	89.35	0.06	100.00	0.06	89.35	0.08
mc_3_9_5	62706	96.79	0.09	100.00	0.09	96.79	0.16
mc_4_8_2	1566	97.93	0.02	100.00	0.02	97.93	0.02
mc_4_8_3	9923	85.72	0.05	100.00	0.05	85.72	0.08
mc_4_8_5	9420	72.94	0.06	100.00	0.06	72.94	0.13
mc_4_10_2	6437	89.62	0.03	100.00	0.05	89.62	0.06
mc_4_10_3	11507	92.16	0.08	100.00	0.08	92.16	0.14
mc_4_10_5	28142	79.20	0.16	100.00	0.17	79.20	0.31
mc_4_15_2	12424	97.39	0.33	100.00	0.34	97.39	0.41
mc_4_15_3	35807	92.37	0.61	100.00	0.56	92.37	0.97
mc_4_15_5	41685	92.67	1.67	100.00	1.42	92.67	3.34
mc_5_8_2	6250	93.68	0.03	100.00	0.03	93.68	0.06
mc_5_8_3	5551	100.00	0.06	100.00	0.08	100.00	0.09
mc_5_8_5	31353	76.84	0.16	100.00	0.16	76.84	0.31
mc_5_10_2	3928	86.69	0.06	100.00	0.06	86.69	0.06
mc_5_10_3	10064	71.85	0.09	100.00	0.11	71.85	0.14
mc_5_10_5	56300	79.51	0.22	100.00	0.17	79.51	0.38
mc_5_12_2	4316	99.11	0.22	100.00	0.19	99.11	0.24
mc_5_12_3	13947	94.53	0.39	100.00	0.36	94.53	0.56
mc_5_12_5	45687	74.10	1.00	100.00	0.80	74.10	1.66
mc_5_15_2	7784	99.78	0.34	100.00	0.33	99.78	0.45
mc_5_15_3	32833	91.87	1.67	100.00	1.41	91.87	2.31
mc_5_15_5	24186	89.19	1.75	100.00	1.49	89.19	3.42

Table 10.4. The strength of the MDAP formulations on the first group of the RMCMWP instances cont.

Instance Name	Optimum Value	MDAP1		MDAP2 with Equation 5.21		MDAP2	
		Duality Gap(%)	CPU	Duality Gap(%)	CPU	Duality Gap(%)	CPU
mc_5_20_2	9813	99.82	1.30	100.00	1.11	99.82	1.69
mc_5_20_3	20461	98.41	1.72	100.00	1.50	98.41	2.44
mc_5_20_5	58846	87.77	22.61	100.00	18.80	87.77	55.62
mc_5_30_2	56665	98.93	8.84	100.00	7.53	98.93	8.44
mc_5_30_3	78443	87.60	20.20	100.00	21.42	87.60	29.73
mc_5_30_5	224750	86.17	61.47	100.00	165.48	86.17	255.24
mc_6_10_2	3082	98.29	0.08	100.00	0.06	98.29	0.08
mc_6_10_3	6427	71.45	0.13	100.00	0.11	71.45	0.17
mc_6_10_5	11459	61.50	0.33	100.00	0.23	61.50	0.45
mc_8_10_2	7004	87.83	0.11	100.00	0.11	87.83	0.13
mc_8_10_3	10420	93.03	0.20	100.00	0.17	93.03	0.30
mc_8_10_5	21288	67.34	0.42	100.00	0.36	67.34	0.86
mc_10_10_2	3601	79.70	0.14	100.00	0.13	79.70	0.22
mc_10_10_3	13564	87.79	0.28	100.00	0.27	87.79	0.41
mc_10_10_5	5390	89.59	0.64	100.00	0.55	89.59	1.75
mc_10_15_2	2878	91.38	1.08	100.00	1.03	91.38	1.45
mc_10_15_3	6980	87.82	3.58	100.00	3.17	87.82	4.80
mc_10_15_5	13233	76.10	20.84	100.00	13.28	76.10	32.45
mc_10_20_2	10367	99.28	5.03	100.00	5.11	99.28	6.33
mc_10_20_3	5496	82.61	7.27	100.00	5.55	82.61	17.13
mc_10_20_5	18080	99.98	42.78	100.00	31.05	82.57	367.32
mc_10_24_2	4182	66.77	280.96	100.00	168.75	99.91	69.56
mc_10_24_3	11103	99.55	75.97	100.00	66.06	87.48	178.23
mc_10_24_5	25276	85.91	440.60	100.00	303.65	74.33	1037.46
mc_10_30_2	21333	85.23	394.55	100.00	99.31	99.77	40.27
mc_10_30_3	51096	87.30	1121.09	100.00	790.61	93.03	263.68
mc_10_30_5	82881	82.64	929.59	100.00	1208.63	82.64	1944.65
Average	21077.97	87.98	57.52	100.00	48.70	88.18	72.27

Table 10.5. The performance of the MDA heuristics on the first class of the RMCMWP instances.

Instance Name	MDAP1	MDAP2	RMDAP1			RMDAP2		
	CPU	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU
mc_2_2_2	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.02
mc_2_2_3	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.03
mc_2_2_5	0.02	0.02	0.00	0.00	0.02	0.00	0.00	0.53
mc_2_4_2	0.00	0.02	0.00	0.00	0.03	0.00	0.00	0.05
mc_2_4_3	0.00	0.00	0.00	0.00	138.00	0.00	0.00	420.55
mc_2_4_5	0.00	0.02	0.00	0.00	0.33	0.00	0.00	0.16
mc_3_5_2	0.02	0.03	0.00	0.00	0.20	0.00	0.00	3.02
mc_3_5_3	0.03	0.03	8.93	0.00	0.88	9.02	0.00	2.91
mc_3_5_5	0.14	0.14	8.59	0.57	0.34	8.67	0.57	2.08
mc_3_9_2	0.44	0.44	0.75	0.00	0.39	0.76	0.00	2.20
mc_3_9_3	0.83	0.89	6.12	0.00	0.63	6.30	0.00	3.80
mc_3_9_5	1.06	1.02	0.00	0.00	1.61	0.01	0.00	4.86
mc_4_8_2	0.27	0.16	4.33	0.00	0.39	4.34	0.00	2.75
mc_4_8_3	1.13	0.86	11.52	4.00	0.59	11.64	4.00	2.42
mc_4_8_5	0.27	0.30	0.57	0.00	0.78	0.99	0.00	4.52
mc_4_10_2	0.70	0.84	11.71	0.87	0.64	8.86	1.60	3.09
mc_4_10_3	0.49	0.59	0.84	0.00	0.95	0.98	0.00	5.45
mc_4_10_5	5.47	5.92	11.82	0.01	0.94	13.08	0.00	6.22
mc_4_15_2	8.34	8.55	4.03	0.00	2.25	4.38	0.00	9.38
mc_4_15_3	20.78	20.95	4.09	0.71	2.83	4.19	0.71	10.58
mc_4_15_5	38.64	49.64	3.84	1.94	3.42	4.83	1.94	9.25
mc_5_8_2	0.16	0.16	0.00	0.00	0.05	0.00	0.00	0.56
mc_5_8_3	1.69	1.86	16.45	7.89	0.64	17.93	7.89	4.56
mc_5_8_5	4.78	4.67	13.73	4.84	1.24	15.25	4.84	4.06
mc_5_10_2	0.63	0.70	13.21	8.40	0.66	13.27	1.17	2.50
mc_5_10_3	0.95	0.95	3.57	0.00	0.88	3.94	0.00	4.67
mc_5_10_5	5.25	6.59	8.07	6.61	1.34	8.07	1.56	5.22
mc_5_12_2	3.97	4.61	11.53	0.00	1.67	11.90	0.00	4.17
mc_5_12_3	16.78	18.70	15.78	0.44	1.77	15.90	0.44	6.92
mc_5_12_5	30.31	37.03	9.49	0.98	4.47	11.36	0.00	7.25
mc_5_15_2	2.67	3.09	1.71	0.00	2.22	1.85	0.00	5.70
mc_5_15_3	21.38	18.63	1.96	0.00	3.75	3.53	0.00	29.58
mc_5_15_5	32.58	36.19	7.31	0.00	4.16	8.12	0.00	9.00

Table 10.5. The performance of the MDA heuristics on the first class of the RMCMWP instances cont.

Instance Name	MDAP1	MDAP2	RMDAP1			RMDAP2		
	CPU	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU
mc_5_20_2	12.48	16.41	3.07	0.00	3.56	3.21	0.00	9.16
mc_5_20_3	54.28	60.50	4.03	8.37	4.91	4.34	5.77	14.03
mc_5_20_5	6832.14	10434.50	9.58	4.19	11.03	14.69	6.78	32.08
mc_5_30_2	256.95	233.41	2.39	1.13	9.19	3.39	0.89	25.91
mc_5_30_3	2271.83	2059.56	4.08	2.67	18.34	6.21	0.00	29.25
mc_5_30_5	7291.67	7604.22	2.03	0.30	25.74	2.59	0.24	71.14
mc_6_10_2	1.89	2.50	19.41	4.22	0.70	20.23	4.22	4.13
mc_6_10_3	1.49	1.50	16.42	5.09	0.78	17.31	5.09	3.63
mc_6_10_5	4.11	4.14	15.55	0.00	1.23	15.93	0.00	4.52
mc_8_10_2	2.02	1.72	34.14	20.52	1.00	34.35	10.95	4.28
mc_8_10_3	19.19	7.47	57.94	67.72	0.97	60.05	30.89	6.69
mc_8_10_5	31.53	35.59	22.86	22.01	1.59	21.45	8.20	6.66
mc_10_10_2	3.91	9.67	31.56	37.68	0.78	31.84	17.55	4.28
mc_10_10_3	111.38	109.06	36.94	6.78	1.70	41.76	13.28	4.81
mc_10_10_5	26.44	22.22	34.08	41.06	2.58	35.97	28.31	5.47
mc_10_15_2	124.22	90.67	35.76	61.22	3.38	36.43	40.17	8.20
mc_10_15_3	123.45	126.52	15.39	16.88	4.97	16.02	12.92	13.94
mc_10_15_5	3088.95	2690.47	23.21	12.48	10.05	23.84	7.90	25.91
mc_10_20_2	221.25	212.88	12.60	28.98	7.17	13.00	25.69	14.34
mc_10_20_3	1393.53	1413.77	21.79	21.96	7.13	22.81	19.10	14.48
mc_10_20_5	135920.00	106411.00	15.67	16.88	22.73	18.82	9.15	64.77
mc_10_24_2	5267.80	5476.11	13.05	15.21	15.25	14.04	9.28	46.17
mc_10_24_3	26998.60	33298.50	14.48	21.38	21.92	14.98	13.54	62.33
mc_10_24_5	43662.60	52343.10	9.41	8.54	32.61	13.89	8.54	111.88
mc_10_30_2	5385.80	5172.08	18.06	0.00	22.34	18.19	0.00	53.97
mc_10_30_3	53900.10	55699.50	14.24	10.30	29.77	15.50	7.56	63.39
mc_10_30_5	238939.00	287311.00	12.08	5.56	50.73	13.68	6.74	101.70
Average	8869.11	9517.86	11.33	7.97	8.17	12.06	5.29	23.09

MDAP2 is less efficient than the MDAP1, their LR results indicate that the converse holds for the accuracy of the upper bounds. In fact, the upper bounds of RMDAP2 are better than that of RMDAP1 with an increase in computational expenses. RMDAP1 produces slightly better lower bounds than the RMDAP2. This is expected from the LPR results which indicate that MDAP1 is tighter than the MDAP2 formulation. MDAP2 with Equation 5.21 produces a trivial lower bound of zero which is absolutely worse than the ones produced by MDAP2. Fortunately, better upper bounding performance of the MDAP2 over MDAP1 has motivated us to use both formulations for the MDA heuristics on the MCMWP instances with Euclidean distances.

10.2.2.2. Euclidean Distance. The results obtained with the DA heuristics (ℓ_1-MDA1, ℓ_1-MDA2, ℓ_∞-MDA1, ℓ_∞-MDA2, CL-MDA1, CL-MDA2) and with their relaxed versions (ℓ_1-RMDA1, ℓ_1-RMDA2, ℓ_∞-RMDA1, ℓ_∞-RMDA2, CL-RMDA1, CL-RMDA2) are summarized in Table 10.6, Table 10.7 and Table 10.8 for first class of MCMWP test instances. The first column indicates the instance name and the last rows provide column averages. We use the formula given by Equation 10.1 to calculate percent deviations from the reference value Z_R. For the "UB(%)" columns Z_M is replaced by the value of the feasible solutions computed by the heuristics (i.e., an upper bound on the optimal value Z^*), which is changed to the lower bound computed by using the approximation. There are two points which should be emphasized. First of all we do not report any results with the $\ell_{1\infty}$-norm based approximation methods $\ell_{1\infty}$-MDA1, $\ell_{1\infty}$-MDA2, $\ell_{1\infty}$-RMDA1 and $\ell_{1\infty}$-RMDA2 since they are very inefficient due to the extremely large number of intersection points. Second, we should remind that the customer based discrete approximations CL-MDA1, CL-MDA2, CL-RMDA1 and CL-RMDA2 provide only upper bounds on Z^*. The reference values Z_R are taken as the benchmark lower (upper) bounds given in Table A.1 and Table A.2 to calculate percent deviations UB (%) (LB (%)). Moreover, the results obtained with the second class of MCMWP instances are given in Table 10.9 and Table 10.10.

As it can be noticed, CL-MDA1 and CL-MDA2 are the most accurate discrete approximation heuristics. However, CL-MDA2 performs slightly better than CL-MDA1 at the expense of higher CPU times. One major weakness of both CL-MDA1 and CL-MDA2 is their extreme inefficiency. Notice that in Table 10.6, although we report the percent deviations obtained with the CL-MDA2, the solutions obtained with the CL-MDA1 and the CL-MDA2

are the same for the first class of the test instances. However, this is not the case for the second class of instances. That is to say we could not obtain upper bounds within CPU time limits using CL-MDA1 and CL-MDA2. Even four-hour CPU time limit is not sufficient to obtain a feasible solution with the CL-MDA1 and CL-MDA2 for some of the second (very large) class of instances. These cases are indicated with "N/A" in the tables. Furthermore, note also that no result with ℓ_1-MDA1, ℓ_1-MDA2, ℓ_∞-MDA1 and ℓ_∞-MDA2 is reported for the second class instances because of the same reason.

Recall that to alleviate the excessive CPU times required for the solution of MDAP1 and MDAP2, we propose LR schemes. They become especially useful for large instances. The use of LR considerably increases the efficiency of ℓ_1-RMDA1, ℓ_1-RMDA2, ℓ_∞-RMDA1, ℓ_∞-RMDA2, CL-RMDA1, and CL-RMDA2. Namely, they compute good solutions in reasonable CPU times on large instances. Finally, that among all relaxed MDA heuristics the most accurate one is CL-RMDA2 with an overall average percent deviation of 6.05%.

At first look, we can say that even the least accurate MDA heuristic ℓ_1-RMDA2 yields lower percent relative deviations than the most accurate location-allocation heuristic C-MRR: 17.29% versus 25.59%. In addition, the overall average total CPU times for them are 365.78 and 355.78 seconds, respectively. This implies a superiority of the DA heuristics over the ALA heuristics. The tightest of the lower bounds is computed by ℓ_∞-MDA1 and ℓ_∞-MDA2 with an overall average percent deviation of 10.26% and 10.25%, respectively; however, they have the highest CPU times. Notice that the difference between these two equivalent formulations stems from the four hour of CPU time limit. A "N/A" indicates that the validity of the lower bound values calculated by block norm based MDA heuristics is not guaranteed when four hours of time limit is exceeded. On the other hand, the upper bounds are clearly valid as long as a feasible solution is produced within the CPU time limit. What is more, for the second class of the test instances, the ℓ_∞-RMDA1 produces the tightest lower bound values. Generally speaking, considering the ℓ_1 and ℓ_∞-norm based approximations we observe that the ℓ_1-norms based approximation yields more efficient but less accurate results than the ℓ_∞-norm based approximation does, except the ℓ_1-RMDA1 heuristic.

In summary, these experiments have encouraged us to use the lower bounds produced by ℓ_∞-MDA1 and ℓ_∞-RMDA1 heuristics within the BB methods developed. In particular, we favor the more efficient ℓ_∞-RMDA1 to be used within a BB algorithm since the computational expense of ℓ_∞-MDA1 becomes excessive with increasing instance sizes.

Table 10.6. The performance of the CL-MDA heuristics on the first class of the MCMWP instances.

Instance	CL-MDA1		CL-MDA2		CL-RMDA1		CL-MDA2	
Name	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_2_2_2	0.00	0.06	0.00	0.02	0.00	0.03	0.00	0.08
mc_2_2_3	0.00	0.13	0.00	0.00	0.00	0.03	0.00	0.09
mc_2_2_5	0.00	0.06	0.00	0.00	0.00	0.08	0.00	0.13
mc_2_4_2	0.10	0.08	0.10	0.02	0.10	0.14	0.10	0.17
mc_2_4_3	0.10	0.08	0.10	0.00	0.10	0.38	0.10	19.88
mc_2_4_5	0.10	0.17	0.10	0.02	0.10	1.34	0.10	1.17
mc_3_5_2	0.10	0.17	0.10	0.00	0.10	0.44	0.10	0.58
mc_3_5_3	0.10	0.11	0.10	0.05	0.10	0.61	0.10	12.31
mc_3_5_5	0.10	0.28	0.10	0.06	0.10	1.39	0.10	13.50
mc_3_9_2	0.10	0.16	0.10	0.08	0.10	4.48	1.71	16.34
mc_3_9_3	0.10	0.36	0.10	0.13	0.10	1.61	0.10	27.06
mc_3_9_5	0.90	0.23	0.90	0.14	0.90	11.27	0.90	65.14
mc_4_8_2	0.04	0.27	0.04	0.05	0.03	0.34	0.03	17.72
mc_4_8_3	0.10	0.42	0.10	0.13	1.57	1.94	0.10	25.45
mc_4_8_5	0.05	0.48	0.05	0.08	0.05	4.02	0.05	45.20
mc_4_10_2	0.10	0.38	0.10	0.11	0.87	3.31	2.47	28.20
mc_4_10_3	0.44	0.52	0.44	0.20	0.44	2.55	0.44	47.23
mc_4_10_5	0.03	1.11	0.03	0.75	2.19	4.27	2.19	68.97
mc_4_15_2	0.10	0.80	0.10	0.44	0.10	2.09	0.10	81.47
mc_4_15_3	0.62	1.16	0.62	1.06	0.65	5.70	0.65	65.72
mc_4_15_5	0.01	2.14	0.01	1.61	0.01	10.38	0.01	69.03
mc_5_8_2	0.10	0.31	0.10	0.05	0.10	0.70	0.10	1.05
mc_5_8_3	0.10	0.52	0.10	0.22	10.17	1.28	10.17	10.59
mc_5_8_5	6.82	0.88	6.82	0.41	10.55	5.52	8.87	17.13
mc_5_10_2	0.10	0.42	0.10	0.14	0.10	6.97	0.10	12.47
mc_5_10_3	0.10	0.64	0.10	0.24	0.10	4.69	0.10	12.92
mc_5_10_5	0.29	1.27	0.29	0.73	0.29	4.81	0.29	21.81
mc_5_12_2	3.40	0.72	3.40	0.28	3.40	1.56	3.40	8.36
mc_5_12_3	0.34	1.25	0.34	0.78	0.34	2.13	0.34	9.59
mc_5_12_5	11.28	1.44	11.28	1.22	11.29	5.70	11.28	24.48
mc_5_15_2	0.01	0.52	0.01	0.34	0.01	7.03	0.01	10.08
mc_5_15_3	0.04	1.03	0.04	0.50	0.04	10.31	0.04	17.50
mc_5_15_5	4.79	2.25	4.79	1.30	4.79	5.91	4.79	26.83

Table 10.6. The performance of the CL-MDA heuristics on the first class of the MCMWP instances cont.

Instance	CL-MDA1		CL-MDA2		CL-RMDA1		CL-MDA2	
Name	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_5_20_2	1.35	0.94	1.35	0.67	1.35	9.27	1.35	21.94
mc_5_20_3	2.05	4.09	2.05	4.94	4.42	8.42	2.38	24.63
mc_5_20_5	0.02	11.91	0.02	11.36	0.02	16.88	0.15	46.03
mc_5_30_2	0.00	5.22	0.00	4.31	0.85	13.27	0.00	36.70
mc_5_30_3	0.00	24.88	0.00	23.58	0.00	11.34	0.00	52.81
mc_5_30_5	0.01	28.48	0.01	28.97	0.01	46.98	0.01	86.94
mc_6_10_2	0.11	0.56	0.11	0.39	5.18	1.03	0.11	11.06
mc_6_10_3	0.09	0.70	0.09	0.27	0.09	4.86	0.09	19.77
mc_6_10_5	0.11	1.30	0.11	0.72	0.11	2.64	0.11	17.52
mc_8_10_2	0.05	0.84	0.05	0.39	15.86	3.19	15.28	10.58
mc_8_10_3	0.37	1.97	0.37	1.88	37.88	3.78	13.18	25.81
mc_8_10_5	0.00	4.05	0.00	4.50	11.94	7.95	4.86	28.56
mc_10_10_2	0.00	4.86	0.00	4.34	27.86	1.53	16.59	11.11
mc_10_10_3	0.00	11.97	0.00	14.36	0.87	6.30	1.85	22.78
mc_10_10_5	0.01	9.64	0.01	8.36	30.41	9.83	11.40	31.31
mc_10_15_2	0.14	16.75	0.14	16.67	14.96	5.42	48.07	19.23
mc_10_15_3	0.01	8.09	0.01	8.41	4.85	7.77	2.84	33.95
mc_10_15_5	0.20	49.69	0.20	22.25	12.24	11.08	6.16	54.08
mc_10_20_2	0.23	5.64	0.23	5.09	20.55	8.77	0.23	34.34
mc_10_20_3	0.01	75.80	0.01	70.75	34.14	6.59	11.82	39.88
mc_10_20_5	0.14	226.47	0.14	314.38	14.39	26.38	4.24	87.08
mc_10_24_2	0.47	43.70	0.47	51.42	2.59	10.42	8.23	31.11
mc_10_24_3	0.01	113.75	0.01	145.75	22.42	18.44	13.99	58.59
mc_10_24_5	0.01	65.52	0.01	70.80	12.19	23.94	3.05	119.89
mc_10_30_2	0.16	89.14	0.16	77.39	0.16	9.58	0.16	56.83
mc_10_30_3	0.16	803.56	0.16	1130.72	15.25	24.72	11.55	86.00
mc_10_30_5	0.06	634.89	0.06	944.30	3.55	55.98	3.55	160.06
Average	0.61	37.75	0.61	49.63	5.72	7.82	3.84	33.45

Table 10.7. The performance of the ℓ_1-MDA heuristics on the first class of the MCMWP instances.

Instance Name	ℓ_1-MDA1			ℓ_1-MDA2			ℓ_1-RMDA1			ℓ_1-RMDA2		
	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU
mc_2_2_2	29.30	0.00	0.00	29.29	0.00	0.02	29.30	0.00	0.06	29.30	0.00	0.08
mc_2_2_3	29.29	0.00	0.00	29.29	0.00	0.00	29.29	0.00	0.08	29.29	0.00	0.06
mc_2_2_5	29.29	0.00	0.00	29.29	0.00	0.00	29.29	0.00	0.48	29.29	0.00	0.08
mc_2_4_2	10.81	0.10	0.02	10.81	0.10	0.00	10.81	0.10	0.11	10.81	0.10	0.17
mc_2_4_3	15.51	0.10	0.02	15.51	0.10	0.00	15.51	0.10	0.45	22.08	0.10	0.27
mc_2_4_5	9.91	0.10	0.02	9.91	0.10	0.02	9.92	0.10	0.28	9.93	0.10	0.30
mc_3_5_2	9.57	0.10	0.02	9.57	0.10	0.02	9.57	0.10	0.64	9.59	0.10	0.27
mc_3_5_3	5.58	0.10	0.03	5.58	0.10	0.03	14.06	0.10	1.38	14.24	0.10	0.63
mc_3_5_5	16.27	9.12	0.14	16.27	9.12	0.14	23.48	0.10	1.20	23.95	0.10	0.53
mc_3_9_2	14.69	2.01	0.45	14.69	2.01	0.42	15.35	0.10	1.09	15.48	0.10	1.14
mc_3_9_3	13.80	4.63	0.83	13.80	4.63	0.91	19.12	4.63	2.31	19.40	0.10	0.83
mc_3_9_5	15.98	0.90	1.06	15.98	0.90	1.03	16.00	0.90	3.48	16.06	0.90	1.94
mc_4_8_2	6.12	1.38	0.27	6.12	1.38	0.16	10.23	0.03	1.17	10.63	0.03	0.63
mc_4_8_3	14.73	0.10	1.13	14.73	0.10	0.86	24.60	1.57	1.67	24.75	8.51	0.86
mc_4_8_5	13.95	0.05	0.27	13.95	0.05	0.30	14.50	0.05	1.39	15.29	0.05	1.11
mc_4_10_2	18.10	1.67	0.72	18.10	1.67	0.84	24.30	5.61	2.91	27.87	5.61	1.34
mc_4_10_3	18.10	0.44	0.55	18.10	0.44	0.58	18.80	0.44	5.64	20.56	12.52	2.23
mc_4_10_5	15.28	3.55	5.49	15.28	3.55	5.91	25.37	2.19	3.80	28.70	2.19	1.50
mc_4_15_2	7.80	0.10	8.34	7.80	0.10	8.63	11.53	0.10	4.22	13.49	0.10	2.19
mc_4_15_3	12.36	0.65	20.77	12.36	0.65	22.06	16.01	0.65	5.86	17.46	1.44	2.11
mc_4_15_5	13.61	0.01	38.61	13.61	0.01	47.92	17.44	0.01	9.75	20.17	0.01	4.36

Table 10.7. The performance of the ℓ_1-MDA heuristics on the first class of the MCMWP instances cont.

Instance Name	ℓ_1-MDA1			ℓ_1-MDA2			ℓ_1-RMDA1			ℓ_1-RMDA2		
	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU
mc_5_8_2	9.06	0.10	0.17	9.06	0.10	0.16	9.08	0.10	0.72	9.08	0.10	0.56
mc_5_8_3	3.56	0.10	1.69	3.56	0.10	1.97	19.51	10.17	1.23	26.11	10.17	0.97
mc_5_8_5	10.39	6.82	4.78	10.39	6.82	4.72	22.64	10.55	4.77	26.48	8.87	1.58
mc_5_10_2	11.70	0.88	0.63	11.70	0.88	0.73	23.40	0.10	4.14	24.70	16.48	1.44
mc_5_10_3	11.31	0.10	0.95	11.31	0.10	1.00	14.52	0.10	5.02	16.99	0.10	3.22
mc_5_10_5	8.35	0.29	5.24	8.35	0.29	6.56	15.24	0.29	4.06	19.67	0.29	2.39
mc_5_12_2	5.22	3.40	3.97	5.22	3.40	4.56	16.27	3.40	2.28	17.27	3.40	1.17
mc_5_12_3	8.95	2.16	17.05	8.95	2.16	18.83	23.40	0.34	3.77	24.28	3.83	1.25
mc_5_12_5	10.14	11.32	30.39	10.14	11.32	37.52	19.21	11.32	7.95	23.14	11.32	4.31
mc_5_15_2	16.39	0.01	2.72	16.39	0.01	3.27	17.88	0.01	5.08	18.01	0.01	2.00
mc_5_15_3	9.33	0.04	21.47	9.33	0.04	19.28	11.17	0.04	16.39	15.02	0.04	6.72
mc_5_15_5	9.83	4.79	32.53	9.83	4.79	34.99	17.17	4.79	4.95	19.40	4.79	2.61
mc_5_20_2	17.18	1.35	12.47	17.18	1.35	16.17	19.75	1.35	9.86	20.47	1.35	3.25
mc_5_20_3	15.01	5.18	54.39	15.01	5.18	60.81	18.50	4.73	8.78	20.67	6.06	4.05
mc_5_20_5	9.66	2.65	9752.55	N/A	3.96	14410.10	18.73	0.02	17.58	29.13	4.44	7.75
mc_5_30_2	14.83	0.29	239.89	14.83	0.29	235.09	16.93	0.00	20.00	18.70	0.97	5.38
mc_5_30_3	13.67	0.69	2816.36	13.67	0.69	2247.64	17.25	0.00	22.19	20.74	0.68	8.14
mc_5_30_5	12.44	0.15	9248.92	N/A	0.15	14431.00	14.29	0.01	43.28	16.56	0.15	14.41
mc_6_10_2	13.20	0.11	2.13	13.20	0.11	2.75	30.09	5.18	2.89	30.77	5.18	0.88
mc_6_10_3	11.03	0.09	2.02	11.03	0.09	1.66	25.63	11.91	5.67	27.33	0.09	2.05
mc_6_10_5	13.32	0.11	4.63	13.32	0.11	4.56	26.71	0.11	4.22	27.45	0.11	3.05

Table 10.7. The performance of the ℓ_1-MDA heuristics on the first class of the MCMWP instances cont.

Instance	ℓ_1-MDA1			ℓ_1-MDA2			ℓ_1-RMDA1			ℓ_1-RMDA2		
Name	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU
mc_8_10_2	13.58	0.05	2.80	13.58	0.05	1.94	43.14	15.86	4.27	44.73	19.46	1.52
mc_8_10_3	18.40	1.43	21.30	18.40	1.43	8.30	65.70	37.88	3.80	68.96	22.98	2.81
mc_8_10_5	17.83	0.00	35.66	17.83	0.00	39.16	34.89	19.57	6.55	39.29	11.46	2.27
mc_10_10_2	21.98	0.00	4.42	21.98	0.00	9.69	46.77	28.00	3.66	48.66	25.20	1.39
mc_10_10_3	13.49	0.87	111.42	13.49	0.87	109.70	45.43	1.85	8.38	53.90	2.11	2.27
mc_10_10_5	22.34	0.01	29.75	22.34	0.01	24.67	49.00	28.91	6.33	52.32	25.33	2.70
mc_10_15_2	14.55	0.14	164.02	14.55	0.14	98.86	45.41	50.51	4.61	48.47	39.92	3.59
mc_10_15_3	19.50	0.01	126.11	19.50	0.01	128.77	31.94	3.76	11.17	33.50	11.87	3.56
mc_10_15_5	9.35	0.81	5808.30	9.35	0.81	3237.47	30.27	12.59	15.86	35.38	7.85	6.03
mc_10_20_2	11.90	0.24	218.47	11.90	0.24	241.05	23.05	5.86	18.47	24.66	25.94	4.33
mc_10_20_3	11.99	0.01	1397.48	11.99	0.01	1463.45	31.21	31.32	9.73	32.46	27.23	4.27
mc_10_20_5	N/A	0.22	14512.67	N/A	0.57	14544.50	25.08	12.42	32.28	34.73	22.71	12.92
mc_10_24_2	9.77	1.21	9096.97	9.77	1.21	10203.30	21.60	15.97	17.25	28.35	27.70	6.91
mc_10_24_3	N/A	2.87	14434.72	N/A	2.87	14454.50	22.95	17.44	24.14	28.01	2.02	11.55
mc_10_24_5	N/A	0.01	14605.84	N/A	0.01	14675.30	18.75	4.01	43.38	27.75	14.14	21.80
mc_10_30_2	14.82	2.07	9484.25	14.82	2.07	8167.95	30.25	3.67	23.63	32.61	7.91	7.98
mc_10_30_3	N/A	0.29	14529.70	N/A	0.29	14442.90	27.29	13.62	49.39	31.18	35.99	16.78
mc_10_30_5	N/A	1.14	14709.13	N/A	0.97	14751.20	24.47	2.61	67.02	30.42	10.83	22.14
Average	13.71	1.28	2027.11	13.81	1.31	2137.26	23.32	6.45	9.88	25.93	7.52	3.91

Table 10.8. The performance of the ℓ_∞-MDA heuristics on the first class of the MCMWP instances.

Instance	ℓ_∞-MDA1			ℓ_∞-MDA2			ℓ_∞-RMDA1			ℓ_∞-RMDA2		
Name	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU
mc_2_2_2	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.08
mc_2_2_3	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.09
mc_2_2_5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.20
mc_2_4_2	7.42	0.10	0.02	7.42	0.10	0.02	7.42	0.10	0.11	7.42	0.10	0.42
mc_2_4_3	12.17	0.10	0.02	12.17	0.10	0.02	12.17	0.10	1.00	46.69	0.10	0.22
mc_2_4_5	7.43	0.10	0.02	7.43	0.10	0.02	7.46	0.10	0.28	7.47	0.10	0.67
mc_3_5_2	9.64	0.10	0.03	9.64	0.10	0.02	9.64	0.10	0.63	9.64	0.10	0.64
mc_3_5_3	14.33	0.10	0.03	14.33	0.10	0.03	27.19	0.10	1.25	27.53	0.10	1.64
mc_3_5_5	9.60	0.10	0.14	9.60	0.10	0.14	15.94	0.10	1.77	16.22	0.10	1.59
mc_3_9_2	16.80	6.33	0.61	16.80	6.33	0.73	20.53	0.10	3.36	20.69	0.10	4.20
mc_3_9_3	9.18	0.10	0.63	9.18	0.10	0.58	12.40	0.10	1.17	12.67	0.10	1.66
mc_3_9_5	20.08	0.90	0.81	20.08	0.90	0.98	20.10	0.90	1.95	20.15	0.90	3.50
mc_4_8_2	20.14	0.03	0.11	20.14	0.03	0.11	22.90	0.03	0.73	22.98	0.03	1.28
mc_4_8_3	7.44	0.10	1.13	7.44	0.10	1.17	13.39	0.10	3.06	13.67	0.10	4.45
mc_4_8_5	8.67	0.05	0.31	8.67	0.05	0.31	12.50	0.05	1.88	12.52	0.05	2.50
mc_4_10_2	5.95	0.10	2.52	5.95	0.10	3.02	17.39	5.61	2.33	17.55	2.47	2.17
mc_4_10_3	5.90	0.44	2.64	5.90	0.44	2.13	10.23	12.52	2.72	11.80	0.44	6.09
mc_4_10_5	9.98	6.79	10.00	9.98	6.79	10.73	14.52	2.19	5.28	15.78	2.19	6.39
mc_4_15_2	15.34	0.10	7.27	15.34	0.10	7.52	17.15	0.10	3.94	17.33	0.10	4.28
mc_4_15_3	11.53	0.65	89.20	11.53	0.65	104.53	18.43	0.65	5.55	18.98	0.65	8.69
mc_4_15_5	9.01	0.01	175.97	9.01	0.01	194.59	16.14	0.01	10.78	15.81	0.01	18.20

Table 10.8. The performance of the ℓ_∞-MDA heuristics on the first class of the MCMWP instances cont.

Instance	ℓ_∞-MDA1			ℓ_∞-MDA2			ℓ_∞-RMDA1			ℓ_∞-RMDA2		
Name	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU
mc_5_8_2	16.31	8.44	0.39	16.31	8.44	0.38	16.50	0.10	1.92	16.61	0.10	2.48
mc_5_8_3	18.99	0.10	1.91	18.99	0.10	1.63	31.68	10.17	1.63	32.89	10.17	2.30
mc_5_8_5	11.26	8.87	6.05	11.26	8.87	4.92	29.31	8.87	3.97	30.37	6.82	4.98
mc_5_10_2	8.48	0.10	0.44	8.48	0.10	0.44	15.71	0.10	3.66	16.63	0.10	3.98
mc_5_10_3	12.19	0.99	1.22	12.19	0.99	1.25	16.56	0.99	4.25	17.30	0.99	6.39
mc_5_10_5	13.86	0.29	6.97	13.86	0.29	8.33	20.55	0.29	6.16	22.10	0.29	6.72
mc_5_12_2	15.09	3.40	4.63	15.09	3.40	4.84	23.47	3.40	1.88	23.63	3.40	2.77
mc_5_12_3	11.95	0.34	19.61	11.95	0.34	20.73	23.94	0.34	3.45	24.64	0.34	5.58
mc_5_12_5	14.80	11.32	48.95	14.80	11.32	68.75	24.40	11.28	8.67	24.58	11.28	18.13
mc_5_15_2	8.64	0.01	6.38	8.64	0.01	6.34	13.21	0.01	6.16	13.43	0.01	4.80
mc_5_15_3	11.17	0.04	18.55	11.17	0.04	18.00	12.11	0.04	11.55	13.44	0.04	20.22
mc_5_15_5	12.40	4.79	66.44	12.40	4.79	53.39	19.09	4.79	6.77	19.73	4.79	8.77
mc_5_20_2	11.59	1.35	29.00	11.59	1.35	29.09	14.07	1.35	9.78	15.41	1.35	11.50
mc_5_20_3	12.04	2.04	560.27	12.04	2.99	218.95	17.14	4.42	13.00	18.91	2.05	12.78
mc_5_20_5	11.11	0.15	2764.91	11.11	0.15	2914.95	17.52	0.40	16.34	21.92	1.51	36.69
mc_5_30_2	9.12	0.00	1164.33	9.12	0.00	473.59	11.12	0.00	19.72	12.05	0.00	27.66
mc_5_30_3	9.09	0.00	9068.31	9.09	0.00	11509.10	14.06	2.06	21.94	15.44	0.00	32.77
mc_5_30_5	10.68	0.12	17849.00	N/A	0.12	14575.40	12.56	0.01	39.23	13.27	0.01	64.56
mc_6_10_2	9.87	0.11	1.74	9.87	0.11	1.89	25.07	0.11	2.36	26.68	0.11	3.05
mc_6_10_3	8.95	0.09	2.72	8.95	0.09	2.22	18.77	0.09	3.45	20.07	0.09	5.08
mc_6_10_5	8.75	0.11	6.02	8.75	0.11	5.73	23.12	0.11	3.64	23.50	0.11	5.06

Table 10.8. The performance of the ℓ_∞-MDA heuristics on the first class of the MCMWP instances cont.

Instance	ℓ_∞-MDA1			ℓ_∞-MDA2			ℓ_∞-RMDA1			ℓ_∞-RMDA2		
Name	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU
mc_8_10_2	8.48	0.05	3.73	8.48	0.05	3.03	42.24	23.38	4.50	42.54	14.60	4.02
mc_8_10_3	5.99	0.37	44.02	5.99	0.37	37.00	62.24	32.09	5.25	62.98	19.77	4.09
mc_8_10_5	6.75	0.00	94.02	6.75	0.00	59.69	33.72	4.86	8.42	30.57	1.65	7.75
mc_10_10_2	3.73	1.49	38.69	3.73	1.49	30.84	38.77	17.92	4.67	40.42	9.85	5.20
mc_10_10_3	10.36	0.00	139.80	10.36	0.00	192.34	41.30	0.75	8.27	47.08	3.64	6.02
mc_10_10_5	5.31	3.56	72.19	5.31	3.56	86.91	41.55	37.85	8.97	43.08	14.37	8.39
mc_10_15_2	8.42	0.14	187.95	8.42	0.14	152.72	42.92	40.46	6.11	44.79	36.41	5.53
mc_10_15_3	7.80	5.76	1543.94	7.80	5.76	943.30	26.60	14.78	10.83	27.97	12.10	9.94
mc_10_15_5	13.10	0.20	1912.67	13.10	0.20	1932.06	31.45	6.16	17.27	33.11	7.96	22.16
mc_10_20_2	9.32	0.23	950.06	9.32	0.23	872.50	24.27	28.66	10.88	26.88	11.05	15.55
mc_10_20_3	12.09	0.14	9562.39	12.09	0.14	11196.60	30.69	11.73	10.61	32.05	8.64	13.66
mc_10_20_5	N/A	1.83	14437.11	N/A	1.83	14451.80	24.36	10.95	35.88	30.74	7.35	52.24
mc_10_24_2	15.72	0.47	4258.23	15.72	0.47	5713.02	27.50	2.59	14.81	28.33	2.59	17.95
mc_10_24_3	N/A	0.01	14435.33	N/A	0.01	14430.60	26.88	12.22	28.13	28.86	14.15	34.50
mc_10_24_5	N/A	0.01	14605.48	N/A	0.01	14572.90	19.08	9.49	42.64	25.10	12.43	55.61
mc_10_30_2	N/A	0.16	14435.48	N/A	0.16	14428.60	24.65	1.07	22.91	25.79	2.00	37.28
mc_10_30_3	N/A	2.02	14472.02	N/A	2.02	14737.70	74.76	18.96	18.61	23.63	5.56	62.97
mc_10_30_5	N/A	0.77	14966.81	N/A	0.77	15052.80	19.89	9.86	95.63	22.45	2.61	119.38
Average	10.26	1.27	2301.32	10.25	1.28	2319.02	21.81	5.93	9.87	22.56	3.97	13.96

Table 10.9. The performance of the CL-MDA heuristics on the second class of the MCMWP instances.

Instance Name	CL-MDA1		CL-MDA2		CL-RMDA1		CL-RMDA2	
	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_10_100_2	0.42	5349.89	0.42	9251.61	0.71	118.98	0.42	122.59
mc_10_100_3	0.05	14530.70	0.05	14574.50	7.38	173.47	4.94	228.56
mc_10_100_5	3.38	15163.50	0.12	15185.00	2.94	548.39	4.43	616.16
mc_15_150_2	1.07	15986.30	1.07	16016.50	11.73	376.39	13.55	347.50
mc_15_150_3	10.38	18286.70	9.98	18352.20	6.63	788.22	10.28	746.78
mc_15_150_5	N/A	N/A	N/A	N/A	2.45	1692.45	4.59	2056.73
mc_20_100_2	1.10	14797.10	0.50	14854.10	31.25	209.88	28.83	143.23
mc_20_100_3	0.15	15578.90	2.78	15607.20	22.85	281.73	13.37	333.84
mc_20_100_5	0.04	18155.20	2.74	18252.50	9.47	859.38	7.10	877.47
mc_30_100_2	0.30	15626.60	3.07	15697.10	118.47	260.69	82.75	254.84
mc_30_100_3	0.03	17464.10	4.41	17567.10	27.77	573.91	22.89	597.72
mc_30_100_5	N/A	N/A	N/A	N/A	19.21	1219.17	4.58	1426.63
mc_30_150_2	18.80	21547.00	7.28	21694.40	19.61	637.23	1.05	828.95
mc_30_150_3	N/A	N/A	N/A	N/A	3.89	1223.47	3.59	1572.42
mc_30_150_5	N/A	N/A	N/A	N/A	0.00	4002.47	2.30	4484.83
mc_45_150_2	N/A	N/A	N/A	N/A	102.41	1274.30	37.24	1846.33
mc_45_150_3	N/A	N/A	N/A	N/A	31.76	2828.09	0.11	3370.69
mc_45_150_5	N/A	N/A	N/A	N/A	16.40	7019.06	0.04	7281.70
Average	3.25	15680.54	2.95	16095.66	24.16	1338.18	13.45	1507.61

Table 10.10. The performance of ℓ_1 and ℓ_∞-MDA heuristics on the second class of the MCMWP instances.

Instance	ℓ_1-RMDA1			ℓ_1-RMDA2			ℓ_∞-RMDA1			ℓ_∞-RMDA2		
Name	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU	LB(%)	UB(%)	CPU
mc_10_100_2	12.55	6.62	504.39	23.14	6.62	216.66	13.30	1.62	682.78	18.97	5.09	1151.44
mc_10_100_3	13.70	5.65	892.19	26.35	17.34	318.55	12.94	9.03	861.78	19.24	12.19	1934.16
mc_10_100_5	19.07	6.81	1402.19	38.13	6.70	687.56	17.38	6.82	2289.31	24.57	4.02	3835.09
mc_15_150_2	17.23	10.23	1957.16	29.36	17.27	659.33	17.14	21.34	2404.20	23.42	12.01	5241.48
mc_15_150_3	17.78	11.85	2655.98	33.30	23.48	995.86	17.81	9.08	4378.42	25.71	19.96	6833.89
mc_15_150_5	21.44	2.22	5581.27	37.58	18.04	1896.50	21.14	5.09	8276.73	28.75	8.07	14401.09
mc_20_100_2	19.09	36.95	1217.33	29.10	39.93	433.84	18.21	34.55	1186.27	24.59	42.14	2245.41
mc_20_100_3	20.60	20.23	1359.58	34.50	30.73	669.70	21.47	22.01	1630.02	32.36	22.66	4556.33
mc_20_100_5	29.93	7.46	2569.03	44.95	21.02	1001.08	28.85	16.30	3793.36	38.22	7.20	8666.08
mc_30_100_2	29.26	112.78	1206.06	56.68	97.44	573.73	29.77	146.91	1609.67	91.07	123.47	758.23
mc_30_100_3	34.69	31.80	2357.16	50.31	65.91	897.20	33.54	31.85	2921.25	44.41	54.78	8270.81
mc_30_100_5	53.94	20.60	5751.53	68.11	15.16	1764.78	52.99	13.88	5326.78	65.43	19.08	11873.38
mc_30_150_2	40.88	13.48	3774.69	49.77	50.85	1394.86	41.38	16.96	4293.05	45.40	11.30	7131.17
mc_30_150_3	37.06	10.49	5941.67	51.33	28.81	1812.91	35.29	19.63	8648.42	47.96	24.11	14413.45
mc_30_150_5	37.43	9.87	9347.75	54.87	29.23	3504.81	37.29	5.73	14296.22	54.74	30.35	14412.98
mc_45_150_2	89.75	101.11	5568.25	100.00	246.04	1945.39	89.90	90.88	8065.89	100.00	172.04	2885.45
mc_45_150_3	71.32	15.87	9515.73	91.94	27.73	3168.52	72.51	7.09	12195.13	92.12	91.03	4796.81
mc_45_150_5	65.02	18.75	14421.22	77.66	18.98	5575.02	64.57	28.71	14404.42	94.94	58.12	9142.50
Average	35.04	24.60	4223.51	49.84	42.29	1528.68	34.75	27.08	5403.54	48.44	39.87	6808.32

10.2.3. Modified Subgradient Algorithm

According to the recovery procedures employed at the final step of the MS algorithm we have totally proposed four approaches: solving the SCLP with CG procedure via D.C. programming (CG with D.C.), solving the SCLP with CG via WPLD (CG with WPLD), solving the UDAP with ℓ_1-norm (ℓ_1-UDAP) and solving the UDAP with ℓ_∞-norm (ℓ_∞-UDAP). The computer code developed by (Boyacı, 2009) is used to solve the PS by D.C. programming. The resulting CM problem is solved by the OA algorithm (Chen *et al.*, 1998) which is coded in "*C#*" by (Boyacı, 2009). We call the code from a *C++* environment and adapt it to work for the MCMWP. In all these approaches we have run the UDAH employing the customer locations as the candidate point set at each step of the MS algorithm as long as $Z_{UDAH}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3) \leq Z_{UB}^{best}$ holds. Whenever $Z_{UDAH}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ exceeds Z_{UB}^{best} at some step of the MS algorithm we resort to one of the four recovery procedures to update the Lagrangean multiplier vectors $\boldsymbol{\varphi}$ and $\boldsymbol{\mu}^3$. Fortunately, in none of the test instances, $Z_{UDAH}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ exceeded Z_{UB}^{best} values during the MS algorithm. Therefore, we did not perform a recovery procedure and we did not update the Lagrangean multipliers until the last step of the MS algorithm.

In Table 10.11, we give a summary of our results obtained with these four recovery procedures employed at the final step of the MS algorithm. The first class (small to large) of the test instances are computationally tested. However, the second class of the test instances are not considered due to their large sizes and computational requirements. The initial upper bound values used for the updates in the MS algorithm are obtained by CL-RMDA1 heuristic. The first column shows the instance names. In the next four blocks of triple columns, we report the results obtained with each approach (i.e., CG with D.C. programming, CG with WPLD, ℓ_1-UDAP and ℓ_∞-UDAP, respectively). These are relative percent deviation of lower and upper values obtained with the corresponding recovery approach and CPU times in seconds, respectively. "LB (%)" is calculated using Equation 10.1 where Z_R is the benchmark upper bound value given in Table A.1 and Z_M is the corresponding lower bound value produced by the method. Similarly, "UB (%)" shows the percent deviation of an upper bound from benchmark lower bound value given in Table A.1. This time heuristic upper bounds replace Z_M and Z_R is the benchmark lower bound value. The computational times in seconds are given under the columns "CPU". The numbers in the last row are simply the column averages.

Table 10.11. The performance of the MS algorithm on the first group of the MCMWP instances.

Instance Name	CG with D.C.			CG with WPLD			ℓ_1-UDAP			ℓ_∞-UDAP		
	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU
mc_2_2_2	100.00	0.00	0.72	100.00	0.00	0.61	100.00	0.00	0.59	100.00	0.00	0.48
mc_2_2_3	2.96	0.00	1.39	2.96	0.00	1.20	43.61	0.00	0.66	2.96	0.00	0.66
mc_2_2_5	51.21	0.00	1.44	51.19	0.00	1.13	100.00	0.00	0.88	51.54	0.00	0.97
mc_2_4_2	43.56	0.10	121.53	43.56	0.10	2.64	54.09	0.10	1.67	47.98	0.10	1.69
mc_2_4_3	9.95	0.10	5.28	9.95	0.10	5.77	23.22	0.10	3.64	18.76	0.10	3.50
mc_2_4_5	8.36	0.10	29.38	7.40	0.10	12.88	16.16	0.10	6.75	12.30	0.10	8.48
mc_3_5_2	25.43	0.10	11.67	25.43	0.10	8.28	30.48	0.10	7.78	27.31	0.10	7.86
mc_3_5_3	68.00	0.10	66.63	68.48	0.10	4.73	72.59	0.10	3.64	72.97	0.10	3.61
mc_3_5_5	53.24	0.10	100.34	53.37	0.10	52.70	54.76	0.10	32.02	58.49	0.10	33.05
mc_3_9_2	17.87	0.10	34.14	17.87	0.10	26.55	26.05	0.10	11.41	32.22	0.10	11.44
mc_3_9_3	21.41	0.10	57.81	21.46	0.10	45.30	33.23	0.10	26.02	25.22	0.10	25.08
mc_3_9_5	9.89	0.90	260.05	9.96	0.90	284.66	25.27	0.90	127.19	32.02	0.90	122.55
mc_4_8_2	54.62	0.10	33.69	59.12	0.10	29.02	55.03	0.10	20.56	66.40	0.10	20.00
mc_4_8_3	33.27	0.10	43.63	33.22	0.10	40.58	42.47	0.10	29.05	38.17	0.10	28.84
mc_4_8_5	33.38	0.10	166.36	33.68	0.10	173.17	41.65	0.10	82.36	40.66	0.10	85.41
mc_4_10_2	35.97	0.87	34.52	37.32	5.61	19.41	59.21	0.87	8.83	46.15	0.87	8.36
mc_4_10_3	20.90	0.44	122.47	20.96	0.44	61.63	39.43	0.44	30.89	29.29	0.44	32.34
mc_4_10_5	39.31	2.26	356.88	39.62	0.10	550.45	48.49	0.10	173.58	38.27	0.10	173.67
mc_4_15_2	23.68	0.10	112.45	23.75	0.10	106.48	38.05	0.10	56.92	34.23	0.10	59.05
mc_4_15_3	22.31	6.89	210.66	22.40	6.89	299.06	31.94	6.89	73.53	35.50	6.89	72.77
mc_4_15_5	21.19	6.33	1727.73	21.27	6.33	7785.28	30.54	6.33	568.02	30.48	6.33	512.45

Table 10.11. The performance of the MS algorithm on the first group of the MCMWP instances cont.

Instance Name	CG with D.C.			CG with WPLD			ℓ_1-UDAP			ℓ_∞-UDAP		
	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU
mc_5_8_2	37.91	0.10	53.53	37.52	0.10	22.83	35.43	0.10	12.30	49.51	0.10	12.63
mc_5_8_3	68.08	10.17	64.56	68.26	10.17	47.09	70.01	10.17	31.98	76.78	10.17	27.23
mc_5_8_5	65.81	10.55	225.92	67.05	8.87	172.05	68.57	8.87	147.30	75.32	8.87	128.98
mc_5_10_2	41.00	0.10	80.70	41.01	0.10	67.39	51.12	0.10	43.08	41.18	0.10	45.06
mc_5_10_3	51.91	0.10	116.30	52.28	0.10	111.77	62.11	0.10	64.23	58.94	0.10	53.89
mc_5_10_5	43.99	10.47	473.11	44.48	9.11	464.92	58.80	9.11	203.38	47.49	9.11	188.09
mc_5_12_2	49.83	3.40	118.41	49.83	3.40	100.06	50.45	3.40	68.08	53.81	3.40	69.39
mc_5_12_3	36.51	0.34	312.98	36.92	0.34	234.67	45.92	0.34	135.08	46.51	0.34	129.55
mc_5_12_5	51.41	11.29	1251.89	51.55	11.29	1448.44	60.14	11.29	329.06	60.62	11.29	299.16
mc_5_15_2	20.55	4.02	980.06	20.92	3.82	130.17	31.55	3.82	120.81	29.42	3.82	109.42
mc_5_15_3	24.51	4.54	410.72	24.63	4.54	414.31	31.18	4.54	298.16	35.35	4.54	270.69
mc_5_15_5	24.85	10.78	1122.14	24.88	10.78	24821.83	37.00	10.78	500.88	36.40	10.78	423.39
mc_5_20_2	20.32	8.83	258.34	20.32	8.83	200.92	45.91	8.83	70.41	34.53	8.83	74.50
mc_5_20_3	14.65	12.63	1021.38	14.66	12.63	1743.77	28.82	12.63	551.44	24.18	12.63	548.14
mc_5_20_5	21.80	9.47	2439.88	22.06	9.74	23224.55	31.46	9.47	1364.42	30.91	9.47	1364.11
mc_5_30_2	10.68	9.17	1916.69	10.74	9.17	2200.19	26.92	8.56	1145.20	18.24	8.56	1106.39
mc_5_30_3	12.46	8.00	5567.83	12.49	8.00	21037.83	25.74	8.00	2882.38	18.72	8.00	2642.70
mc_5_30_5	14.80	10.55	17750.95	14.82	10.55	420383.31	25.11	10.55	9893.78	22.29	10.55	10455.09
mc_6_10_2	37.20	9.51	77.69	35.23	5.08	42.64	56.45	5.08	31.42	41.24	5.08	31.55
mc_6_10_3	63.04	9.32	298.11	63.16	9.32	216.69	71.57	9.32	141.95	67.96	9.32	128.72
mc_6_10_5	58.77	9.59	769.39	59.51	9.59	669.05	66.04	9.59	389.31	62.04	9.59	386.98

Table 10.11. The performance of the MS algorithm on the first group of the MCMWP instances cont.

Instance Name	CG with D.C.			CG with WPLD			ℓ_1-UDAP			ℓ_∞-UDAP		
	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU	LB (%)	UB (%)	CPU
mc_8_10_2	93.05	21.52	59.69	93.60	21.52	35.77	94.96	24.90	21.58	95.55	24.90	22.61
mc_8_10_3	96.99	29.01	227.38	96.39	22.36	95.42	98.43	22.36	68.59	98.20	22.36	56.78
mc_8_10_5	78.35	16.49	1623.38	78.34	14.17	1366.13	80.89	9.63	749.06	75.44	9.63	677.94
mc_10_10_2	100.00	40.42	491.50	99.26	40.42	12.31	100.00	40.42	9.88	100.00	40.42	10.30
mc_10_10_3	96.83	14.34	603.41	95.17	12.53	229.20	99.69	12.53	133.52	97.31	12.53	121.75
mc_10_10_5	98.97	68.12	869.92	99.93	51.16	440.50	99.57	62.87	340.77	100.00	62.87	344.27
mc_10_15_2	89.21	59.43	374.48	89.16	64.95	152.05	93.28	64.95	81.78	92.20	64.95	80.70
mc_10_15_3	63.61	10.12	825.31	65.31	13.56	590.48	80.13	10.12	406.44	70.88	10.12	470.30
mc_10_15_5	67.14	28.63	3756.36	67.34	28.63	4296.28	69.06	29.08	2020.84	71.96	29.08	1817.91
mc_10_20_2	45.35	34.64	1061.02	45.27	36.28	875.86	47.51	36.28	444.70	51.35	36.28	439.58
mc_10_20_3	61.42	31.97	3634.14	61.70	31.97	3328.27	66.56	31.97	2560.66	64.05	31.97	2717.08
mc_10_20_5	52.18	23.20	43241.05	52.45	29.99	64599.55	54.09	23.42	45069.34	54.60	23.42	39895.41
mc_10_24_2	37.59	19.39	2540.86	38.00	19.39	2260.20	45.27	19.39	1335.89	48.51	19.39	1427.25
mc_10_24_3	35.06	30.05	8059.45	35.30	36.53	10550.91	42.95	30.05	5823.92	44.84	30.05	5598.38
mc_10_24_5	38.26	15.74	80214.39	38.41	15.74	126694.14	43.67	15.74	42962.64	42.90	15.74	50524.81
mc_10_30_2	49.52	13.55	3399.83	49.89	9.70	2448.75	57.13	9.70	2231.52	50.84	9.70	2143.66
mc_10_30_3	38.37	16.69	36478.20	38.49	16.69	39045.78	46.97	16.69	26574.39	39.98	16.69	23660.59
mc_10_30_5	34.94	14.54	438928.63	35.08	15.96	445683.14	42.95	15.03	352701.55	36.52	15.03	352403.28
Average	44.06	10.99	11086.64	44.24	10.81	20166.18	53.49	10.61	8387.13	50.13	10.61	8368.67

The acceleration strategies considerably improve the computational requirements of the LR scheme. Most of the MCMWP test instances could not be solved in reasonable CPU times without using heuristic acceleration strategy (namely using UDAH). Furthermore, we observe that when we apply the first acceleration strategy, that is, using $Z_{UDAH}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$ values which are computed with the UDAH, have never exceeded Z_{UB}^{best} value during the MS algorithm. In other words, we only apply a recovery approach in the final step of the MS algorithm to ensure that $Z_{LB}^{best} \leq Z^*$ is satisfied. Moreover, we also observe that UDAH totally requires 72.93% of the overall CPU time of the MS algorithm spent for all test instances. This implies that almost 30% of the CPU time spent by the recovery procedures should be repeated hundreds of times if we use them at each step of the MS algorithm. In other words, comparing these ratios approves the advantage of using heuristic bounds (i.e., $Z_{UDAH}(\boldsymbol{\varphi}, \boldsymbol{\mu}^3)$) within the MS algorithm.

In addition, we observe considerable improvements in the efficiency of the LR scheme when we apply the acceleration strategy with the FSG algorithm. Recall that in the second acceleration strategy we employ two lower bounding schemes which restrict the number of columns generated by the FSG algorithm into the SCLP. We can eliminate up to 86.6% (with an overall average of 28.99%) of the columns before considering their addition to the SCLP, for all test instances by applying the second acceleration strategy. Observe that handling the pricing subproblem by solving the D.C. programming problem is the winner among all approaches and it produces best lower bounds in 48 out of 60 instances. The recovery approaches ℓ_1-UDAP and ℓ_∞-UDAP perform better in terms of upper bounds.

We observe that the percent deviation of the lower bound value of the proposed approaches, namely the CG with D.C., CG with WPLD, ℓ_1-UDAP and ℓ_∞-UDAP, from the benchmark upper bound values are 44.06%, 44.24%, 53.49% and 50.13%, respectively. Furthermore, the best performing ALA heuristic (C-MRR heuristic) upper bounds are worse than the upper bounds produced by our methods with percentages 9.93%, 10.01%, 10.3% and 10.3% from CG with D.C. programming, CG with WPLD, ℓ_1-UDAP and ℓ_∞-UDAP results, respectively. As a final observation, notice that the upper bound values obtained with the LR approaches are closer to the optimal value than the lower bound values they provide. Hence, we can say the proposed approaches can be considered as accurate LR heuristics. Unfortunately, the DA heuristics outperform the MS algorithm in terms of both accuracy and efficiency. On the other hand, MS algorithm can be very well adapted to

many other optimization problems as long as an efficient heuristic solutions can be obtained for the resulting Lagrangean subproblems. Indeed, the convergence of the MS algorithm is guaranteed and a lower bound can be achieved as well as an upper bound.

10.2.4. Confidence Intervals

The confidence intervals (CIs) are calculated as explained in Chapter 7 and listed in Appendix B with Table B.1–8. We provide CIs computed using C-MCALA, randomized MDA using MDAP1 (MDA1 heuristic) and C-MDRR heuristics, respectively. That is to say, for each of the MCALA, MDA1 and MDRR heuristics, the initial facility locations are randomly selected within the convex hull of the customer locations. The C-MCALA and C-MDRR heuristics are initialized with I random facility locations and the MDA1 heuristic started with I random candidate facility locations. We employ commonly used SPEs given by Equation 7.13, 7.16 and 7.19 to calculate Weibull location, scale and shape parameters, respectively. We analyze two sampling schemes: the McRoberts' Approach (MRA) (McRoberts, 1971) and the Los and Lardinois' Approach (LLA) (Los and Lardinois, 1982). In the MRA, intermediate solutions obtained during the run of our randomly initialized heuristics constitute the samples. However, (Los and Lardinois, 1982) claim that the sample generation method of MRA may harm the independence of the samples and they suggest the use of $M \times \overline{M}$ distinct observations. Actually, the requirement that all observations should be distinct is not necessary as discussed in Chapter 7. In the LLA sampling scheme the samples can be constructed of $\overline{M}$ observations which are not necessarily distinct (Wilson *et al.*, 2004).

Using both MRA and LLA sampling approaches, we generate samples from three parent populations which include randomly initialized solutions obtained with the MCALA, MDA1 and MDRR heuristics. In our MRA implementation, we consider samples of size $M = 20$, $M = 30$ and $M = 40$ each consisting of the intermediate feasible solutions of a randomly initialized heuristic (i.e., MCALA, MDA1 or MDRR). In our LLA implementation, again samples of size $M = 20$, $M = 30$ and $M = 40$ are taken, where each of them consists of $\overline{M} = 10$ randomly initialized heuristic solution outputs obtained with one the MCALA, MDA1 or MDRR heuristics.

In order to test the independence of the samples we employ the runs test over the

sample minimums with 95% confidence level. On the instances which pass the independence test, the Kolmogorov-Smirnov (K-S) test with a 95% confidence level is applied to check the hypothesis that the sample minimums fit the Weibull distribution (Beyer, 1974). In case, the sample minimums pass both of these tests, a CI can be determined based on the Fisher-Tippett theorem. Otherwise, in case the sample minimums fail at least one of the tests, we do not report a confidence interval. We should state that the A-D test is also performed for our calculations. However, we were not able to produce a CI for the most of the instances. Therefore, we confine ourselves with only the K-S test to check the fitness of the samples to the Weibull distribution.

In order to validate the CI estimation using EVT, we focus on the MCMWP where three heuristics are employed on the test instances. Once we produce the CIs, their validity can be confirmed by comparing the lower and upper end points of the intervals with the optimal value of a test instance. For that purpose we consider the results obtained on the RMCMWP test instances. Note that the optimum solutions for the RMCMWP instances can be calculated by solving the MDAP1 formulation as discussed. After the validation of our CI estimation using EVT, we employ it in estimating CIs of the objective values of MCMWP (with Euclidean distances) test instances for which we generally do not know optimum values.

The validation of our CI estimation approach using EVT, is first applied on a subset (consisting of 30 instances) of the first class of RMCMWP test instances for which we can obtain optimal solution values. According to our experiments we can say that the CI estimation approach using EVT performs well for the RMCMWP. First, all samples pass the independence test. Second, in only 6 out of $30 \times 3 \times 6 = 540$ samples, a CI is not found due to the failure of the K-S test. Only 3 intervals produce lower bounds larger than the optimum value. The EVT approach employing the Golden-Alt procedure yields covering intervals in 98.33% of the samples (531 out of 540) generated by three heuristics, in total. The results are summarized in Table 10.12 for the RMCMWP. The heuristics used are shown in the first column. The second column stands for the sampling approaches (i.e., MRA and LLA samplings). The sizes of extreme value samples are presented in the third column. The fourth column denotes the mean CI width which is calculated by taking the average of values calculated according to Equation 10.2 over all intervals produced for the corresponding heuristic and sampling method pair. Here, Z_{LB}^{final} and Z_{UB}^{final} indicate the

Table 10.12. Summary of the confidence interval approach on a subset of 30 RMCMWP instances.

Heuristic	Sampling Method	No. Of Samples	Interval Width (%)	Absolute Gap (%)	No. Of Covering Intervals
MCALA	MRA	20	38.40	0.08	30
		30	37.66	0.63	30
		40	39.07	1.46	30
	Average		38.37	0.72	
MCALA	LLA	20	20.18	1.05	30
		30	20.83	2.13	30
		40	21.00	2.68	30
	Average		20.67	1.95	
MDA1	MRA	20	23.07	2.59	30
		30	24.36	5.22	30
		40	24.67	7.06	30
	Average		24.03	4.96	
MDA1	LLA	20	11.23	2.28	29
		30	10.79	3.14	28
		40	11.12	3.32	27
	Average		11.05	2.91	
MDRR	MRA	20	40.04	0.09	30
		30	37.85	0.55	30
		40	39.83	1.43	30
	Average		39.24	0.69	
MDRR	LLA	20	23.32	1.87	29
		30	22.76	2.25	29
		40	22.38	3.08	29
	Average		22.82	2.40	

lower and upper limits of the corresponding CIs, respectively. Z_R denotes the corresponding optimum value of the test instance. The fifth column presents the mean absolute gap in percentages between the lower limit of the CI and the optimum value for the produced CIs. Similar to the interval width, they are calculated by taking the average using Equation 10.1 over all intervals. The average of interval widths and absolute gaps are also presented for each heuristic and sampling method pair. The sixth column includes the number of instances which have passed the K-S test and for which the optimum is covered by the interval.

The results indicate that both the MCALA and MDRR heuristics employing the MRA sampling, outperform the other cases in terms of both the mean absolute gap between the optimum value and the interval lower limit, and the number of covering intervals produced. The MDA1 heuristic yields tighter CIs than other heuristics using each of the sampling methods. For all heuristics, we observe that the MRA sampling produces both smaller

mean absolute gaps between interval lower bounds and the optimum and greater number of covering intervals than the LLA sampling. On the other hand, the LLA sampling yields tighter intervals than the MRA sampling does. What is more, the performance of the heuristics are generally higher for the sample size of $M = 20$, than for the sample size of $M = 30$ or $M = 40$ in terms of all performance measures (i.e., mean interval width, absolute gap and number of covering intervals). Finally, we should note that the CI approach using EVT applied to the RMCMWP instances produces lower bounds within 0.09% of the optimum solution value on average. Hence, we can say that the CI approach using EVT outputs quite good lower bounds on the objective values of the RMCMWP.

Encouraged from the results on RMCMWP instances presented above, we apply the CI approach using EVT in order to obtain acceptable limits on the objective values of the MCMWP (with Euclidean distances) test instances. Although the optimum values of the MCMWP instances are not initially known, CI approaches, which give successful results for the RMCMWP test instances, can still be applied in order to obtain interval estimates for the optimum objective values of the MCMWP. For that purpose, we run each of the C-MCALA, randomized MDA1 and C-MDRR heuristics 20000 times. For the MCMWP, the quality of CIs are evaluated with respect to the benchmark upper bounds (Z_R) provided in Table 10.4.

We only present a summary of these results here. Out of $30 \times 3 \times 6 = 540$ samples formed for the MCMWP instances, 12 samples fail to pass the independence test and 53 samples do not belong to the Weibull distribution due to the failure of the K-S test. Consequently, the Golden-Alt procedure could not produce CIs 12.03% of the samples (65 out of 540) for the MCMWP instances. On the remaining 475 CIs produced by the EVT approach, in 8% of the intervals (38 out of 475), lower limits are larger than the benchmark global minimum value. Since the solutions are not known for most of the instances, we can not exactly say that the optimum is covered by the remaining 92% of the intervals (437 out of 475). However, we believe that the lower bounds are good approximations for the optimum solutions of most of the test instances as in the RMCMWP cases. All the independence test failures of samples have occurred using the MDA1 heuristic with the LLA sampling method. We again observe that the samples fail to pass the independence test for the cases in which the corresponding heuristic generates a relatively less number of local optima.

The results are summarized in Table 10.13. All but the fourth and fifth columns are organized in the same way as in Table 10.12. The fourth column denotes the mean interval width which is calculated by taking the average of Equation 10.2 over all intervals produced for the corresponding heuristic and sampling method pair. Z_R is selected as the corresponding benchmark upper bound value of the test instance. The fifth column presents the mean absolute gaps in percentages between the lower limits of the CIs (Z_M) and the benchmark upper bound values (Z_R) for the produced CIs. Similar to the interval width, they are calculated by taking the average of Equation 10.1 over all intervals. The MDRR heuristic applied with the MRA sampling produces the closest mean lower bounds to the benchmark global minimums. On the other hand, MDA1 heuristic generates the tightest intervals for each sampling method except LLA with $M = 20$. We observe that the MRA sampling generates wider intervals than the LLA does. However, the MRA performs better in terms of the mean absolute gaps between the benchmark upper bounds and the interval lower bounds. Although there are some exceptional instances, the smaller the sample sizes are, the tighter the widths of the CIs and the absolute gaps between the lower bounds and the benchmark minimums are. As a final remark, we should note that the CI approach applied to the MCMWP instances produces lower bounds within 3.54% on average. Thus, we can say that the CI approach using EVT outputs quite reasonable lower bounds on the objective values of the MCMWP.

In the light of our previous computational experiments, we perform some additional calculations on the remaining first class (the instances from mc_10_15_2 to mc_10_30_5) and the second class of the MCMWP instances. In particular, we prefer to use the MRA sampling approach with C-MCALA and C-MDRR heuristics because the performance of the randomized MDA1 is not very well and it requires excessive CPU times. The summary of additional CIs are presented in Table 10.14. They share the same outline with Table 10.12 and Table 10.13. For these additional experiments, we took 1000 random solutions for each MCALA and MDRR heuristics as their parent populations. As can be noticed, on larger instances both MCALA and MDRR heuristics can produce CIs which pass both the independence and Weibull fitness tests. Moreover, the number of intervals covering benchmark upper bound values is very high on these instances (5 out of 180 instances in total). The interval widths between the lower and upper confidence limits are relatively higher on these larger test instances. The performance of the CI approach is quite reasonable and they conform with our earlier calculations.

Table 10.13. Summary of the confidence interval approach on a subset of 30 MCMWP instances.

Heuristic	Sampling Method	No. Of Samples	Interval Width (%)	Absolute Gap (%)	No. Of Covering Intervals
MCALA	MRA	20	26.28	7.11	30
		30	27.54	7.59	30
		40	28.07	7.30	30
	Average		27.30	7.33	
MCALA	LLA	20	15.39	3.54	22
		30	18.01	8.45	18
		40	18.32	8.86	19
	Average		17.24	6.95	
MDA1	MRA	20	25.21	6.84	25
		30	29.37	9.36	24
		40	29.44	11.66	24
	Average		28.01	9.29	
MDA1	LLA	20	16.42	6.51	20
		30	17.03	7.27	19
		40	17.12	7.41	19
	Average		16.86	7.06	
MDRR	MRA	20	30.11	9.85	30
		30	29.77	8.77	29
		40	28.98	6.63	28
	Average		29.62	8.42	
MDRR	LLA	20	24.72	9.86	22
		30	25.08	10.78	22
		40	24.03	9.21	20
	Average		24.61	9.95	

Table 10.14. Summary of the performance of the confidence interval approach on 12 MCMWP instances from mc_10_15_2 to mc_10_30_5.

Heuristic	Sampling Method	No. Of Samples	Interval Width (%)	Absolute Gap (%)	No. Of Covering Intervals
MCALA	MRA	20	38.52	2.13	12
		30	39.60	1.11	12
		40	40.46	1.98	12
	Average		39.53	1.74	
MDRR	MRA	20	38.22	1.50	11
		30	38.24	2.14	11
		40	38.78	1.79	11
	Average		38.41	1.81	

Table 10.15. Summary of the performance of the confidence interval approach on the second group of the MCMWP instances.

Heuristic	Sampling Method	No. Of Samples	Interval Width (%)	Absolute Gap (%)	No. Of Covering Intervals
MCALA	MRA	20	71.42	1.19	18
		30	71.55	1.64	18
		40	71.90	0.43	18
	Average		71.62	1.09	
MDRR	MRA	20	54.16	0.88	18
		30	54.33	1.56	17
		40	54.70	1.45	17
	Average		54.40	1.30	

It should be underlined that the CI approach we employ does not guarantee a valid lower bound on the true optimal values. Lower confidence limits are only estimates of the lower bounds over the optimal value. On the other hand, the upper confidence limits impose absolutely an upper bound on the optimal value since it is already a minimum value of samples which consists of at least one feasible solution for the MCMWP. We should also emphasize that it is possible to obtain different confidence intervals by different samples. The ones we report here are the tightest ones in the sense that lower and upper confidence limits are closest.

10.2.5. Branch-and-Bound Methods

In this section we provide BB based methods which include ABB and LBB algorithms as well as the BS heuristic for both CMWP and MCMWP. To make a complete comparison between the performance of ABB and LBB algorithms, the CMWP and MCMWP are separately addressed.

10.2.5.1. The Capacitated Multi-facility Weber Problem. We make a series of computational tests with the proposed ABB and LBB algorithms. We reimplement the RLT based lower bounding formulation of (Sherali *et al.*, 2002) for the sake of a fair comparison with the proposed BB algorithms. We employ ℓ_∞-norm as the lower bounding block norm within the DAP and in its relaxed version RDAP. We confined ourselves to use the ℓ_∞-norm because the size of the lower bounding MILPs increase rapidly when block norms are employed. Furthermore, the quality of the lower bounds produced by the ℓ_1-norm is worse than of the ℓ_∞-norm. This is also experimentally verified by the DA heuristic performances on the MCMWP. For example, when we employ the weighted ℓ_1-norm we observe that its performance is worse

than the ℓ_∞-norm. We impose a CPU time limit of five hours to report the best feasible solution for the SABB algorithm. On the other hand, the SLBB and SLBBCE algorithms are run for two hours for the CMWP. This CPU time limit is enough to show the superior performance of the LBB algorithms over the SABB algorithm. In addition, it is observed that the improvements obtained on both lower and upper bounds are marginal with the LBB algorithms beyond some predefined time limit. A similar pattern is also observed with the SABB algorithm.

In Table 10.16–18 we give a summary of our experimental results obtained with the SABB algorithm on the CMWP test instances in the first and second classes, respectively. The first three columns present instance names (the number of facilities and customers are given in parenthesis for the first class of standard test instances), the initial upper bound values as described in Chapter 8, and best known upper bound values which are used as benchmark upper bounds. The best known upper bound values are taken as the best outcome of the SABB and SLBB algorithms among the ones run with different lower bounding procedures and branching variable selection strategies. We should emphasize that the best known solutions to the first class of the test instances (standard instances) reported by (Sherali *et al.*, 2002) and our results are slightly different after the decimals. We think that the difference stems from the typographic errors in (Sherali *et al.*, 2002). Another reason for this difference can be the round off errors in our or their calculations especially for the solution of the WPs. In any case, these differences are insignificant and does not affect the accuracy of the percent deviations.

The next three blocks of columns give the results obtained with the lower bounding procedures (i.e., RLT Based Lower Bounding, ℓ_∞-norm Lower Bounding of the DAP and ℓ_∞-norm Lower Bounding of the DAP with LR) which we use together with different branching variable selection strategies or rules. The top row of these lower bounding procedures indicates two additional things: whether Z_{SAS} given by Equation 8.15 is used as an additional lower bounding for tightening the corresponding lower bound or not and the branching variable selection strategy (i.e., BrS1, BrS2 or BrS3). For example, "+ Z_{SAS} with BrS3" indicates that both Z_{SAS} and the suggested lower bound (i.e., RLT, ℓ_∞-DAP or ℓ_∞-RDAP) is employed within the SABB algorithm where the allocation variables are selected using BrS3 given by Equation 8.22. The duality gaps, given under the columns titled as "GAP (%)", are presented in the second columns dedicated to each of three blocks of columns. For each lower bounding procedure, we keep the record of their initial lower bound value calculated

at the root node of the BB tree. Note that the SABB algorithm follows a DFS strategy and only the lower bound computed at the root node gives a valid lower bound on the optimum of the CMWP. As a consequence, the duality gaps are calculated by using Equation 10.2 where Z_R is replaced with the best known upper bound values given in the second column of the tables, Z_{LB}^{final} is replaced with the initial lower bounds produced and Z_{UB}^{final} is the final upper bound produced by the SABB algorithm. Moreover, the accuracy of the SABB algorithm is measured with the percent deviation of the final upper bound value (i.e., Z_M) from the best known solution (i.e., Z_R) in Equation 10.1 under the columns "UB (%)". The corresponding CPU times in minutes are reported under the columns "CPU". Note that a CPU time of less than 300 minutes indicates that ϵ-optimality of the best feasible solution found by the algorithm is proven. Clearly, the lower the percent deviation "UB (%)" and the duality gap "GAP (%)" are, the better the SABB algorithm performance is. The last rows of instance classes gives the average values of the columns as usual.

RLT based lower bounding always applies the straightforward lower bounding Z_{SAS} to improve the lower bounds. (Sherali *et al.*, 2002) indicate that the RLT based lower bounding yields the best performance when the branching variable selection rule BrS2 (Equation 8.17) is employed. Therefore, we implement this lower bounding scheme in order to make a fair comparison with the proposed approaches.

We observe that the branching variable selection rule BrS1 beats BrS2 when BrS1 is used with the ℓ_∞-norm based lower bounding procedure. As a result, we only present the results obtained with the branching variable selection rule BrS1 (Equation 8.16). According to our computational experiments, SABB algorithm using "$+Z_{SAS}+$ BrS3" outperforms both of SABB algorithms using "BrS1" and "$+Z_{SAS}+$ BrS1" in terms of efficiency. In addition, the SABB algorithm results obtained with "$+Z_{SAS}+$ BrS3" are quite promising for the second class of instances. Notice that the block norm based lower bounding approaches employed within the SABB algorithm outperforms the RLT based approaches significantly in terms of both accuracy and efficiency. In particular, the average accuracy of the SABB algorithm with RLT, ℓ_∞-DAP and ℓ_∞-RDAP are 9.61%, 0.13% and 0.05% and their average CPU times are 218.71, 217.60 and 220.09 minutes over all instances, respectively. On the other hand, RLT based lower bounding procedure solves 13 out of 18 of the first class of CMWP instances. this number is 12 and 11 for the SABB algorithm with ℓ_∞-DAP and ℓ_∞-RDAP lower bounding procedures, respectively. Unfortunately, only 2 out of 30 second class of CMWP instances can be solved to ϵ-optimality by the SABB algorithm.

Table 10.16. The performance of the ABB algorithm on the first group of the ECMWP instances.

Instance Name	Init UB	Bench UB	RLT + Z_{SAS}+ BrS2			DAP + BrS1			DAP +Z_{SAS}+ BrS1			DAP +Z_{SAS}+ BrS3		
			UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU
P1 (2,2)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
P2 (2,4)	310.04	247.28	0.00	6.41	0.00	0.00	8.60	0.00	0.00	8.60	0.00	0.00	8.60	0.00
P3 (2,4)	375.58	214.37	0.00	48.89	0.00	0.00	10.43	0.00	0.00	10.43	0.00	0.00	10.43	0.00
P4 (3,5)	24.00	24.00	0.00	71.98	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
P5 (3,5)	462.48	73.96	0.00	89.71	0.00	0.00	18.87	0.00	0.00	18.87	0.00	0.00	18.87	0.00
P6 (3,9)	240.19	221.40	0.00	100.00	0.05	0.00	10.57	0.05	0.00	10.57	0.05	0.00	10.57	0.06
P7 (3,9)	1017.78	871.63	0.00	94.65	0.03	0.00	3.97	0.03	0.00	3.97	0.03	0.00	3.97	0.03
P8 (4,8)	802.49	609.22	0.00	100.00	0.20	0.00	20.72	0.30	0.00	20.72	0.29	0.00	20.72	0.27
P9 (5,15)	11228.85	8169.79	0.00	83.28	300.00	0.00	7.72	300.01	0.00	7.72	300.00	0.00	7.72	300.01
P10 (5,20)	25948.27	12846.13	0.00	80.83	300.00	0.00	9.68	300.04	0.00	9.68	300.03	0.00	9.68	300.01
P11 (5,20)	1790.18	1107.18	0.00	100.00	128.75	0.00	8.60	9.30	0.00	8.60	9.10	0.00	8.60	9.02
P12 (5,30)	31785.26	23990.02	0.00	100.00	300.00	2.49	13.09	301.24	2.49	13.09	300.69	0.88	11.48	300.19
P15 (5,10)	5893.38	2595.47	0.00	93.94	0.19	0.00	12.73	1.32	0.00	12.73	1.10	0.00	12.73	0.90
P16 (6,10)	8837.29	7797.20	0.00	100.00	8.18	0.00	9.02	71.00	0.00	9.02	69.67	0.00	9.02	67.62
P17 (7,10)	11549.59	6967.89	0.00	100.00	34.73	0.00	7.49	300.00	0.00	7.49	300.00	0.00	7.49	300.00
P18 (8,10)	3800.75	1564.45	0.00	100.00	300.00	0.00	7.32	46.39	0.00	7.32	47.42	0.00	7.32	35.00
P19 (9,10)	3911.34	3250.68	0.00	100.00	300.00	0.00	9.50	300.00	0.00	9.50	300.00	0.00	9.50	300.00
P20 (10,10)	8923.45	7719.79	0.00	74.08	300.00	0.00	15.75	300.00	0.00	15.75	300.00	0.00	15.75	300.00
Average	6494.50	4348.36	0.00	80.21	109.56	0.14	9.67	107.21	0.14	9.67	107.13	0.05	9.58	106.28

Table 10.16. The performance of the ABB algorithm on the first group of the ECMWP instances cont.

Instance Name	Init UB	Bench UB	RDAP (br1)			RDAP $+Z_{SAS}+$ BrS1			RDAP $+Z_{SAS}+$ BrS3		
			UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB(%) (%)	GAP (%)	CPU
P1 (2,2)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
P2 (2,4)	310.04	247.28	0.00	8.60	0.01	0.00	8.60	0.01	0.00	8.60	0.00
P3 (2,4)	375.58	214.37	0.00	10.43	0.02	0.00	10.43	0.01	0.00	10.43	0.01
P4 (3,5)	24.00	24.00	0.00	35.10	0.02	0.00	35.10	0.01	0.00	35.10	0.01
P5 (3,5)	462.48	73.96	0.00	18.87	0.02	0.00	18.87	0.02	0.00	18.87	0.01
P6 (3,9)	240.19	221.40	0.00	12.30	0.12	0.00	12.30	0.08	0.00	12.30	0.07
P7 (3,9)	1017.78	871.63	0.00	3.97	0.14	0.00	3.97	0.09	0.00	3.97	0.08
P8 (4,8)	802.49	609.22	0.00	24.68	1.06	0.00	24.68	0.74	0.00	24.68	0.72
P9 (5,15)	11228.85	8169.79	0.00	8.02	300.00	0.00	8.02	300.00	0.00	8.02	300.00
P10 (5,20)	25948.27	12846.13	0.00	9.96	300.00	0.00	9.96	300.00	0.00	9.96	300.00
P11 (5,20)	1790.18	1107.18	0.00	13.77	3.88	0.00	13.77	2.88	0.00	13.77	2.64
P12 (5,30)	31785.26	23990.02	0.00	14.08	300.00	0.00	14.08	300.01	0.00	14.08	300.01
P15 (5,10)	5893.38	2595.47	0.00	15.38	5.92	0.00	15.38	3.06	0.00	15.38	2.41
P16 (6,10)	8837.29	7797.20	0.00	14.49	93.43	0.00	14.49	59.03	0.00	14.49	56.50
P17 (7,10)	11549.59	6967.89	0.00	12.80	300.00	0.00	12.80	300.00	0.00	12.80	300.00
P18 (8,10)	3800.75	1564.45	0.00	21.91	300.00	0.00	21.91	300.00	0.00	21.91	300.00
P19 (9,10)	3911.34	3250.68	0.07	23.90	300.00	0.19	24.02	300.00	0.00	23.82	300.00
P20 (10,10)	8923.45	7719.79	2.09	17.98	300.00	2.78	18.67	300.00	0.00	15.89	300.00
Average	6494.50	4348.36	0.12	14.79	122.48	0.17	14.84	120.33	0.00	14.67	120.14

Table 10.17. The performance of the ABB algorithm on the second group of the ECMWP instances with unit transportation costs.

Instance Name	Init UB	Bench UB	RLT $+Z_{SAS}$+ BrS2			DAP + BrS1			DAP $+Z_{SAS}$+ BrS1			DAP $+Z_{SAS}$+ BrS3		
			UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU
hg_4_8	153.69	133.57	0.00	100.00	2.38	0.00	4.92	0.51	0.00	4.92	0.52	0.00	4.92	0.40
hg_4_24	1183.01	966.91	0.00	100.00	300.00	0.00	12.61	300.01	0.00	12.61	300.03	0.00	12.61	300.35
hg_5_20	7170.30	6006.64	1.88	101.88	300.00	0.00	11.10	300.06	0.00	11.10	300.05	0.00	11.10	300.79
hg_5_25	8082.28	6321.19	0.38	100.38	300.00	0.00	9.87	316.85	0.00	9.87	311.31	0.38	10.25	312.65
hg_5_30	12518.70	11478.25	0.00	100.00	300.00	3.04	16.32	316.28	3.04	16.32	315.93	3.04	16.32	416.56
hg_5_40	17492.34	17070.09	0.98	100.98	300.00	0.00	10.62	359.87	0.00	10.62	338.44	0.00	10.62	338.49
hg_6_20	6251.37	4665.63	0.00	100.00	300.00	0.00	8.80	300.07	0.00	8.80	300.30	0.00	8.80	300.38
hg_6_25	5185.19	4958.71	0.00	100.00	300.00	0.00	7.58	310.10	0.00	7.58	311.16	0.00	7.58	318.53
hg_6_30	10817.14	9564.39	10.83	110.83	300.00	0.50	12.04	300.17	0.50	12.04	300.15	0.50	12.04	318.61
hg_7_20	4555.39	3562.43	8.61	108.61	300.00	0.00	7.09	300.59	0.00	7.09	300.06	0.00	7.09	300.44
hg_7_25	4766.11	4021.82	11.97	111.97	300.00	0.00	10.04	301.21	0.00	10.04	301.12	0.08	10.12	308.30
hg_8_20	4093.08	2803.62	14.96	114.96	300.00	0.00	8.26	300.25	0.00	8.26	300.36	0.00	8.26	305.30
hg_8_25	3783.79	3235.83	16.93	116.93	300.00	0.00	7.07	303.83	0.00	7.07	340.43	0.00	7.07	303.25
hg_9_20	3751.37	2381.30	28.75	128.75	300.00	0.00	8.58	300.16	0.00	8.58	300.18	0.00	8.58	300.18
hg_10_20	3241.95	1870.99	27.39	127.39	300.00	0.00	8.60	301.02	0.00	8.60	300.48	0.00	8.60	300.28
Average	6203.05	5269.42	8.18	108.18	280.16	0.24	9.57	287.40	0.24	9.57	288.03	0.27	9.60	294.97

Table 10.17. The performance of the ABB algorithm on the second group of the ECMWP instances with unit transportation costs cont.

Instance Name	Init UB	Bench UB	RDAP + BrS1			RDAP $+Z_{SAS}+$ BrS1			RDAP $+Z_{SAS}+$ BrS3		
			UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU
hg_4_8	153.69	133.57	0.00	25.19	2.20	0.00	25.19	1.97	0.00	25.19	1.48
hg_4_24	1183.01	966.91	0.00	13.13	300.00	0.00	13.13	300.00	0.00	13.13	300.00
hg_5_20	7170.30	6006.64	0.00	11.55	300.00	0.00	11.55	300.01	0.00	11.55	300.00
hg_5_25	8082.28	6321.19	0.00	11.57	300.01	0.00	11.57	300.00	0.00	11.57	300.01
hg_5_30	12518.70	11478.25	0.00	15.14	300.00	0.00	15.14	300.00	0.00	15.14	300.01
hg_5_40	17492.34	17070.09	0.00	11.94	300.02	0.00	11.94	300.00	0.00	11.94	300.02
hg_6_20	6251.37	4665.63	0.00	10.46	300.00	0.00	10.46	300.01	0.00	10.46	300.00
hg_6_25	5185.19	4958.71	0.00	13.04	300.00	0.00	13.04	300.01	0.00	13.04	300.00
hg_6_30	10817.14	9564.39	0.00	13.61	300.00	0.00	13.61	300.02	0.00	13.61	300.00
hg_7_20	4555.39	3562.43	0.00	9.39	300.00	0.00	9.39	300.00	0.00	9.39	300.00
hg_7_25	4766.11	4021.82	0.00	13.61	300.00	0.00	13.61	300.01	0.00	13.61	300.00
hg_8_20	4093.08	2803.62	0.00	11.56	300.00	0.00	11.56	300.01	0.00	11.56	300.01
hg_8_25	3783.79	3235.83	0.00	13.68	300.00	0.00	13.68	300.01	0.00	13.68	300.01
hg_9_20	3751.37	2381.30	0.00	16.45	300.00	0.00	16.45	300.01	0.00	16.45	300.01
hg_10_20	3241.95	1870.99	0.00	13.57	300.00	0.00	13.57	300.01	0.00	13.57	300.00
Average	6203.05	5269.42	0.00	13.59	280.15	0.00	13.59	280.14	0.00	13.59	280.10

Table 10.18. The performance of the ABB algorithm on the second group of the ECMWP instances with non-unit transportation costs.

Instance Name	Init UB	Bench UB	RLT $+Z_{SAS}$+ BrS2			DAP $+Z_{SAS}$+ BrS1			DAP $+Z_{SAS}$+ BrS1			DAP $+Z_{SAS}$+ BrS3		
			UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU
ht_4_8	824.47	570.25	0.00	100.00	0.60	0.00	2.85	0.16	0.00	2.85	0.16	0.00	2.85	0.11
ht_4_24	5385.54	4232.24	0.00	100.00	300.00	0.00	14.49	300.07	0.00	14.49	300.04	0.00	14.49	300.05
ht_5_20	33429.18	22146.54	0.00	100.00	300.00	0.00	11.16	300.02	0.00	11.16	300.03	0.00	11.16	300.33
ht_5_25	60379.77	38236.43	13.50	113.50	300.00	0.00	9.44	300.90	0.00	9.44	300.13	0.00	9.44	300.22
ht_5_30	61415.86	42899.92	7.41	107.41	300.00	0.00	8.33	300.09	0.00	8.33	302.15	0.00	8.33	301.48
ht_5_40	132502.37	104983.19	12.51	112.51	300.00	0.00	7.77	329.00	0.00	7.77	301.57	0.00	7.77	305.26
ht_6_20	20803.08	14693.26	7.11	107.11	300.00	0.00	12.63	300.30	0.00	12.63	300.15	0.00	12.63	300.14
ht_6_25	32061.22	19036.28	6.48	106.48	300.00	0.00	7.57	300.33	0.00	7.57	300.38	0.00	7.57	300.07
ht_6_30	65800.01	59665.36	4.34	104.34	300.00	0.62	10.46	307.19	0.62	10.46	302.28	0.62	10.46	308.95
ht_7_20	32593.24	15075.27	17.35	117.35	300.00	0.00	10.37	300.06	0.00	10.37	300.08	0.00	10.37	300.26
ht_7_25	26557.91	16086.30	52.65	152.65	300.00	0.00	13.39	300.72	0.00	13.39	300.02	0.00	13.39	300.14
ht_8_20	18856.13	6499.80	44.41	144.41	300.00	0.00	7.49	300.08	0.00	7.49	300.12	0.00	7.49	300.34
ht_8_25	24698.60	9486.23	51.53	151.53	300.00	1.14	8.76	300.35	1.14	8.76	302.28	3.46	11.07	300.90
ht_9_20	15367.09	6551.37	60.69	160.69	300.00	0.00	8.28	300.06	0.00	8.28	301.70	0.00	8.28	301.61
ht_10_20	10776.51	4702.16	60.44	160.44	300.00	0.00	14.27	300.52	0.00	14.27	300.21	0.00	14.27	300.69
Average	36096.73	24324.31	22.56	122.56	280.04	0.12	9.82	282.66	0.12	9.82	280.75	0.27	9.97	281.37

Table 10.18. The performance of the ABB algorithm on the second group of the ECMWP instances with non-unit transportation costs cont.

Instance Name	Init UB	Bench UB	RDAP + BrS1			RDAP $+Z_{SAS}$+ BrS1			RDAP $+Z_{SAS}$+ BrS3		
			UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU	UB (%)	GAP (%)	CPU
ht_4_8	824.47	570.25	0.00	15.46	0.10	0.00	15.46	0.11	0.00	15.46	0.09
ht_4_24	5385.54	4232.24	0.00	15.97	300.00	0.00	15.97	300.00	0.00	15.97	300.00
ht_5_20	33429.18	22146.54	0.00	13.57	300.00	0.00	13.57	300.00	0.00	13.57	300.00
ht_5_25	60379.77	38236.43	0.00	11.14	300.00	0.00	11.14	300.00	0.00	11.14	300.00
ht_5_30	61415.86	42899.92	0.00	12.80	300.01	0.00	12.80	300.00	0.00	12.80	300.01
ht_5_40	132502.37	104983.19	0.00	8.52	300.03	0.00	8.52	300.03	0.00	8.52	300.02
ht_6_20	20803.08	14693.26	0.00	16.04	300.00	0.00	16.04	300.00	0.00	16.04	300.00
ht_6_25	32061.22	19036.28	0.00	10.56	300.01	0.00	10.56	300.00	0.00	10.56	300.00
ht_6_30	65800.01	59665.36	0.00	14.42	300.01	0.00	14.42	300.02	0.00	14.42	300.00
ht_7_20	32593.24	15075.27	0.00	13.73	300.00	0.00	13.73	300.00	0.00	13.73	300.00
ht_7_25	26557.91	16086.30	0.00	13.98	300.00	0.00	13.98	300.00	0.00	13.98	300.00
ht_8_20	18856.13	6499.80	0.00	24.09	300.00	0.00	24.09	300.01	0.00	24.09	300.01
ht_8_25	24698.60	9486.23	3.46	20.98	300.01	3.46	20.98	300.00	2.44	19.97	300.01
ht_9_20	15367.09	6551.37	0.00	26.98	300.01	0.00	26.98	300.00	0.00	26.98	300.00
ht_10_20	10776.51	4702.16	0.00	32.91	300.00	0.00	32.91	300.01	0.00	32.91	300.00
Average	36096.73	24324.31	0.23	16.74	280.01	0.23	16.74	280.01	0.16	16.68	280.01

Our computational results justify that the performance of the exact solution of the DAP formulation reduces significantly as the instance size increases. It is also possible to assess the quality of the lower bounding procedures (i.e., RLT, ℓ_∞-DAP, ℓ_∞-RDAP). This can be observed from the percent deviation of the gap between the initial lower bound value and the best known upper bound value. Clearly, RLT produces the weakest lower bounds and the worst duality gaps. Meanwhile, ℓ_∞-DAP performs is the winner. It is followed by the ℓ_∞-RDAP. On the other hand, although the RLT based approach is quite efficient, its performance deteriorates significantly on the second class of instances (i.e., large instances). In this case, employing a LR procedure, which produces weaker lower bounds than its exact solution in reasonable CPU times, within the ABB algorithm can be a better choice considering the trade-off between the efficiency and accuracy. We see that the use of Z_{SAS} slightly improves the efficiency of the ABB algorithm. We can say that the performance of the algorithms on both their accuracy and efficiency indicates that unit cost instances are generally more difficult to solve than their non-unit cost counterparts.

For the LBB algorithm experiments, we first examine the lower bound Z_{LP}^{SLBB} given by Equation 9.2. Then, we test the performance of the Z_{DAP}^{SLBB} by using a hybrid lower bounding approach. In this hybrid lower bounding approach, both Z_{LP}^{SLBB} and Z_{DAP}^{SLBB} are calculated and their maximum is taken as the lower bound value for a facility-rectangle combination until the location space is partitioned into several rectangles. Since the calculation of Z_{DAP}^{SLBB} is computationally more expensive, the lower bounding procedure is performed using only Z_{LP}^{SLBB} afterwards. On *small* instances we prefer to use the exact solution of the MILP with ℓ_∞-norm approximation for Z_{DAP}^{SLBB}. The LR of the MILP is solved on *medium* and *large* instances to determine Z_{DAP}^{SLBB}. There are two reasons to use the hybrid lower bounding approach as mentioned. One of them is the observation that block norm based lower bounds are initially very tight on the LBB algorithm. As a result, the number of subproblems can be reduced significantly without further exploration at the initial nodes of the BB tree. The other reason is that once the location space is partitioned into a sufficient number of rectangles, the Z_{LP}^{SLBB} lower bounds become tighter than the Z_{DAP}^{SLBB}. Therefore, we use both Z_{LP}^{SLBB} and Z_{DAP}^{SLBB} until the number of rectangles reaches $3 \times J$ and then we switch to use only Z_{LP}^{SLBB} as the hybrid lower bounding approach within the SLBB and SLBBCE algorithms.

A comparison of the SLBB and SLBBCE algorithms is presented in Table 10.19–

21 which have almost the same outline as the tables given for the SABB algorithm. To compare the performance of the SLBB algorithm we impose a two hours of run time limit. We separate two blocks of columns for both the SLBB and SLBBCE algorithms. When we compare SLBB and SLBBCE algorithms, it can be observed that the SLBB outperforms the SLBBCE algorithm in terms of average duality gaps between their final upper and lower bounds. The average accuracy of both algorithms are almost the same with a slight superiority of the SLBBCE algorithm. On the other hand, SLBBCE algorithm is more efficient than the SLBB algorithm on the average.

Notice that the SABB algorithm with ℓ_∞-RDAP lower bounding beats the SLBB and SLBBCE algorithms using only Z_{LP}^{SLBB} as the lower bounding procedure. In particular, the duality gaps are quite weak when only Z_{LP}^{SLBB} is employed for the LBB algorithms (i.e., SLBB and SLBBCE algorithms). However, when the hybrid lower bounding approach (i.e., $Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$) is used, it produces better results in terms of accuracy with a slight decrease in its efficiency. Both of the SLBB and SLBBCE algorithms can solve 10 out of 18 instances for the first class of the CMWP instances. On the other hand, only 2 out of 30 instances can be solved by the LBB algorithms on the second class of CMWP instances. Both of the SLBB and SLBBCE algorithms using only Z_{LP}^{SLBB} and $Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$ lower bounding approaches yield the same number of ϵ-optimum solutions for the test instances.

The LBB algorithms with hybrid lower bounding approach produces the best duality gap percentages and accuracy on both the first (standard) and second class of CMWP test instances while their efficiency is also better than the ones of the SABB algorithm. We should note that even if the running time of the SLBB algorithm is less than the SABB algorithms, SLBB algorithm yields at least the same accuracy with the ABB algorithms. Consequently, we can conclude that SLBB algorithm with the hybrid lower bounding approach is the winner in terms of both accuracy and efficiency. The results obtained with the LBB algorithms also confirm that the homogenous subgroup of the test instances are more difficult to solve than the non-unit subgroup in terms of average duality gap percentages "GAP (%)", average accuracy "UB (%)" values and average CPU times (in minutes).

It can be noticed that the second class consists of 30 CMWP test instances. The remaining 64 instances are large or very large instances. Clearly, neither the ABB algorithms nor the LBB algorithms are applicable for those instances. As a remedy, we have applied the BS heuristic on these instances. Another approach may be to resort to DA heuristics for

Table 10.19. The performance of the LBB algorithms on the standard CMWP instances.

Instance Name	SLBB						SLBBCE					
	Z_{LP}^{SLBB}			$Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$			Z_{LP}^{SLBB}			$Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$		
	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
P1 (2,2)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.01
P2 (2,4)	0.00	0.31	8.36	0.00	0.11	1.03	0.00	0.11	0.09	0.00	0.11	0.07
P3 (2,4)	0.00	0.10	0.23	0.00	0.10	0.22	0.00	0.10	0.21	0.00	0.10	0.16
P4 (3,5)	0.00	0.10	0.13	0.00	0.00	0.10	0.00	0.13	0.04	0.00	0.00	0.05
P5 (3,5)	0.00	0.10	0.03	0.00	0.10	0.03	0.00	0.11	0.02	0.00	0.11	0.02
P6 (3,9)	0.00	0.10	0.23	0.00	0.10	0.27	0.00	0.13	0.18	0.00	0.13	0.23
P7 (3,9)	0.00	0.10	1.03	0.00	0.10	0.78	0.00	0.12	0.41	0.00	0.12	0.14
P8 (4,8)	0.00	0.10	36.22	0.00	0.10	38.32	0.00	0.10	18.55	0.00	0.10	13.50
P9 (5,15)	0.00	35.81	120.00	0.00	7.67	120.00	0.00	39.76	10.55	0.00	7.85	8.02
P10 (5,20)	0.00	29.88	120.00	0.01	9.68	120.00	0.00	36.71	120.00	0.00	9.68	120.00
P11 (5,20)	0.00	0.10	120.00	0.00	0.10	120.00	0.00	0.10	120.00	0.00	0.10	120.00
P12 (5,30)	0.00	29.96	120.00	0.00	10.18	120.00	0.00	64.68	50.52	0.00	10.68	35.96
P15 (5,10)	0.00	0.10	120.00	0.00	0.10	120.00	0.00	0.11	120.00	0.00	0.11	120.00
P16 (6,10)	0.00	53.99	120.00	0.00	10.03	120.00	0.00	73.30	120.01	0.00	9.59	120.00
P17 (7,10)	0.00	76.41	120.00	0.48	9.62	120.00	0.00	86.52	120.00	0.00	9.22	120.00
P18 (8,10)	0.00	49.63	120.00	0.00	8.10	120.00	0.00	80.12	120.00	0.00	8.10	120.00
P19 (9,10)	0.00	72.18	120.00	0.40	9.89	120.00	0.00	88.55	120.00	0.00	9.50	120.00
P20 (10,10)	0.00	74.14	120.00	0.00	15.75	120.00	0.00	84.64	120.01	0.00	15.75	120.03
Average	0.00	23.51	69.23	0.05	4.54	68.93	0.00	30.85	57.81	0.00	4.51	56.57

Table 10.20. The performance of the LBB algorithms on the second group of the ECMWP instances with unit transportation costs.

Instance Name	SLBB						SLBBCE					
	Z_{LP}^{SLBB}			$Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$			Z_{LP}^{SLBB}			$Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$		
	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
hg_4_8	0.00	0.10	8.46	0.00	0.10	7.13	0.00	0.10	11.18	0.00	0.10	6.72
hg_4_24	0.00	17.14	120.00	0.00	11.73	120.00	0.00	41.76	120.00	0.00	12.72	120.00
hg_5_20	0.00	36.06	120.00	0.00	10.58	120.00	0.00	71.19	120.00	0.00	11.10	120.00
hg_5_25	0.00	25.75	120.00	0.00	9.94	120.00	0.00	55.47	120.00	0.00	10.33	120.00
hg_5_30	0.00	49.77	120.00	0.00	13.30	120.00	0.00	77.06	120.00	0.00	14.12	120.03
hg_5_40	0.00	44.88	120.00	0.00	11.08	120.00	0.00	73.07	120.00	0.00	11.98	120.02
hg_6_20	0.00	58.71	120.00	0.00	8.76	120.00	0.00	93.47	120.00	0.00	8.88	120.00
hg_6_25	0.00	49.26	120.00	0.00	8.80	120.00	0.00	89.52	120.00	0.00	8.97	120.00
hg_6_30	0.00	64.72	120.00	0.00	11.78	120.00	0.00	95.15	120.00	0.00	12.13	120.01
hg_7_20	0.00	78.90	120.00	0.00	7.12	120.00	0.00	98.48	120.00	0.00	7.12	120.00
hg_7_25	0.00	67.48	120.00	0.00	10.40	120.00	0.00	99.13	120.00	0.00	10.81	120.01
hg_8_20	0.00	87.79	120.00	0.00	9.04	120.00	0.00	99.86	120.00	0.00	9.04	120.01
hg_8_25	0.00	78.61	120.00	0.00	7.10	120.00	0.00	99.10	120.01	0.00	7.25	120.04
hg_9_20	2.09	96.58	120.00	0.00	8.59	120.00	0.00	100.00	120.00	0.00	10.91	120.10
hg_10_20	10.39	107.69	120.00	0.00	8.74	120.00	1.34	101.26	120.01	0.00	10.76	120.05
Average	0.83	57.56	112.56	0.00	9.14	112.48	0.09	79.64	112.75	0.00	9.75	112.47

Table 10.21. The performance of the LBB algorithms on the second group of the ECMWP instances with non-unit transportation costs.

Instance Name	SLBB						SLBBCE					
	Z_{LP}^{SLBB}			$Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$			Z_{LP}^{SLBB}			$Z_{LP}^{SLBB} + Z_{DAP}^{SLBB}$		
	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
ht_4_8	0.00	0.10	4.29	0.00	0.10	4.15	0.00	0.10	3.20	0.00	0.10	2.28
ht_4_24	0.00	10.65	120.00	0.00	8.71	120.00	0.00	16.44	120.00	0.00	10.87	120.00
ht_5_20	0.00	20.25	120.00	0.00	7.19	120.00	0.00	50.21	120.00	0.00	1.91	120.00
ht_5_25	0.00	21.09	120.00	0.00	8.49	120.00	0.00	53.81	120.00	0.00	9.55	120.00
ht_5_30	0.00	30.47	120.00	0.00	7.73	120.01	0.00	71.64	120.00	0.00	8.35	120.00
ht_5_40	0.00	42.28	120.00	0.00	7.75	120.00	0.00	78.05	120.00	0.00	7.79	120.01
ht_6_20	0.00	52.00	120.00	0.00	9.58	120.00	0.00	91.23	120.00	0.00	12.65	120.00
ht_6_25	0.00	33.76	120.00	0.00	4.18	120.00	0.00	80.80	120.00	0.00	3.54	120.00
ht_6_30	0.00	61.88	120.00	0.00	10.23	120.00	0.00	96.90	120.01	0.00	10.52	120.00
ht_7_20	0.00	79.74	120.00	0.00	11.38	120.00	0.00	99.85	120.01	0.00	11.44	120.00
ht_7_25	0.00	75.71	120.00	0.00	12.65	120.00	0.00	99.81	120.00	0.00	13.56	120.01
ht_8_20	0.00	94.95	120.00	0.00	9.22	120.00	0.00	100.00	120.00	0.00	10.92	120.05
ht_8_25	0.00	97.42	120.01	0.00	7.63	120.00	0.00	100.00	120.01	0.00	7.63	120.01
ht_9_20	8.49	104.26	120.01	0.00	9.40	120.00	2.83	102.83	120.02	0.00	13.54	120.02
ht_10_20	0.00	100.00	120.00	0.00	15.36	120.00	17.50	117.50	120.00	2.51	32.22	120.00
Average	0.57	54.97	112.29	0.00	8.64	112.28	1.36	77.28	112.22	0.17	10.31	112.16

accurate solutions. However, the solution of the resulting MILPs is not possible especially on large instances. In fact, LR approach applied to these MILPs may not always yield quite accurate solutions. Briefly, we can say that the BS heuristic performs very well in particular for large instances.

Table 10.22–24 summarize our results with the BS heuristic. As an additional stopping condition, we impose a restriction for the minimum area limit of regions considered with the BS heuristic. The area of the rectangles, which is to be partitioned, are limited to be larger than 0.001 with the purpose to accelerate the BS heuristic and make sure that the partitioning ends in a finite number of steps. We use the hybrid lower bounding procedure described for the LBB algorithms. That is to say, we use both LP based and block norm based lower bounding procedures for the BS heuristic. It is observed that using only the LP based lower bounds does not have promising performance. The same settings with the LBB algorithms are preserved for the BS heuristic. The hybrid bounding continues with the LP and block norm based lower bounding procedures until there are $J \times K$ rectangles for the LBB algorithm and then we switch to use only LP based lower bounding procedure. The first columns state the instance names. The second column presents the benchmark upper bound used as the reference value Z_R for the percent deviation of upper bounds. We run DA heuristics with a discretization over customer locations by solving DAP formulation with zero lower bounds $\widehat{l}_{ij} = 0$ and $\widehat{u}_{ij} = \min\{s_i, q_j\}$. Namely, the CL-DA heuristic with exact solution of DAP formulation over the customer locations. Note that MCALA is replaced with a CALA as a final step to perform improvements for the CMWP. We impose a four-hour time limit for the solution of the DAP formulation. However, in some very large instances either it is not possible to obtain a solution or it takes up to almost nine hours to produce a feasible solution. We also resort to the LR of DAP with a discretization over customers (CL-RDA heuristic). Then, we select the best solution produced by these heuristics and the BS heuristic as the benchmark upper bounds. The rest of the columns are dedicated to various combinations of beam width W and evaluation function parameter Ψ which are indicated in the second rows within parenthesis as (Ψ,W). We report upper bound percent deviations from the benchmark upper bound and CPU times in minutes. Beam width W is set to 1 and 3, and the evaluation function parameter Ψ is set to 0, 0.25 and 0.5 in the experiments. In addition, $\Psi = 0.75$ and $\Psi = 1.0$ is also tested within the experiments. However, their outcome is not as satisfactory as the ones which are reported here. Additionally, a beam width of $W > 3$ requires excessive CPU times. As a result, we prefer to use this setting

for (Ψ,W). The combination of (Ψ,W) affects the performance of the BS heuristic on the CMWP. We observe that beam width of $W = 3$ yields more accurate solutions with $\Psi = 0.25$ for both the first and second class of the test instances.

For the first class of the CMWP instances, BS heuristic is neither more efficient nor more accurate than the DA heuristics (i.e., CL-DA and CL-RDA heuristics). On the other hand, BS yields more accurate solutions on the large CMWP instances (instances in the second class) than the CL-RDA heuristic does. CL-DA heuristic yields the most accurate solutions. However, for large instances, BS can find good feasible solutions where CL-DA can only produce solutions with drastic CPU times. Actually, BS heuristic lies in between the DA heuristics CL-DA and CL-RDA in terms of both efficiency and accuracy. BS requires more CPU time than the CL-RDA heuristic does. To sum up, on large instances BS is a promising alternative when the exact solution of the MILPs are not possible for CL-DA with the CMWP.

10.2.5.2. The Multi-commodity Capacitated Multi-facility Weber Problem. In Table 10.25 we present the results on the first class of MCMWP instances obtained with the MABB algorithm which employs RLT based lower bounding procedure accompanied with pure location based lower bounding Z_{MSAS} given by Equation 8.61 and three MBrSs (i.e., MBrS1, MBrS2 and MBrS3 defined by Equation 8.70, 8.71 and 8.72 respectively). The first column stands for the instance names. The second column provides the initial upper bound values. The calculation of the initial upper bounds are described in Chapter 8 while the lower bounds are determined by the RLT based lower bounding procedure for the MCMWP. Note that initially Z_{MSAS} is equal to zero since $l_{ijk} = 0$ for all allocation variables. The next three blocks of columns are dedicated to the results obtained with each of the three MBrSs (i.e., MBrS1, MBrS2 and MBrS3). Total computational times in minutes are given under the columns "CPU". A CPU time limit of four hours is imposed to run the MABB algorithm. "UB (%)" denotes the percent deviation of the final upper bound Z_{UB}^{final} (Z_M) from the benchmark lower bound value (namely, the reference value Z_R) given in Table A.1. They are calculated by the Equation 10.1. "GAP (%)" stands for the duality gap between the lower and upper bounds determined by using Equation 10.2 where Z_{UB}^{Final} and Z_{LB}^{Final} are the final upper and lower bounds of the corresponding BB algorithm, respectively. Z^R is the benchmark lower bound value given in Table A.1 and Table A.2. Finally, the last rows give the overall average of the test instances.

Table 10.22. The performance of the BS heuristic on the standard CMWP instances.

Instance Name	Bench UB	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
		(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
P1 (2,2)	0.00	0.00	0.01	0.00	0.01	0.00	0.01	0.00	0.01	0.00	0.01	0.00	0.01
P2 (2,4)	247.28	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02
P3 (2,4)	214.37	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02
P4 (3,5)	24.00	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.03	0.00	0.02	0.00	0.03
P5 (3,5)	73.96	0.00	0.03	0.00	0.03	0.00	0.02	0.00	0.03	0.00	0.02	0.00	0.03
P6 (3,9)	221.40	0.00	0.03	0.00	0.03	0.00	0.03	0.00	0.03	0.00	0.03	0.00	0.03
P7 (3,9)	871.63	0.00	0.03	0.00	0.03	0.00	0.03	0.00	0.03	0.00	0.03	0.00	0.03
P8 (4,8)	609.22	0.00	0.07	0.00	0.11	0.00	0.07	0.00	0.10	0.00	0.08	0.00	0.08
P9 (5,15)	8169.79	0.00	0.08	0.00	0.10	0.00	0.08	0.00	0.10	0.00	0.08	0.00	0.11
P10 (5,20)	12846.13	0.28	0.11	0.28	0.12	0.28	0.10	0.10	0.12	0.01	0.10	0.10	0.12
P11 (5,20)	1107.18	0.00	0.26	0.00	0.28	0.00	0.26	0.00	0.27	0.00	0.26	0.00	0.26
P12 (5,30)	23990.02	0.48	0.18	0.48	0.24	0.48	0.15	0.48	0.22	0.48	0.16	0.48	0.19
P15 (5,10)	2595.47	2.32	0.11	0.00	0.11	2.32	0.10	1.22	0.13	2.32	0.10	2.32	0.12
P16 (6,10)	7797.20	1.64	0.09	0.00	0.24	0.00	0.09	0.00	0.23	0.00	0.09	0.00	0.24
P17 (7,10)	6967.89	0.77	0.12	0.77	0.26	0.77	0.12	0.77	0.23	0.77	0.12	0.77	0.22
P18 (8,10)	1564.45	0.00	0.21	0.00	0.32	0.00	0.18	0.00	0.32	0.00	0.19	0.00	0.20
P19 (9,10)	3250.68	15.74	0.15	0.52	0.46	13.69	0.15	0.07	0.43	4.04	0.17	1.79	0.34
P20 (10,10)	7719.79	3.12	0.23	0.79	0.29	0.55	0.24	3.76	0.37	1.99	0.26	3.76	0.37
Average	4348.36	1.46	0.10	0.36	0.15	0.96	0.09	0.64	0.15	0.66	0.10	0.75	0.13

Table 10.23. The performance of the BS heuristic on the second group of the CMWP instances with unit transportation costs.

Instance Name	Bench UB	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
		(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
hg_4_8	133.57	0.00	0.04	0.00	0.09	0.00	0.09	0.00	0.07	0.00	0.08	0.00	0.09
hg_4_15	533.77	0.00	0.06	0.00	0.10	0.00	0.10	0.00	0.10	0.00	0.09	0.00	0.10
hg_4_24	966.91	0.00	0.11	0.00	0.15	0.00	0.14	0.00	0.15	0.00	0.14	0.00	0.14
hg_5_20	6006.64	0.00	0.15	0.00	0.20	0.00	0.19	0.00	0.34	0.00	0.18	0.00	0.21
hg_5_25	6321.19	0.00	0.19	0.00	0.26	0.00	0.22	0.00	0.26	0.00	0.22	0.00	0.25
hg_5_30	11478.25	0.00	0.26	0.00	0.67	0.57	0.29	0.00	0.64	0.57	0.29	0.00	0.47
hg_5_40	17070.09	0.00	0.42	0.00	0.63	0.00	0.46	0.00	0.51	0.00	0.43	0.00	0.50
hg_5_50	15322.04	0.00	0.67	0.00	0.69	0.00	0.67	0.00	0.71	0.00	0.65	0.00	0.69
hg_5_100a	23927.70	0.30	2.83	0.00	2.86	0.30	2.77	0.30	2.90	0.30	2.69	0.30	2.91
hg_5_100b	23812.27	0.00	3.24	0.00	3.65	0.00	3.26	0.00	3.25	0.00	3.14	0.00	3.25
hg_5_100c	28334.84	0.65	3.61	0.00	5.62	0.65	3.52	0.65	3.62	0.65	3.44	0.65	3.52
hg_5_100d	26622.10	0.00	2.71	0.00	4.94	0.00	2.63	0.00	4.73	0.00	2.37	0.00	2.51
hg_5_100e	28211.98	0.00	2.87	0.00	5.40	0.00	2.76	0.00	4.87	0.00	2.73	0.00	4.43
hg_5_250	109711.85	0.00	26.51	0.00	52.17	0.00	25.58	0.00	51.39	0.17	25.04	0.00	43.58
hg_6_20	4665.63	0.00	0.27	0.00	0.42	0.00	0.31	0.00	0.43	0.00	0.31	0.00	0.43
hg_6_25	4958.71	0.00	0.27	0.00	0.36	0.00	0.32	0.00	0.35	0.00	0.31	0.00	0.35
hg_6_30	9564.39	1.27	0.37	0.25	0.91	1.27	0.41	0.00	0.61	1.27	0.39	0.00	0.62
hg_7_20	3562.43	0.00	0.27	0.00	0.53	0.00	0.32	0.00	0.37	0.00	0.30	0.00	0.34
hg_7_25	4021.82	0.33	0.35	0.33	0.44	0.33	0.37	0.33	0.40	0.33	0.37	0.33	0.40
hg_7_30	7916.86	0.00	0.45	0.00	1.03	0.00	0.49	0.00	0.78	0.00	0.49	0.00	0.65
hg_8_20	2803.62	0.00	0.29	0.00	0.43	0.00	0.36	0.00	0.41	0.00	0.36	0.00	0.39
hg_8_25	3235.83	0.48	0.42	2.87	0.76	0.48	0.49	0.48	0.57	5.81	0.46	0.48	0.54
hg_8_30	7141.48	0.47	0.62	0.47	1.73	0.56	0.66	0.56	1.11	0.56	0.66	0.74	1.00
hg_9_20	2381.30	0.00	0.43	2.09	1.07	0.00	0.50	2.09	1.10	4.01	0.44	0.00	1.17
hg_9_25	2857.26	0.00	0.55	0.00	1.50	0.00	0.58	0.00	1.30	0.00	0.55	0.00	0.95
hg_9_30	6028.25	2.26	0.76	2.26	2.46	2.13	0.88	2.13	1.03	2.13	0.87	2.13	1.06

Table 10.23. The performance of the BS heuristic on the second group of the CMWP instances with unit transportation costs cont.

Instance Name	Bench UB	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
		(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
hg_10_20	1870.99	0.14	0.44	0.00	1.11	0.00	0.48	0.00	0.73	0.00	0.46	0.00	0.73
hg_10_25	2652.54	3.15	0.66	3.15	1.31	3.15	0.75	3.71	1.59	4.81	0.68	3.71	1.53
hg_10_30	5424.01	2.01	1.05	2.01	2.31	2.01	1.04	1.10	1.47	2.01	1.16	1.10	1.43
hg_10_40	7516.13	0.00	1.94	0.00	2.77	0.00	1.89	0.00	2.32	0.00	1.93	0.00	2.06
hg_10_50	9769.90	0.00	3.28	0.00	8.38	3.56	3.38	1.54	4.25	2.53	3.26	1.30	3.79
hg_10_100	30101.22	0.36	17.54	0.16	39.28	0.16	17.57	0.15	28.40	0.16	16.71	0.15	23.90
hg_20_40	2969.07	15.44	6.44	11.89	15.96	7.16	6.29	13.54	9.94	6.29	6.13	10.83	12.85
hg_20_50	4079.25	19.27	13.73	21.69	25.71	14.87	13.39	8.83	28.15	9.44	12.90	12.43	17.94
hg_20_250	47256.56	0.48	267.11	0.85	268.05	0.48	263.03	0.00	256.36	0.48	249.60	0.71	240.34
hg_25_250a	45586.28	1.84	270.17	3.10	250.36	1.84	263.69	3.10	248.78	3.15	251.31	3.15	260.00
hg_25_250b	52570.12	0.18	243.52	1.22	259.02	0.18	263.30	1.22	268.04	1.22	259.29	1.22	268.66
hg_25_250c	46587.81	3.20	255.67	3.20	269.94	3.20	250.32	3.20	248.95	3.20	270.04	3.20	246.78
hg_25_250d	40605.47	4.28	259.04	5.13	252.53	5.13	245.94	5.13	269.52	5.13	250.55	5.13	265.57
hg_25_250e	54781.01	0.23	287.26	0.23	255.55	0.23	262.51	0.23	262.80	0.23	252.82	0.23	260.39
hg_25_500a	112788.70	2.19	74.64	3.92	236.42	3.89	67.57	3.09	197.12	3.18	75.62	3.09	150.00
hg_25_500b	92350.51	2.88	78.72	4.17	200.08	2.17	95.34	3.42	210.00	3.13	62.39	2.76	204.95
hg_25_500c	90562.32	4.55	81.61	3.19	179.97	4.81	86.17	3.79	240.31	4.81	78.83	3.65	211.80
hg_25_500d	108319.68	3.29	51.94	3.72	184.80	4.29	72.91	2.10	208.12	2.35	58.86	2.45	168.68
hg_25_500e	92443.86	3.19	73.64	3.19	195.33	3.19	76.02	2.82	200.93	3.18	96.83	1.89	192.68
hg_30_100	9722.42	7.47	148.48	5.82	243.59	7.47	146.76	5.82	243.40	6.76	143.93	7.47	241.29
hg_50_250	25505.31	1.82	242.58	1.82	314.54	1.82	317.16	1.82	249.16	1.82	240.62	1.82	247.26
Average	26405.40	1.74	51.66	1.84	70.13	1.61	53.27	1.51	69.41	1.69	50.66	1.51	65.81

Table 10.24. The performance of the BS heuristic on the second group of the CMWP instances with non-unit transportation costs.

Instance Name	Bench UB	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
		(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
ht_4_8	570.25	0.00	0.05	0.00	0.09	0.00	0.08	0.00	0.09	0.00	0.09	0.00	0.09
ht_4_15	3004.46	0.00	0.61	0.00	0.61	0.00	0.61	0.00	0.61	0.00	0.61	0.00	0.61
ht_4_24	4232.24	0.00	1.33	0.00	1.33	0.00	1.33	0.00	1.33	0.00	1.33	0.00	1.33
ht_5_20	22146.54	0.00	1.34	0.00	1.34	0.00	1.34	0.00	1.34	0.00	1.34	0.00	1.34
ht_5_25	38236.43	0.00	1.41	0.00	1.41	0.00	1.41	0.00	1.41	0.00	1.41	0.00	1.41
ht_5_30	42899.92	0.00	2.73	0.00	2.73	0.00	2.73	0.00	2.73	0.00	2.73	0.00	2.73
ht_5_40	104983.19	0.00	3.20	0.00	3.20	0.00	3.20	0.00	3.20	0.00	3.20	0.00	3.20
ht_5_50	71433.34	0.00	4.34	0.00	4.34	0.00	4.34	0.00	4.34	0.00	4.34	0.00	4.34
ht_5_100a	119564.85	0.00	16.31	0.00	16.31	0.00	16.31	0.00	16.31	0.00	16.31	0.00	16.31
ht_5_100b	128098.72	0.00	14.94	0.00	14.94	0.00	14.94	0.00	14.94	0.00	14.94	0.00	14.94
ht_5_100c	149585.44	0.00	13.13	0.00	13.13	0.00	13.13	0.00	13.13	0.00	13.13	0.00	13.13
ht_5_100d	158094.46	0.00	13.39	0.00	13.39	0.00	13.39	0.00	13.39	0.00	13.39	0.00	13.39
ht_5_100e	219839.84	0.00	11.72	0.00	11.72	0.00	11.72	0.00	11.72	0.00	11.72	0.00	11.72
ht_5_250	891589.16	0.02	61.67	0.00	61.67	0.00	61.67	0.00	61.67	0.00	61.67	0.02	61.67
ht_6_20	14693.26	0.00	2.25	0.00	2.25	0.00	2.25	0.00	2.25	0.00	2.25	0.00	2.25
ht_6_25	19036.28	0.00	2.02	0.00	2.02	0.00	2.02	0.00	2.02	0.00	2.02	0.00	2.02
ht_6_30	59665.36	0.35	3.98	0.00	3.98	0.35	3.98	0.00	3.98	0.35	3.98	0.00	3.98
ht_7_20	15075.27	0.00	2.47	0.00	2.47	6.59	2.47	0.00	2.47	6.59	2.47	6.59	2.47
ht_7_25	16086.30	0.00	2.67	0.00	2.67	0.00	2.67	0.00	2.67	0.00	2.67	0.00	2.67
ht_7_30	33627.21	0.00	3.83	0.00	3.83	0.00	3.83	0.00	3.83	0.00	3.83	0.00	3.83
ht_8_20	6499.80	0.00	3.13	0.00	3.13	0.00	3.13	0.00	3.13	2.23	3.13	0.00	3.13
ht_8_25	9486.23	0.00	3.78	0.00	3.78	0.08	3.78	0.00	3.78	0.08	3.78	0.08	3.78
ht_8_30	18081.73	7.14	8.72	4.73	8.72	4.73	8.72	4.73	8.72	4.73	8.72	4.73	8.72
ht_9_20	6551.37	2.83	4.59	0.00	4.59	2.83	4.59	6.05	4.59	6.05	4.59	6.05	4.59
ht_9_25	5983.27	4.01	3.95	4.01	3.95	4.01	3.95	4.01	3.95	4.01	3.95	4.01	3.95
ht_9_30	10643.15	0.06	5.34	0.06	5.34	0.06	5.34	0.06	5.34	0.06	5.34	0.06	5.34

Table 10.24. The performance of the BS heuristic on the second group of the CMWP instances with non-unit transportation costs cont.

Instance Name	Bench UB	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
		(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
		UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
ht_10_20	4702.16	1.84	2.75	0.09	2.75	1.84	2.75	1.84	2.75	1.84	2.75	1.84	2.75
ht_10_25	3950.70	36.18	5.88	28.02	5.88	37.45	5.88	36.18	5.88	23.90	5.88	35.89	5.88
ht_10_30	6863.79	3.18	5.28	3.18	5.28	3.18	5.28	0.63	5.28	0.63	5.28	5.73	5.28
ht_10_40	37834.75	1.11	8.38	1.11	8.38	1.11	8.38	1.11	8.38	1.11	8.38	1.11	8.38
ht_10_50	47125.80	2.75	15.58	0.00	15.58	0.00	15.58	0.00	15.58	0.00	15.58	0.00	15.58
ht_10_100	171126.36	1.22	32.11	1.22	32.11	0.98	32.11	0.98	32.11	1.06	32.11	1.06	32.11
ht_20_40	8240.14	16.72	40.30	37.42	40.30	30.77	40.30	25.67	40.30	30.77	40.30	25.67	40.30
ht_20_50	13535.76	7.57	52.44	7.57	52.44	17.47	52.44	8.98	52.44	17.47	52.44	7.43	52.44
ht_20_250	221817.88	7.08	686.00	7.08	686.00	7.08	686.00	6.77	686.00	7.08	686.00	6.77	686.00
ht_25_250a	208587.12	13.31	1097.89	13.31	1097.89	9.15	1097.89	11.86	1097.89	9.15	1097.89	11.86	1097.89
ht_25_250b	278454.58	2.51	1152.69	2.51	1152.69	1.77	1152.69	2.51	1152.69	1.77	1152.69	2.51	1152.69
ht_25_250c	257526.84	0.86	1260.77	0.86	1260.77	1.34	1260.77	0.86	1260.77	1.34	1260.77	0.86	1260.77
ht_25_250d	195125.60	6.94	946.25	6.94	946.25	1.93	946.25	1.93	946.25	1.93	946.25	1.93	946.25
ht_25_250e	289012.15	1.13	931.09	1.13	931.09	0.00	931.09	1.13	931.09	1.13	931.09	1.13	931.09
ht_25_500a	646653.93	7.09	3056.73	4.08	3056.73	4.04	3056.73	3.62	3056.73	3.84	3056.73	1.38	3056.73
ht_25_500b	512483.15	6.43	3409.83	9.55	3409.83	4.91	3409.83	8.52	3409.83	4.44	3409.83	8.52	3409.83
ht_25_500c	484237.33	11.19	3697.81	6.29	3697.81	8.92	3697.81	3.36	3697.81	6.66	3697.81	0.00	3697.81
ht_25_500d	643832.22	3.62	4718.78	6.16	4718.78	4.75	4718.78	4.65	4718.78	4.75	4718.78	4.65	4718.78
ht_25_500e	491602.33	11.73	3373.33	10.66	3373.33	6.85	3373.33	8.01	3373.33	6.85	3373.33	4.99	3373.33
ht_30_100	26560.19	25.10	377.66	20.69	377.66	25.98	377.66	19.53	377.66	17.21	377.66	19.53	377.66
ht_50_250	87077.90	0.00	5142.06	0.00	5142.06	5.45	5142.06	0.00	5142.06	5.45	5142.06	4.80	5142.06
Average	144809.76	3.87	642.69	3.76	642.69	4.12	642.69	3.47	642.69	3.67	642.69	3.60	642.69

The presentation of Table 10.26 is similar to the one of Table 10.25. Table 10.26 provides the performance of the ℓ_∞-norm based lower bounding procedure (namely ℓ_∞-RMDAP1) within the MABB algorithm. Notice that for the MABB algorithm, we prefer a weaker (looser) but more efficient lower bounding procedure. The motivation behind the use of LR of MDAP1 with ℓ_∞-norm is its approved efficiency on the CMWP instances. The lower bounds are always tightened with Z_{MSAS} inspired by the results obtained for the CMWP.

For the sake of conciseness, we will denote the MABB algorithm with the RLT based lower bounding procedure as RLT-MABB and the MABB algorithm with ℓ_∞-norm based lower bounding procedure as ℓ_∞-MABB in the sequel. The computational results favor the ℓ_∞-MABB algorithm when compared to RLT-MABB algorithm. Although there are a few instances for which RLT-MABB algorithm is superior to ℓ_∞-MABB, the ℓ_∞-MABB algorithm performs much better than the RLT-MABB algorithm in terms of both accuracy and duality gaps. On the average, the RLT-MABB and ℓ_∞-MABB algorithms yield upper bound values with percent deviations from Z_R by 3.92% and 0.61% and, duality gaps 55.66% and 9.66%, respectively. Although, the RLT based lower bounds are weak, the RLT-MABB algorithm can solve 11 out of 60 test instances with 0.1% closeness to optimality. On the other hand, the ℓ_∞-MABB algorithm outputs 8 out of 60 test instances within 0.001 of the optimal value. The RLT-MABB algorithm requires less CPU time than the ℓ_∞-MABB algorithm does on the average. Meanwhile, the CPU times reach the four-hour limit on most of the instances when both RLT-MABB and ℓ_∞-MABB algorithms are used. It is observed that the performance of the MABB algorithm deteriorates with the increasing number of commodities. Furthermore, the total number of allocation variables has also a negative effect on the MABB algorithm's performance.

Based on the observations, we can say that MBrS2 does not perform well with neither RLT nor the ℓ_∞-norm based lower bounding procedure. Moreover, MBrS1 seems to be slightly better than MBrS3. Therefore, we can also claim that the performance of the MABB algorithm is significantly affected by the sequence of allocation variables selected for branching. The winner is the ℓ_∞-MABB algorithm employing the ℓ_∞-norm based lower bounding procedure with MBrS1.

In addition to the MABB algorithm, LBB algorithms (i.e., MLBB and MLBBCE algorithms) are also tested as part of the experiments on the test instances. Both LP (Z_{LP}^{MLBB}

given by Equation 9.3) and ℓ_∞-norm (Z_{MDAP}^{MLBB}) based lower bounding procedures are initially used together within the MLBB and MLBBCE algorithms. However, the calculation of ℓ_∞-norm based lower bounding procedure continues until $3 \times N \times K$ rectangles are generated by both MLBB and MLBBCE algorithms. This setting improves the performance of both the MLBB and MLBBCE algorithms as it is the case for the CMWP. In summary, the block norm based lower bounding is expensive and the performance of the algorithm deteriorates when it is calculated at every bounding step together with the LP based bounding procedure. Actually, the block norm based lower bounding procedure produces sufficiently tight bounds in the early stages of the LBB and LBBCE algorithms. However, as the algorithm proceeds LP based lower bound values exceeds the block norm based lower bound values. Hence, we continue only with the LP based lower bounding procedure when the number of rectangles reaches a predefined upper limit.

Table 10.27 gives the results obtained with both the MLBB and MLBBCE algorithms. Table 10.27 has the same format with Table 10.19 – 10.21. At the root node of the BB tree Z_{MDAP3} is equal to Z_{MDAP}^{MLBB}. Therefore, the initial lower bounds are the same for both ABB and LBB algorithms using ℓ_∞-norm at the root nodes.

Our computational experiments indicate that the MLBB algorithm outperforms the MLBBCE algorithm on the average in terms of duality gaps. However, there exist four instances for which MLBBCE algorithm yields better lower bounds than the MLBB algorithm does. The MLBBCE algorithm seems to be more efficient than the MLBB algorithm on the average but its performance deteriorates significantly for instances with more than 10 facilities. In particular, the number of facility-rectangle combinations increases quickly during the run of the MLBBCE algorithm. This limits the use of MLBBCE algorithm on instances with small number of facilities (i.e., $I \leq 10$). Once a rectangle is partitioned, the subproblems produced by the MLBBCE algorithm can be solved by multiple processors simultaneously as a remedy. We believe the performance of the MLBBCE algorithm may be improved by a parallel implementation.

The MLBB algorithm yields duality gaps less than 0.15 of the optimum on 90% (54 out of 60) of the test instances. On the other hand, this ratio is 65% (39 out of 60) for the MLBBCE algorithm and 85% (51 out of 60) for the best performing MABB algorithm (ℓ_∞-MABB using MBrS1). On the average, the MLBB algorithm finds duality gaps within

Table 10.25. The performance of the MABB algorithm with the RLT bounds on the first group of MCMWP instances.

Instance	Init	RLT + MBrS1			RLT + MBrS2			RLT + MBrS3		
Name	UB	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_2_2_2	84.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_2_3	524.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_2_5	100.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_4_2	932.57	0.10	0.29	0.00	0.10	0.29	0.01	0.10	0.29	0.00
mc_2_4_3	720.26	0.10	0.42	0.03	0.10	0.19	0.05	0.10	0.65	0.03
mc_2_4_5	1737.97	0.10	0.12	0.04	0.10	0.10	0.22	0.10	0.12	0.03
mc_3_5_2	2082.54	0.10	0.22	0.01	0.10	0.28	0.01	0.10	0.13	0.01
mc_3_5_3	790.67	0.10	0.10	0.71	0.10	0.10	25.69	0.10	0.12	0.70
mc_3_5_5	4786.91	0.10	0.10	60.04	0.10	17.73	240.00	0.10	0.10	58.84
mc_3_9_2	2322.95	0.10	5.32	240.00	0.10	31.99	240.00	0.10	5.92	240.00
mc_3_9_3	7335.42	0.10	11.95	240.00	0.10	30.23	240.00	0.10	12.51	240.00
mc_3_9_5	57182.27	0.90	17.12	240.01	0.90	29.07	240.01	0.90	17.85	240.00
mc_4_8_2	1592.78	0.03	0.10	0.95	0.03	0.11	1.60	0.03	0.10	0.95
mc_4_8_3	8919.72	0.10	11.19	240.00	0.10	26.20	240.00	0.10	10.93	240.00
mc_4_8_5	9793.59	0.05	26.18	240.00	0.05	43.25	240.00	0.05	25.92	240.01
mc_4_10_2	8482.43	0.10	5.34	240.00	0.10	22.09	240.00	0.10	5.28	240.00
mc_4_10_3	13110.09	0.44	11.23	240.00	0.44	43.99	240.00	0.44	10.88	240.00
mc_4_10_5	37022.02	0.03	41.15	240.00	0.03	55.77	240.00	0.03	41.30	240.00
mc_4_15_2	11476.23	0.10	31.93	240.00	0.10	67.01	240.00	0.10	32.91	240.00
mc_4_15_3	46104.69	0.62	41.09	240.00	0.62	61.93	240.00	0.62	41.02	240.00
mc_4_15_5	34120.48	0.01	88.59	240.01	0.01	98.20	240.00	0.01	88.30	240.01

Table 10.25. The performance of the MABB algorithm with the RLT bounds on the first group of MCMWP instances cont.

Instance	Init	RLT + MBrS1			RLT + MBrS2			RLT + MBrS3		
Name	UB	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_5_8_2	7035.34	0.10	0.10	3.28	0.10	2.36	240.00	0.10	0.10	3.46
mc_5_8_3	9716.66	0.10	39.52	240.00	0.10	82.31	240.00	0.10	37.91	240.00
mc_5_8_5	33238.26	6.82	42.20	240.00	6.82	76.76	240.00	6.82	43.93	240.00
mc_5_10_2	4827.05	0.10	0.61	240.00	0.10	24.59	240.00	0.10	1.34	240.00
mc_5_10_3	19683.12	0.10	30.38	240.01	0.10	46.79	240.01	0.10	31.27	240.00
mc_5_10_5	57738.93	0.29	33.22	240.01	0.29	34.04	240.01	0.29	33.09	240.01
mc_5_12_2	5150.44	3.40	87.09	240.00	3.40	98.25	240.01	3.40	87.19	240.00
mc_5_12_3	14734.72	0.34	61.55	240.00	0.34	73.97	240.00	0.34	60.92	240.00
mc_5_12_5	53643.85	11.28	36.11	240.01	11.28	46.51	240.02	11.28	36.70	240.01
mc_5_15_2	6582.82	0.01	33.11	240.01	0.01	70.48	240.00	0.01	33.79	240.01
mc_5_15_3	47265.38	0.04	78.46	240.01	0.04	91.13	240.01	0.04	78.21	240.01
mc_5_15_5	32277.56	4.79	73.20	240.01	4.79	94.74	240.00	4.79	72.84	240.01
mc_5_20_2	12506.04	1.35	45.29	240.00	1.35	84.86	240.00	1.35	44.86	240.00
mc_5_20_3	24425.72	2.05	99.49	240.03	2.05	102.05	240.01	2.05	99.36	240.01
mc_5_20_5	48244.38	0.02	61.35	240.03	0.15	80.32	240.03	0.02	61.40	240.02
mc_5_30_2	56593.66	0.00	95.76	240.00	0.00	99.08	240.01	0.00	95.66	240.01
mc_5_30_3	95743.88	0.00	98.65	240.02	0.00	98.83	240.03	0.00	98.62	240.02
mc_5_30_5	226741.59	0.01	94.26	240.03	0.01	95.92	240.05	0.01	94.11	240.05
mc_6_10_2	4804.01	0.11	35.98	240.00	0.11	50.59	240.00	0.11	36.59	240.00
mc_6_10_3	10965.43	0.09	42.35	240.01	0.09	58.25	240.00	0.09	42.57	240.00
mc_6_10_5	14115.76	0.11	58.97	240.01	0.11	68.32	240.00	0.11	59.39	240.01

Table 10.25. The performance of the MABB algorithm with the RLT bounds on the first group of MCMWP instances cont.

Instance	Init	RLT + MBrS1			RLT + MBrS2			RLT + MBrS3		
Name	UB	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_8_10_2	12752.86	0.05	63.41	240.00	0.05	75.14	240.00	0.05	63.45	240.00
mc_8_10_3	21974.01	1.43	55.86	240.01	5.23	79.39	240.00	5.87	59.90	240.00
mc_8_10_5	29002.70	4.80	85.13	240.00	3.81	85.08	240.02	4.80	85.07	240.01
mc_10_10_2	7185.13	4.90	102.50	240.00	7.59	103.77	240.00	4.90	102.39	240.00
mc_10_10_3	15572.58	4.67	93.59	240.01	5.19	95.48	240.01	4.67	93.88	240.00
mc_10_10_5	13219.58	30.65	130.65	240.02	36.40	136.40	240.00	30.65	130.65	240.00
mc_10_15_2	6617.89	31.38	131.38	240.00	14.46	114.46	240.00	31.38	131.38	240.01
mc_10_15_3	9347.60	4.33	98.73	240.01	2.07	97.25	240.03	2.84	92.96	240.02
mc_10_15_5	19704.50	6.85	106.22	240.03	6.85	106.69	240.01	6.93	106.37	240.07
mc_10_20_2	26253.55	42.86	142.86	240.01	36.32	136.32	240.01	42.86	142.86	240.00
mc_10_20_3	13032.02	17.33	117.33	240.04	6.54	106.09	240.02	19.03	119.03	240.02
mc_10_20_5	22089.73	7.24	107.24	240.11	7.59	107.57	240.13	4.15	104.15	240.12
mc_10_24_2	5809.89	21.48	121.48	240.02	10.42	110.42	240.03	21.48	121.48	240.04
mc_10_24_3	14744.54	14.97	114.97	240.01	22.11	122.11	240.06	14.97	114.97	240.07
mc_10_24_5	38410.05	15.96	113.13	240.07	5.88	103.31	240.38	19.36	116.43	240.10
mc_10_30_2	48785.94	0.16	100.16	240.00	1.72	101.72	240.00	0.16	100.16	240.02
mc_10_30_3	64196.14	7.72	107.72	240.01	12.39	112.39	240.02	7.72	107.72	240.02
mc_10_30_5	102820.83	6.91	106.91	240.10	16.02	116.02	240.10	12.91	112.91	240.21
Average	25112.93	4.30	55.66	197.10	3.92	65.31	204.48	4.48	55.83	197.08

Table 10.26. The performance of the MABB algorithm with the ℓ_∞-RMDAP1 bounds on the first group of the MCMWP instances.

Instance	Init	ℓ_∞-RMDAP1 + MBrS1			ℓ_∞-RMDAP1 + MBrS2			ℓ_∞-RMDAP1 + MBrS3		
Name	UB	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_2_2_2	84.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_2_3	524.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_2_5	100.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_4_2	932.57	0.10	0.14	0.10	0.10	0.12	0.14	0.10	0.28	0.12
mc_2_4_3	720.26	0.10	0.10	1.24	0.10	0.10	1.46	0.10	0.10	1.18
mc_2_4_5	1737.97	0.10	0.10	214.27	0.10	0.84	240.00	0.10	0.10	223.97
mc_3_5_2	2082.54	0.10	0.10	0.36	0.10	0.14	0.48	0.10	0.20	0.41
mc_3_5_3	790.67	0.10	0.10	145.33	0.10	1.92	240.00	0.10	0.10	156.06
mc_3_5_5	4786.91	0.10	5.91	240.00	0.10	7.49	240.00	0.10	5.99	240.01
mc_3_9_2	2322.95	0.10	1.46	240.00	0.10	3.46	240.01	0.10	1.54	240.00
mc_3_9_3	7335.42	0.10	6.76	240.00	0.10	7.62	240.01	0.10	6.82	240.01
mc_3_9_5	57182.27	0.90	18.76	240.00	0.90	19.38	240.00	0.90	18.74	240.01
mc_4_8_2	1592.78	0.03	0.10	228.37	0.03	2.76	240.00	0.03	0.13	240.00
mc_4_8_3	8919.72	0.10	5.20	240.00	0.10	5.53	240.00	0.10	5.25	240.00
mc_4_8_5	9793.59	0.05	8.02	240.01	0.05	8.12	240.00	0.05	8.04	240.00
mc_4_10_2	8482.43	0.10	0.95	240.00	0.10	2.46	240.00	0.10	1.05	240.00
mc_4_10_3	13110.09	0.44	4.01	240.00	0.44	5.71	240.00	0.44	4.03	240.00
mc_4_10_5	37022.02	0.03	8.75	240.00	0.03	9.08	240.00	0.03	8.77	240.00
mc_4_15_2	11476.23	0.10	13.41	240.00	0.10	14.06	240.01	0.10	13.49	240.00
mc_4_15_3	46104.69	0.62	10.75	240.01	0.62	10.84	240.01	0.62	10.75	240.01
mc_4_15_5	34120.48	0.01	8.07	240.03	0.01	8.08	240.01	0.01	8.07	240.01

Table 10.26. The performance of the MABB algorithm with the ℓ_∞-RMDAP1 bounds on the first group of the MCMWP instances cont.

Instance	Init	ℓ_∞-RMDAP1 + MBrS1			ℓ_∞-RMDAP1 + MBrS2			ℓ_∞-RMDAP1 + MBrS3		
Name	UB	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_5_8_2	7035.34	0.10	4.78	240.00	0.10	8.53	240.00	0.10	4.95	240.00
mc_5_8_3	9716.66	0.10	13.24	240.00	0.10	16.06	240.00	0.10	13.12	240.00
mc_5_8_5	33238.26	6.82	27.25	240.00	6.82	27.56	240.00	6.82	27.28	240.00
mc_5_10_2	4827.05	0.10	4.31	240.00	0.10	5.26	240.00	0.10	4.37	240.00
mc_5_10_3	19683.12	0.10	12.10	240.01	0.10	13.83	240.00	0.10	13.51	240.01
mc_5_10_5	57738.93	0.29	18.37	240.01	0.29	19.31	240.00	0.29	18.38	240.00
mc_5_12_2	5150.44	3.40	13.95	240.01	3.40	14.48	240.00	3.40	14.00	240.00
mc_5_12_3	14734.72	0.34	11.51	240.01	0.34	12.29	240.00	0.34	11.53	240.00
mc_5_12_5	53643.85	11.28	21.98	240.01	11.28	20.23	240.01	11.28	21.97	240.02
mc_5_15_2	6582.82	0.01	5.98	240.01	0.01	6.32	240.00	0.01	5.99	240.01
mc_5_15_3	47265.38	0.04	10.91	240.01	0.04	11.01	240.00	0.04	10.95	240.02
mc_5_15_5	32277.56	4.79	13.03	240.00	4.79	16.30	240.02	4.79	13.04	240.03
mc_5_20_2	12506.04	1.35	8.90	240.00	1.35	9.20	240.01	1.35	8.92	240.01
mc_5_20_3	24425.72	2.05	12.22	240.01	2.05	12.21	240.01	2.05	12.22	240.03
mc_5_20_5	48244.38	0.02	11.21	240.06	0.02	11.86	240.02	0.02	11.20	240.04
mc_5_30_2	56593.66	0.00	8.95	240.01	0.00	8.93	240.02	0.00	8.95	240.05
mc_5_30_3	95743.88	0.00	9.09	240.01	0.00	9.09	240.03	0.00	9.09	240.05
mc_5_30_5	226741.59	0.01	10.60	240.06	0.01	10.81	240.01	0.01	10.61	240.04
mc_6_10_2	4804.01	0.11	6.53	240.00	0.11	7.21	240.00	0.11	6.53	240.00
mc_6_10_3	10965.43	0.09	8.54	240.00	0.09	8.88	240.00	0.09	8.64	240.00
mc_6_10_5	14115.76	0.11	14.52	240.01	0.11	15.22	240.00	0.11	15.01	240.01

Table 10.26. The performance of the MABB algorithm with the ℓ_∞-RMDAP1 bounds on the first group of the MCMWP instances cont.

Instance	Init	ℓ_∞-RMDAP1 + MBrS1			ℓ_∞-RMDAP1 + MBrS2			ℓ_∞-RMDAP1 + MBrS3		
Name	UB	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_8_10_2	12752.86	0.05	7.75	240.00	0.05	11.21	240.04	0.05	8.69	240.01
mc_8_10_3	21974.01	0.37	13.34	240.00	0.37	18.16	240.01	0.37	13.56	240.01
mc_8_10_5	29002.70	0.00	12.85	240.01	0.00	17.64	240.00	0.00	12.98	240.02
mc_10_10_2	7185.13	0.00	7.23	240.00	0.00	8.90	240.00	0.00	7.14	240.00
mc_10_10_3	15572.58	0.00	31.45	240.01	0.00	34.15	240.00	0.00	31.63	240.01
mc_10_10_5	13219.58	0.01	7.65	240.00	0.01	14.01	240.00	0.01	7.66	240.00
mc_10_15_2	6617.89	0.14	11.39	240.01	0.14	22.89	240.01	0.14	13.14	240.00
mc_10_15_3	9347.60	0.01	9.27	240.01	0.01	12.05	240.01	0.01	9.48	240.00
mc_10_15_5	19704.50	0.20	17.52	240.01	0.20	20.59	240.00	0.20	17.99	240.02
mc_10_20_2	26253.55	0.23	12.39	240.01	0.23	14.64	240.01	0.23	12.38	240.02
mc_10_20_3	13032.02	0.01	13.02	240.03	0.01	18.46	240.01	0.01	13.27	240.01
mc_10_20_5	22089.73	0.22	15.04	240.04	0.22	18.63	240.06	0.22	15.02	240.03
mc_10_24_2	5809.89	0.47	15.82	240.03	0.47	17.84	240.02	0.47	15.85	240.01
mc_10_24_3	14744.54	0.01	17.40	240.01	0.01	19.21	240.05	0.01	17.40	240.04
mc_10_24_5	38410.05	0.01	13.12	240.05	0.01	14.00	240.12	0.01	13.16	240.02
mc_10_30_2	48785.94	0.16	8.86	240.01	0.16	16.09	240.05	0.16	9.09	240.03
mc_10_30_3	64196.14	0.16	13.21	240.02	0.16	13.50	240.07	0.16	13.18	240.08
mc_10_30_5	102820.83	0.06	11.89	240.15	0.06	16.44	240.06	0.06	11.99	240.05
Average	25112.93	0.61	9.66	213.84	0.61	11.35	216.05	0.61	9.79	214.37

6.97% while MLBBCE and best performing MABB algorithm yield duality gaps of 11.08% and 9.66%, respectively. In total, the MLBB and MLBBCE algorithms solve 19 out of 60 and 21 out of 60 instances within 0.001 of optimum, respectively. On the other hand, this number is 11 out of 60 for the MABB algorithm. As a verdict, the location based algorithms perform better than the allocation based algorithms in the light of our extensive experiments.

BS heuristic is also applied on the MCMWP test instances with the same settings defined for the CMWP. Our results are summarized in Table 10.28 for the first class of MCMWP instances and in Table 10.29 for the second class of the test instances. Table 10.28 and Table 10.29 have the same outline with Table 10.22–24. As the reference values Z_R, benchmark lower bounds in Table A.1 and A.2 are used. BS heuristic performance is more promising for the MCMWP instances than the CMWP instances. Actually, there is a trade-off between the performance of the BS heuristic and the beam width W. Clearly, the larger the beam width is the larger the accuracy and running times are. In general, $W = 3$ yields more accurate solutions for the BS heuristic. What is more, we can state that for different values of Ψ, the BS heuristic may perform differently. Hence the most suitable Ψ value should be calibrated by some trial and error experiments. According to our computational experiments, we observe that the combination $(\Psi,W) = (0.25,3)$ has a superior performance than the other combinations for the first class of MCMWP instances while $(\Psi,W) = (0.5,3)$ is better for the second class of the test instances.

According to our computational experiments, BS finds very close solutions (less than 1%) to the best performing heuristic CL-MDA1 for the first group of MCMWP test instances. Their accuracies are 0.61% and 0.91% on the average for the CL-MDA1 and BS heuristic, respectively. On the other hand, CL-MDA1 is more efficient than the BS heuristic in the average. The BS heuristic is 2.93% more accurate than the best performing LR heuristic CL-RMDA2 on the average with an expense of additional CPU time requirements.

On the second class of test instances (very large instances), BS heuristic yields 7.45% percent deviations from the Z_R. This deviation is 13.45% for the CL-RMDA2 heuristic which is more efficient than the BS heuristic. On these instances, the best performing heuristic CL-MDA2 does not find solutions in 7 out of 18 test instances even with a four-hour time limit which is stated in Table 10.9. Consequently, BS heuristic yields competitive results with the DA heuristics which have the best accuracy and efficiency for the MCMWP on large instances.

Table 10.27. The performance of the MLBB and MLBBCE algorithms on the first group of the MCMWP instances.

Instance	MLBB ($Z_{LP}^{MLBB} + Z_{MDAP}^{MLBB}$)			MLBBCE ($Z_{LP}^{MLBB} + Z_{MDAP}^{MLBB}$)		
Name	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_2_2_2	0.00	0.00	0.00	0.00	0.00	0.01
mc_2_2_3	0.00	0.00	0.02	0.00	0.00	0.01
mc_2_2_5	0.00	0.00	0.00	0.00	0.00	0.01
mc_2_4_2	0.10	0.10	0.13	0.10	0.10	0.10
mc_2_4_3	0.10	0.10	0.44	0.10	0.10	0.47
mc_2_4_5	0.10	0.10	1.25	0.10	0.10	0.52
mc_3_5_2	0.10	0.10	1.47	0.10	0.10	1.30
mc_3_5_3	0.10	0.10	8.74	0.10	0.11	4.86
mc_3_5_5	0.10	0.10	6.91	0.10	0.10	6.32
mc_3_9_2	0.10	0.10	80.73	0.10	0.10	97.26
mc_3_9_3	0.10	0.10	5.94	0.10	0.11	4.98
mc_3_9_5	0.90	0.98	240.00	0.90	0.90	240.00
mc_4_8_2	0.03	0.10	16.97	0.03	0.11	11.81
mc_4_8_3	0.10	0.10	34.61	0.10	0.10	37.54
mc_4_8_5	0.05	0.10	21.66	0.05	0.10	7.88
mc_4_10_2	0.10	0.10	133.05	0.10	0.10	79.94
mc_4_10_3	0.44	0.44	240.00	0.44	0.67	240.00
mc_4_10_5	0.03	0.10	240.00	0.03	0.10	155.73
mc_4_15_2	0.10	0.10	233.52	0.10	0.11	240.00
mc_4_15_3	0.62	6.48	240.00	0.62	11.02	240.00
mc_4_15_5	0.01	5.95	240.00	0.01	8.72	240.00
mc_5_8_2	0.10	0.10	38.33	0.10	0.11	13.26
mc_5_8_3	0.10	0.10	26.53	0.10	0.10	20.55
mc_5_8_5	6.82	6.82	240.00	6.82	15.83	240.00
mc_5_10_2	0.10	0.10	33.19	0.10	0.10	38.51
mc_5_10_3	0.10	0.10	57.90	0.10	0.10	68.35
mc_5_10_5	0.29	14.79	240.00	0.29	19.55	240.11
mc_5_12_2	3.40	3.40	240.00	3.40	4.46	240.00
mc_5_12_3	0.34	1.59	240.00	0.34	0.34	240.00
mc_5_12_5	11.28	14.66	240.00	11.28	26.24	240.00
mc_5_15_2	0.01	4.76	240.00	0.01	3.68	240.01
mc_5_15_3	0.04	4.34	240.00	0.04	9.90	240.00
mc_5_15_5	4.79	10.19	240.00	4.79	16.49	240.00

Table 10.27. The performance of the MLBB and MLBBCE algorithms on the first group of the MCMWP instances cont.

Instance	MLBB ($Z_{LP}^{MLBB} + Z_{MDAP}^{MLBB}$)			MLBBCE ($Z_{LP}^{MLBB} + Z_{MDAP}^{MLBB}$)		
Name	UB(%)	GAP(%)	CPU	UB(%)	GAP(%)	CPU
mc_5_20_2	1.35	8.22	240.00	1.35	10.78	240.00
mc_5_20_3	2.05	11.44	240.00	2.05	12.62	240.01
mc_5_20_5	0.02	8.53	240.02	0.02	12.43	240.00
mc_5_30_2	0.00	7.63	240.01	0.00	8.96	240.20
mc_5_30_3	0.00	7.40	240.00	0.00	9.11	240.01
mc_5_30_5	0.01	9.55	240.04	0.01	10.68	240.01
mc_6_10_2	0.11	6.46	240.00	0.11	0.11	164.11
mc_6_10_3	0.09	9.57	240.00	0.09	11.21	240.00
mc_6_10_5	0.11	14.89	240.02	0.11	17.38	240.00
mc_8_10_2	0.05	11.26	240.00	0.05	11.20	240.00
mc_8_10_3	0.37	16.05	240.00	0.37	22.72	240.06
mc_8_10_5	0.00	13.51	240.18	0.00	22.11	240.10
mc_10_10_2	0.00	14.94	240.02	0.00	30.31	241.08
mc_10_10_3	0.00	25.33	240.00	0.00	35.41	240.02
mc_10_10_5	0.01	17.22	240.18	0.01	39.06	243.75
mc_10_15_2	0.14	15.36	240.00	0.14	47.63	240.11
mc_10_15_3	0.01	9.24	240.04	0.01	18.63	240.20
mc_10_15_5	0.20	14.84	240.11	0.20	27.40	242.77
mc_10_20_2	0.23	13.03	240.00	0.23	19.69	240.06
mc_10_20_3	0.01	14.28	240.37	1.62	28.68	241.63
mc_10_20_5	0.14	13.28	240.07	1.09	24.26	240.31
mc_10_24_2	0.47	16.30	240.02	0.47	26.84	240.06
mc_10_24_3	0.01	18.75	240.01	3.00	26.66	240.64
mc_10_24_5	0.01	12.39	240.21	0.01	17.70	240.53
mc_10_30_2	0.16	10.76	240.06	0.16	21.49	240.25
mc_10_30_3	0.16	11.64	240.03	0.16	17.13	240.86
mc_10_30_5	0.06	10.37	240.23	0.06	14.81	241.26
Average	0.61	6.97	171.72	0.70	11.08	168.13

Table 10.28. The performance of the BS heuristic on the first group of the MCMWP instances.

Instance Name	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
	(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_2_2_2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_2_3	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.02
mc_2_2_5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
mc_2_4_2	0.10	0.03	0.10	0.06	0.10	0.03	0.10	0.06	0.10	0.03	0.10	0.06
mc_2_4_3	0.10	0.04	0.10	0.08	0.10	0.04	0.10	0.08	0.10	0.04	0.10	0.08
mc_2_4_5	0.10	0.64	0.10	0.86	0.10	0.63	0.10	0.86	0.10	0.63	0.10	0.89
mc_3_5_2	0.10	0.99	0.10	1.79	0.10	0.97	0.10	1.78	0.10	0.97	0.10	1.78
mc_3_5_3	0.10	0.47	0.10	0.80	0.10	0.34	0.10	0.78	0.10	0.34	0.10	0.72
mc_3_5_5	0.10	1.16	0.10	1.11	0.10	1.16	0.10	1.44	0.10	1.16	0.10	1.25
mc_3_9_2	0.10	0.12	0.10	0.45	0.10	0.12	0.10	0.41	0.10	0.12	0.10	0.42
mc_3_9_3	0.10	1.75	0.10	1.88	0.10	1.74	0.10	1.88	0.10	1.74	0.10	1.89
mc_3_9_5	0.90	0.33	0.90	0.71	0.90	0.33	0.90	0.71	0.90	0.33	0.90	0.70
mc_4_8_2	0.04	1.37	0.04	1.72	0.04	1.35	0.04	1.71	0.04	1.36	0.04	1.71
mc_4_8_3	0.10	2.00	0.10	2.40	0.10	1.99	0.10	2.39	0.10	2.31	0.10	2.40
mc_4_8_5	0.05	3.34	0.05	6.36	0.05	3.35	0.05	5.95	0.05	3.34	0.05	5.93
mc_4_10_2	0.87	0.27	0.10	0.73	0.87	0.27	0.10	0.73	0.87	0.27	0.10	0.73
mc_4_10_3	0.44	3.02	0.44	3.26	0.44	0.36	0.44	3.06	0.44	0.36	0.44	0.70
mc_4_10_5	0.03	0.97	0.03	1.96	0.03	3.59	0.03	5.01	0.03	3.58	0.03	4.62
mc_4_15_2	0.10	1.82	0.10	2.72	0.10	1.09	0.10	2.71	0.10	1.10	0.10	2.73
mc_4_15_3	0.62	1.12	0.62	4.08	0.62	1.12	0.62	2.04	0.62	1.12	0.62	4.09
mc_4_15_5	0.01	2.26	0.01	3.91	0.01	2.26	0.01	3.89	0.01	2.26	0.01	3.88

Table 10.28. The performance of the BS heuristic on the first group of the MCMWP instances cont.

Instance Name	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
	(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_5_8_2	0.10	3.64	0.10	4.78	0.10	3.62	0.10	4.89	0.10	3.63	0.10	4.90
mc_5_8_3	0.10	3.17	0.10	3.65	0.10	3.15	0.10	4.47	0.10	0.98	0.10	7.51
mc_5_8_5	6.82	2.88	6.82	6.73	6.82	2.88	6.82	9.27	6.82	1.14	6.82	4.07
mc_5_10_2	0.10	2.64	0.10	8.58	0.10	2.63	0.10	8.59	0.10	2.64	0.10	8.56
mc_5_10_3	0.10	4.40	0.10	10.18	0.10	4.40	0.10	10.22	0.10	4.40	0.10	8.26
mc_5_10_5	0.29	5.71	0.29	8.49	0.29	2.19	0.29	8.69	0.29	2.19	0.29	9.12
mc_5_12_2	3.40	1.01	3.40	3.49	3.40	1.01	3.40	3.51	3.40	1.02	3.40	3.49
mc_5_12_3	0.34	4.87	0.34	5.11	0.34	4.87	0.34	5.72	0.34	3.79	0.34	4.89
mc_5_12_5	11.28	1.81	11.28	4.22	11.28	1.82	11.28	4.21	11.28	1.82	11.28	4.26
mc_5_15_2	0.01	0.69	0.01	1.35	0.01	0.68	0.01	1.36	0.01	0.69	0.01	1.36
mc_5_15_3	0.04	4.39	0.04	7.12	0.04	4.38	0.04	7.13	0.04	4.38	0.04	7.12
mc_5_15_5	4.79	8.14	4.79	10.92	4.79	8.14	4.79	10.87	4.79	7.77	4.79	16.51
mc_5_20_2	1.35	0.85	1.35	1.59	1.35	0.85	1.35	1.59	1.35	0.85	1.35	1.58
mc_5_20_3	2.05	1.90	2.05	3.35	2.05	1.68	2.05	3.65	2.05	1.69	2.05	4.02
mc_5_20_5	0.02	10.99	0.02	23.25	0.02	10.99	0.02	25.31	0.02	10.98	0.02	8.09
mc_5_30_2	0.00	10.26	0.00	23.48	0.00	10.27	0.00	23.48	0.00	10.29	0.00	23.46
mc_5_30_3	0.00	3.13	0.00	6.66	0.00	2.66	0.00	7.69	0.00	3.90	0.00	8.47
mc_5_30_5	0.01	6.39	0.01	13.52	0.01	6.39	0.01	13.48	0.01	6.39	0.01	14.68
mc_6_10_2	0.11	0.27	0.11	5.82	0.11	0.22	0.11	5.54	0.11	2.41	0.11	5.92
mc_6_10_3	0.09	3.52	0.09	3.40	0.09	3.52	0.09	2.71	0.09	3.52	0.09	8.02
mc_6_10_5	0.11	6.34	0.11	17.31	0.11	6.34	0.11	13.61	0.11	6.02	0.11	12.85

Table 10.28. The performance of the BS heuristic on the first group of the MCMWP instances cont.

Instance Name	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
	(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_8_10_2	0.05	2.41	0.05	5.61	0.05	1.83	0.05	5.37	8.23	2.98	0.05	21.01
mc_8_10_3	0.37	4.25	0.37	7.93	0.37	4.26	0.37	3.88	11.84	7.24	1.43	23.10
mc_8_10_5	0.00	2.29	0.00	40.99	0.00	14.73	0.00	37.85	0.00	14.75	0.00	45.01
mc_10_10_2	9.24	7.36	6.47	7.46	6.39	4.62	1.84	5.84	3.34	2.94	0.00	12.10
mc_10_10_3	0.17	4.79	0.00	14.76	0.00	5.73	0.00	12.40	0.00	5.58	0.00	11.50
mc_10_10_5	6.54	15.57	2.26	36.27	6.54	19.81	2.26	51.96	6.54	19.84	0.01	27.30
mc_10_15_2	22.83	3.88	4.65	10.08	9.67	6.98	4.65	8.39	9.67	5.60	21.77	8.72
mc_10_15_3	2.92	15.92	1.54	11.85	7.28	5.76	1.54	17.07	6.85	3.09	3.46	12.92
mc_10_15_5	0.81	14.73	2.16	32.80	0.81	14.63	0.81	35.55	6.97	15.74	0.81	25.05
mc_10_20_2	10.39	3.11	0.23	10.11	0.23	4.55	0.23	9.74	0.23	3.61	0.23	6.17
mc_10_20_3	6.54	11.12	5.50	28.67	4.38	25.39	6.54	46.70	4.73	9.39	4.87	59.61
mc_10_20_5	3.36	25.19	6.07	35.33	3.36	12.67	0.22	42.90	6.66	9.61	0.22	62.84
mc_10_24_2	2.59	9.95	0.97	19.81	1.21	6.77	1.21	21.47	1.21	6.77	1.21	17.77
mc_10_24_3	1.30	8.13	1.30	48.05	0.56	8.10	0.01	42.48	3.94	10.09	3.94	37.12
mc_10_24_5	3.05	16.00	1.17	72.33	0.01	15.85	0.01	38.41	0.01	13.13	1.17	93.19
mc_10_30_2	0.16	6.98	0.16	37.07	0.16	7.23	0.16	29.17	0.16	7.23	0.16	28.42
mc_10_30_3	3.31	6.87	0.29	29.94	1.89	38.10	0.29	35.01	5.32	5.67	4.77	19.90
mc_10_30_5	1.36	46.21	0.06	62.98	1.14	69.69	0.06	109.02	4.94	67.98	0.77	65.99
Average	1.83	5.22	1.13	12.00	1.32	6.17	0.91	12.75	1.93	5.21	1.32	12.99

Table 10.29. The performance of the BS heuristic on the second group of the MCMWP instances.

Instance Name	Beam Width (W) and Evaluation Function Parameter (Ψ) Combination (Ψ,W)											
	(0,1)		(0,3)		(0.25,1)		(0.25,3)		(0.5,1)		(0.5,3)	
	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU	UB(%)	CPU
mc_10_100_2	0.42	19.82	0.42	34.40	0.42	18.73	0.42	22.14	0.42	19.45	0.42	22.83
mc_10_100_3	4.78	31.16	0.05	68.21	3.43	30.33	3.43	47.82	6.52	30.91	2.79	47.38
mc_10_100_5	1.37	61.73	1.15	151.55	1.37	60.47	1.37	107.33	1.81	61.21	2.29	83.87
mc_15_150_2	8.59	98.99	4.39	236.48	4.53	86.80	5.29	214.95	9.08	94.48	5.29	177.00
mc_15_150_3	2.55	135.37	2.55	244.93	0.07	122.97	4.80	158.77	6.19	138.88	4.80	201.75
mc_15_150_5	1.79	235.02	1.39	252.78	0.07	240.15	3.65	251.58	2.93	240.22	3.65	250.78
mc_20_100_2	6.16	77.60	6.99	174.90	15.50	75.78	3.39	159.99	7.09	86.68	9.79	101.87
mc_20_100_3	6.83	122.54	10.89	243.89	14.84	128.08	10.96	245.36	14.71	139.35	10.96	244.84
mc_20_100_5	4.21	192.84	5.43	241.19	5.46	208.72	5.40	242.37	8.07	229.34	5.40	240.21
mc_30_100_2	42.40	132.02	36.62	242.95	50.68	134.31	35.97	240.36	77.85	149.23	27.69	241.73
mc_30_100_3	11.04	229.46	17.97	247.47	22.50	240.04	22.74	243.89	30.75	244.37	5.19	242.98
mc_30_100_5	0.31	252.73	3.83	241.04	3.94	248.10	3.94	259.69	3.21	259.11	3.94	245.73
mc_30_150_2	3.80	240.69	1.73	250.50	5.66	241.18	9.55	248.59	5.66	240.29	0.17	243.70
mc_30_150_3	2.19	245.90	3.59	252.95	6.67	249.12	0.16	241.44	5.64	241.55	4.78	258.53
mc_30_150_5	5.74	269.31	10.46	246.55	7.77	250.42	6.86	257.44	7.69	275.30	6.86	246.80
mc_45_150_2	31.58	254.67	38.88	245.37	0.22	250.47	24.32	247.96	0.22	246.12	24.32	255.06
mc_45_150_3	10.16	266.30	10.16	258.39	24.62	247.91	35.94	266.24	24.62	259.35	24.62	256.76
mc_45_150_5	8.71	259.66	11.30	244.39	0.07	257.72	0.07	247.56	0.07	292.81	0.07	246.50
Average	8.48	173.66	9.32	215.44	9.32	171.74	9.90	205.75	11.81	180.48	7.95	200.46

11. CONCLUSIONS

In this dissertation, we address a multi-commodity and capacitated extension of the MLAP. In particular, we deal with a variant of the continuous MLAP, namely the MCMWP, that is new to the literature. We propose heuristics and exact solution methods for the MCMWP where the distances are measured using ℓ_r-norm with $1 \leq r < \infty$.

First, we suggest new location-allocation heuristics for the MCMWP. The location-allocation heuristics are basically the adaptations and enhancements of Cooper's alternate location-allocation and Luis *et al.*'s region rejection heuristics (Cooper, 1964; Luis *et al.*, 2009). Among them the C-MRR and the D-MCALA, which are straightforward generalizations of Cooper's ALA heuristic, perform the best. All MCALA, MRR and MDRR are randomly initialized heuristics which are very efficient. They may produce accurate solutions when they are performed many times. Nevertheless, their performance depends on the initial selection of the facility locations (or an initial feasible transportation problem). In practice, there may be a sequence of random locations which gives rise to a superior performance of one ALA heuristic over the others. That is to say, randomized ALA heuristics have an oscillating performance that the range between their best and worst case is quite large. Such a randomness is exploited to produce practical estimates on the objective value within a confidence interval framework in this work.

Second, Discrete Approximation (DA) heuristics are implemented for the MCMWP. They are inspired from the studies by (Hansen *et al.*, 1998) for the MWP and (Aras *et al.*, 2007) for the CMWP. DA heuristics reduce the location space into a finite number of candidate facility locations and require the optimum solution of a MILP problem. We incorporate the theoretical results by (Thisse *et al.*, 1984) on block norms and extended the applicability of the DA heuristics such that they can now be used to produce lower bounds, as well. Basically, we suggest two discretization strategies which select the candidate locations using the block norms and customer locations. In the experiments, we implement block norms based strategy for the ℓ_1 and ℓ_∞-norms. We observe that ℓ_∞-norm is a better choice than the ℓ_1-norm in computing the lower bounds; but the upper bounds obtained using the customer locations have been the most accurate ones. In particular, it can be expected that ℓ_1-norm produces better lower bounds than the ℓ_∞-norm when $1 < r < 2$. In fact, the relation $2^{(1-r)/r}\ell_1 \leq \ell_r$ becomes tighter as r approaches towards 1. On the other hand, ℓ_∞-norm is a

better lower bounding approximation for ℓ_r-norm when $r > 2$. Another, choice is to rotate the customer locations on the plane with an angle between 0^o and 45^o and test whether a better lower bounding block norm, which also has four fundamental directions as ℓ_1 and ℓ_∞-norms, can be found using the relations given in Chapter 5.

When comparing the ALA and DA heuristics, we can say that the latters generally perform better than the formers in terms of accuracy. However, their major weakness is the drastic CPU time requirement, especially for large instances. Therefore, keeping in mind this inconvenience, we devised specially tailored LR strategies in order to increase the efficiency of the discrete approximation procedures. According to our computational experiments, we can say that among all relaxed discrete approximation heuristics, CL-RMDA2 produces the most accurate upper bounds within reasonable CPU times. Considering the trade-off between the accuracy and efficiency we can recommend ℓ_1-RMDA1 or ℓ_∞-RMDA1 as lower bounding approaches. However, when accurate upper bounds are crucial we definitely recommend the CL-MDA1 at the expense of its inefficiency.

As the third heuristic approach, we propose a LR scheme and the MS algorithm for the MCMWP. The proposed LR scheme requires the solution of a variant of the well-known MWP, as a subproblem of the MCMWP. The LR subproblem was handled with two strategies: the CG procedure and the lower bounding block norm approach. In the first strategy, we suggest an equivalent SC problem formulation for the LR subproblem. Then the LP relaxation of the equivalent SC problem, namely SCLP, is solved by CG procedure. We examine two different approaches, which are successfully applied on the MWP by (Krau, 1997) and (Righini and Zaniboni, 2007), in solving the Pricing Subproblem (PS) of the CG procedure. In one approach, we solve a D.C. programming problem. In the other approach, we solve the WPLD to perform the CG procedure. In the second strategy to solve the LR subproblem, we propose using the block norm approach. For that purpose, we make use of an approximating MILP of the LR subproblem, namely the UDAP. Then, using the ℓ_1-norm and ℓ_∞-norm in the objective function of the UDAP, we propose two lower bounding approaches for the LR subproblem. In total, four lower bounding approaches are employed within the MS algorithm for the MCMWP. We can say that among all lower bounding approaches, handling the PS by solving the D.C. programming problem yields the most accurate lower bounds at the expense of its inefficiency. Unfortunately, the MS algorithm for the MCMWP, which employs any of these four approaches at each step, requires excessive CPU times even

for small MCMWP instance. When we compare the MS algorithm with the ALA and DA heuristics, the MS is very inefficient. Its accuracy is better than the ALA heuristics but worse than the DA heuristics. Although, the MS algorithm does not perform well on the MCMWP instances, it can be applied on other COPs for which the resulting subproblems can be solved by efficient heuristics.

In another approach we have applied the Fisher and Tippett's theorem to produce statistical estimates and confidence intervals on the optimal solution of the MCMWP. Initialized by random starting solutions, three approximate solution procedures (i.e., MCALA, MDA1 and MDRR) are devised for that purpose and they are combined with statistical estimation procedures. We see that both the sampling method and the sample size affect the efficiency of the confidence interval approach using EVT. Generally, the MRA sampling produces more reliable confidence intervals, smaller absolute gaps and greater number of covering intervals than the LLA does on test instances. In addition, a small sample size (i.e., $M = 20$) is often enough to obtain reasonable bounds with the Golden and Alt's approach on the optimum (or benchmark minimum) value for the MCMWP. It is also interesting to observe that an optimal (or good) initial assignment of random candidate locations to facilities (e.g., the MDA1 or MDRR heuristic) does not always produce better results than a totally random initialization of the facilities (e.g., MCALA heuristic). Indeed, although the MDA1 heuristic yields narrower confidence intervals than the others do, the MCALA and MDRR heuristics yield more reliable intervals, smaller absolute gaps and greater number of covering intervals than the MDA1 heuristic. In the overall, the MCALA can be a better choice for the MCMWP since it produces the largest number of covering intervals in all cases.

Exact solution methods are implemented for two continuous MLAPs: the CMWP and MCMWP. We develop two types of BB algorithm: One of them works on the allocation space and the other one works on the location space. The SABB algorithm is based on the study by (Sherali *et al.*, 2002) for the CMWP. We replicate (Sherali *et al.*, 2002)'s RLT based bounding procedure for the CMWP. Additionally, we embed our block norm based bounding procedures within the SABB algorithm and use several branching variable selection strategies with a DFS strategy. We can say that our SABB implementation using block norm based bounding procedures beats (Sherali *et al.*, 2002)'s algorithm using RLT based bounding procedure for the CMWP. For the MCMWP, the MABB algorithm is the first attempt to solve it exactly. We devise three lower bounding procedures: block norm and

RLT based lower bounding procedures which are tightened with pure location based lower bounds Z_{MSAS}. We follow a BFS strategy and investigate the performance of three different MBrSs (i.e., MBrS1, MBrS2 and MBrS3) for the MABB algorithm. The block norm based bounding procedures also produced better results than the ones produced by RLT based bounding procedures for the MCMWP.

The LBB algorithms partition the location space and, for all we know, it is the first time that such an algorithm is employed for continuous MLAPs. For both CMWP and MCMWP, we employ two lower bounding procedures: LP and block norm based lower bounding procedures. We pursue a BFS strategy and used a specially tailored branching strategy working on the location space. We also suggest a complete enumeration strategy (i.e., LBBCE) that can be used within the LBB algorithm. The LBBCE algorithm can be enhanced further with a parallel implementation to solve new subproblems. In general, the LBB algorithm shows a superior performance than the ABB algorithm in terms of both the efficiency and the accuracy.

Lastly, we focus on the BS heuristics employing the LBB algorithm. The generic BS heuristic is designed to solve discrete optimization problems such as sequencing and scheduling problems. For the CMWP and MCMWP, we adapt the BS heuristic within a continuous partitioning scheme (i.e., partitioning of the location space). Its performance is higher than the DA heuristics using LR schemes. Moreover, the BS heuristic is also suitable for very large instances on which the most accurate DA heuristic MDA1 fails to produce any bounds. We believe that the framework we use for the BS heuristic can also be extended to other optimization problems which requires a continuous partitioning.

As a further research direction, one can consider the MCMWP2 formulation and apply a MS algorithm by relaxing constraints given by Equation 2.10 and 2.11. The resulting Lagrange subproblem can be decomposed into a variant of K CMWPs. Then, the MS algorithm can yield both lower and upper bounds on the MCMWP. However, we should note that such a LR scheme and using MS algorithm may not be an efficient method regarding the inherent difficulty of solving the CMWP exactly. In case the constraints given by Equation 2.8 is also relaxed from the MCMWP2, then the LR subproblems are again MWP variants which can be solved by methods similar to the ones presented in Chapter 6.3.

In this work, we only consider the most primitive BB approach for the LBB algorithm.

Yet, another research direction may be the generation of valid inequalities which can be used within the LBB algorithm to improve the performance of the proposed lower bounding procedures.

To the best of our knowledge, the LBB algorithm is a novel approach for the MLAPs and its performance is promising. As an open research avenue, one can also develop the LBB algorithm for various location-allocation problems. As for example, there may be passage limitations over some regions on the customer's plane and/or some regions may be forbidden for locating a facility. In this case, for the LP based bounding procedure, the definition of the closest distance over a region changes. In other words, one should also consider the barriers or forbidden regions to calculate the lower bounding distance function. As far as we know, there does not exist such a single or multi-commodity capacitated continuous MLAP in the literature. Heuristic or exact solution methods deserve further research for these more restricted problems.

As another open research avenue, one can consider the situation where customer locations are randomly distributed. In this case, customer locations may have a bivariate probability distribution. The problem reduces to solving the MCMWP with the expected value of distances between customers and facilities. The studies by (Durmaz *et al.*, 2009) and (Altınel *et al.*, 2009) propose new ALA and DA heuristics for the CMWP with probabilistic customer locations (PCMWP). Similar ALA and DA heuristics can also be developed for the MCMWP with probabilistic customer locations (PMCMWP). However, an exact solution method for solving the PCMWP does not exist in the literature. Exact solution methods suggested in this dissertation can be adapted to the PCMWP and as well as to the PMCMWP in order to fill this gap in the literature.

Last but not least, it is possible to obtain various variants of location problems by substituting the allocation space with another set of constraints. In particular, the transportation constraints of the MCMWP can be replaced with the constraint sets of Minimum Spanning Tree Problem, Minimum Cost Network Flow Problem, Travelling Salesman Problem and Vehicle Routing Problem which result in Location-Minimum Spanning Tree Problem, Location-Network Flow Problem, Location-Travelling Salesman Problem and Location-Routing Problem, respectively. Each of these problems is computationally difficult to solve and also deserves special research interest.

APPENDIX A: BENCHMARK BOUNDS

Table A.1 and Table A.2 summarize the outcomes of the solution methods suggested for the MCMWP. In Table A.1 and Table A.2, the first column shows the instance names. Each of the remaining columns correspond to the outcome of a solution (both exact and approximate) method which are the best outcome produced by each solution method. For example, "BEST LB" and "BEST UB" under the columns dedicated to "ABB" algorithm stand for the best lower bound value and the best upper bound value produced by the MABB algorithm. Note that there are several lower bounding (i.e. RLT based, block norm based and MSAS) procedures and branching variable selection strategies (i.e., MBrS1, MBrS2 and MBrS3). The values reported here are selected as the best ones of the lower and upper bounds produced by the MABB algorithm with different bounding procedure and MBrS combinations.

Table A.1. Summary of the suggested solution procedures on the first class of the MCMWP instances and their benchmark bounds.

Instance	ALA	DA		MS		CI		ABB		LBB		BS	Benchmark	
Name	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST UB	LB	UB
mc_2_2_2	64	64	64	0	64	N/A	N/A	64	64	64	64	64	64	64
mc_2_2_3	464	464	464	450.2853	464	N/A	N/A	464	464	464	464	464	464	464
mc_2_2_5	100	100	100	48.80861	100	N/A	N/A	100	100	100	100	100	100	100
mc_2_4_2	631.898	585	631.8977	356.6235	631.8977	N/A	N/A	631.1207	631.8977	631.27	631.9	631.8977	631.27	631.8977
mc_2_4_3	489.588	430	489.5879	440.8572	489.5879	N/A	N/A	489.0871	489.5879	489.1	489.59	489.5879	489.1	489.5879
mc_2_4_5	1733.84	1605	1733.84	1605.532	1733.844	N/A	N/A	1732.105	1733.844	1732.11	1733.84	1733.844	1732.11	1733.84
mc_3_5_2	1124.37	1016.82	1124.37	838.4189	1124.373	N/A	N/A	1123.226	1124.373	1123.25	1124.37	1124.373	1123.25	1124.37
mc_3_5_3	428.393	404.47	428.39	137.077	428.3926	N/A	N/A	427.9642	428.3926	427.96	428.39	428.3926	427.9642	428.39
mc_3_5_5	3970.19	3589	3970.185	1856.29	3970.185	N/A	N/A	3966.213	3970.185	3966.21	3970.19	3970.185	3966.213	3970.185
mc_3_9_2	1506	1284.81	1506	1236.852	1506.002	N/A	N/A	1484	1506.002	1504.5	1506	1506.002	1504.5	1506
mc_3_9_3	4919.41	4468	4919.41	3865.991	4919.413	N/A	N/A	4587.098	4919.413	4914.49	4919.41	4919.413	4914.49	4919.41
mc_3_9_5	52770.7	44339.8	52770.66	47553.07	52770.65	N/A	N/A	43814.63	52770.65	52299.49	52770.66	52770.66	52299.49	52770.65
mc_4_8_2	1179.55	1107.33	1179.55	535.2725	1179.554	1179.14	1179.55	1178.374	1179.554	1178.37	1179.55	1179.554	1179.14	1179.55
mc_4_8_3	8228.35	7616	8228.35	5494.52	8228.352	8220.12	8228.35	7801	8228.352	8220.12	8228.35	8228.352	8220.12	8228.35
mc_4_8_5	7741.16	7070	7741.16	5156.978	7741.161	7737.652	7741.16	7120.278	7741.161	7733.42	7741.16	7741.161	7737.652	7741.16
mc_4_10_2	5600.25	5227	5557.59	3558.39	5600.247	5486.66	5689.31	5505.096	5557.593	5552.04	5557.59	5557.594	5552.04	5557.59
mc_4_10_3	9934.84	9349	9934.84	7858.456	9934.843	8714.865	9934.846	9537.72	9934.843	9891.81	9934.84	9934.843	9891.81	9934.84
mc_4_10_5	23488.9	21144	23488.86	14498.62	23488.86	23480.99	24475.26	21434.77	23488.86	23465.37	23488.86	23488.86	23480.99	23488.86
mc_4_15_2	9528.17	8785.09	9528.168	7272.244	9528.168	9518.64	9528.169	8252.095	9528.168	9518.64	9528.17	9528.168	9518.64	9528.168
mc_4_15_3	28897.3	25560	28890.78	22446.2	28890.78	28711.79	28890.78	25805	28890.78	27029.41	28890.78	28890.78	28711.79	28890.78
mc_4_15_5	34120.5	31047.5	34120.48	26889.28	34120.48	34117.05	34120.48	31368.05	34120.48	32089.12	34120.48	34120.48	34117.05	34120.48
mc_5_8_2	4859.95	4419.42	4859.947	3137.943	4859.947	N/A	N/A	4855.083	4859.947	4855.09	4859.95	4859.947	4855.09	4859.947
mc_5_8_3	4453.75	3925.15	4070.027	1299.091	4479.654	N/A	N/A	3536.545	4070.027	4065.96	4070.03	4070.027	4065.96	4070.027
mc_5_8_5	24739.5	22169.9	24739.47	8458.094	25213.86	N/A	N/A	18428.14	24739.47	23159.03	24739.47	24739.47	23159.03	24739.47
mc_5_10_2	3145.64	2879	3145.638	1855.855	3145.638	2472.82	3800.37	3126.498	3145.638	3142.49	3145.64	3145.638	3142.49	3145.638
mc_5_10_3	8095.29	7116.32	8024.09	3858.914	8024.09	7476.604	8095.293	7054.066	8024.09	8016.06	8024.09	8024.09	8016.06	8024.09
mc_5_10_5	43438.3	39810.1	43438.3	24329.58	43438.31	43314.11	43438.83	35481.61	43438.31	37031.82	43438.31	43438.31	43314.11	43438.3
mc_5_12_2	3219.87	3051.87	3219.866	1615.246	3219.866	N/A	N/A	2785.411	3219.866	3113.91	3219.87	3219.866	3113.91	3219.866
mc_5_12_3	11810.5	9862.02	10831.77	6877.035	10831.77	N/A	N/A	9589.713	10831.77	10795.08	10831.77	10831.77	10795.08	10831.77
mc_5_12_5	35949.1	32305.6	35949.09	17466.72	35953.21	N/A	N/A	29412.9	35949.09	31213.87	35949.09	35949.09	32305.6	35949.09

Table A.1. Summary of the suggested solution procedures on the first class of the MCMWP instances and their benchmark bounds cont.

Instance	ALA	DA		MS		CI		ABB		LBB		BS	Benchmark	
Name	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST UB	LB	UB
mc_5_15_2	6582.82	6014	6582.82	5229.834	6582.823	6582.335	6582.82	6189.464	6582.823	6340.82	6582.82	6582.824	6582.335	6582.82
mc_5_15_3	25605.1	23216.4	25605.15	19330.3	25605.15	25595.63	26949.93	22813.82	25605.15	24493.78	25605.15	25605.15	25595.63	25605.1
mc_5_15_5	18966.8	17102.1	18966.75	14253.91	18966.75	18100.53	18966.76	16608.05	18966.75	17121.84	18966.75	18966.75	18100.53	18966.75
mc_5_20_2	8378.72	7408	8378.717	6676.153	8378.716	8267	8378.717	7642.835	8378.716	7699.04	8378.72	8378.717	8267	8378.716
mc_5_20_3	17071.9	14974	17024	14530.35	17024.01	16682.84	17693	14986.25	17024.01	15115.12	17024.01	17024.01	16682.84	17024
mc_5_20_5	46980.5	41754.7	46217.85	36140.39	46280.78	46210.19	48263.9	41041.07	46217.85	42275.93	46217.85	46217.85	46210.19	46217.85
mc_5_30_2	49392.4	42754	47042.64	42017.43	47176.39	47041.89	51385.1	42843	47042.64	43455.59	47042.64	47042.64	47041.89	47042.64
mc_5_30_3	65297.2	58409	64248.36	56244.24	64248.36	64247.67	65454.87	58408.14	64248.36	59491.82	64248.36	64248.36	64247.67	64248.36
mc_5_30_5	181701	162127	181502.4	154644.1	181502.4	181477.3	181701	162258.1	181502.4	164180.2	181502.39	181502.4	181477.3	181502.4
mc_6_10_2	2637.79	2263	2510.71	1626.095	2635.321	2496.466	2637.79	2347	2510.711	2507.99	2510.71	2510.711	2507.99	2510.71
mc_6_10_3	5108.09	4651	5108.086	1888.098	5108.086	5103.516	5108.087	4672.39	5108.086	4619.88	5108.09	5108.086	5103.516	5108.086
mc_6_10_5	9614.58	8530	9347.91	3854.594	9347.915	9337.35	9614.581	7992.322	9347.915	7957.67	9347.91	9347.915	9337.35	9347.91
mc_8_10_2	6564.35	5245	5730.92	398.0166	6424.68	5728.04	6466.868	5286.988	5730.92	5089.56	5730.92	5730.92	5728.04	5730.92
mc_8_10_3	9952.76	8488	9029.16	325.8833	10385.7	8996.271	10009.4	7829.397	9029.16	7585.08	9029.16	9029.162	8996.271	9029.16
mc_8_10_5	19505.5	17084	18319.8	4498.973	18729.77	18319.44	21795.95	15965.25	18319.85	15844.16	18319.85	18319.85	18319.44	18319.8
mc_10_10_2	4895.11	3142	3263.65	24	4412.07	3263.54	4684.837	3030.522	3263.65	2776.04	3263.65	3263.65	3263.54	3263.65
mc_10_10_3	11762.6	9938	11086.63	535.6398	11183.29	11086.49	12708	7600.166	11086.62	8278.27	11086.63	11086.63	11086.49	11086.62
mc_10_10_5	8733.59	4647	4907.52	50.70326	7024.206	4906.801	8720.206	4531.905	4907.523	4062.64	4907.52	4907.523	4906.801	4907.52
mc_10_15_2	3641.61	2181	2381.59	258.2762	3477.162	2378.319	4243.635	2110.729	2381.592	2016.25	2381.59	2488.917	2378.319	2381.59
mc_10_15_3	7378.53	5653	6131.147	2231.295	6224.859	6130.627	7365.24	5562.895	6131.147	5564.67	6131.15	6224.859	6130.627	6131.147
mc_10_15_5	12777.6	9357.144	10321.72	3391.331	12036.27	10301.2	13909.46	8517.1	10321.72	8793.23	10321.72	10384.74	10301.2	10321.72
mc_10_20_2	11579.7	7545	8320.806	4553.735	10158.82	8301.803	13216.85	7292.662	8320.805	7239.23	8320.81	8320.806	8301.803	8320.805
mc_10_20_3	6516.99	3886.26	4415.93	1703.515	5128.521	4415.304	6192.174	3841.124	4415.931	3785.41	4415.93	4608.669	4415.304	4415.93
mc_10_20_5	17636.9	10873.51	14375.29	6874.268	15362.7	14354.55	17028.77	12228.99	14385.49	12469.28	14375.29	14385.49	14354.55	14375.29
mc_10_24_2	4016	2957.121	3277.326	2045.24	3530.569	3261.975	4436.01	2761.297	3277.326	2745.57	3277.33	3293.693	3261.975	3277.326
mc_10_24_3	10168.3	6703.4	8700.496	5649.953	9346.457	8699.385	10905.31	7187.063	8700.496	7069.76	8700.5	8700.496	8699.385	8700.496
mc_10_24_5	25367.8	16175.07	19908.36	12291.08	20187.2	19906.64	25909.67	17297.13	19908.36	17441.3	19908.36	19908.36	19906.64	19908.36
mc_10_30_2	20886.9	15084.71	17708.64	8939.518	17708.64	17679.59	22587.5	16142.52	17708.64	15807.07	17708.64	17708.64	17679.59	17708.64
mc_10_30_3	49210.2	32497.05	42551.91	26226.83	43881.58	42485.22	49911	36954.2	42551.91	37606.72	42551.91	42607.62	42485.22	42551.91
mc_10_30_5	84381.7	54597.64	68156.19	44344.18	69971.12	68116.57	86136.6	60053.76	68156.19	61090.84	68156.19	68156.19	68116.57	68156.19
Average	18149.1	15100.91	17104.55	11696.27	17401.86	20712.02	22716.34	15052.53	17104.72	15638.55	17104.548	17113.53	16972.98	17104.54

Table A.2. Summary of the suggested solution procedures on the second class of the MCMWP instances and their benchmark bounds.

Instance	ALA	DA		CI		BS	Benchmark	
Name	BEST UB	BEST LB	BEST UB	BEST LB	BEST UB	BEST UB	LB	UB
mc_10_100_2	237180	197337.16	225647.69	224695.33	246927.00	225647.7	224695.3	225647.7
mc_10_100_3	247016	197718.19	227102.00	226995.75	264732.00	227102.2	226995.8	227102
mc_10_100_5	386244	295577.80	357735.00	357299.40	385787.00	361403.1	357299.4	357735
mc_15_150_2	215326	141341.53	170582.00	168775.91	215734.00	176182	168775.9	170582
mc_15_150_3	168423	115195.74	149311.82	140021.75	171931.00	140115	140021.8	140115
mc_15_150_5	292748	198569.43	257198.43	251617.22	291110.00	251795.3	251617.2	251795.3
mc_20_100_2	52010	30109.04	36813.80	36630.63	54480.30	37870.92	36630.63	36813.8
mc_20_100_3	86471.1	42492.48	53520.20	53440.11	84200.70	57090.62	53440.11	53520.2
mc_20_100_5	119036	68957.65	96918.90	96878.90	115660.00	100961.9	96878.9	96918.9
mc_30_100_2	15243	3683.67	5207.59	5192.13	15584.80	6629.754	5192.13	5207.59
mc_30_100_3	44752.4	17117.85	25757.00	25749.63	45963.80	27085.65	25749.63	25757
mc_30_100_5	78390.8	29003.41	64329.48	61511.87	81909.50	61702.69	61511.87	61702.69
mc_30_150_2	41698.5	16553.29	28245.20	27951.81	41084.60	28000	27951.81	28000
mc_30_150_3	65694.6	29363.53	46934.20	45307.56	68518.02	45378.71	45307.56	45378.71
mc_30_150_5	139965	68216.39	108783.57	108782.00	135681.00	115030	108782	108783.6
mc_45_150_2	8566.1	347.72	4644.20	3384.07	9414.90	3391.515	3384.07	3391.515
mc_45_150_3	20507.4	3495.28	12189.19	12176.40	19427.50	13413.55	12176.4	12189.19
mc_45_150_5	63529.8	14448.57	40783.49	40766.99	65431.30	40794.85	40766.99	40783.49
Average	126822.3	81640.48	106205.76	104843.19	128532.08	106644.2	104843.2	105079.1

APPENDIX B: CONFIDENCE INTERVALS

Now we present the detailed computational experiments which are summarized in Chapter 7 with Table 10.12–15. Confidence intervals obtained with the MCALA, MDA1 and MDRR heuristics on the RMCMWP instances and the MCMWP (with Euclidean distance) instances are presented in Tables B.1–8, respectively. In Table B.1–8, the first columns include the labels of the test instances. The number of distinct local optima produced by each heuristic, i.e., MCALA, MDA1 and MDRR, is shown in the second, third and fourth columns, respectively. The letter `F` indicates that the sample has failed to pass the independence test and the symbol "N/A" is used to indicate samples which do not fit the Weibull distribution. In Table B.1–8, the extreme value sample sizes, which are selected as $M = 20$, $M = 30$ and $M = 40$, are presented in the fifth column. Confidence intervals obtained with the MCALA, MDA1 and MDRR heuristics using the MRA and LLA samplings are given in the remaining columns. The cells representing large instances which can not be solved because of excessive CPU time requirements are marked with "N/A".

Table B.1. Confidence intervals obtained with random MCALA heuristic on a subset of 30 MCMWP instances with the rectilinear distance.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_4_8_2	219	20	[1565.29,1924.01]	[1562.26,1744]
		30	[1563.67,1963]	[1562.26,1744]
		40	[1563.53,1963]	[1562.26,1744]
mc_4_8_3	342	20	[9862.98,13428]	[9867.69,10306]
		30	[9529.92,13428]	[9867.69,10306]
		40	[8981.42,12576]	[9867.69,10306]
mc_4_8_5	1188	20	[9411.99,11007]	[9237.59,10201]
		30	[9411.99,11007]	[9384.8,10201]
		40	[9257.22,11007]	[9229.69,10201]
mc_4_10_2	397	20	[6433.67,8891]	[6429.16,6842]
		30	[6342.2,8775]	[6429.16,6842]
		40	[6237.79,8775]	[6429.16,6842]
mc_4_10_3	381	20	[11498.92,15190]	[10858.39,12610]
		30	[11458.65,14349]	[11349.39,12610]
		40	[11458.65,14349]	[11349.39,12610]
mc_4_10_5	945	20	[28130.25,31909]	[28139.67,29333]
		30	[27264.73,32273]	[28139.67,29333]
		40	[27122.59,32202]	[28048.42,29335]
mc_4_15_2	646	20	[12400.27,17051]	[12399.1,15164]
		30	[12412.26,16823]	[11967.31,14695]
		40	[12341.52,16971]	[11716.31,14695]
mc_4_15_3	2640	20	[35786.78,43448]	[35699.69,39314]
		30	[35782.7,42889]	[34023.82,37709]
		40	[35788.72,43203]	[32716.95,37048]
mc_4_15_5	4306	20	[41657.9,50190]	[41662.68,45317]
		30	[41277.98,52850]	[41679.36,44609]
		40	[39363.29,48528]	[41662.68,45317]
mc_5_10_2	437	20	[3925.4,5764]	[3405.41,4592]
		30	[3918.31,5288]	[2406.03,3974]
		40	[3913.29,5739]	[2406.03,3974]
mc_5_10_3	1084	20	[10060.58,17054]	[9979.54,16306]
		30	[10062.11,18984]	[9912.56,16205]
		40	[10062.86,18429]	[9300.92,15894]
mc_5_10_5	1371	20	[56190.36,65581]	[56258.84,57337]
		30	[52787.9,63101]	[56248.62,57385]
		40	[49379.44,60557]	[55906.16,56991]
mc_5_15_2	552	20	[7782.34,9353]	[7776.22,7784]
		30	[7758.03,9353]	[7776.22,7784]
		40	[7781.31,9379]	[7776.22,7784]
mc_5_15_3	7064	20	[32831.31,44312]	[32829.38,38366]
		30	[32769.33,44083]	[32823.63,38366]
		40	[32794.94,46280]	[32723.58,37421]
mc_5_15_5	6257	20	[24177.85,36344]	[24183.45,29562]
		30	[24153.42,34396]	[24184.22,29398]
		40	[24185.21,36326]	[24181.14,29534]

Table B.1. Confidence intervals obtained with random MCALA heuristic on a subset of 30 MCMWP instances with the rectilinear distance cont.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_5_20_2	2280	20	[9800.98,14284]	[9808,10997]
		30	[9770.22,14263]	[9768.53,11466]
		40	[9775.81,15192]	[9590,10997]
mc_5_20_3	1896	20	[20454.21,23193]	[20459.62,21714]
		30	[20452.9,22103]	[20282.43,21714]
		40	[20338.28,22634]	[20385.15,21714]
mc_5_20_5	6802	20	[58843.13,69522]	[58844.78,64354]
		30	[58844.78,67501]	[58844.28,63658]
		40	[58840.98,67898]	[58845.12,64342]
mc_5_30_2	7310	20	[56646.66,76477]	[56664.16,64098]
		30	[56654.47,75867]	[56664.55,64118]
		40	[56635.39,74826]	[56658.83,65021]
mc_5_30_3	11827	20	[78437.99,96930]	[78427.73,87451]
		30	[78438.41,97088]	[78442.32,88324]
		40	[78433.4,96977]	[78403.6,87451]
mc_5_30_5	12697	20	[224740.64,271616]	[224749.37,243052]
		30	[224735.98,266003]	[224749.26,244214]
		40	[224718.52,267429]	[224724.5,242228]
mc_6_10_2	367	20	[3069.65,4194]	[3025.3,3485]
		30	[3062.24,3756]	[2937.39,3358]
		40	[2960.24,3756]	[2945.21,3358]
mc_6_10_3	1295	20	[6417.45,7546]	[6426.56,7062]
		30	[6423.45,7546]	[6426.66,7062]
		40	[6370.45,7546]	[6426.66,7062]
mc_6_10_5	1443	20	[11455.3,14992]	[10777.84,12162]
		30	[11449.01,14992]	[10777.84,12162]
		40	[11454.02,14909]	[10726.84,12162]
mc_8_10_2	3090	20	[7003.59,11357]	[7003.51,8957]
		30	[7001.88,11011]	[7000.95,9047]
		40	[7001.02,11171]	[7003.37,9160]
mc_8_10_3	5353	20	[10418.69,18134]	[10418.46,14545]
		30	[10410.18,17493]	[10408.2,14781]
		40	[10415.86,18108]	[10393.63,14590]
mc_8_10_5	8970	20	[21287.16,32528]	[21287.87,27652]
		30	[21287.37,33607]	[21287.24,28325]
		40	[21287.15,33570]	[21287.98,28650]
mc_10_10_2	2587	20	[3600.53,7044]	[3600.76,5908]
		30	[3599.74,6793]	[3600.97,5940]
		40	[3599.81,6812]	[3600.82,5908]
mc_10_10_3	4026	20	[13563.27,17475]	[13563.29,16174]
		30	[13563.51,17475]	[13563.47,16174]
		40	[13563.65,17376]	[13563.91,15926]
mc_10_10_5	4020	20	[5387.62,11839]	[5387.1,10773]
		30	[5387.93,12335]	[5387.74,11076]
		40	[5389.99,12335]	[5380.66,11076]

Table B.2. Confidence intervals obtained with the randomized MDA1 heuristic on a subset of 30 MCMWP instances with the rectilinear distance.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_4_8_2	46	20	[1494.09,1913]	[1495.56,1708]
		30	[1494.09,1913]	[1397.34,1656]
		40	[1494.09,1913]	[1386.43,1566]
mc_4_8_3	38	20	[9883.68,10320]	[9913.08,9923]
		30	[9102.08,9923]	[9913.08,9923]
		40	[9102.08,9923]	[9913.08,9923]
mc_4_8_5	85	20	[9145.38,11135]	[9410.58,9420]
		30	[6946.58,9420]	[9410.58,9420]
		40	[6946.58,9420]	[9410.58,9420]
mc_4_10_2	47	20	[6029.98,7019]	[6379.61,6711]
		30	[5820.07,6914]	[6379.61,6711]
		40	[5676.16,6842]	[6396.29,6711]
mc_4_10_3	41	20	[9988.49,11507]	[11495.49,11507]
		30	[9988.49,11507]	[11495.49,11507]
		40	[9988.49,11507]	[11495.49,11507]
mc_4_10_5	88	20	[27169.5,29500]	[27765.3,28740]
		30	[25607.5,29500]	[27765.3,28740]
		40	[25607.5,29500]	[26755.86,28142]
mc_4_15_2	89	20	[9514.58,12424]	[12411.58,12424]
		30	[7265.58,12424]	[12411.58,12424]
		40	[7172.58,12424]	[12411.58,12424]
mc_4_15_3	98	20	[33880.09,39909]	[35625.88,36060]
		30	[32850.95,37048]	[35632.94,36060]
		40	[32490.78,36218]	[35632.94,36060]
mc_4_15_5	238	20	[41662.68,45317]	[41643.32,41685]
		30	[41662.68,45317]	[41643.32,41685]
		40	[41658.68,45317]	[41643.32,41685]
mc_5_10_2	14	20	[3622.03,3974]	N/A
		30	[3622.03,3974]	N/A
		40	[3622.03,3974]	N/A
mc_5_10_3	180	20	[9752.92,17076]	[7763.79,10215]
		30	[9179.99,17212]	[7967.33,10478]
		40	[4886.85,14463]	[7763.79,10215]
mc_5_10_5	113	20	[56205.25,59752]	[56225.64,58291]
		30	[55691.51,59495]	[56210.84,58291]
		40	[56093.25,59752]	[56179.33,58284]
mc_5_15_2	91	20	[7744.22,7784]	[7776.22,7784]
		30	[7744.22,7784]	[7776.22,7784]
		40	[7744.22,7784]	[7776.22,7784]
mc_5_15_3	284	20	[32793.33,35668]	[29965.17,32833]
		30	[32793.33,35668]	[29965.17,32833]
		40	[32793.33,35668]	[29965.17,32833]
mc_5_15_5	350	20	[23536.88,27713]	[23627.04,24504]
		30	[22021.94,25723]	[23424.89,24425]
		40	[21857.97,25723]	[23168.81,24186]

Table B.2. Confidence intervals obtained with the randomized MDA1 heuristic on a subset of 30 MCMWP instances with the rectilinear distance cont.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_5_20_2	294	20	[9759.4,13596]	[9300.85,10154]
		30	[9759.4,13596]	[9300.85,10154]
		40	[9779.33,13570]	[9117.92,10154]
mc_5_20_3	310	20	[20449.9,22103]	[20434.88,21463]
		30	[20449.9,22103]	[20434.88,21463]
		40	[20449.9,22103]	[20434.88,21463]
mc_5_20_5	239	20	[58840.23,63175]	[57508.94,61600]
		30	[58819.66,63519]	[57154.54,61700]
		40	[58833.92,63519]	N/A
mc_5_30_2	809	20	[56658.06,62940]	[56454.39,59609]
		30	[56658.06,62940]	[55713.15,59609]
		40	[56609.23,62637]	[56000.39,59609]
mc_5_30_3	1811	20	[78432.62,83348]	[78136.17,81504]
		30	[77923.16,83348]	[77762.88,81317]
		40	[77093.39,84615]	[78247.96,81504]
mc_5_30_5	433	20	[219956.04,224965]	[224740.04,224965]
		30	[219956.04,224965]	[224740.04,224965]
		40	[216217.04,224965]	[224740.04,224965]
mc_6_10_2	58	20	[3079.04,3693]	[2729.76,3521]
		30	[3079.04,3693]	[2729.76,3521]
		40	[3079.04,3693]	[2729.76,3521]
mc_6_10_3	84	20	[6421.78,7062]	[6426.15,6851]
		30	[6414.84,7070]	[6426.15,6851]
		40	[6423.22,7269]	[6426.15,6851]
mc_6_10_5	68	20	[11435.81,14414]	[11046.8,13585]
		30	[11018.63,14414]	[12277.05,12162]
		40	[10950.42,13585]	N/A
mc_8_10_2	531	20	[7002.84,9162]	[6997,7004]
		30	[6981.52,9235]	[6020,7004]
		40	[6835.24,9162]	[6020,7004]
mc_8_10_3	601	20	[10375.46,14730]	[10407.14,11938]
		30	[10351.51,13210]	[10391.87,11809]
		40	[10031.42,13581]	[10332.87,11809]
mc_8_10_5	1418	20	[21284.95,29756]	[21256.67,26026]
		30	[21284.67,29481]	[21229.97,26026]
		40	[21249.19,29481]	[21229.97,26026]
mc_10_10_2	953	20	[3598.68,6293]	[3593.98,5913]
		30	[3600.87,6749]	[3600.73,5476]
		40	[3599.32,6293]	[3600.44,5476]
mc_10_10_3	1248	20	[13563.74,16722]	[13546.38,14616]
		30	[13563.18,16340]	[13546.38,14616]
		40	[13556.3,16516]	[13535.38,14616]
mc_10_10_5	1439	20	[5389.25,10725]	[5383.48,8986]
		30	[5388.9,10896]	[5387.28,9061]
		40	[5389.37,10725]	[5389.41,8986]

Table B.3. Confidence intervals obtained with the randomized MDRR heuristic on a subset of 30 MCMWP instances with the rectilinear distance.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_4_8_2	218	20	[1564.32,1977]	[1474.43,1566]
		30	[1544.09,1913]	[1474.43,1566]
		40	[1552.46,1976]	[1386.43,1566]
mc_4_8_3	362	20	[9920.47,13748]	[8991.69,10306]
		30	[9917.86,13748]	[8991.69,10306]
		40	[9915.71,13735]	[8245.69,10306]
mc_4_8_5	1230	20	[9417,11002]	[9112.94,10065]
		30	[9411.99,11007]	[8984.94,10065]
		40	[9259.99,11007]	[8243.58,9420]
mc_4_10_2	364	20	[6422.79,9097]	[6427.32,6842]
		30	[5973.75,8413]	[6411.11,6894]
		40	[5593.09,7906]	[6411.11,6894]
mc_4_10_3	265	20	[11481.26,14736]	[11349.39,12610]
		30	[11482.3,14676]	[10858.39,12610]
		40	[11466.29,14676]	[11349.39,12610]
mc_4_10_5	684	20	[28099.07,31895]	[28114.28,30209]
		30	[28133.09,31909]	[27837.38,29335]
		40	[27264.73,32273]	[27837.67,29335]
mc_4_15_2	827	20	[12421.92,17030]	[12419.37,15112]
		30	[12410.74,16824]	[12420.7,15112]
		40	[12410.73,16824]	[12306.92,15112]
mc_4_15_3	1996	20	[35799.8,42203]	[35367.73,38343]
		30	[35754.98,43110]	[33443.59,37086]
		40	[35427.57,42031]	[33461.11,37474]
mc_4_15_5	4326	20	[41660.06,49018]	[41674.37,43998]
		30	[41231.06,47943]	[41664.01,44179]
		40	[40977.74,48926]	[41670.54,44241]
mc_5_10_2	134	20	[3922.47,5538]	[3469.46,5424]
		30	[3926.95,5538]	[3469.46,5424]
		40	[3926.26,5538]	[3408.31,4686]
mc_5_10_3	630	20	[10023.85,19387]	[9995.24,16804]
		30	[10052,18757]	[9394.71,16205]
		40	[10059.03,19266]	[9319.69,16306]
mc_5_10_5	822	20	[56276.24,64410]	[56292.74,58189]
		30	[54973.33,62681]	[56284.12,58352]
		40	[52787.9,63101]	[56273.43,58117]
mc_5_15_2	346	20	[7760.33,9376]	[7776.22,7784]
		30	[7720.77,9353]	[7776.22,7784]
		40	[7693.62,9379]	[7776.22,7784]
mc_5_15_3	5218	20	[32824.12,47174]	[32830.43,39224]
		30	[32832.14,46280]	[32822.88,39112]
		40	[32827.85,45835]	[32731.51,38490]
mc_5_15_5	3614	20	[24160.4,33600]	[24180.58,29163]
		30	[24164.24,33642]	[24178.68,28846]
		40	[24108.4,34148]	[24184.05,28909]

Table B.3. Confidence intervals obtained with the randomized MDRR heuristic on a subset of 30 MCMWP instances with the rectilinear distance cont.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_5_20_2	1796	20	[9812.99,15192]	[9811.35,12530]
		30	[9812.73,15214]	[9810.01,12687]
		40	[9810.81,15192]	[9809.12,12530]
mc_5_20_3	1175	20	[20455.02,22742]	[20448.64,21714]
		30	[20449.69,23193]	[20448.25,21714]
		40	[20441.26,22742]	[20453.17,21714]
mc_5_20_5	4862	20	[58843.12,69215]	[58839.86,63299]
		30	[58844.02,67744]	[58842.81,64047]
		40	[58844.21,67744]	[58844.02,64266]
mc_5_30_2	8564	20	[56661.93,76152]	[56664.55,66082]
		30	[56588.62,78123]	[56662.13,64118]
		40	[56656.87,76314]	[56621.76,64118]
mc_5_30_3	10158	20	[78429.61,93703]	[78441.45,86774]
		30	[78442.41,94737]	[78439.14,86340]
		40	[78437.23,94759]	[78395.94,87938]
mc_5_30_5	12161	20	[224737.76,270670]	[224746.43,243276]
		30	[224748.73,261621]	[224749.26,240742]
		40	[224380.53,260322]	[224728.57,239699]
mc_6_10_2	222	20	[3073.95,4050]	[3057.65,3485]
		30	[3073.95,4050]	[3069.36,3485]
		40	[3062.24,3756]	[3074.71,3485]
mc_6_10_3	520	20	[6426.28,8449]	[6415.77,7195]
		30	[6423.45,7546]	[6419.6,7062]
		40	[6220.82,7435]	[6418.37,7062]
mc_6_10_5	551	20	[11434.92,15373]	[11449.97,13799]
		30	[11364.08,15203]	[11401.46,13915]
		40	[11426.13,15203]	[11400.33,13778]
mc_8_10_2	950	20	[7002.66,11283]	[7001.34,9778]
		30	[7001.3,11057]	[7003.44,9710]
		40	[7003.7,11524]	[7003.24,9595]
mc_8_10_3	1107	20	[10411.73,16787]	[10341.2,13799]
		30	[10267.06,16787]	[10270.58,13799]
		40	[9547.08,19918]	[10341.2,13799]
mc_8_10_5	2144	20	[21283.67,34775]	[21281.83,28833]
		30	[21282.44,34465]	[21278.65,29082]
		40	[21282.47,33209]	[21282.23,29093]
mc_10_10_2	612	20	[3594.94,7311]	[3118.86,5987]
		30	[3600.22,7125]	[3598.15,5987]
		40	[3600.81,7083]	[3545.72,5987]
mc_10_10_3	799	20	[13561.55,17352]	[14379.97,17475]
		30	[13562.35,17241]	[15228.95,17475]
		40	[13559.69,17475]	[15681.04,17241]
mc_10_10_5	1652	20	[5386.03,12884]	[5341.41,11534]
		30	[5389.89,11534]	[5367.01,11534]
		40	[5383.73,12377]	[5328.87,11534]

Table B.4. Confidence intervals obtained with the randomized MCALA heuristic on a subset of 30 MCMWP instances.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_4_8_2	49	20	[1134.02,1364.08]	[1176.86,1271.11]
		30	[1112.31,1364.08]	[993.85,1179.55]
		40	[1134.02,1364.08]	[993.85,1179.55]
mc_4_8_3	75	20	[6121.65,9198.07]	N/A
		30	[5937.13,9106.36]	N/A
		40	[5938.32,9106.36]	N/A
mc_4_8_5	198	20	[7700.98,8746.15]	N/A
		30	[7705.85,8746.15]	N/A
		40	[7705.85,8746.15]	N/A
mc_4_10_2	76	20	[4286.73,5689.31]	[5210.65,5689.31]
		30	[5303.60,6900.90]	N/A
		40	[5303.60,6900.90]	N/A
mc_4_10_3	55	20	[8714.82,9934.85]	N/A
		30	[8714.82,9934.85]	N/A
		40	[8714.82,9934.85]	N/A
mc_4_10_5	129	20	[23480.99,25268.60]	[23401.87,24475.26]
		30	[23332.10,25075.31]	[23411.38,24475.26]
		40	[23480.99,25268.60]	[23306.78,24475.26]
mc_4_15_2	93	20	[4854.59,9528.17]	N/A
		30	[4854.59,9528.17]	N/A
		40	[4777.79,9528.17]	N/A
mc_4_15_3	150	20	[28173.72,30336.44]	[28014.78,28890.78]
		30	[28173.72,30336.44]	[28014.78,28890.78]
		40	[28166.56,30332.70]	[27751.82,28897.34]
mc_4_15_5	265	20	[33412.56,35754.25]	[32452.58,34120.48]
		30	[32229.38,35754.25]	[30872.23,34120.48]
		40	[31961.39,35754.25]	[30256.57,34120.48]
mc_5_10_2	64	20	[2388.56,3800.37]	N/A
		30	[2388.56,3800.37]	N/A
		40	[2388.56,3800.37]	N/A
mc_5_10_3	220	20	[7476.60,13886.28]	[6067.57,9981.92]
		30	[7031.16,13886.28]	[2296.21,8095.30]
		40	[7209.22,13753.12]	[2296.21,8095.29]
mc_5_10_5	159	20	[41973.50,44047.83]	[43314.11,43702.83]
		30	[41900.61,44047.83]	[42786.39,43438.83]
		40	[41654.70,44047.83]	[42436.76,43949.43]
mc_5_15_2	120	20	[5866.65,7319.68]	[6582.33,6954.48]
		30	[5866.64,7319.68]	[6512.07,6973.72]
		40	[5866.64,7319.68]	[6512.07,6973.72]
mc_5_15_3	680	20	[24071.87,29613.06]	[25289.65,26949.93]
		30	[25532.04,31593.52]	[25414.31,26949.93]
		40	[23937.67,30793.45]	[25098.08,26949.93]
mc_5_15_5	684	20	[17404.93,23458.29]	[18100.47,18966.76]
		30	[15008.48,22168.32]	[18100.47,18966.76]
		40	[17119.33,24214.73]	[17985.33,18966.76]

Table B.4. Confidence intervals obtained with the randomized MCALA heuristic on a subset of 30 MCMWP instances cont.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_5_20_2	268	20	[8063.66,10133.70]	[7760.66,8378.72]
		30	[7557.35,9895.20]	[7760.66,8378.72]
		40	[7787.39,10332.67]	[7760.66,8378.72]
mc_5_20_3	377	20	[16682.84,17692.99]	[17675.31,17692.99]
		30	[16468.55,17692.99]	[17675.31,17692.99]
		40	[16247.65,17692.99]	[17675.31,17692.99]
mc_5_20_5	371	20	[46190.16,50550.99]	N/A
		30	[46141.07,50550.99]	N/A
		40	[46210.19,50550.99]	N/A
mc_5_30_2	980	20	[46443.90,54240.05]	[47037.66,51385.11]
		30	[46964.09,55813.98]	[47038.08,51385.11]
		40	[44497.82,54009.85]	[47004.04,51385.11]
mc_5_30_3	2085	20	[64223.97,69068.36]	[63339.02,65454.87]
		30	[64234.22,70165.71]	[64228.57,66743.50]
		40	[63356.48,70165.71]	[63685.49,66411.84]
mc_5_30_5	586	20	[180207.82,192929.95]	[177883.44,183649.38]
		30	[177274.91,191226.71]	[175141.10,181700.98]
		40	[180957.76,195908.98]	[174290.43,181700.98]
mc_6_10_2	91	20	[2343.33,2887.30]	[2290.62,2637.79]
		30	[2496.47,2982.33]	[2290.62,2637.79]
		40	[2488.22,3060.64]	[2212.31,2637.79]
mc_6_10_3	254	20	[5103.52,5946.22]	[5037.55,5913.87]
		30	[5057.30,5913.87]	[3839.84,5108.09]
		40	[5057.30,5913.87]	[3839.84,5108.09]
mc_6_10_5	291	20	[8210.28,11315.34]	[8656.69,9614.58]
		30	[8016.40,11315.34]	[8656.84,9614.58]
		40	[7834.10,11224.70]	[7955.97,9640.47]
mc_8_10_2	886	20	[5595.51,7745.36]	[5721.14,6466.87]
		30	[5077.75,7225.74]	[5721.14,6466.87]
		40	[5387.18,7678.92]	[5701.53,6466.87]
mc_8_10_3	2095	20	[8774.10,12462.21]	[8890.02,11236.10]
		30	[8996.27,12853.06]	[8787.27,11236.10]
		40	[8961.19,12853.06]	[8772.59,11259.59]
mc_8_10_5	4300	20	[18228.09,23615.46]	[18293.39,21795.95]
		30	[18287.73,24122.19]	[18295.01,21795.95]
		40	[18152.27,24122.19]	[18244.59,21795.95]
mc_10_10_2	1110	20	[3214.51,5085.74]	[3262.86,4684.84]
		30	[3223.75,5085.74]	[3260.06,4684.84]
		40	[3242.20,5096.34]	[3255.44,4684.84]
mc_10_10_3	1585	20	[11079.78,13057.84]	[11085.08,12708]
		30	[11034.86,13057.84]	[11086.49,12708]
		40	[11014.52,13057.84]	[11086.49,12708]
mc_10_10_5	3381	20	[4850.43,8720.21]	[4802.69,8891.50]
		30	[4807.20,8720.21]	[5607.98,8720.21]
		40	[4906.80,8891.50]	[5546.58,8891.50]

Table B.5. Confidence intervals obtained with the randomized MDA1 heuristic on a subset of 30 MCMWP instances.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_4_8_2	25	20	[1179.14,1179.55]	[1178.37,1179.55]
		30	[1172.63,1472.32]	[1178.37,1179.55]
		40	[1150.92,1472.32]	[1178.37,1179.55]
mc_4_8_3	23	20	[7342.11,8228.35]	[8220.12,8228.35]
		30	[7342.11,8228.35]	[8220.12,8228.35]
		40	[5254.77,8228.35]	[8220.12,8228.35]
mc_4_8_5	34	20	[6112.61,7741.16]	[6394.87,7741.16]
		30	[6112.61,7741.16]	N/A
		40	[6112.61,7741.16]	N/A
mc_4_10_2	39	20	[5486.66,6988.11]	[5683.62,5689.31]
		30	[5313.62,7086.21]	[5683.62,5689.31]
		40	[4695.41,6590.59]	[5683.62,5689.31]
mc_4_10_3	7	20	[11133.76,11144.9]	F
		30	N/A	F
		40	N/A	F
mc_4_10_5	23	20	[24592.24,26085.1]	[22842.88,25445.9]
		30	[24602.04,26085.1]	[22842.88,25445.9]
		40	[22996.01,25445.9]	[22842.88,25445.9]
mc_4_15_2	37	20	[9518.64,9528.17]	N/A
		30	[4853.91,9528.17]	N/A
		40	[4261.61,9528.17]	N/A
mc_4_15_3	30	20	[28711.79,30014]	N/A
		30	[28711.79,30014]	N/A
		40	[28711.79,30014]	N/A
mc_4_15_5	35	20	[34099.44,35754.3]	[32452.58,34120.5]
		30	[34090.19,35754.3]	[32452.58,34120.5]
		40	[34099.44,35754.3]	[32452.58,34120.5]
mc_5_10_2	7	20	[2472.82,3800.37]	F
		30	[2472.82,3800.37]	F
		40	[2472.82,3800.37]	F
mc_5_10_3	35	20	[11121.29,17992.7]	[15213.78,17992.7]
		30	[11121.29,17992.7]	[14931.48,17992.7]
		40	[13043.56,17992.7]	[14923.98,17992.7]
mc_5_10_5	39	20	[41654.65,44047.8]	[43254.1,43702.8]
		30	[41654.65,44047.8]	[41973.45,44047.8]
		40	[45713.22,46078.1]	[41973.45,44047.8]
mc_5_15_2	9	20	[5872.16,6737.96]	[6421.1,6582.82]
		30	[5872.16,6737.96]	[5839.38,6582.82]
		40	[5872.16,6737.96]	[5839.38,6582.82]
mc_5_15_3	86	20	[25424.79,31891]	[25595.63,30266]
		30	[25517.59,31918]	[25595.63,30266]
		40	[25475.97,31891]	[25595.63,30266]
mc_5_15_5	97	20	[13348.63,22168.3]	[18020.6,20104.5]
		30	[13333.43,22160.7]	[17980.42,20084.4]
		40	[13103.83,22168.3]	[17980.42,20084.4]

Table B.5. Confidence intervals obtained with the randomized MDA1 heuristic on a subset of 30 MCMWP instances cont.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_5_20_2	56	20	[8196.58,10343.6]	[8267,9895.2]
		30	[8169.8,10343.6]	[6283.88,8903.14]
		40	[8196.58,10343.6]	[6283.88,8903.14]
mc_5_20_3	43	20	[17477.41,17693]	F
		30	[17477.41,17693]	F
		40	[17321.01,17693]	F
mc_5_20_5	21	20	[52739.18,52922]	F
		30	[52739.18,52922]	F
		40	[52869.08,52922]	F
mc_5_30_2	172	20	[47013.32,55829.9]	[47025.33,53375.8]
		30	[47013.32,55829.9]	[47038.04,53375.8]
		40	[47018.38,55261.9]	[47041.89,53375.8]
mc_5_30_3	385	20	[64228.06,71939.4]	[64210.87,68826.5]
		30	[64247.67,71939.4]	[64163.77,69213.2]
		40	[64229.6,71939.4]	[63633.8,67048.8]
mc_5_30_5	124	20	[181010.09,195909]	[181477.3,181701]
		30	[180888.09,195909]	[181477.3,181701]
		40	[177364.09,195909]	[181477.3,181701]
mc_6_10_2	36	20	[2480.95,3060.64]	[2825.83,2945.35]
		30	[2331.87,2982.33]	[2878.12,2945.35]
		40	[2324.78,2982.33]	[2800.48,2945.35]
mc_6_10_3	42	20	[4793.2,6070.78]	[4881.72,6070.78]
		30	[5172.54,6070.78]	[4881.72,6070.78]
		40	[5287.41,6070.78]	[4881.72,6070.78]
mc_6_10_5	39	20	[8518.33,11384.7]	[9337.35,9640.47]
		30	[5588.98,9919.2]	[9337.35,9640.47]
		40	[5588.98,9919.2]	[9337.35,9640.47]
mc_8_10_2	154	20	[5728.04,8126.1]	[5721.14,6466.87]
		30	[5724.97,8126.1]	[5721.14,6466.87]
		40	[5703.72,7910.56]	[5721.14,6466.87]
mc_8_10_3	166	20	[7945.32,12323.7]	[8835.76,11840.2]
		30	[8589.06,12682.5]	[8835.76,11840.2]
		40	[8926.15,12682.5]	[8835.76,11840.2]
mc_8_10_5	471	20	[18319.44,24764.6]	[18284.99,22209.7]
		30	[18318.81,24764.6]	[18308.63,22209.7]
		40	[18303.52,25265.2]	[18305.89,22209.7]
mc_10_10_2	136	20	[3263.54,5801.43]	[3172.12,4684.84]
		30	[3263.29,5801.43]	[3163.29,4684.84]
		40	[3263.29,5801.43]	[3163.29,4684.84]
mc_10_10_3	51	20	[10923.51,13622.8]	N/A
		30	[10859.22,14528.2]	N/A
		40	[11082.52,14186.2]	N/A
mc_10_10_5	312	20	[4890.37,9333.41]	[4862.75,9333.41]
		30	[4906.69,9333.41]	[4848.14,9544.64]
		40	[4906.69,9333.41]	[4570.47,9333.41]

Table B.6. Confidence intervals obtained with the randomized MDRR heuristic on a subset of 30 MCMWP instances.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_4_8_2	44	20	[1134.01592,1364.08]	[885.60045,1179.55]
		30	[1123.974497,1364.08]	[885.60045,1179.55]
		40	[1112.30592,1364.08]	[885.60045,1179.55]
mc_4_8_3	72	20	[8105.593086,10226.3]	[7528.75165,8228.35]
		30	[7955.710637,10226.3]	[7528.75165,8228.35]
		40	[8177.0737,10226.3]	[7528.75165,8228.35]
mc_4_8_5	178	20	[7737.651793,8746.15]	[6589.84884,7741.16]
		30	[7705.85385,8746.15]	[6589.84884,7741.16]
		40	[7703.049933,8746.15]	[6589.84884,7741.16]
mc_4_10_2	72	20	[4286.72069,5689.31]	N/A
		30	[5315.7191,6900.9]	N/A
		40	[5309.677634,6900.9]	N/A
mc_4_10_3	34	20	[8714.86515,9934.85]	N/A
		30	[8714.86515,9934.85]	N/A
		40	[8714.86515,9934.85]	N/A
mc_4_10_5	102	20	[23298.71736,24890.3]	[23406.6247,24475.3]
		30	[23275.87553,24890.3]	[23406.6247,24475.3]
		40	[23474.35614,25268.6]	[23406.6247,24475.3]
mc_4_15_2	109	20	[4853.91183,9528.17]	N/A
		30	[4777.81183,9528.17]	N/A
		40	N/A	N/A
mc_4_15_3	79	20	[28599.13303,30014]	N/A
		30	[26479.5027,28897.3]	N/A
		40	[26479.5027,28897.3]	N/A
mc_4_15_5	214	20	[34117.05237,35754.3]	N/A
		30	[33344.9457,35754.3]	N/A
		40	[30078.9795,34120.5]	N/A
mc_5_10_2	27	20	[2448.46963,3800.37]	[2472.81963,3800.37]
		30	[2388.55963,3800.37]	[2472.81963,3800.37]
		40	[2388.55963,3800.37]	N/A
mc_5_10_3	151	20	[1033.237,10663]	[2496.537,10663]
		30	[1033.237,10663]	[2496.537,10663]
		40	[4776.544597,13886.3]	[2496.537,10663]
mc_5_10_5	97	20	[42798.52355,44685.5]	[43248.2145,44685.5]
		30	[42788.55194,44685.5]	[43248.2145,44685.5]
		40	[41110.2062,43949.4]	[41973.4522,44047.8]
mc_5_15_2	42	20	[6421.05298,7597.02]	[5839.37718,6582.82]
		30	[6421.05298,7597.02]	[5839.37718,6582.82]
		40	[6421.05298,7597.02]	[5839.37718,6582.82]
mc_5_15_3	413	20	[23521.13411,28458.6]	[25314.6219,27978.1]
		30	[23560.04858,29750.5]	[25314.6219,27978.1]
		40	[23560.0495,29750.5]	[25314.6219,27978.1]
mc_5_15_5	228	20	[17440.0859,19814.1]	[18100.5332,18966.8]
		30	[15746.3332,18966.8]	[18100.5332,18966.8]
		40	[15746.3332,18966.8]	[18100.5332,18966.8]

Table B.6. Confidence intervals obtained with the randomized MDRR heuristic on a subset of 30 MCMWP instances cont.

Instance Names	Loc. Opt. Num.	Samp. Num.	Confidence Intervals	
			MRA	LLA
mc_5_20_2	158	20	[7290.193713,10343.6]	[7880.409068,10343.6]
		30	[7801.495764,10343.6]	[6385.8116,8988.4]
		40	[7289.0564,10343.6]	[8245.497873,10343.6]
mc_5_20_3	92	20	[16247.607,17693]	[17675.307,17693]
		30	[16247.607,17693]	[17675.307,17693]
		40	[16142.707,17693]	[17675.307,17693]
mc_5_20_5	175	20	[43557.5361,48263.9]	N/A
		30	[43557.5361,48263.9]	N/A
		40	[43557.5361,48263.9]	N/A
mc_5_30_2	676	20	[47033.16971,55907.1]	[46742.55052,51385.1]
		30	[46016.49484,55475.8]	[46811.7149,51385.1]
		40	[46757.31296,56238.5]	[46722.3221,52477.9]
mc_5_30_3	774	20	[64146.89011,70099.4]	[63685.3882,66411.8]
		30	[63508.12038,70099.4]	[63688.7882,66411.8]
		40	[63881.73561,71683.8]	[64239.58865,67406.6]
mc_5_30_5	330	20	[181416.2428,193256]	[177744.351,183649]
		30	[180680.07,192930]	[176314.351,183649]
		40	[180849.985,193015]	[176309.351,183649]
mc_6_10_2	58	20	[2414.46767,2982.33]	[2290.61221,2637.79]
		30	[2414.46767,2982.33]	[2212.30221,2637.79]
		40	[2481.31936,3060.64]	[2212.30221,2637.79]
mc_6_10_3	106	20	[5095.794286,5946.22]	[4577.228748,5946.22]
		30	[5094.23742,5946.22]	[4577.228748,5946.22]
		40	[5102.08784,5946.22]	[4577.228748,5946.22]
mc_6_10_5	79	20	[6879.319076,14264.1]	[8084.7323,11167.7]
		30	[9814.538537,14115.8]	[8084.7323,11167.7]
		40	[10984.25072,14239.5]	[8084.7323,11167.7]
mc_8_10_2	185	20	[5251.567179,7618.18]	[5661.541911,6848.78]
		30	[5727.211199,8298.13]	[5654.05122,6848.78]
		40	[5677.032806,8263.74]	[5654.05122,6848.78]
mc_8_10_3	304	20	[8224.5598,11840.2]	[8390.2906,10009.4]
		30	[8685.7035,12396.5]	[8390.2906,10009.4]
		40	[8685.7035,12396.5]	[8438.5877,10112.3]
mc_8_10_5	575	20	[18281.55069,24786.8]	[18248.92179,24230.3]
		30	[18234.51927,24786.8]	[18305.73619,24230.3]
		40	[18304.82261,24986.5]	[18255.30639,24230.3]
mc_10_10_2	212	20	[3234.907203,5963.98]	[3124.185158,5552.07]
		30	[3221.059398,5919.75]	[3124.185158,5552.07]
		40	[3218.989708,5919.75]	[3329.831315,5558.06]
mc_10_10_3	133	20	[11055.98513,14186.2]	[13232.2138,14186.2]
		30	[11055.98513,14186.2]	[12998.64538,14020.9]
		40	[11042.73932,14020.9]	[12864.34538,14020.9]
mc_10_10_5	695	20	[4802.241468,9133.5]	[4811.562205,9333.41]
		30	[4886.987501,9133.5]	[4679.846228,9333.41]
		40	[4888.649777,9133.5]	[4761.185833,9333.41]

Table B.7. Confidence intervals obtained on 12 MCMWP instances from mc_10_15_2 to mc_10_30_5.

Instance Name	Loc. Opt. Num.		Samp. Num.	CI (with MRA sampling)	
	MCALA	MDRR		MCALA	MDRR
mc_10_15_2	134	61	20	[2201.11,4243.64]	[2471.548467,4521.72]
			30	[2341.77,4430.20]	[2595.10828,4521.72]
			40	[2378.32,4330.61]	[2595.10828,4521.72]
mc_10_15_3	456	337	20	[6130.63,7612.19]	[6055.896894,7378.53]
			30	[6120.99,7612.19]	[5979.996731,7365.24]
			40	[6086.27,7757.54]	[5991.678961,7378.53]
mc_10_15_5	617	355	20	[10301.20,13909.46]	[10274.93385,14275.5]
			30	[10227.56,13975.56]	[10109.76066,14331.6]
			40	[9942.73,14048.50]	[9946.987447,14363.9]
mc_10_20_2	590	380	20	[8176.21,13216.85]	[7958.889685,13227.3]
			30	[8177.69,13251.39]	[8046.202456,13227.3]
			40	[8301.80,13216.85]	[8145.4727,13227.3]
mc_10_20_3	507	335	20	[4263.74,6242.70]	[4411.78621,6609.82]
			30	[4295.29,6372.97]	[4375.567393,6558.38]
			40	[3991.70,6192.17]	[4415.304393,6609.82]
mc_10_20_5	733	390	20	[13951.04,17028.77]	[13816.49277,17306.8]
			30	[14304.37,17583.93]	[14245.2456,17889.8]
			40	[14354.55,17786.20]	[14352.32642,18220.3]
mc_10_24_2	648	477	20	[3231.81,4962.32]	[3256.264229,4464.4]
			30	[3261.97,5148.39]	[3207.184942,4436.01]
			40	[3168.74,5082.50]	[3208.64399,4436.01]
mc_10_24_3	880	699	20	[8551.46,10950.88]	[8699.384793,11132.2]
			30	[8693.4811074.45]	[8586.190734,11074.5]
			40	[8467.05,10905.31]	[8575.591244,11157.3]
mc_10_24_5	876	706	20	[19265.54,25909.67]	[19711.87498,26936.9]
			30	[19899.25,26574.68]	[19875.57072,26936.9]
			40	[19842.96,26574.68]	[19906.6425,27480.6]
mc_10_30_2	260	244	20	[17546.44,22587.51]	[17655.3125,22587.5]
			30	[17314.06,22587.51]	[17655.3125,22587.5]
			40	[17679.59,23390.36]	[17655.3125,22587.5]
mc_10_30_3	763	729	20	[42126.96,50616.18]	[41701.289,49911]
			30	[41379.07,50238.58]	[41343.889,49911]
			40	[41430.19,50877.30]	[42485.22269,51175.8]
mc_10_30_5	865	820	20	[67173.91,86497.44]	[67997.90941,86136.6]
			30	[68116.57,86476.41]	[67955.49377,87000.5]
			40	[68040.91,87856.67]	[67920.78177,86918.1]

Table B.8. Confidence intervals on the second class of the MCMWP instances.

Instance Name	Loc. Opt. Num.		Samp. Num.	CI (with MRA sampling)	
	MCALA	MDRR		MCALA	MDRR
mc_10_100_2	946	563	20	[224043.34,260168.18]	[222068.37,246927]
			30	[224695.33,261102.63]	[224331,249732]
			40	[224647.70,261102.63]	[224236.95,249732]
mc_10_100_3	999	887	20	[226995.75,270249.7]	[226811.17,264732]
			30	[226064.30,269098.47]	[226909.31,264732]
			40	[226444.31,272687.21]	[226953.13,265733]
mc_10_100_5	997	853	20	[357299.4,400184.71]	[349779.37,385787]
			30	[352757.96,397675.32]	[355161.27,393595]
			40	[356920.28,401555.59]	[355200.97,393835]
mc_10_150_2	998	787	20	[167864.79,221988.53]	[167442.27,215734]
			30	[166923.87,221988.53]	[167802.18,215734]
			40	[168775.91,225404.42]	[168373.27,215734]
mc_10_150_3	1000	966	20	[139777.58,172152.05]	[139677.72,172996]
			30	[139115.06,172152.5]	[148171.70,171931]
			40	[140021.75,173145.38]	[148400.75,171931]
mc_10_150_5	1000	986	20	[250930.51,294147.39]	[251305.26,291110]
			30	[251617.22,294147.39]	[251353.26,291110]
			40	[249574.34,294147.39]	[250227.01,291110]
mc_20_100_2	1000	905	20	[36630.63,58771.89]	[36554.71,54480.3]
			30	[36504.43,58459.72]	[36624.72,54480.3]
			40	[36423.64,58459.72]	[36620.11,54480.3]
mc_20_100_3	1000	889	20	[53284.25,91099.35]	[53227.37,84200.7]
			30	[53420.3,91099.35]	[52194.72,84200.7]
			40	[53225.38,91099.35]	[53440.11,84200.7]
mc_20_100_5	1000	986	20	[96559.65,130934.92]	[96666.78,115660]
			30	[96846.10,130934.92]	[96722.39,115660]
			40	[96878.9,131415.87]	[96821.65,115660]

Table B.8. Confidence intervals on the second class of the MCMWP instances cont.

Instance Name	Loc. Opt. Num.		Samp. Num.	CI (with MRA sampling)	
	MCALA	MDRR		MCALA	MDRR
mc_30_100_2	999	943	20	[5189.67,16590.99]	[5171.22,15584.8]
			30	[5192.13,16590.99]	[4958.46,15584.8]
			40	[5192.08,16590.99]	[4822.78,15673.1]
mc_30_100_3	1000	996	20	[25714.23,53156.77]	[25725.09,45963.8]
			30	[25749.63,53156.77]	[25418.52,45963.8]
			40	[25742.12,53156.77]	[25731.12,45963.8]
mc_30_100_5	1000	992	20	[61053.18,94248.97]	[61469,81909.5]
			30	[61414.98,94248.97]	[61492.09,81909.5]
			40	[61511.87,94248.97]	[61506.09,81909.5]
mc_30_150_2	1000	985	20	[27854.02,54716.48]	[26355.38,41084.6]
			30	[27951.81,54716.48]	[26409.02,41084.6]
			40	[27893.84,54716.48]	[27677.8,42372]
mc_30_150_3	1000	998	20	[43550.80,68518.02]	[45282.42,70417.3]
			30	[45307.56,71376.43]	[44079.66,70507]
			40	[45301.03,71376.43]	[44343.28,70417.3]
mc_30_150_5	999	979	20	[108781.86,153897.60]	[108773.95,135681]
			30	[108683.26,153897.60]	[108715.86,135681]
			40	[108782,153897.60]	[108780.78,135681]
mc_45_150_2	1000	973	20	[3384.07,11912.79]	[3380.81,9432.88]
			30	[2664.92,11062.19]	[3381,9414.9]
			40	[3334.58,11818.27]	[3367.94,9414.9]
mc_45_150_3	1000	992	20	[10958.04,23024.34]	[12170.25,19427.5]
			30	[12143.38,24814.20]	[12168.42,19427.5]
			40	[12176.40,24814.20]	[12176.29,19427.5]
mc_45_150_5	999	999	20	[40494.54,72333.01]	[40693.068,65431.3]
			30	[40698.66,72333.01]	[40609.76,65431.3]
			40	[40766.99,72333.01]	[39126.28,66367.8]

REFERENCES

Akyüz, M. H., İ. K. Altınel, and T. Öncan, 2011, "Allocation and location space based branch and bound algorithms for the capacitated multi-facility Weber problem", In: A. C. Martínez, I. E. Miranda, B. L. Jiménez, J. P. Albandoz, C. González, A. M. R. Chía, and C. V. Franco (Eds.), *Proceedings of OR Peripatetic Post-Graduate Programme*, pp. 101-106, Cádiz, 13-17 September 2011, Servicio de Publicaciones de la Universidad de Cádiz, Spain.

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2008, "The multi-commodity capacitated multi-facility Weber problem: heuristics and confidence intervals", Technical Report FBE-IE-11/2008-12, Institute for Graduate Studies in Science and Engineering, Boğaziçi University.

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2009a, "Efficient lower and upper bounds for the multi-commodity capacitated multi-facility Weber problem with rectilinear distances", In: S. Voss, J. Pahl, and S. Schwarse (Eds.), *Logistik Management – Systeme Methoden Integration*, pp. 229-245, Hamburg, 2-4 September 2009, Physica Verlag, Germany.

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2009b, "The multi-commodity capacitated multi-facility Weber problem: heuristics and confidence intervals", In: S. I. Ao, O. Castillo, C. Douglas, D. D. Feng, and J.-A. Lee (Eds.), *Proceedings of the International MultiConference of Engineers and Computer Scientists*, pp. 2042-2047, Hong Kong, 18-20 March 2009, Newswood Limited, Hong Kong.

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2010a, "Efficient approximate solution methods for the multi-commodity capacitated multi-facility Weber problem", Technical Report FBE-IE-02/2010-03, Institute for Graduate Studies in Science and Engineering, Boğaziçi University.

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2010b, "Efficient lower bounding schemes for the multi-commodity capacitated multi-facility Weber problem", In: M. Collan (Ed.), *Proceedings of the* 2^{nd} *International Conference on Applied Operational Research*, pp. 470-485, Turku, 25-27 August 2010, TADBIR, Finland.

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2010c, "The multi-commodity capacitated multi-

facility Weber problem: heuristics and confidence intervals", *IIE Transactions*, Vol. 42, No. 11, pp. 825–841.

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2011, "Solving the multi-commodity capacitated multi-facility Weber problem using Lagrangean relaxation and a subgradient-like algorithm", *Journal of the Operational Research Society*, (doi:10.1057/jors.2011.81).

Akyüz, M. H., T. Öncan, and İ. K. Altınel, 2012, "Efficient approximate solution methods for the multi-commodity capacitated multi-facility Weber problem", *Computers and Operations Research*, Vol. 39, No. 2, pp. 225–237.

Al-Loughani, I., 1997, *Algorithmic approaches for solving the Euclidean distance location-allocation problems*, Ph.D. Thesis, Virginia Polytechnic Institute and State University.

Alpaydın, E., İ. K. Altınel, and N. Aras, 1996, "Parametric distance functions vs. non-parametric neural networks for estimating road travel distances", *European Journal of Operational Research*, Vol. 93, No. 2, pp. 230–243.

Altınel, İ. K., N. Aras, and B. J. Oommen, 2000, "Fast, efficient and accurate solutions to the Hamiltonian path problem using neural approaches", *Computers and Operations Research*, Vol. 27, No. 5, pp. 461–494.

Altınel, İ. K., E. Durmaz, N. Aras, and K. C. Özkısacık , 2009, "A location-allocation heuristic for the capacitated multi-facility Weber problem with probabilistic customer locations", *European Journal of Operational Research*, Vol. 198, No. 3, pp. 790–799.

Anderson, T. W. and D. A. Darling, 1952, "Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes", *The Annals of Mathematical Statistics*, Vol. 23, No. 2, pp. 193–212.

Aneja, Y. P. and M. Parlar, 1994, "Algorithms for Weber facility location in the presence of forbidden regions and/or barriers to travel", *Transportation Science*, Vol. 28, No. 1, pp. 70–76.

Aras, N., İ. K. Altınel, and M. Orbay, 2007, "New heuristic methods for the capacitated multi-facility Weber problem", *Naval Research Logistics*, Vol. 54, No. 1, pp. 21–32.

Aras, N., M. Orbay, and İ. K. Altınel, 2008, "Efficient heuristics for the rectilinear distance capacitated multi-facility Weber problem", *Journal of the Operational Research Society*, Vol. 59, No. 1, pp. 64–79.

Aras, N., S. Yumuşak, and İ. K. Altınel, 2006, "Solving the capacitated multi-facility Weber problem by simulated annealing, threshold accepting and genetic algorithms", In: K. F. Doerner, M. Gendreau, P. Greistorfer, W. J. Gutjahr, R. F. Hartl, and M. Reimann (Eds.), *Metaheuristics–Progress in Complex Systems Optimization*. pp. 91-112, Springer, New York.

Bazaraa, M. S., J. J. Jarvis, and H. D. Sherali, 2010, *Linear Programming and Network Flows*, 4^{th} edition, Wiley, New York.

Bazaraa, M. S., H. D. Sherali, and C. M. Shetty, 1993, *Nonlinear Programming: Theory and Algorithms*, 2^{nd} edition, John Wiley & Sons, New Jersey.

Beyer, W. H., 1974, *CRC Handbook of Tables for Probability and Statistics*, 2^{nd} edition, CRC Press Inc., Florida.

Boender, C. G. E., A. H. G. R. Kan, G. T. Timmer, and L. Stougie, 1982, "A stochastic method for global optimization", *Mathematical Programming*, Vol. 22, No. 2, pp. 125–140.

Bongartz, I., P. H. Calamai, and A. R. Conn, 1994, "A projection method for the ℓ_p norm location-allocation problems", *Mathematical Programming*, Vol. 66, No. 3, pp. 283–312.

Boyacı, B., 2009, *Solving the capacitated multifacility Weber problem approximately*, M.S. Thesis, Boğaziçi University, İstanbul.

Brandeau, M. L. and S. S. Chiu, 1993, "Sequential location and allocation: worst case performance and statistical estimation", *Location Science*, Vol. 1, No. 4, pp. 289–298.

Brimberg, J., R. Chen, and D. Chen, 1998, "Accelerating convergence in the Fermat-Weber location problem", *Operations Research Letters*, Vol. 22, No. 4, pp. 151–157.

Brimberg, J., P. Hansen, and N. Mladenović, 2006, "Decomposition strategies for large-scale continuous location-allocation problems", *IMA Journal of Management Mathematics*, Vol. 17, No. 4, pp. 307–316.

Brimberg, J., P. Hansen, N. Mladenović, and S. Salhi, 2008, "A survey of solution methods for the continuous location-allocation problem", *International Journal of Operations Research*, Vol. 5, No. 1, pp. 1–12.

Brimberg, J., P. Hansen, N. Mladenović, and E. Taillard, 2000, "Improvements and comparison of heuristics for solving the multisource Weber problem", *Operations Research*, Vol. 48, No. 3, pp. 444–460.

Brimberg, J. and R. F. Love, 1993, "Global convergence of a generalized iterative procedure for the minisum location problem with ℓ_p distances", *Operations Research*, Vol. 41, No. 6, pp. 1153–1163.

Brimberg, J. and R. F. Love, 1995, "Estimating the distances", In: Z. Drezner (Ed.), *Facility location: A survey of applications and methods*, pp. 9-32, Springer, Heidelberg.

Brimberg, J., J. H. Walker, and R. F. Love, 2007, "Estimation of travel distances with the weighted l_p norm: some empirical results", *Journal of Transport Geography*, Vol. 15, No. 1, pp. 62–72.

Bruijs, P. A., 1984, "On the quality of heuristic solutions to a 19×19 quadratic assignment problem", *European Journal of Operational Research*, Vol. 17, No. 1, pp. 21–30.

Canel, C., B. M. Khumawala, J. Law, and A. Loh, 2001, "An algorithm for the capacitated, multi-commodity multi-period facility location problem", *Computers and Operations Research*, Vol. 28, No. 5, pp. 411 – 427.

Chen, P.-C., P. Hansen, B. Jaumard, and H. Tuy, 1992, "Weber's problem with attraction and repulsion", *Journal of Regional Science*, Vol. 32, No. 4, pp. 467–486.

Chen, P.-C., P. Hansen, B. Jaumard, and H. Tuy, 1998, "Solution of the multisource Weber and conditional Weber problems by D.C. programming", *Operations Research*, Vol. 46, No. 4, pp. 548–562.

Chen, R., 1983, "Solution of minisum and minimax location-allocation problems with Euclidean distances", *Naval Research Logistics Quarterly*, Vol. 30, No. 3, pp. 449–459.

Clarke, G. and J. W. Wright, 1964, "Scheduling of vehicles from a central depot to a number of depots", *Operations Research*, Vol. 12, No. 4, pp. 568–581.

Cooke, P., 1979, "Statistical inference for bounds of random variables", *Biometrika*, Vol. 66, No. 2, pp. 367–374.

Cooper, L., 1963, "Location-allocation problems", *Operations Research*, Vol. 11, No. 3, pp. 331–343.

Cooper, L., 1964, "Heuristic methods for location-allocation problems", *SIAM Review*, Vol. 6, No. 1, pp. 37–53.

Cooper, L., 1967, "Solutions of generalized locational equilibrium models", *Journal of Regional Science*, Vol. 7, No. 1, pp. 1–18.

Cooper, L., 1972, "The transportation-location problem", *Operations Research*, Vol. 20, No. 1, pp. 94–108.

Cooper, L., 1976, "An efficient heuristic for the transportation-location problem", *Journal of Regional Science*, Vol. 16, No. 3, pp. 309–315.

CPLEX, 2007, ILOG CPLEX 11.0 User's Manual.

Croce, F. D., M. Ghirardi, and R. Tadei, 2004, "Recovering beam search: enhancing the beam search approach for combinatorial optimization problems", *Journal of Heuristics*, Vol. 10, No. 1, pp. 89–104.

Dannenbring, D. G., 1977, "Procedures for estimating optimal solution values for large combinatorial problems", *Management Science*, Vol. 23, No. 12, pp. 1272–1283.

Derigs, U., 1985, "Using confidence limits for the global optimum in combinatorial optimization", *Operations Research*, Vol. 33, No. 5, pp. 1024–1049.

Drezner, Z., 1984, "The planar two-center two-median problems", *Transportation Science*, Vol. 18, No. 4, pp. 352–361.

Drezner, Z., T. Drezner, and G. O. Wesolowsky, 2009, "Location with acceleration-deceleration distance", *European Journal of Operational Research*, Vol. 198, No. 1, pp. 157–164.

Drezner, Z., K. Klamroth, A. Schöbel, and G. O. Weselowsky, 2002, "The Weber problem", In: Z. Drezner and H. W. Hamacher (Eds.), *Facility location: applications and theory*, pp. 1-36, Springer, Heidelberg.

Drezner, Z., A. Mehrez, and G. O. Wesolowsky, 1991, "The facility location problem with limited distances", *Transportation Science*, Vol. 25, No. 3, pp. 183–187.

Drezner, Z. and A. Suzuki, 2004, "The big triangle small triangle method for the solution of nonconvex facility location problems", *Operations Research*, Vol. 52, No. 1, pp. 128–135.

Drezner, Z., G. O. Wesolowsky, and T. Drezner, 2004, "The gradual covering problem", *Naval Research Logistics*, Vol. 51, No. 6, pp. 841–855.

Drezner, Z., G. O. Wesolowsky, and T. Drezner, 2007, "A general global optimization approach for solving location problems in the plane", *Journal of Global Optimization*, Vol. 37, No. 2, pp. 305–319.

Dubey, S. D., 1967, "Some percentile estimators for Weibull parameters", *Technometrics*, Vol. 9, No. 1, pp. 119–129.

Durier, R. and C. Michelot, 1994, "On the set of optimal points to the Weber problem: further results", *Transportation Science*, Vol. 28, No. 2, pp. 141–149.

Durmaz, E., N. Aras, and İ. K. Altınel, 2009, "Discrete approximation heuristics for the capacitated continuous location-allocation problem with probabilistic customer locations", *Computers and Operations Research*, Vol. 36, No. 7, pp. 2139 – 2148.

Eilon, S., C. D. T. Watson-Gandy, and N. Christofides, 1971, *Distribution management: mathematical modelling and practical analysis*, Charles Griffin & Co. Ltd., London.

Engelhardt, M. and L. J. Bain, 1977, "Simplified statistical procedures for the Weibull or extreme-value distribution", *Technometrics*, Vol. 19, No. 3, pp. 323–331.

Fisher, R. A. and L. H. C. Tippett, 1928, "limiting forms of the frequency distribution of the largest or smallest member of a sample", In: *Proceedings of the Cambridge Philosophical Society,* Vol. 24, pp. 180-190.

Fliege, J. and S. Nickel, 2000, "An interior point method for multifacility location problems with forbidden regions", *Studies in Locational Analysis*, Vol. 14, pp. 23–46.

Francis, R. L., L. F. McGinnis Jr., and J. A. White, 1992, *Facility layout and location: An analytical approach*, 2^{nd} edition, International Series in Industrial and Systems Engineering, Prentice Hall, New Jersey.

Frenk, J. B. G., M. T. Melo, and S. Zhang, 1994, "A Weiszfeld method for a generalized ℓ_p distance minisum location model in continuous space", *Location Science*, Vol. 2, No. 2, pp. 111–127.

Gamal, M. D. H. and S. Salhi, 2001, "Constructive heuristics for the uncapacitated continuous location-allocation problem", *Journal of the Operational Research Society*, Vol. 52, No. 7, pp. 821–829.

Gamal, M. D. H. and S. Salhi, 2003, "A cellular heuristic for the multisource Weber problem", *Computers and Operations Research*, Vol. 30, No. 11, pp. 1609–1624.

Ganesan, V. K. and A. I. Sivakumar, 2006, "Scheduling in static jobshops for minimizing mean flowtime subject to minimum total deviation of job completion times", *International Journal of Production Economics*, Vol. 13, No. 2, pp. 5633–647.

Gendron, B., J. Potvin, and P. Soriano, 2003, "A parallel hybrid heuristic for the multicommodity capacitated location problem with balancing requirements", *Parallel Computing*, Vol. 29, No. 5, pp. 591–606.

Geoffrion, A. M., 1967, "Integer programming by implicit enumeration and Balas' method", *SIAM Review*, Vol. 9, No. 2, pp. 178–190.

Golden, B. L., 1977, "A statistical approach to the TSP", *Networks*, Vol. 7, No. 3, pp. 209–225.

Golden, B. L., 1978, "Point estimation of a global optimum for large combinatorial problems", *Communications in Statistics Simulation and Computation*, Vol. 7, No. 4, pp. 361–367.

Golden, B. L. and F. B. Alt, 1979, "Interval estimation of a global optimum for large combinatorial problems", *Naval Research Logistic Quarterly*, Vol. 26, No. 1, pp. 69–77.

Golden, B. L. and W. R. Stewart, 1985, "Emprical analysis of heuristics", In: E. L. Lawler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Shmoys (Eds.), *The traveling salesman problem: a guided tour of combinatorial optimization*, pp. 207-250, Wiley, Chichester.

Gonsalvez, D. J., N. G. Hall, W. T. Rhee, and S. P. Siferd, 1987, "Heuristic solutions and confidence intervals for the multicovering problem", *European Journal of Operarional Research*, Vol. 31, No. 1, pp. 94–101.

Gumbel, E., 1958, *Statistics of Extremes*, Columbia University Press, New York.

Hall, N. G., S. Ghosh, R. D. Kankey, S. Narasimhan, and W. T. Rhee, 1988, "Bin packing problems in one dimension: heuristic solutions and confidence intervals", *Computers and Operations Research*, Vol. 15, No. 2, pp. 171–177.

Hamacher, H. W. and S. Nickel, 1998, "Classification of location models", *Location Science*, Vol. 6, No. 1-4, pp. 229 – 242.

Hansen, P. and N. Mladenović, 2001, "Variable neighborhood search: principles and applications", *European Journal of Operational Research*, Vol. 130, No. 3, pp. 449–467.

Hansen, P., N. Mladenović, and E. Taillard, 1998, "Heuristic solution of the multisource Weber problem as a p-median problem", *Operations Research Letters*, Vol. 22, No. 2-3, pp. 55–62.

Hansen, P., D. Peeters, D. Richard, and J. F. Thisse, 1985, "The minisum and minimax location problems revisited", *Operations Research*, Vol. 33, No. 6, pp. 1251–1265.

Hansen, P., D. Peeters, and J. F. Thisse, 1981, "On the location of an obnoxious facility", *Sistemi Urbani*, Vol. 3, pp. 299–317.

Hansen, P., J. Perreur, and J. F. Thisse, 1980, "Location theory, dominance and convexity: some further results", *Operations Research*, Vol. 28, No. 5, pp. 1241–1250.

Harter, H. L. and A. H. Moore, 1965, "Maximum-likelihood estimation of the parameters of Gamma and Weibull populations from complete and from censored samples", *Technometrics*, Vol. 7, No. 4, pp. 639–643.

Hartley, H. O. and R. C. Pfaffenberger, 1969, "Statistical control for non-linear optimization", Themis Technical Report No 18, Institute of Statistics, Texas A&M University, USA.

Held, M., P. Wolfe, and H. P. Crowder, 1974, "Validation of subgradient optimization", *Mathematical Programming*, Vol. 6, No. 1, pp. 62–88.

Horst, R. and H. Tuy, 1996, *Global Optimization: deterministic approaches*, 3^{rd} edition, Springer, Berlin.

Idrissi, H., P. Loridan, and C. Michelot, 1988, "Approximation of solutions for location problems", *Journal of Optimization Theory and Applications*, Vol. 56, No. 1, pp. 127–143.

Karkazis, J. and T. B. Boffey, 1981, "The multi-commodity facilities location problem", *The Journal of the Operational Research Society*, Vol. 32, No. 9, pp. 803–814.

Klingman, D., A. Napier, and J. Stutz, 1974, "NETGEN: a program for generating large scale capacitated assignment, transportation, and minimum cost flow problems", *Management Science*, Vol. 20, No. 5, pp. 814–821.

Korovkin, P. P., 1986, *Inequalities*, Mir Publishers, Moskow.

Krau, S., 1997, *Extensions du problème de Weber*, Ph.D. Thesis, University of Montréal.

Kudva, G., J. F. Elkamel, J. F. Pekny, and G. V. Reklaitis, 1994, "Heuristic algorithm for scheduling batch and semi-continuous plants with production deadlines, intermediate storage limitations and equipment changeover costs", *Computers and Chemical Engineering*, Vol. 18, No. 9, pp. 859–875.

Kuenne, R. E. and R. M. Soland, 1972, "Exact and approximate solutions to the multisource Weber problem", *Mathematical Programming*, Vol. 3, No. 1, pp. 193–209.

Kuhn, H. W., 1967, "On a pair of dual nonlinear programs", In: J. Abadie, (Ed.), *Methods of nonlinear programming*, pp. 38-54, North Holland, Amsterdam.

Kuhn, H. W., 1973, "A note on Fermat's problem", *Mathematical Programming*, Vol. 4, No. 1, pp. 98–107.

Kuhn, H. W. and R. E. Kuenne, 1962, "An efficient algorithm for the numerical solution of the generalized Weber problem in spatial economics", *Journal of Regional Science*, Vol. 4, No. 2, pp. 21–34.

Law, A. M. and W. D. Kelton, 1991, *Simulation Modeling and Analysis*, 2^{nd} edition, McGraw Hill, Inc., New York.

Liau, T., H. O. Hartley, and R. L. Sielken Jr., 1973, "Confidence limits for the global optimum", Themis Technical Report No 43, Institute of Statistics, Texas A&M University, USA.

Los, M. and C. Lardinois, 1982, "Combinatorial programming, statistical optimization and the optimal transportation network problem", *Transportation Research B*, Vol. 16, No. 2, pp. 89–124.

Love, R. F. and H. Juel, 1982, "Properties and solution methods for large location-allocation problems", *Journal of the Operational Research Society*, Vol. 33, No. 5, pp. 443–452.

Love, R. F., J. G. Morris, and G. O. Wesolowsky, 1988, *Facilities Location: Models & Methods*, North-Holland, New York.

Lowerre, B. T., 1976, *The HARPY speech recognition system*, Ph.D. Thesis, Carnegie-Mellon University.

Luis, M., S. Salhi, and G. Nagy, 2009, "Region-rejection based heuristics for the capacitated multi-source Weber problem", *Computers and Operations Research*, Vol. 36, No. 6, pp. 2007–2017.

Luis, M., S. Salhi, and G. Nagy, 2011, "A guided reactive GRASP for the capacitated multi-source Weber problem", *Computers and Operations Research*, Vol. 38, No. 7, pp. 1014–1024.

Marin, A. and J. Salmerón, 1996, "Tactical design of rail freight networks. Part II: Local search methods with statistical analysis", *European Journal of Operational Research*, Vol. 94, No. 1, pp. 43–53.

Martello, S. and P. Toth, 1990, *Knapsack problems: algorithms and computer implementations*, John Wiley & Sons, New York.

Mathirajan, M. and A. I. Sivakumar, 2006, "Minimizing total weighted tardiness on heterogeneous batch processing machines with incompatible job families", *International Journal of Advanced Manufacturing Technology*, Vol. 28, No. 9, pp. 1038–1047.

McRoberts, K. L., 1971, "A search model for evaluating combinatorially explosive problems", *Operations Research*, Vol. 19, No. 6, pp. 1331–1349.

Megiddo, N. and K. J. Supowit, 1984, "On the complexity of some common geometric location problems", *SIAM Journal on Computing*, Vol. 13, No. 1, pp. 182–196.

Miehle, W., 1958, "Link-length minimization in networks", *Operations Research*, Vol. 6, No. 2, pp. 232–243.

Monroe, H. M., 1982, *Confidence limits for the global optimum*, Ph.D. Thesis, Texas A&M University.

Muralidhar, K. and S. H. Zanakis, 1992, "A simple minimum-bias percentile estimator of the location parameter for the Gamma, Weibull and Log-Normal distributions", *Decision Sciences*, Vol. 23, No. 4, pp. 862–879.

Neebe, A. W. and B. M. Khumawala, 1981, "An improved algorithm for the multi-commodity location problem", *The Journal of the Operational Research Society*, Vol. 32, No. 2, pp. 143–149.

Nelder, J. A. and R. Mead, 1965, "A simplex method for function minimization", *Computer Journal*, Vol. 7, No. 4, pp. 308–313.

Nickel, S. and J. Puerto, 2005, *Location theory: a unified approach*, Springer, Heidelberg.

Ostresh, L. M. J., 1975, "An efficient algorithm for solving the two center location-allocation problem", *Journal of Regional Science*, Vol. 15, No. 2, pp. 209–216.

Ostresh, L. M. J., 1977, "The multifacility location problem: applications and descent theorems", *Journal of Regional Science*, Vol. 17, No. 3, pp. 409–419.

Ovacık, I. M., S. Rajagopalan, and R. Uzsoy, 2000, "Integrating interval estimates of global optima and local search methods for combinatorial optimization problems", *Journal of Heuristics*, Vol. 6, No. 4, pp. 481–500.

Ow, P. S. and T. E. Morton, 1988, "Filtered beam search in scheduling", *International Journal of Production Research*, Vol. 26, No. 1, pp. 35–62.

Ow, P. S. and T. E. Morton, 1989, "The single machine early/tardy problem", *Management Science*, Vol. 35, No. 2, pp. 177–191.

Pirkul, H. and V. Jayaraman, 1998, "A multi-commodity, multi-plant, capacitated facility location problem: formulation and efficient heuristic solution", *Computers and Operations Research*, Vol. 25, No. 10, pp. 869–878.

Plastria, F., 1992, "GBSSS-the generalized big square small square method for planar single-facility location", *European Journal of Operational Research*, Vol. 62, No. 2, pp. 163–174.

Plastria, F., 1995, "Continuous location problems", In: Z. Drezner (Ed.), *Facility location: a survey of applications and methods*, pp. 37-79, Springer, Heidelberg.

Polyak, B. T., 1967, "A general method of solving extremum problems", *Soviet Mathematics Doklady*, Vol. 8, pp. 593 – 597.

Polyak, B. T., 1969, "Minimization of unsmooth functionals", *USSR Computational Mathematics and Mathematical Physics*, Vol. 9, No. 3, pp. 14 – 29.

Rawls, J., 1971, *A theory of justice*, Harward University Press, Cambridge.

Righini, G. and L. Zaniboni, 2007, "A branch-and-price algorithm for the multi-source Weber problem", *International Journal of Operations Research*, Vol. 2, No. 2, pp. 188–207.

Robson, D. S. and J. H. Whitlock, 1964, "Estimation of a truncation point", *Biometrika*, Vol. 51, No. 1-2, pp. 33–39.

Rockafellar, R. T., 1970, *Convex Analysis*, Princeton University Press, New Jersey.

Rosen, J. B. and G. L. Xue, 1993, "On the convergence of a hyperboloid approximation procedure for the perturbed Euclidean multifacility location problem", *Operations Research*, Vol. 41, No. 6, pp. 1164–1171.

Rosing, K. E., 1992, "An optimal method for solving the (generalized) multi-Weber problem", *European Journal of Operational Research*, Vol. 58, No. 3, pp. 414–426.

Ryan, D. M. and B. A. Foster, 1981, "An integer programming approach to scheduling". In: A. Wren (Ed.), *Computer Scheduling of Public Transport: Urban Passenger Vehicle and Crew Scheduling*, pp. 269-280, North-Holland, Amsterdam.

Salhi, S. and M. D. H. Gamal, 2003, "A genetic algorithm based approach for the uncapacitated continuous location-allocation problem", *Annals of Operations Research*, Vol. 123, No. 1-4, pp. 203–222.

Sastry, S. and J. Pi, 1991, "Estimating the minimum of partitioning and floorplanning problems", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 10, No. 2, pp. 273–282.

Selim, S., 1979, *Biconvex programming and deterministic and stochastic location allocation problems*, Ph.D. Thesis, Georgia Institute of Technology.

Sherali, H. D. and W. P. Adams, 1999, *A reformulation-linearization technique for solving discrete and continuous nonconvex problems*, Kluwer Academic Publishers, Dordrecht.

Sherali, H. D., I. Al-Loughani, and S. Subramanian, 2002, "Global optimization procedures for the capacitated Euclidean and ℓ_p distance multifacility location-allocation problem", *Operations Research*, Vol. 50, No. 3, pp. 443–448.

Sherali, H. D. and F. L. Nordai, 1988, "NP-hard, capacitated, balanced p-median problems on a chain graph with a continuum of link demands", *Mathematics of Operations Research*, Vol. 13, No. 1, pp. 32–49.

Sherali, H. D., S. Ramachandran, and S. I. Kim, 1994, "A localization and reformulation discrete programming approach for the rectilinear distance location-allocation problems", *Discrete Applied Mathematics*, Vol. 49, No. 1-3, pp. 357–378.

Sherali, H. D. and C. H. Tunçbilek, 1992, "A squared-Euclidean distance location-allocation problem", *Naval Research Logistics*, Vol. 39, No. 4, pp. 447–469.

Thisse, J. F., J. E. Ward, and R. E. Wendell, 1984, "Some properties of location problems with block and round norms", *Operations Research*, Vol. 32, No. 6, pp. 1309–1327.

Van Der Watt, P., 1980, "A note on estimation of bounds of random variables", *Biometrika*, Vol. 67, No. 3, pp. 712–714.

Vasko, F. J. and G. R. Wilson, 1984, "An efficient heuristic for large set covering problems", *Naval Research Logistics Quarterly*, Vol. 31, No. 1, pp. 163–171.

Ward, J. E. and R. E. Wendell, 1980, "A new norm for measuring distance which yields linear location problems", *Operations Research*, Vol. 28, No. 3, pp. 836–844.

Ward, J. E. and R. E. Wendell, 1985, "Using block norms for location modeling", *Operations Research*, Vol. 33, No. 5, pp. 1074–1090.

Weiszfeld, E., 1937, "Sur le point lequel la somme des distances de n points donné est minimum", *Tôhoku Mathematics Journal*, Vol. 43, No. 1, pp. 355–386.

Wendell, R. E. and A. P. Hurter, 1973, "Location theory, dominance and convexity", *Operations Research*, Vol. 21, No. 1, pp. 314–320.

Wesolowsky, G. O., 1993, "The Weber problem: history and perspectives", *Location Science*, Vol. 1, No. 1, pp. 5–23.

Wilson, A. D., R. E. King, and J. R. Wilson, 2004, "Case study on statistically estimating the minimum makespan for flow line scheduling problems", *European Journal of Operational Research*, Vol. 155, No. 2, pp. 439–454.

Witzgall, C., 1964, "Optimal location of central facility: mathematical models and concepts", Report No 8388, National Bureau of Standards (NBS), Maryland, USA.

Wolsey, L. A., 1998, *Integer Programming*, John Wiley & Sons, New York.

Wyckoff, J., L. J. Bain, and M. Engelhardt, 1980, "Some complete and censored sampling results for the three-parameter Weibull distribution", *Journal of Statistical Computation and Simulation*, Vol. 11, No. 2, pp. 139–151.

Zainuddin, Z. M. and S. Salhi, 2007, "A perturbation-based heuristic for the capacitated multisource Weber problem", *European Journal of Operational Research*, Vol. 179, No. 3, pp. 1194–1207.

Zanakis, S. H., 1977, "Computational experience of some nonlinear optimization algorithms in deriving maximum likelihood estimates for the three-parameter Weibull distribution", *TIMS Studies in Management Sciences*, Vol. 7, pp. 63–77.

Zanakis, S. H., 1979, "Extended pattern search with transformations for the three parameter Weibull MLE problem", *Management Science*, Vol. 25, No. 11, pp. 1149–1161.

Zanakis, S. H. and N. R. Mann, 1982, "A good simple percentile estimator of the Weibull shape parameter for use when all three parameters are unknown", *Naval Research Logistics Quarterly*, Vol. 29, No. 3, pp. 419–428.