

GALATASARAY UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

**LOCATION IN MULTIPLE REGIONS HAVING
NON-IDENTICAL NORMS**



Gökhan ALTAY

JANUARY 2025

LOCATION IN MULTIPLE REGIONS HAVING NON-IDENTICAL NORMS

**(FARKLI MESEFE NORMLARINA SAHİP BİRDEN FAZLA BÖLGEDE TESİS
YERLEŞİMİ)**

by

Gökhan ALTAY, M.S.

Thesis

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

in

INDUSTRIAL ENGINEERING

in the

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

JANUARY 2025

ACKNOWLEDGEMENTS

Firstly, I would like to express gratitude to my thesis supervisor Prof. Dr. Temel ÖNCAN, for his support, motivation, guidance, patience and time. Secondly, I would also like to express my sincere thanks for Assoc. Prof. Dr. M. Hakan AKYÜZ for his patience, his support and time.

I would also like to thank my thesis committee member Prof. Dr. İ. Kuban ALTINEL for his valuable remarks, suggestions and guidance. I also thank my thesis committee member Assoc. Prof Dr. Tuncay GÜRBÜZ for his valuable suggestions. I am also grateful to the Assoc. Prof. Dr. M. Ebru ANGÜN and Prof. Dr. Semra AĞRALI for accepting to join my thesis committee.

Lastly, many thanks to my brother Can ALTAY for his immense support and encouragement. Also thanks to my mother Nigar ALTAY and father Tayfun ALTAY for their motivation and encouragement.

Gökhan ALTAY

May 2024

TABLE OF CONTENTS

LIST OF SYMBOLS	vii
LIST OF FIGURES	viii
LIST OF TABLES	x
ABSTRACT.....	xiii
ÖZET	xv
1 INTRODUCTION	1
2 LITERATURE REVIEW	4
2.1 Single Facility Location Problem with Multiple Regions with Different Norms	4
2.2 Heuristic Methods for SMDN	6
2.3 Location Routing Problem	7
3 SINGLE FACILITY LOCATION PROBLEM WITH MULTIPLE REGIONS HAVING DIFFERENT NORMS.....	9
3.1 Single Facility Location Problem with Two Regions Having Different Norms	9
3.2 Single Facility Location Problem with Three Regions Having Different Norms.....	12
3.2.1 SMDN3 with Vertical Separating Lines	13
3.2.1.1 Mathematical formulation of SMDN3 with vertical separating lines .	15
3.2.2. SMDN3 with Oblique Separating Lines	18
3.2.2.1 Gate-point formulations of SMDN3- ℓ_1	20
3.2.2.2 Gate-point formulations of SMDN3- ℓ_2	25
4 SOLUTION METHODS FOR SMDN3.....	30
4.1 An Exact Solution Method	30
4.1.1 Big Square Small Square Algorithm	31

4.2 Heuristic Methods.....	34
4.2.1 Modified Weiszfeld Algorithm	35
4.2.2 Discrete Approximation Algorithm.....	36
4.2.3 Meta-heuristic Solution Approaches	37
4.2.3.1 Circular Neighborhood	37
4.2.3.2 Focused Neighborhood.....	38
4.2.3.3 Tabu Search	40
4.2.3.4 Simulated Annealing	42
4.2.4 Beam Search Algorithm	43
5 LOCATION ROUTING PROBLEM CONSIDERING MULTIPLE	
REGIONS WITH DIFFERENT NORMS.....	46
5.1 LRP-M with Vertical Separating Line	46
5.1.1 A MINLP Model for LRP-M with Vertical Separating Line	48
5.1.2 A MILP Model for the Discrete Approximation of the LRP-M with Vertical Separating Line	50
5.2 LRP-M with Oblique Separating Line	51
5.2.1 A MINLP Model for the Continuous LRP-M with Oblique Separating Line	52
5.2.2 Discrete Approximation of LRP-M with Oblique Separating Line	55
5.2.2.1 A MINLP model of the LRP-M with oblique separating line using discrete gate-points	55
5.2.2.2 A MILP model of the LRP-M with oblique separating line using discrete facility locations.....	57
6 COMPUTATIONAL STUDIES.....	58
6.1 Computational Experiments for SMDN3 Using Commercial Solvers, BSSS, MW and DA	58
6.1.1 Test Bed.....	58
6.1.2 Computational Results	59
6.1.2.1 SMDN3 with vertical separating lines	60
6.1.2.2 SMDN3 with oblique separating lines	63
6.2 Computational Experiments for SMDN3 Using BSSS, BS and Meta- heuristic Algorithms.....	68
6.2.1 Test Bed	68

6.2.2 Parameter Calibration.....	69
6.2.3 Computational Results	71
6.3 Computational Experiments for LRP-M with Two Regions	76
6.3.1 Test Bed	77
6.3.2 Computational Results.....	77
7 CONCLUSIONS	79
REFERENCES.....	82
BIOGRAPHICAL SKETCH.....	87



LIST OF SYMBOLS

ARSA	: Adaptive Random Search Algorithm
B&B	: Branch-and-bound
B&B	: Branch-and-cut
B&P	: Branch-and-price
BSSS	: Big Square Small Square
CMWP	: Capacitated Multi-facility Weber Problem
CN	: Circular Neighborhood Search
EA	: Evolutionary Algorithm
GA	: Genetic Algorithm
FN	: Focused Neighborhood Search
LB	: Lower bound
LRP	: Location Routing Problem
LRP-M	: Location Routing Problem in multiple regions having different norms
MILP	: Mixed Integer Linear Programming
MINLP	: Mixed Integer Nonlinear Programming
MSWP	: Multi-Source Weber Problem
MW	: Modified Weiszfeld Algorithm
NLP	: Nonlinear Programming
PA	: Particle Swarm Optimization Algorithm
QQ	: Quantile quantile
SA	: Simulated Annealing Algorithm
SFLP	: Single Facility Location Problem
SMDN	: Single Facility Location Problem in multiple regions having non-identical norms
SMDN2	: SMDN in two regions having non-identical norms
SMDN3	: SMDN in three regions having non-identical norms
TS	: Tabu Search
TSP	: Travelling Salesman Problem
TTT	: Time-to-target
UB	: Upper bound
VRP	: Vehicle Routing Problem
VNS	: Variable Neighborhood Search
WP	: Weber Problem

LIST OF FIGURES

Figure 3.1: An illustration of SMDN2	10
Figure 3.2: Sub-cases of SMDN2 with ℓ_1 and ℓ_2 norms	12
Figure 3.3: Cases of SMDN3 considering vertical borderlines	13
Figure 3.4: An illustrative example of SMDN3- ℓ_1 where Ω_3 is split into five sub-regions....	21
Figure 3.5: The shortest paths for the SMDN3- ℓ_1 considering different slanted separating line pairs	22
Figure 3.6: An illustrative example of SMDN3- ℓ_2 with three sub-regions of Ω_3	26
Figure 3.7: he shortest paths for the SMDN3- ℓ_2 separated by different line pairs with various slopes	28
Figure 4.1: An illustration of separate distances taken for lower bound	34
Figure 4.2: Circular Neighborhood $N_C(\mathbf{x}, r, \theta)$	38
Figure 4.3: Focused Neighborhood $N_F(\mathbf{x})$	39
Figure 4.4: Successive Implementation of the Focused Neighborhood.....	39
Figure 4.5: The shortest path distances between customer locations and the borders of the rectangle.....	45
Figure 5.1: LRP-M with vertical separating line	47
Figure 5.2: LRP-M with oblique separating line	52
Figure 6.1: Rectangular hull of customers and placement of separating lines in SMDN3.....	70
Figure 6.2: TTT and QQ plot of TS for the SMDN3- ℓ_2 instance with 250, 000 customers considering two separating lines with slopes $m_1 < m_2 < 0$	75
Figure 6.3: TTT and QQ plot of SA for the SMDN3- ℓ_2 instance with 250, 000 customers considering two separating lines with slopes $m_1 < m_2 < 0$	76

Figure 6.4: Number of iterations vs average percentage gap plots for the SMDN3- ℓ_2 instance with 250, 000 customers considering two separating lines with slopes $m_1 < m_2 < 0$ 76



LIST OF TABLES

Table 3.1: Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_1 < m_2 < 0$	23
Table 3.2: Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_2 < m_1 < 0$	24
Table 3.3: Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_1 < 0 < m_2$	25
Table 3.4: Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_1 < 0 < m_2$	25
Table 3.5: Gate-points of the shortest paths for the SMDN3- ℓ_2 with $m_1 < m_2 < 0$	27
Table 3.6: Gate-points of the shortest paths for the SMDN3- ℓ_2 with $m_1 = m_2 < 0$	29
Table 4.1: Notation used for the TS.....	40
Table 4.2: Notation used for the SA	42
Table 5.1: Notation for LRP-M with vertical separating line.....	47
Table 5.2: Notation for LRP-M with oblique separating line	53
Table 6.1: Computational results for SMDN3- ℓ_1 with vertical borderlines	61
Table 6.2: Computational results of improved exact solution methods for SMDN3- ℓ_1 with vertical borderlines.....	62
Table 6.3: Computational results for SMDN3- ℓ_2 with vertical borderlines	62
Table 6.4: Computational results of improved exact solution methods for SMDN3- ℓ_2 with vertical borderlines.....	63
Table 6.5: Average results of BSSS, DA and MW for each case of SMDN3 with oblique lines	64
Table 6.6: Computational results for the SMDN3- ℓ_1 with oblique borderlines having slopes of $0 > m_2 > m_1$	65

Table 6.7: Computational results for the SMDN3- ℓ_2 with oblique borderlines having slopes of $0 > m_2 > m_1$	65
Table 6.8: Computational results for the SMDN3- ℓ_1 with oblique borderlines having slopes of $0 > m_1 > m_2$	66
Table 6.9: Computational results for the SMDN3- ℓ_2 with oblique borderlines having slopes of $0 > m_1 > m_2$	66
Table 6.10: Computational results for the SMDN3- ℓ_1 with oblique borderlines having slopes of $m_2 > 0 > m_1$	67
Table 6.11: Computational results for the SMDN3- ℓ_2 with oblique borderlines having slopes of $m_2 > 0 > m_1$	67
Table 6.12: Computational results for the SMDN3- ℓ_2 with parallel oblique borderlines having slopes of $0 > m_1 = m_2$	68
Table 6.13: Parametric calibration.....	70
Table 6.14: Average of the computational results of heuristic methods and BSSS for SMDN3.....	72
Table 6.15: Computational results of heuristic methods for the SMDN3- ℓ_1 with vertical borderlines.....	73
Table 6.16: Computational results of heuristic methods for the SMDN3- ℓ_2 with vertical borderlines.....	73
Table 6.17: Computational results of heuristic methods for the SMDN3- ℓ_1 with oblique borderlines as $0 > m_2 > m_1$	73
Table 6.18: Computational results of heuristic methods for the SMDN3- ℓ_2 with oblique borderlines as $0 > m_2 > m_1$	74
Table 6.19: Computational results of heuristic methods for the SMDN3- ℓ_1 with oblique borderlines as $m_2 > 0 > m_1$	74
Table 6.20: Computational results of heuristic methods for the SMDN3- ℓ_2 with oblique borderlines as $m_2 > 0 > m_1$	74
Table 6.21: Experiments with the models for LRP-M with vertical separating line	78

Table 6.22: Experiments with the models for LRP-M with oblique separating line 78



ABSTRACT

This study focuses on the facility location problems in multiple regions having different norms on a continuous plane. These regions are divided by straight borderlines with arbitrary slopes. While facility location problem has been previously studied in the literature for two regions, this study aims to take a further step by generating the gate-point formulations for three regions. It is observed that even finding geodesic path requires significant effort and the optimal gate-point formulations do not always have closed-form solutions.

Considering the single facility problem in three regions with different norms, the performance of the Mixed-Integer Nonlinear Programming formulation and three Nonlinear Programming formulations that solve the problem for each region separately are evaluated using commercial solvers for the cases where the separating lines are vertical. In addition, the Big Square Small Square algorithm (BSSS) is specifically designed to solve every case of the single facility problem in three regions with different norms as an efficient exact geometric branch-and-bound algorithm. From extensive computations, BSSS arises to be the best exact solution approach for each case. Additionally, Discrete Approximation scheme and Modified Weiszfeld algorithm are applied to pursue further efficiency while sustaining reasonable accuracy.

Furthermore, large-scale instances of the single facility problem in three regions with different norms with up to 250,000 customers are considered. Two meta-heuristic algorithms, i.e., Tabu Search and Simulated Annealing, are developed to achieve accurate solutions while maintaining decent efficiency. Furthermore, as an accurate BSSS-based heuristic method, Beam Search algorithm is developed. It has been observed from the computational results that Simulated Annealing is the most efficient while Beam Search is the most accurate one.

Lastly, the location routing problem in two regions having different norms for a single route and single facility is addressed and formulated as a Mixed-Integer Nonlinear Programming problem for multiple cases. To this end, alternative formulations discretizing both gate-points and facility locations are proposed. Solving the Mixed-Integer Linear Programming problem obtained by discretizing the facility locations is observed to be a viable choice considering the intractability of the original problem.

Keywords : Facility location, Distance norms, Meta-heuristics, Branch and bound algorithms



ÖZET

Bu çalışma, farklı mesafe normlarına sahip olan çoklu bölgedeki sürekli düzlem tesis yerleşimi problemlerine odaklanmaktadır. Bu bölgeler, herhangi bir eğimi olan düz sınır çizgileriyle ayrılmaktadır. Bu problem literatürde iki bölge için kapsamlı bir şekilde çalışılmış ve en kısa yolların geçiş noktalarının belirlenmesi problemi olarak ele alınmıştır. Bu çalışma, bu problemin genelleştirilmesine katkıda bulunarak üç bölge için geçiş noktalarının belirlenmesi problemini modellemiştir. Bu aşamada, farklı mesafe normları kullanılan üç bölge için, iki nokta arasındaki en kısa yolun bulunmasının dikkate değer bilgisayarlı süreler gerektirdiği ve bazı durumlarda geçiş noktalarının matematiksel formülasyonunun olmadığı gözlemlenmiştir.

Bu çalışmanın ilk bölümü, farklı mesafe normları kullanan üç bölgede tek tesis yerleşimi üzerinedir. Bu problemin sınır çizgilerinin dik olduğu özel durum için Karışık Tamsayılı Doğrusal Olmayan Programlama modeli ile bölgeleri ayrı ayrı değerlendiren üç adet Doğrusal Olmayan Programlama modellerinin performansı, çözümler kullanılarak değerlendirildi. Ayrıca daha kullanışlı ve hızlı bir kesin çözüm algoritması olarak Büyük Kare Küçük Kare algoritması bu problemin bütün özel durumları için geliştirilmiştir. Bu problem üzerinde yapılan hesaplamalı sonuçlara göre, Büyük Kare Küçük Kare algoritması en iyi kesin çözüm yaklaşımıdır. Ayrıca bu problem için daha hızlı performans gösterip kabul edilebilir yaklaşıklıkta sonuçlar veren iki sezgisel yöntem olarak Dönüştürülmüş Weiszfeld ve Kesikli Yaklaşıklama algoritmaları önerilmiştir.

Daha sonra, ele alınan tek tesis yerleşimi problemini daha büyük veri kümelerinde (250,000 müşteriye kadar) çözmek amacıyla daha etkin iki meta-sezgisel yaklaşım (Tabu Arama ve Benzetimli Tablama) geliştirilmiştir. Ayrıca Büyük Kare ve Küçük Kare algoritmasından hareketle geliştirilen Işın Arama algoritması tasarlanmıştır. Çalışmanın sonuçlarına göre, sezgisel yaklaşımlar arasında Işın Arama algoritması kabul edilebilir bir etkinlikte en iyi sonuçları verirken, Benzetimli Tavlama algoritması en hızlı ve kabul

edilebilir doğrulukta sonuçlara ulaşmaktadır.

Son olarak, farklı mesafe normuna sahip iki bölgede Yerleşim-Rotalama (tek tesis ve tek rota) problemi ele alınmıştır. Bu problemin sürekli düzlem üzerindeki çözümü için Karışık Tamsayılı Linear Olmayan Programlama modeli önerilmiş, farklı durumlar için geçiş noktalarının veya tesis yerleşiminin ayrı ayrı kesikli yakınlştırılması ile oluşan alternatif modellerin performansı sınanmıştır. Problemin her durumu için tesis yerleşiminin kesikli yakınlştırılması ile elde edilen karışık tamsayılı model, en iyi performansı vermektedir.

Anahtar Kelimeler : Tesis yerleşimi, Mesafe normları, Meta-sezgisel yöntemler, Dalsınır algoritması

1 INTRODUCTION

This thesis focuses on various facility location problems with multiple regions having different norms. We begin by explaining the single facility location problem (SFLP) with multiple regions having different norms, which has been studied extensively over the past few decades (Love et al., 1988 ; Francis et al., 1992 ; Moradi and Bidkhori, 2009). We extend our work on SFLP in multiple regions with different norms (SMDN), which can be considered as a generalization of the well-known Weber problem (WP). The objective of SMDN is to find the optimal location of a single facility on a continuous plane, which is divided into multiple regions where the distance to travel in each region is approximated using different ℓ_p norms. In this study, we focus primarily on the regions with ℓ_1 (*Manhattan*) and ℓ_2 (*Euclidean*) norms as they are commonly encountered in real-world applications, particularly in urban and rural areas. For example, fiber optic cables are usually laid along the road network in urban areas, which can possibly be approximated by using ℓ_1 norm. However, in rural regions, this restriction does not apply, hence ℓ_2 norm is a more plausible metric for distance approximation. Therefore, following these distance approximation metrics, opening a distribution center in a rural area near a city or finding the optimal location of a main facility in a telecommunication network or pipeline system or facility location problems in emergency services such as an ambulance network considering multiple regions using different distance metrics can be seen as a real-life application of the SMDN.

To the best of our knowledge, the studies in the literature are mostly based on the SMDN with two regions. Even though some authors such as s Brimberg et al. (2003) and Brimberg et al. (2005) have come up with results for the SFLP considering more than two regions, their work considers only some special cases, such as 3 subregions using ℓ_1 or ℓ_2 norms as distance approximations divided by two vertical separating lines.

We first focus on the various cases of SMDN with more than two regions. We deal mostly with the SMDN with three regions (SMDN3). First, we derive mathematical results for the SMDN with three regions divided by two straight separating lines with various slopes. To this end, we consider any significant case for the SMDN3 while formulating the gate-points with ℓ_1 and ℓ_2 norms. As a result, we provide a characterization of the gate-point locations

for the shortest paths within SMDN in three regions. We also show that the difficulty of solving the SMDN in three regions depends considerably on the slopes of each separating line from our computational results. Lastly, we concentrate on the cases of more than three regions; we observe that the solution of the SMDN is significantly more difficult since the gate-points do not have a closed form solution unless a special case such as vertical separating lines is involved.

In order to tackle SMDN3, we start by developing a geometrical branch-and-bound (B&B) algorithm, namely a Big Square Small Square (BSSS) algorithm specially tailored for SMDN3. Furthermore, we propose two heuristic procedures, i.e. Discrete Approximation (DA) and Modified Weiszfeld (MW) schemes for SMDN3. We also employ commercial solvers such as KNITRO and BARON to solve the mathematical formulations of a special case of SMDN3 where the separating lines are vertical or horizontal. In this regard, two distinct approaches are evaluated: A Mixed-Integer Nonlinear Programming (MINLP) formulation and a method that decomposes the problem into three separate Nonlinear Programming (NLP) models each addressing a single region individually. Nevertheless, we observe that the BSSS algorithm provides promising results in comparison to both suggested heuristic methods DA and MW and commercial solvers such as BARON and KNITRO.

Subsequently, we extend our focus on SMDN3 for large-sized data sets with up to 250,000 customers efficiently while maintaining reasonable accuracy. To this end, we conduct our tests using instances from the literature with the number of customers varying from 50 to 85,900 and randomly generated instances with 100,000 to 250,000 customers. We propose two efficient metaheuristics, i.e. Simulated Annealing and Tabu Search algorithms for SMDN3 and we suggest a branch-and-bound based heuristic method, namely the Beam Search algorithm that employs the methodology of BSSS to efficiently reach accurate results. Computational experiments for these heuristic methods are repeated 10 times for each instance and their average is reported. Computations on different cases of SMDN3 are included in these experiments.

Lastly, we turn our attention to the location routing problem with two regions having different norms (LRP-M). We address two different cases of LRP-M with two regions. The first case considers vertical separating line and the second case addresses an oblique separating line (general case).

We first propose mathematical programming models for both versions, i.e. a MINLP model using distance functions similar to Parlar (1994) together with Gavish-Graves sub-tour eli-

mination constraints (Gavish and Graves, 1978). For the sake of simplicity, the scope of this work is limited to a single route, namely the Traveling Salesman Problem (TSP). Note that in the single-route LRP-M, optimal facility location is located in one of the customer locations. Therefore, we suggest a discrete approximation approach and propose a Mixed-Integer Linear Programming (MILP) formulation considering customer locations as candidate sites. Then, we address the second version, which is the LRP-M with oblique separating line. For this version, we consider discrete gate-points case and we devise a MINLP formulation.

The remainder of this thesis is organized as follows. Chapter 2 provides a brief review of the literature, Chapter 3 introduces the SMDN and emphasizes on gate-point formulations and the mathematical model of SMDN3. Chapter 4 includes the solution approaches proposed to solve SMDN3. Chapter 5 presents LRP-M with various mathematical formulations. Chapter 6 reports the computational results. Finally, Chapter 7 incorporates the concluding remarks and future research directions.

2 LITERATURE REVIEW

2.1 Single Facility Location Problem with Multiple Regions with Different Norms

Single facility location problem with multiple regions with different norms (SMDN) is a continuous facility location problem which can also be seen as a form of generalization of the well-known Weber Problem (WP). In the literature, there are several studies that are closely related to SMDN. First, Planchart and Hurter (1975) address an extension of the WP in which distances are measured as a linear combination of different norms. Although Planchart and Hurter (1975) do not explicitly consider the case of SMDN, they use a weighted sum of ℓ_1 and ℓ_2 norms to define distance approximations and suggests an algorithm based on duality relations. Michelot and Lefebvre (1987) propose a primal-dual method for WP with different norms. Brimberg et al. (2007) approximate road travel distances using different weighted ℓ_p norms. Furthermore, Brimberg et al. (2009) approximate distances with a mixture of weighted norms to respond to different transport modes. They suggest the Weiszfeld Algorithm for the solution of this problem.

Considering SMDN, Parlar (1994) develops a MINLP model of SMDN for two regions using ℓ_1 and ℓ_2 norms as distance metrics. In addition, the author proves that the objective function of the SMDN is not convex, despite the distance functions being convex. Parlar (1994) proposes the Modified Weiszfeld Algorithm (MW) for the solution of the SMDN and compares the performance of this algorithm with Adaptive Random Search Algorithm (ARSA) (Gottfried and Weisman, 1973). Later, as a more efficient approach, a variant of the Big Square Small Square (BSSS) algorithm is introduced for SMDN with two regions by Zaferanieh et al. (2008). Furthermore, they make a comparison of the results of BSSS with those obtained with MW (Parlar, 1994).

Additionally, Brimberg et al. (2003) address the SMDN that includes two regions with ℓ_1 and ℓ_2 norms separated by a single vertical line. The authors address various cases of two regions with ℓ_p and ℓ_∞ norms divided by a single slanted borderline. Additionally, the authors include a brief study for a specific case of SMDN3 considering three regions with ℓ_2 , ℓ_1 , and ℓ_2 norms, in which the area is divided by two vertical lines. Brimberg et al. (2005)

focus on extensions of their previous work where the interior region is bounded and both regions have different distance norms. They also deal with more complex shaped regions in this work.

Considering closed-form gate-point formulations when solving the shortest path problems between two destinations in SMDN with two regions (SMDN2), Franco et al. (2012) derive the gate-point location formulations between two regions divided by a straight line. The authors solve the gate-point formulations for any specific slope of a separating line using the derivatives of distance functions and line equations. In their work, they consider ℓ_1 or ℓ_∞ norms for the distance approximation of one region and ℓ_2 norm for the other. From their work it can be inferred that the problem of finding the gate-point (or gate-points) is also the same as finding the shortest path between two locations, each of which resides in different regions. In addition, Franco et al. (2018) deal with SMDN2 by extending their previous work (Franco et al., 2012) on the subcases of gate-point formulations for SMDN2 with ℓ_1 and ℓ_q norms with $q > 1$.

Similar to the structure of multiple regions with multiple norms, Mitchell and Papadimitriou (1987) tackle the shortest path in multiple regions using the weighted Euclidean norm, where each region uses different weight. In their problem, it is possible to solve more complex shaped regions compared to the studies on SMDN. This problem can be considered in modeling robot movements, where different weights can symbolize different speeds in different regions.

In addition, Plastria (2019) addresses the problem of shortest path with gauges, which is a generalization of SMDN where the distances are not necessarily symmetric. His work also makes important remarks on a special case of *triple-link* shortest path in two regions with different norms divided by a single separating line, which have not been explicitly mentioned previously. This case occurs when both locations are in the same region and when traveling between two gate-points on the separating line constitutes a shortcut.

Lastly, the contributions of this study are as follows. Using the gate-point formulations of SMDN2 from the literature (Franco et al., 2012) and solving the derivatives of the new distance functions, gate-point formulations of SMDN3 with ℓ_1 and ℓ_2 norms are derived for every possible cases and exceptional cases without any closed-form gate-point formulations are handled by using Golden Section Search algorithm. In addition, using these new gate-point formulations, a new MINLP formulation of SMDN3 with vertical lines, which is inspired from Parlar (1994), is formed. Considering SMDN3 with vertical lines, MINLP for-

mulation is also decomposed into three NLP formulations addressing each region separately. The performance of solving three NLP formulations separately is compared to solving one MINLP formulation. BSSS algorithm is tailored as an exact solution approach for solving each case of SMDN3 similar to Zaferanieh et al. (2008). In addition, MW (Parlar, 1994) and DA are developed as alternative efficient algorithms for SMDN3.

2.2 Heuristic Methods for SMDN

This study also focuses on developing efficient heuristic algorithms for SMDN with three regions in larger-scale instances. For the meta-heuristic methods, Gendreau et al. (2010) provide an introduction to various meta-heuristic approaches such as Variable Neighborhood Search (VNS), Simulated Annealing (SA) and Tabu Search (TS). However, note that these meta-heuristic algorithms are most commonly used when addressing discrete problems.

For the continuous location problems, Farham et al. (2018) address restricted planar location problem which includes free regions, barriers and congested regions. They develop three meta-heuristics, namely Simulated Annealing (SA), Particle Swarm Optimization Algorithm (PA), and Evolutionary Algorithm (EA). In their computations, they include various data sets from the restricted planar location problem, Vehicle Routing Problem (VRP), and Traveling Salesman Problem (TSP) literature with up to 33810 customer locations. Furthermore, Irawan et al. (2017) deal with the continuous capacitated single-source multi-facility Weber problem with facility fixed cost. They propose a two-stage heuristic approach and an enhanced VNS. Their computational experiments are performed with up to 1,060 customers and 50 facilities. Brimberg et al. (2014) devise a heuristic algorithm that alternates between continuous and discrete space for Multi-Source Weber Problem (MSWP). They employ a local search approach in the continuous space ; then they use discretization for further improvements. They alternate between these approaches until the solution is no longer improved. They embedded this method in VNS. In addition, for Uncapacitated MSWP, Brimberg et al. (2000) compared various heuristic approaches such as TS, VNS, Genetic Algorithm (GA), p-median heuristics, and relocation heuristics. Their computational experiments include instances with 1060 customer locations and 150 facilities. Moreover, Brimberg et al. (2008) present an extensive survey of heuristic approaches on MSWP.

Furthermore, Estève et al. (2006) develop a Beam Search (BS) algorithm on the scheduling problem and Croce et al. (2004) implement BS on the uncapacitated p-median problem.

Akyüz et al. (2013) devise a BS algorithm to Single and Multi Commodity Capacitated Multi-Facility Weber problem using different branching methods.

Finally, inspired by Gendreau et al. (2010) and Farham et al. (2018), this study proposes two meta-heuristic methods, i.e. SA and TS for solving large-scale instances of SMDN3. Two new local search algorithms, i.e. Circular Neighborhood and Focused Neighborhood algorithms are developed for these meta-heuristic algorithms. In addition, BS (Akyüz et al., 2013) algorithm is tailored for SMDN3 as an efficient BSSS-based heuristic. The performance of these three algorithms is compared to BSSS in the large-scale instances of SMDN3 with up to 250,000 customer locations.

2.3 Location Routing Problem

Although the Location Routing Problem with Multiple Regions having Different Norms (LRP-M) has not been studied in the literature, the location routing problem (LRP) is a well-studied problem. However, since the location problem is a strategic-level problem while the routing problem is a tactical-level problem, they are treated as separate problems by many researchers. LRP problem can be seen as a generalization of Vehicle Routing Problem (VRP) or Travelling Salesman Problem (TSP) in which the facility location is not fixed. Farahani and Hekmatfar (2009) overview LRP together with VRP while giving its variations and applications. Nagy and Salhi (2007), Prodhon and Prins (2014), and Drexl and Schneider (2015) present variations and applications of LRP, mentioning also exact and heuristic solution approaches. Additionally, Schneider and Drexl (2017) introduce a survey on deterministic, static, discrete, single-echelon and single-objective LRP.

Clearly, LRP is strongly $\mathcal{NP}$ -hard since it includes another $\mathcal{NP}$ -hard combinatorial optimization problem, i.e. TSP or VRP. LRP is mostly studied as a discrete location problem since its continuous version is nonlinear; hence, it is much more difficult to solve than its discrete counterpart. Schwardt and Dethloff (2005) propose a self-organising map in solving continuous LRP. Salhi and Nagy (2009) propose an iterative heuristic method for solving continuous LRP. Considering discrete LRP, we can refer to work by Wang et al. (2022) where they develop a Branch-and-Price (B&P) algorithm for location routing problem with pick-up stations in the last-mile distribution systems. Furthermore, Almouhanna et al. (2020) focus on LRP with electric vehicles with a limited distance.

In addition, addressing TSP, recently Kloster et al. (2023) work on multiple TSP with drones.

In their study, each truck starts from the depot and provides the supplies for package stations, then drones are used to provide supplies to the customer locations. They suggest a MILP formulation for smaller instances of their problem and also they propose a decomposition-based heuristic algorithm when handling larger instances. Cacchiani et al. (2020) give various models of time-dependent service times TSP. Hernández-Pérez and Salazar-González (2022) develop a Branch-and-Cut (B&C) algorithm for split demand one commodity pick-up and delivery TSP.

This study formulates continuous LRP with two regions having different norms for single-route which has not been studied in the literature. Two distinct cases of LRP-M are introduced, one using vertical separating line while the other using oblique separating line. Considering vertical separating line case, LRP-M is formulated as a MINLP model and a MILP model by using discrete approximation of the facility locations. Similarly, considering oblique separating line case, LRP-M is also formulated as a MINLP model and a MILP model by using discrete approximation of the facility locations. In addition, an alternative MINLP formulation of LRP-M is developed by using discrete gate-point locations while the facility locations are continuous for LRP-M with oblique lines. Finally, the performance of the commercial solvers are compared for each formulation using instances with up to 100 customer locations for both cases.

3 SINGLE FACILITY LOCATION PROBLEM WITH MULTIPLE REGIONS HAVING DIFFERENT NORMS

This chapter introduces various formulations of the single-facility location problem with multiple regions having different norms (SMDN) and its extensions for more than two regions. The main focus of this study is on SMDN with three regions (SMDN3), which is much more difficult than SMDN with two regions, since shortest-path problems in SMDN with more than three regions usually do not have a closed-form solution.

In the next, SMDN with two regions (SMDN2) is introduced with its key points. Then, SMDN3 is presented starting from formulating its simplest case where the separating lines are vertical or horizontal. Finally, the gate-point formulations of SMDN3 with separating lines with arbitrary slopes are discussed.

3.1 Single Facility Location Problem with Two Regions Having Different Norms

Single facility location with two regions having various norms (SMDN2) is a continuous single facility problem where the shortest distance between two locations is calculated using the respective distance norm (norms) of the region (regions) in the shortest path.

The notation for SMDN2 is given as follows. Let J denote the set of customers, w_j be the demand quantity, and $\mathbf{c}_j = (c_{j1}, c_{j2})$ give the coordinates of the customer j as $j \in J$. Additionally, $\Omega_1(\Omega_2)$ denotes region 1(2) with distance norm $\ell_p(\ell_q)$ distance norm and $J_1(J_2)$ is a subset of J which is a set of customer locations residing in region $\Omega_1(\Omega_2)$. Let $f(\mathbf{x}, \mathbf{c}_j)$ be the distance function that calculates the distance between the unknown facility location $\mathbf{x}(x_1, x_2)$ and $\mathbf{c}_j(c_{j1}, c_{j2})$. Then, f can be split into two separate functions $f^1(\mathbf{x}, \mathbf{c}_j)$ and $f^2(\mathbf{x}, \mathbf{c}_j)$ which calculate the distance between $\mathbf{x}(x_1, x_2)$ and a customer location $\mathbf{c}_j(c_{j1}, c_{j2})$ as $j \in J_1$ and $j \in J_2$, respectively. Since $p \neq q$ holds depending on the regions of a customer, the distance function will be different. The objective function of SMDN2 can be

stated as follows.

$$\min_{\mathbf{x}} \sum_{j=1}^J w_j f(\mathbf{x}, \mathbf{c}_j) = \sum_{j \in J_1} w_j f^1(\mathbf{x}, \mathbf{c}_j) + \sum_{j \in J_2} w_j f^2(\mathbf{x}, \mathbf{c}_j) \quad (3.1)$$

The objective function in (3.1) is shown to be non-convex (Parlar, 1994) despite the convex distance function. The distance of two points $\mathbf{x}$ and $\mathbf{c}_j$ can be measured as the following. $\ell_r(\mathbf{x}, \mathbf{c}_j) = [|x_1 - c_{j1}|^r + |x_2 - c_{j2}|^r]^{\frac{1}{r}}$ for $r = p, q \geq 1$ and $p \neq q$. Distance functions $f^1(\mathbf{x}, \mathbf{c}_j)$ and $f^2(\mathbf{x}, \mathbf{c}_j)$, which calculates the convex distance of the shortest path between $\mathbf{x}$ and $\mathbf{c}_j$ are not always equivalent to ℓ_p or ℓ_q norms since there are also cases involving usage of unknown gate-point $\mathbf{u}_j(u_{j1}, u_{j2})$ on the borderline s . Gate-points are used to calculate the shortest paths between $\mathbf{x}$ and $\mathbf{c}_j$, either when $\mathbf{x}$ and $\mathbf{c}_j$ reside in different regions or when $\mathbf{x}$ and $\mathbf{c}_j$ reside in the same region with lower distance norm ℓ_p as $\ell_p < \ell_q$ and they are located within a specific area close to the separating line s that allows a shorter path to travel over the separating line.

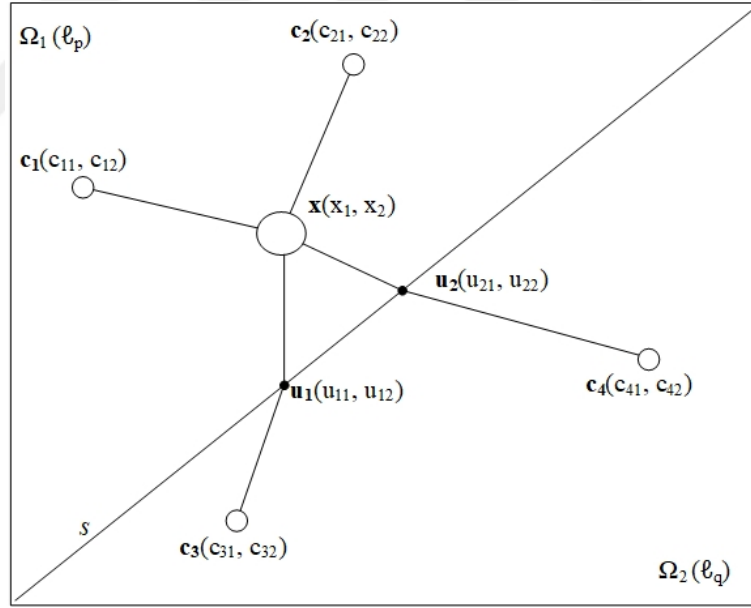


FIGURE 3.1 – An illustration of SMDN2.

Figure 3.1 presents a brief illustration of SMDN with two regions Ω_1 and Ω_2 having ℓ_p and ℓ_q norms. We can observe four demand locations, two of which are located in Ω_1 while the other two reside in Ω_2 . As we can observe, the two regions Ω_1 and Ω_2 are divided by an oblique separating line s . The demand locations $\mathbf{c}_1$ and $\mathbf{c}_2$ ($\mathbf{c}_3$ and $\mathbf{c}_4$) are located in the region Ω_1 (Ω_2). Additionally, it is assumed that the coordinates of the facility location variable $\mathbf{x}(x_1, x_2)$, is placed in the region Ω_1 . $\mathbf{u}_1(u_{11}, u_{12})$ and $\mathbf{u}_2(u_{21}, u_{22})$ present the coordinates of the gate-points that are located at the intersection of the borderline s and the paths from

$\mathbf{x}$ to $\mathbf{c}_3$ and $\mathbf{c}_4$, respectively. Recall that, the shortest paths to the demand points that are situated in each region will be calculated by using their respective norms. For example, the distance between $\mathbf{x}$ and the demand point $\mathbf{c}_3$ is calculated by finding the length of the shortest path between $\mathbf{x}$ and $\mathbf{c}_3$ and using the gate-point $\mathbf{u}_1$ on the borderline s as below.

$$f(\mathbf{x}, \mathbf{c}_3) = \min_{\mathbf{u}_1 \in s} \{\ell_p(\mathbf{x}, \mathbf{u}_1) + \ell_q(\mathbf{u}_1, \mathbf{c}_3)\} \quad (3.2)$$

It can be inferred that calculating the shortest distance between the facility location $\mathbf{x}$ and the demand point $\mathbf{c}_3$, is the same as finding the coordinates of the crossing point $\mathbf{u}_1$ on the borderline s . We know that many authors work with the special or general cases of SMDN2. First, Zaferanieh et al. (2008) provide closed-form formulations of the distances between locations that are situated in different regions and also show that the optimal location of the facility lies within the rectangular hull of the customer locations in SMDN2. Franco et al. (2012) provide a detailed set of gate-point formulations for the shortest path distances for various cases SMDN2 separated by an oblique line with an arbitrary slope.

The main focus of this study revolves around the SMDN considering regions with ℓ_1 and ℓ_2 norms, which is much easier in comparison to the general case of SMDN with ℓ_p and ℓ_r norms when both $p, r > 1$ and $p \neq r$ hold. However, when solving the problem of the shortest path for SMDN2, even in the earlier case, it includes multiple distinct cases of the shortest path between two locations that are specifically indicated by Plastria (2019).

Considering SMDN2 with ℓ_1 and ℓ_2 divided by a single oblique borderline s , there will be three different sub-cases of the shortest path, i.e. *single-link*, *double-link* and *three-link* paths varying on the number of gate points used between two locations. These cases are defined as follows. A *single-link* path is a shortest path between two locations without using any gate-points on the borderline s which occurs when both locations reside in the same region with the norm ℓ_q which satisfies one of the following conditions either $q \geq p$ or $q < p$ and the separating line s does not pass through the rectangular hull of these two locations. A *double-link* shortest paths occurs when two locations are situated in different regions. Finally, the *three-link* path arises only when both locations are situated in the same region with norm ℓ_q as $q < p$ and the separating line s passes through the rectangular hull of these two locations.

The sub-cases of the shortest path between the two locations i.e. *single-link*, *double-link* and *three-link* paths, are depicted with Figure 3.2 for SMDN2 with ℓ_1 and ℓ_2 norms. It can be observed that among three customer locations, both $\mathbf{c}_1$ and $\mathbf{c}_2$ are located in Ω_1 while customer $\mathbf{c}_3$ is located in Ω_2 . As we can observe, the facility is in the region Ω_1 . In Figure

3.2, the shortest path between $\mathbf{x}(x_1, x_2)$ and $\mathbf{c}_2(c_{21}, c_{22})$ is a *single-link* path in the region Ω_1 using the ℓ_1 norm. Similarly, the shortest path between $\mathbf{x}(x_1, x_2)$ and $\mathbf{c}_3(c_{31}, c_{32})$ is a *double-link* path which also travels over separating line with a the gate-point $\mathbf{u}_3(u_{31}, u_{32})$. Finally, the shortest path between $\mathbf{x}(x_1, x_2)$ and $\mathbf{c}_1(c_{11}, c_{12})$ can be observed as a *three-link* path which travels over the separating line s using gate-points $\mathbf{u}_1(u_{11}, u_{12})$ and $\mathbf{u}_2(u_{21}, u_{22})$ on the separating line s . Observe that, for the *three-link* path case, while the shortest path passes through the separating line s between two gate-points $\mathbf{u}_1(u_{11}, u_{12})$ and $\mathbf{u}_2(u_{21}, u_{22})$, the ℓ_2 norm from the region Ω_2 is used.

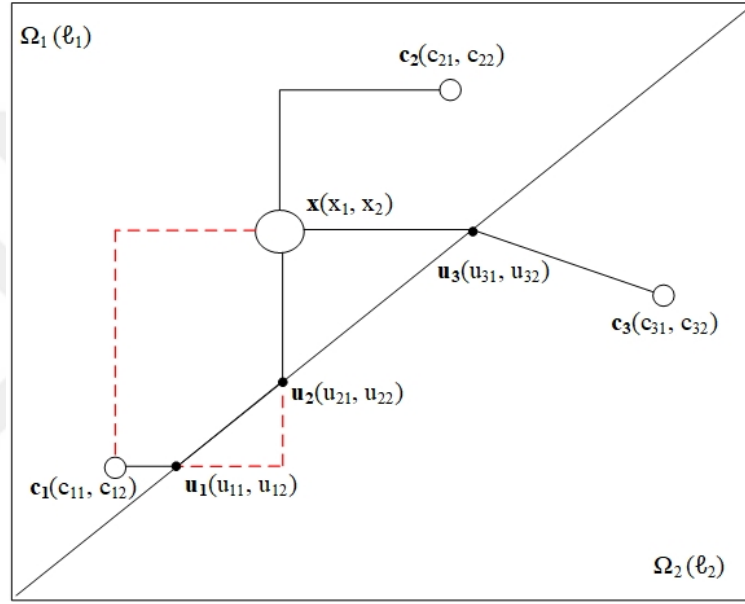


FIGURE 3.2 – Sub-cases of SMDN2 with ℓ_1 and ℓ_2 norms.

3.2 Single Facility Location Problem with Three Regions Having Different Norms

The single facility location problem in three regions with different norms (SMDN3) is a continuous facility problem with three regions Ω_1 , Ω_2 and Ω_3 having different norms separated by two separating lines s_1 and s_2 . This study is based on the cases where three regions are lined horizontally or vertically. In addition, we focus on two cases of SMDN in three regions having norms ℓ_1, ℓ_2, ℓ_1 (SMDN- ℓ_1) and ℓ_2, ℓ_1, ℓ_2 (SMDN- ℓ_2). Note that ℓ_1 and ℓ_2 norms are considered since they are commonly used in the literature.

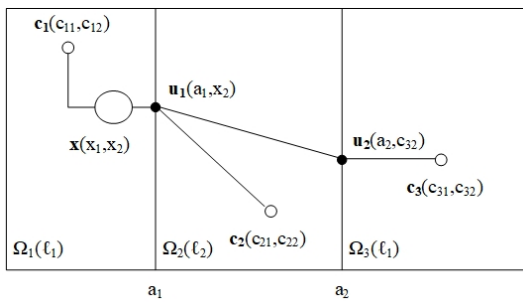
In SMDN3, finding the shortest path between two locations is much more difficult in comparison to SMDN2. Notice that there are cases in SMDN3 with no closed-form formulation of optimal gate-point locations in the shortest path. However, depending on the slopes

of the separating lines s_1 and s_2 the complexity of finding the shortest path changes significantly. The simplest case of SMDN3, which is SMDN3 with vertical (or horizontal) separating lines, is introduced in the literature (Brimberg et al., 2003). In addition to the vertical separating line cases, we propose gate-point formulations of the shortest path for both SMDN- ℓ_1 and SMDN- ℓ_2 problems considering various cases of the slopes of separating lines as $m_1 < m_2 < 0$, $m_2 < m_1 < 0$, $m_1 = m_2 < 0$ and $m_1 < 0, m_2 > 0$. Note that additional combinations of m_1 and m_2 do not produce any different gate-point formulation in the shortest path.

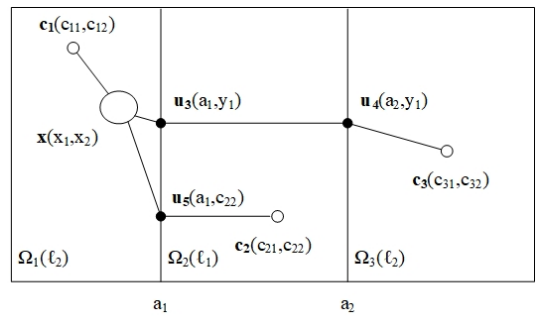
3.2.1 SMDN3 with Vertical Separating Lines

SMDN3 problem with vertical separating lines is the simplest case of SMDN3 since the formulations of gate-point locations are straightforward.

Instances of both SMDN3- ℓ_1 and SMDN3- ℓ_2 separated by vertical separating lines are illustrated with Figure 3.3a and Figure 3.3b, respectively. This is where a_1 and a_2 stand for the location of vertical borderlines by their abscissas. It is assumed that the unknown location of the facility $\mathbf{x}(x_1, x_2)$ is in the region Ω_1 in both Figure 3.3a and Figure 3.3b. In addition, three customer locations are shown, such as $\mathbf{c}_1(c_{11}, c_{12})$, $\mathbf{c}_2(c_{21}, c_{22})$, and $\mathbf{c}_3(c_{31}, c_{32})$, which are located in regions Ω_1 , Ω_2 , and Ω_3 , respectively. Additionally, the gate-points on the separating lines that divide regions Ω_1 and Ω_2 , and regions Ω_2 and Ω_3 , respectively, are indicated by the symbols $\mathbf{u}_1(a_1, x_2)$, $\mathbf{u}_2(a_2, c_{32})$ and $\mathbf{u}_2(a_2, x_2)$, respectively, in Figure 3.3a. Similarly, in Figure 3.3b, the gate-points on the borderline that divides regions Ω_1 and Ω_2 are $\mathbf{u}_5(a_1, c_{22})$ and $\mathbf{u}_3(a_1, y_1)$ and the gate-point on the borderline that divides regions Ω_2 and Ω_3 is $\mathbf{u}_4(a_2, y_1)$.



(a) An instance of the SMDN3- ℓ_1 .



(b) An instance of the SMDN3- ℓ_2 .

FIGURE 3.3 – Cases of SMDN3 considering vertical borderlines.

Note that SMDN2 encompasses the distance functions of the shortest paths between two points located in the same region, as well as adjacent regions. Finding the shortest path between two points that are not in adjacent regions, that is, one in region Ω_1 and the other in Ω_3 is not an easy task unless there is a special case involved. To achieve this, we take into account *three-link* shortest paths for SMDN3- ℓ_1 and SMDN3- ℓ_2 , which are depicted with Figures 3.3a and 3.3b which have different gate-points on each separating line. Taking SMDN- ℓ_1 from Figure 3.3a as an example, the *three-link* path between $\mathbf{x}$ and $\mathbf{c}_3$ passes through both gate points, $\mathbf{u}_1(a_1, x_2)$ and $\mathbf{u}_2(a_2, c_{32})$. This *three-link* path can be formulated using the equation below.

$$f(\mathbf{x}, \mathbf{c}_3) = |a_1 - x_1| + \sqrt{(a_2 - a_1)^2 + (c_{32} - x_2)^2} + |c_{31} - a_2| \quad (3.3)$$

Furthermore, taking into account the SMDN3- ℓ_2 , Figure 3.3b illustrates the *three-link* shortest path crossing through both the gate-points $\mathbf{u}_3(a_1, y_1)$ and $\mathbf{u}_4(a_2, y_1)$ between $\mathbf{x}$ and customer $\mathbf{c}_3$. The following formula is used to find this *three-link* shortest path is as follows.

$$f(\mathbf{x}, \mathbf{c}_3) = \sqrt{(a_1 - x_1)^2 + (y_1 - x_2)^2} + |a_2 - a_1| + \sqrt{(c_{31} - a_2)^2 + (c_{32} - y_1)^2} \quad (3.4)$$

In which y_1 shows the ordinate value of the both gate-points $\mathbf{u}_3$ and $\mathbf{u}_4$ that is calculated within the equation below.

$$y_1 = x_2 - (x_2 - c_{32})|a_1 - x_1| / (|a_1 - x_1| + |c_{31} - a_2|). \quad (3.5)$$

Substituting y_1 in (3.5) can be a significant help to the distance formulation of $f(\mathbf{x}, \mathbf{c}_3)$ as given by Brimberg et al. (2003) as below.

$$f(\mathbf{x}, \mathbf{c}_3) = \sqrt{(|c_{31} - x_1| - |a_2 - a_1|)^2 + (c_{32} - x_2)^2} + |a_2 - a_1| \quad (3.6)$$

3.2.1.1 Mathematical formulation of SMDN3 with vertical separating lines

We introduce a MINLP formulation for the SMDN3- ℓ_1 , in which two vertical lines divide the plane. Our model is inspired by the SMDN2 formulation proposed by Parlar (1994). In addition, to solve this problem for each region independently, we divided this model into three

NLP formulations, namely NLP-1, NLP-2, and NLP-3. It should be noted that the suggested models are also easily adaptable to the other variation of SMDN3, which is SMDN3- ℓ_2 .

Let J define the customer set as $J = \bigcup_{l=1}^3 J_l$ holds when this set J is partitioned into three subsets, J_1 , J_2 , and J_3 . Each subset represents the customers that are situated in regions Ω_1 , Ω_2 , and Ω_3 , respectively. Let the region, in which the facility $\mathbf{x}$ will be opened, be defined by the binary choice variables α and β . The distance of the shortest path between the facility and the customer $\mathbf{c}_j$ for $j \in J_l$ situated in area Ω_l for $l = 1, 2, 3$ is represented by the continuous decision variables λ_j^l . $\mathbf{x} = (x_1, x_2)$ shows the continuous location variable. Furthermore, $0 < a_1 < a_2$ holds as a_1 and a_2 represent the abscissa of the vertical separating lines, as shown with Figure 3.3a. The values $\mathbf{c}_j(c_{j1}, c_{j2})$ and w_j represent the customer coordinates and the demand, respectively, as $j \in J$. Lastly, let M represent a sufficiently large number which equals to the abscissa value of the customer that is positioned at the rightmost of the plane. In summary, SMDN3- ℓ_1 with vertical borderlines can be modelled as a MINLP formulation as follows.

$$\min \sum_{l=1}^3 \sum_{j \in J_l} w_j \lambda_j^l \quad (3.7)$$

Subject to

$$x_1 - a_1 \alpha \geq 0 \quad (3.8)$$

$$x_1 - a_1(1 - \alpha) - \alpha M \leq 0 \quad (3.9)$$

$$x_1 - a_2 \beta \geq 0 \quad (3.10)$$

$$x_1 - a_2(1 - \beta) - \beta M \leq 0 \quad (3.11)$$

$$\beta \leq \alpha \quad (3.12)$$

$$\lambda_j^1 \geq (1 - \alpha) (|x_1 - c_{j1}| + |x_2 - c_{j2}|) \quad \forall j \in J_1 \quad (3.13)$$

$$\lambda_j^1 \geq \alpha(1 - \beta) \left(\sqrt{(x_1 - a_1)^2 + (x_2 - c_{j2})^2} + |a_1 - c_{j1}| \right) \quad \forall j \in J_1 \quad (3.14)$$

$$\lambda_j^1 \geq \beta \left(|x_1 - a_2| + \sqrt{(a_2 - a_1)^2 + (x_2 - c_{j2})^2} + |a_1 - c_{j1}| \right) \quad \forall j \in J_1 \quad (3.15)$$

$$\lambda_j^2 \geq (1 - \alpha) \left(|x_1 - a_1| + \sqrt{(a_1 - c_{j1})^2 + (x_2 - c_{j2})^2} \right) \quad \forall j \in J_2 \quad (3.16)$$

$$\lambda_j^2 \geq \alpha(1 - \beta) \sqrt{(x_1 - c_{j1})^2 + (x_2 - c_{j2})^2} \quad \forall j \in J_2 \quad (3.17)$$

$$\lambda_j^2 \geq \beta \left(|x_1 - a_2| + \sqrt{(a_2 - c_{j1})^2 + (x_2 - c_{j2})^2} \right) \quad \forall j \in J_2 \quad (3.18)$$

$$\lambda_j^3 \geq (1 - \alpha) \left(|x_1 - a_1| + \sqrt{(a_1 - a_2)^2 + (x_2 - c_{j2})^2} + |a_2 - c_{j1}| \right) \quad \forall j \in J_3 \quad (3.19)$$

$$\lambda_j^3 \geq \alpha(1 - \beta) \left(\sqrt{(x_1 - a_2)^2 + (x_2 - c_{j2})^2} + |a_2 - c_{j1}| \right) \quad \forall j \in J_3 \quad (3.20)$$

$$\lambda_j^3 \geq \beta (|x_1 - c_{j1}| + |x_2 - c_{j2}|) \quad \forall j \in J_3 \quad (3.21)$$

$$\alpha, \beta \in \{0, 1\} \quad (3.22)$$

$$x_1, x_2 \geq 0 \quad (3.23)$$

$$\lambda_j^1 \geq 0 \quad \forall j \in J_1 \quad (3.24)$$

$$\lambda_j^2 \geq 0 \quad \forall j \in J_2 \quad (3.25)$$

$$\lambda_j^3 \geq 0 \quad \forall j \in J_3 \quad (3.26)$$

Here, the weighted sum of the distances between the facility and the locations of the customers is minimized using the objective function (3.7). The facility must be opened in one of the regions Ω_1 , Ω_2 , or Ω_3 , according to constraints (3.8)-(3.11). It should be noted that the facility is located in the region Ω_1 when $\alpha = 0$ and $\beta = 0$, in Ω_2 when $\alpha = 1$ and $\beta = 0$, and

in Ω_3 when $\alpha = 1$ and $\beta = 1$. Constraint (3.12) ensures that there are only three possible outcomes $\alpha = \beta = 0$; $\alpha = 1, \beta = 0$; and $\alpha = \beta = 1$ for regions Ω_1, Ω_2 and Ω_3 , respectively, for which the variables α and β can represent the region in which the facility can be placed. Distance of The shortest path between the facility and the demand points which are situated in region Ω_1 for $j \in J_1$, i.e. λ_j^1 is determined by applying constraints (3.13)–(3.15). According to the constraint (3.13), when the facility is in Ω_1 , λ_j^1 is subject to a lower bound. Also, when $\alpha = \beta = 0$, the norm ℓ_1 between the facility $\mathbf{x}$ and the customer $\mathbf{c}_j$ is identical to $\lambda_j^1 \geq |x_1 - c_{j1}| + |x_2 - c_{j2}|$. Otherwise, this constraint is superfluous, since the right-hand side reduces to zero. Similarly, constraints (3.14) and (3.15) are applied when the facility is opened in Ω_2 and Ω_3 , respectively. In summary, the constraints (3.13) - (3.15) bounded the distance λ_j^1 for customer j in the region Ω_1 . Thus, for each customer $\mathbf{c}_j$ such that $j \in J_1$, only one of these three limitations applies. Similarly, when the facility is located in areas Ω_1, Ω_2 , or Ω_3 , constraints (3.16)–(3.18) are used to determine the shortest path distances, λ_j^2 , between the facility and a customer who resides in region Ω_2 for $j \in J_2$. Additionally, the shortest path distances λ_j^3 are calculated by constraints (3.19)–(3.21). α and β are subject to binary limitations under constraints (3.22). $\mathbf{x}$ is guaranteed to be nonnegative by constraints (3.23). Finally, the nonnegativity of the distance variables λ_j^1, λ_j^2 , and λ_j^3 is finally guaranteed by constraints (3.24)–(3.26).

Commercial solvers such as BARON and KNITRO can be employed to solve the MINLP formulation, which is represented by (3.7)–(3.26) when this particular case of SMDN3 considering vertical separating lines is taken into account.

In addition, note that the optimal solution of MINLP formulation can also be obtained by solving three NLP formulations for each region i.e. NLP-1, NLP-2 and NLP-3 representing the cases $\mathbf{x} \in \Omega_1$, $\mathbf{x} \in \Omega_2$ and $\mathbf{x} \in \Omega_3$, respectively. In these formulations, the binary variables α and β are no longer necessary, since the region of $\mathbf{x}$ is defined by a new boundary constraint (or constraints) that defines the region of the unknown facility location $\mathbf{x}$. NLP-1 formulation of SMDN3- ℓ_1 where $\mathbf{x} \in \Omega_1$ is given as follows.

$$\min \sum_{l=1}^3 \sum_{j \in J_l} w_j \lambda_j^l \quad (3.27)$$

Subject to

$$\lambda_j^1 = |x_1 - c_{j1}| + |x_2 - c_{j2}| \quad \forall j \in J_1 \quad (3.28)$$

$$\lambda_j^2 = |x_1 - \mathbf{a}_1| + \sqrt{(\mathbf{a}_1 - c_{j1})^2 + (x_2 - c_{j2})^2} \quad \forall j \in J_2 \quad (3.29)$$

$$\lambda_j^3 = |x_1 - \mathbf{a}_1| + \sqrt{(\mathbf{a}_1 - \mathbf{a}_2)^2 + (x_2 - c_{j2})^2} + |\mathbf{a}_2 - c_{j1}| \quad \forall j \in J_3 \quad (3.30)$$

$$x_1 \leq \mathbf{a}_1 \quad (3.31)$$

$$x_1, x_2 \geq 0 \quad (3.32)$$

$$\lambda_j^1 \geq 0 \quad \forall j \in J_1 \quad (3.33)$$

$$\lambda_j^2 \geq 0 \quad \forall j \in J_2 \quad (3.34)$$

$$\lambda_j^3 \geq 0 \quad \forall j \in J_3 \quad (3.35)$$

Note that the objective function (3.27) is the same as (3.7) that calculates the weighted sum of distances. Constraints (3.28)-(3.30) define the unknown shortest path distance variables λ_j^1 , λ_j^2 and λ_j^3 which gives the distance between $\mathbf{x}$ and $\mathbf{c}_j$ for $j \in J_1$, $j \in J_2$ and $j \in J_3$, respectively. Constraint (3.31) denotes the boundary constraint which ensures that the condition $\mathbf{x} \in \Omega_1$ is satisfied. Constraint (3.32) ensures the non-negativity of $\mathbf{x}$. Lastly, Constraints (3.33)-(3.35) guarantee the non-negativity of the each distance variables λ_j^1 , λ_j^2 and λ_j^3 , respectively.

Similar to the NLP-1 formulation (3.27)-(3.35), the NLP-2 and NLP-3 formulations can be formed simply by defining new boundary constraints for $\mathbf{x} \in \Omega_2$ and $\mathbf{x} \in \Omega_3$, respectively, and changing the distance constraints accordingly.

Let z^* be the best solution from the MINLP formulation and z_1^* , z_2^* and z_3^* be the optimal solutions of NLP-1, NLP-2 and NLP-3, respectively, then the optimal solution of MINLP $z^* = \min(z_1^*, z_2^*, z_3^*)$. Most of the time, solving NLP formulations separately is more efficient in comparison to solving the MINLP model.

3.2.2 SMDN3 with Oblique Separating Lines

Since SMDN3 with vertical separating lines is simply a special case of SMDN3, solving SMDN3 with oblique separating lines is essential both for the generalization of the SMDN3

problem and its real-life applications.

Solving the shortest path problem between two locations and finding a gate-point (or gate-points) on the separating lines in SMDN3 with oblique separating lines with arbitrary slopes is not always trivial in comparison to the SMDN3 with vertical separating lines. Furthermore, the MINLP formulation of SMDN3 with oblique separating lines is proven to be difficult to achieve when it comes to cases with oblique separating lines. In addition, in some cases of SMDN3 having ℓ_2 , ℓ_1 , ℓ_2 norms, the gate-points of the shortest path do not have a closed form formula which makes building a MINLP formulation impossible. Also, for the other cases of SMDN3 with oblique separating lines, there are numerous sub-cases of the shortest path and gate-point locations depending on the location of each customer $\mathbf{c}_j$ and the facility $\mathbf{x}$. Though it is possible to generate a MINLP formulation when there exists a closed-form solution for every gate-point formulation in SMDN3, this will be inefficient compared to MINLP formulation of SMDN3 with vertical lines since binary variables and distance constraints increase significantly. In addition, if there exists a gate-point without a closed-form solution in a shortest path between two location in SMDN3, then forming a MINLP formulation of SMDN3 is not possible.

Developing gate-point formulations for the shortest path in SMDN3 with oblique borderlines was the first goal of this study. The gate-point formulations for the two variants of SMDN3 such as SMDN3- ℓ_1 and SMDN3- ℓ_2 with ℓ_1 , ℓ_2 , ℓ_1 norms and ℓ_2 , ℓ_1 , ℓ_2 norms are characterized for various cases by this study. We consider the cases with $0 > m_2 > m_1$, $0 > m_1 > m_2$, $0 > m_1 = m_2$ and $m_2 > 0 > m_1$ for both SMDN3- ℓ_1 and SMDN3- ℓ_2 . Given two borderlines, $s_1 \equiv y = m_1x + c_1$ and $s_2 \equiv y = m_2x + c_2$, where m_1 and m_2 are the slopes of the lines s_1 and s_2 , we address each scenario independently and provide the formula to determine the gate-point coordinates correspondingly.

Since determining the gate-point is the same as determining the shortest path between two points (Franco et al., 2012), we provide a characterization of the gate-points for the SMDN3. We focus only on determining the shortest path from a point $\mathbf{P}_i \in \Omega_1$ to another point $\mathbf{P}_j$ in a non-adjacent region $\mathbf{P}_j \in \Omega_3$. Given two lines, $s_1 \equiv y = m_1x + c_1$ and $s_2 \equiv y = m_2x + c_2$, which divide the plane into three regions, Ω_1 , Ω_2 and Ω_3 , with ℓ_p , ℓ_q , and ℓ_p norms, respectively. The objective function $f(\mathbf{P}_i, \mathbf{P}_j)$, which indicates the length of the shortest path distance between $\mathbf{P}_i$ and $\mathbf{P}_j$, is defined as follows.

$$f(\mathbf{P}_i, \mathbf{P}_j) = \min_{\mathbf{u}_1 \in s_1; \mathbf{u}_2 \in s_2} \|\mathbf{P}_i, \mathbf{u}_1\|_p + \|\mathbf{u}_1, \mathbf{u}_2\|_q + \|\mathbf{u}_2, \mathbf{P}_j\|_p \quad (3.36)$$

From the equation above, it can be seen that the problem of finding the shortest path is the same as identifying the location of related gate-points.

Lastly, we analyze the gate-point formulations of two different variations of the SMDN3, i.e. the SMDN3- ℓ_1 and the SMDN3- ℓ_2 separately inside the following section. Note that these gate-point formulations are inspired from the of gate-point formulations by Franco et al. (2012) and Franco et al. (2018) for the SMDN2. Gate-point equations are obtained using the line equations of s_1 and s_2 , and the derivatives of the distance functions.

3.2.2.1 Gate-point formulations of SMDN3- ℓ_1

In this section, we focus on the gate-point formulations of SMDN3- ℓ_1 where the plane is partitioned into three regions with two separating lines $s_1 \equiv y = m_1x + c_1$ and $s_2 \equiv y = m_2x + c_2$. We address three cases that arise depending on the slopes of the separating lines s_1 and s_2 , $m_1 < m_2 < 0$, $m_2 < m_1 < 0$ and $m_1 < 0 < m_2$. In addition, notice that gate-point formulations in any alternative cases such as $m_1 > m_2 > 0$, $m_2 > m_1 > 0$ and $m_1 > 0 > m_2$ can be acquired simply by switching the values of m_1 and m_2 from the cases of $m_1 < m_2 < 0$, $m_2 < m_1 < 0$ and $m_1 < 0 < m_2$, respectively.

SMDN3- ℓ_1 with $m_1 < m_2 < 0$:

In the case of SMDN3- ℓ_1 with two separating lines having slopes $m_1 < m_2 < 0$, the shortest path between the points $\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_2 \in \Omega_3$ can be formulated in 5 different ways by further partitioning Ω_3 into five sub-regions, i.e. R_1, R_2, R_3, R_4 and R_5 , which are defined by using three separating points $\mathbf{g}_1, \mathbf{g}_2$ and $\mathbf{g}_3$ on the separating line s_2 as depicted with Figure 3.4. The formulations of the separating points $\mathbf{g}_1, \mathbf{g}_2$, and $\mathbf{g}_3$ are shown below.

$$\mathbf{g}_1 = \left(\frac{m_1^*a_1 + c_2 - (m_1a_1 + c_1)}{m_1^* - m_2}, \frac{m_1^*(m_2a_1 + c_2) - m_2(m_1a_1 + c_1)}{m_1^* - m_2} \right) \quad (3.37)$$

$$\mathbf{g}_2 = \left(\frac{m_1^* \frac{b_1 - c_1}{m_1} + c_2 - b_1}{m_1^* - m_2}, \frac{m_1^*(m_1 \frac{b_1 - c_1}{m_1} + c_2) - m_2b_1}{m_1^* - m_2} \right) \quad (3.38)$$

$$\mathbf{g}_3 = \left(\frac{m_2^* \frac{b_1 - c_1}{m_1} + c_2 - b_1}{m_2^* - m_2}, \frac{m_2^* (m_2 \frac{b_1 - c_1}{m_1} + c_2) - m_2 b_1}{m_2^* - m_2} \right) \quad (3.39)$$

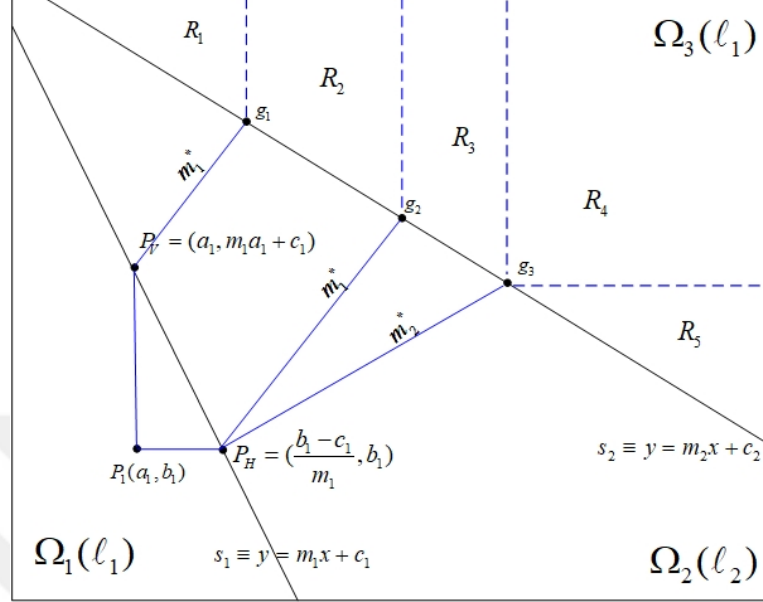


FIGURE 3.4 – An illustrative example of SMDN3- ℓ_1 where Ω_3 is split into five sub-regions.

In addition, an illustration of these five sub-regions is given with Figure 3.4 where we also present the vertical and horizontal projection points on s_1 , i.e. $\mathbf{P}_V$ and $\mathbf{P}_H$, respectively, with their coordinates are defined as below.

$$\mathbf{P}_V = (a_1, m_1 a_1 + c_1) \quad (3.40)$$

$$\mathbf{P}_H = \left(\frac{b_1 - c_1}{m_1}, b_1 \right) \quad (3.41)$$

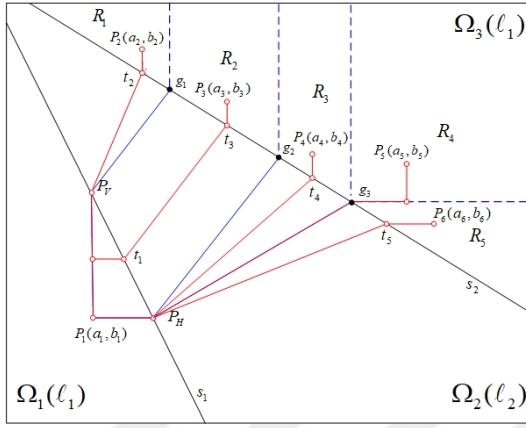
Furthermore, according to the results from Franco et al. (2012) and Franco et al. (2018), optimal slopes of incoming lines m_1^* and m_2^* related to s_1 and s_2 are calculated as follows.

$$m_1^* = \begin{cases} \frac{-m_1 + \sqrt{-2m_1}}{1 + \sqrt{-2m_1}} & \text{if } m_1 < 0 \\ \frac{-m_1 - \sqrt{2m_1}}{1 + \sqrt{2m_1}} & \text{if } m_1 \geq 0 \end{cases} \quad (3.42)$$

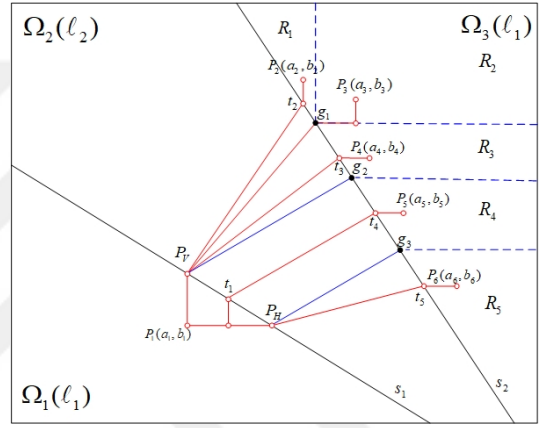
$$m_2^* = \begin{cases} \frac{-m_2 + \sqrt{-2m_2}}{1 + \sqrt{-2m_2}} & \text{if } m_2 < 0 \\ \frac{-m_2 - \sqrt{2m_2}}{1 + \sqrt{2m_2}} & \text{if } m_2 \geq 0, \end{cases} \quad (3.43)$$

As an illustration, the shortest paths between the point $\mathbf{P}_1 = (a_1, b_1) \in \Omega_1$ and five points in five different subregions, i.e. $\mathbf{P}_2 = (a_2, b_2) \in R_1$, $\mathbf{P}_3 = (a_3, b_3) \in R_2$, $\mathbf{P}_4 = (a_4, b_4) \in R_3$, $\mathbf{P}_5 = (a_5, b_5) \in R_4$ and $\mathbf{P}_6 = (a_6, b_6) \in R_5$ are illustrated with Figure 3.5a. The gate-points of the shortest paths between point $\mathbf{P}_1 \in \Omega_1$ and points $\mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4, \mathbf{P}_5$, and $\mathbf{P}_6$ are given with Table 3.1 where the solution of the unknown gate-point t_1 is defined as follows.

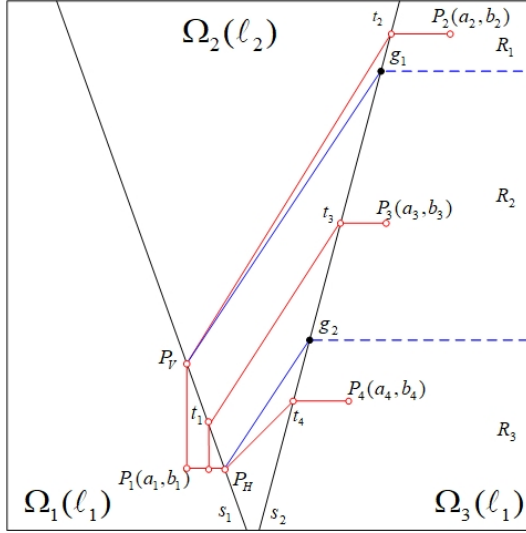
$$\mathbf{t}_1 = \left(\frac{m_1^* a_3 + c_1 - (m_2 a_3 + c_2)}{m_1^* - m_1}, \frac{m_1^* (m_1 a_3 + c_1) - m_1 (m_2 a_3 + c_2)}{m_1^* - m_1} \right) \quad (3.44)$$



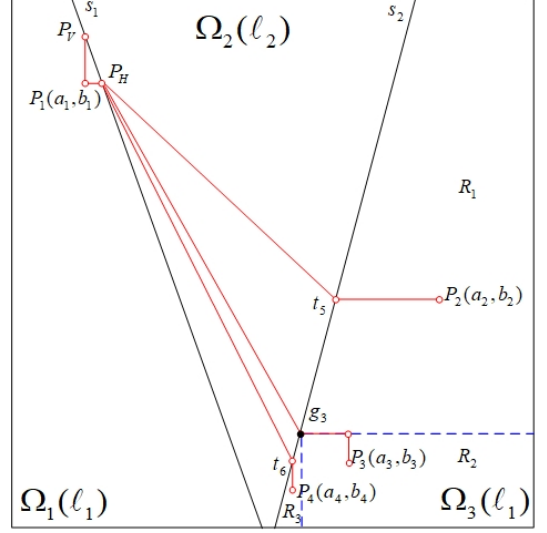
(a) Two separating lines with slopes $m_1 < m_2 < 0$.



(b) Two separating lines with slopes $m_2 < m_1 < 0$.



(c) Two separating lines with slopes $m_1 < 0 < m_2$ and $b_1 < b_2 < b_3 < b_4$.



(d) Two separating lines with slopes $m_1 < 0 < m_2$ and $b_1 > b_2 > b_3 > b_4$.

FIGURE 3.5 – The shortest paths for the SMDN3- ℓ_1 considering different slanted separating line pairs.

After every single case of the formulations of the gate-points in s_1 and s_2 is clarified, the distance of the *three-link* shortest path will be calculated as in Equation (3.36).

TABLE 3.1 – Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_1 < m_2 < 0$.

Path between	Gate-point on s_1	Gate-point on s_2
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_2 \in R_1$	$\mathbf{P}_V$	$\mathbf{t}_2 = (a_2, m_2 a_2 + c_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_3 \in R_2$	$\mathbf{t}_1$	$\mathbf{t}_3 = (a_3, m_2 a_3 + c_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_4 \in R_3$	$\mathbf{P}_H$	$\mathbf{t}_4 = (a_4, m_2 a_4 + c_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_5 \in R_4$	$\mathbf{P}_H$	$\mathbf{g}_3$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_6 \in R_5$	$\mathbf{P}_H$	$\mathbf{t}_5 = (\frac{b_6 - c_2}{m_2}, b_6)$

SMDN3- ℓ_1 with $m_2 < m_1 < 0$:

This section includes the second case of SMDN- ℓ_1 when $m_2 < m_1 < 0$ holds, which has similar characteristics to the first case. Given two separating lines $s_1 \equiv y = m_1 x + c_1$ and $s_2 \equiv y = m_2 x + c_2$ such that $m_2 < m_1 < 0$ holds, the partitioning of the plane is described with Figure 3.5b. Observe that, similar to the first case, Ω_3 is further partitioned into 5 sub-regions, say R_1, R_2, R_3, R_4 and R_5 , which are defined by three separating points $\mathbf{g}_1, \mathbf{g}_2$ and $\mathbf{g}_3$ on the separating line s_2 which are formulated as follows.

$$\mathbf{g}_1 = \left(\frac{m_2^* a_1 + c_2 - (m_1 a_1 + c_1)}{m_2^* - m_2}, \frac{m_2^* (m_2 a_1 + c_2) - m_2 (m_1 a_1 + c_1)}{m_2^* - m_2} \right) \quad (3.45)$$

$$\mathbf{g}_2 = \left(\frac{m_1^* a_1 + c_2 - (m_1 a_1 + c_1)}{m_1^* - m_2}, \frac{m_1^* (m_2 a_1 + c_2) - m_2 (m_1 a_1 + c_1)}{m_1^* - m_2} \right) \quad (3.46)$$

$$\mathbf{g}_3 = \left(\frac{m_1^* \frac{b_1 - c_1}{m_1} + c_2 - b_1}{m_1^* - m_2}, \frac{m_1^* (m_2 \frac{b_1 - c_1}{m_1} + c_2) - m_2 b_1}{m_1^* - m_2} \right) \quad (3.47)$$

Furthermore, formulations of the coordinates of gate-points on the shortest paths between point $\mathbf{P}_1 \in \Omega_1$ and five points $\mathbf{P}_2 \in R_1, \mathbf{P}_3 \in R_2, \mathbf{P}_4 \in R_3, \mathbf{P}_5 \in R_4$ and $\mathbf{P}_6 \in R_5$ are included in Table 3.2 while $\mathbf{t}_1$ is calculated as follows.

$$\mathbf{t}_1 = \left(\frac{m_1^* \frac{b_5 - c_1}{m_1} + c_1 - b_5}{m_1^* - m_1}, \frac{m_1^* (m_1 \frac{b_5 - c_1}{m_1} + c_2) - m_1 b_5}{m_1^* - m_1} \right) \quad (3.48)$$

SMDN- ℓ_1 with $m_1 < 0 < m_2$:

The last case arises when $m_1 < 0 < m_2$ holds for the separating lines. Given any two points $\mathbf{P}_i = (a_i, b_i) \in \Omega_1$ and $\mathbf{P}_j = (a_j, b_j) \in \Omega_3$, we are faced with two different sub-cases where in the first one $b_i < b_j$ is satisfied and in the second one $b_j < b_i$ holds as long as regions are horizontally distributed. These sub-cases are depicted with Figure 3.5c and 3.5d. below.

TABLE 3.2 – Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_2 < m_1 < 0$.

Path between	Gate-points on s_1	Gate-points on s_2
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_2 \in R_1$	$\mathbf{P}_V$	$\mathbf{t}_2 = (a_2, m_2 a_2 + c_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_3 \in R_2$	$\mathbf{P}_V$	$\mathbf{g}_1$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_4 \in R_3$	$\mathbf{P}_V$	$\mathbf{t}_3 = (\frac{b_4 - c_2}{m_2}, b_4)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_5 \in R_4$	$\mathbf{t}_1$	$\mathbf{t}_4 = (\frac{b_5 - c_2}{m_2}, b_5)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_6 \in R_5$	$\mathbf{P}_H$	$\mathbf{t}_5 = (\frac{b_6 - c_2}{m_2}, b_6)$

Note that these sub-cases can also be treated as a single case, but it would be more practical to treat them as separate cases for a more explicit illustration. Properties and the shortest path formulations of these sub-cases are presented below.

Sub-case 1 with $b_i < b_j$:

This sub-case is depicted with Figure 3.5c when finding the shortest path from a point $\mathbf{P}_1 = (a_1, b_1) \in \Omega_1$ to another point $\mathbf{P}_2 = (a_2, b_2) \in \Omega_3$ such that $b_1 < b_2$ holds. Here, Ω_3 is partitioned into three sub-regions, named R_1 , R_2 and R_3 which are defined by using two separating points $\mathbf{g}_1$ and $\mathbf{g}_2$ on the separating line s_2 . Formulations of the coordinates of the separating points are given as follows.

$$\mathbf{g}_1 = \left(\frac{m_1^* a_1 + c_2 - (m_1 a_1 + c_1)}{m_1^* - m_2}, \frac{m_1^* (m_2 a_1 + c_2) - m_2 (m_1 a_1 + c_1)}{m_1^* - m_2} \right) \quad (3.49)$$

$$\mathbf{g}_2 = \left(\frac{m_1^* \frac{b_1 - c_1}{m_1} + c_2 - b_1}{m_1^* - m_2}, \frac{m_1^* (m_2 \frac{b_1 - c_1}{m_1} + c_2) - m_2 b_1}{m_1^* - m_2} \right) \quad (3.50)$$

As an illustration, a random point $\mathbf{P}_1 \in \Omega_1$ and three points $\mathbf{P}_2 \in \Omega_3$, $\mathbf{P}_3 \in \Omega_3$, and $\mathbf{P}_4 \in \Omega_3$ are randomly selected in subregions R_1 , R_2 , and R_3 , respectively, such that $b_1 < b_2 < b_3 < b_4$ holds. Then the formulations of the gate-points of the shortest paths from a point $\mathbf{P}_1 \in \Omega_1$ to points $\mathbf{P}_2 \in R_1$, $\mathbf{P}_3 \in R_2$ and $\mathbf{P}_4 \in R_3$ are given with Table 3.3 where the unknown gate-point $\mathbf{t}_1$ is formulated as follows.

$$\mathbf{t}_1 = \left(\frac{m_1^* \frac{b_3 - c_1}{m_1} + c_1 - b_3}{m_1^* - m_1}, \frac{m_1^* (m_1 \frac{b_3 - c_1}{m_1} + c_1) - m_1 b_3}{m_1^* - m_1} \right) \quad (3.51)$$

Subcase 2 with $b_j < b_i$:

In this sub-case, we try to find the shortest path from a point $\mathbf{P}_i = (a_i, b_i) \in \Omega_1$ to another

TABLE 3.3 – Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_1 < 0 < m_2$.

Path between	Gate-points on s_1	Gate-points on s_2
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_2 \in R_1$	$\mathbf{P}_V$	$\mathbf{t}_2 = (\frac{b_2 - c_2}{m_2}, b_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_3 \in R_2$	$\mathbf{t}_1$	$\mathbf{t}_3 = (\frac{b_3 - c_2}{m_2}, b_3)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_4 \in R_3$	$\mathbf{P}_H$	$\mathbf{t}_4 = (\frac{b_4 - c_2}{m_2}, b_4)$

point $\mathbf{P}_j = (a_j, b_j) \in \Omega_3$ when $b_j < b_i$ is satisfied. Following this sub-case, Ω_3 is partitioned into three sub-regions, named as R_1 , R_2 and R_3 defined by a single separating point $\mathbf{g}_3$ on the separating line s_2 . The formulation of this separating point is illustrated as follows.

$$\mathbf{g}_3 = \left(\frac{m_2^* \frac{b_1 - c_1}{m_1} + c_2 - b_1}{m_2^* - m_2}, \frac{m_2^* (m_2 \frac{b_1 - c_1}{m_1} + c_2) - m_2 b_1}{m_2^* - m_2} \right) \quad (3.52)$$

This sub-case is illustrated with Figure 3.5d. Observe that given $\mathbf{P}_1 \in \Omega_1$ and three points $\mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4 \in \Omega_3$, are selected in sub-regions R_1, R_2 and R_3 , respectively such that $b_1 > b_2 > b_3 > b_4$ holds. Then the gate-points of the shortest paths from $\mathbf{P}_1 \in \Omega_1$ to $\mathbf{P}_2 \in R_1$, $\mathbf{P}_3 \in R_2$ and $\mathbf{P}_4 \in R_3$ are formulated as included in Table 3.4.

TABLE 3.4 – Gate-points of the shortest paths for the SMDN3- ℓ_1 with $m_1 < 0 < m_2$.

Path between	Gate-points on s_1	Gate-points on s_2
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_2 \in R_1$	$\mathbf{P}_H$	$\mathbf{t}_5 = (\frac{b_2 - c_2}{m_2}, b_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_3 \in R_2$	$\mathbf{P}_H$	$\mathbf{g}_3$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_4 \in R_3$	$\mathbf{P}_H$	$\mathbf{t}_6 = (a_4, m_2 a_4 + c_2)$

3.2.2.2 Gate-point formulations of the SMDN3- ℓ_2

Now, we focus on the gate-point formulations of SMDN3- ℓ_2 problem where the plane is partitioned into three regions with two separating lines $s_1 \equiv y = m_1 x + c_1$ and $s_2 \equiv y = m_2 x + c_2$. In this problem, we concentrate on the gate-point formulations for the shortest path for four different sub-cases each presenting a different combination of slopes m_1 and m_2 , satisfying the conditions $m_1 < m_2 < 0$, $m_2 < m_1 < 0$, $m_1 = m_2 < 0$, and $m_1 < 0 < m_2$, respectively. Note that any further sub-cases with different combinations of slopes can be reduced into one of the sub-cases above by simply swapping the slopes of two separating lines. In this section, we first introduce the cases of two separating lines with slopes $m_1 < m_2 < 0$ and $m_2 < m_1 < 0$ for which the calculations of the gate-points

are alike. Second, we present the case of two separating lines with slopes $m_1 = m_2 < 0$. Finally, the case of two separating lines with slopes $m_1 < 0 < m_2$ is addressed.

SMDN3- ℓ_2 with $m_1 < m_2 < 0$ and $m_2 < m_1 < 0$:

Considering the shortest path between two points in nonadjacent regions Ω_1 and Ω_3 , in both cases $m_1 < m_2 < 0$ and $m_2 < m_1 < 0$, optimal gate-points can be found by following the suitable one of three distinct sub-cases, all of which are obtained by further partitioning Ω_3 into sub-regions, i.e. R_1 , R_2 , and R_3 . These sub-regions are defined by two separating points g_2 and g_3 on the separating line s_2 . These sub-regions are described with Figure 3.6 where separating points g_2 and g_3 are the vertical and horizontal projection points of a separating point g_1 on the separating line s_1 , respectively. The coordinates of the separating points g_1 , g_2 , and g_3 are formulated as follows.

$$\mathbf{g}_1 = \left(\frac{m_1^* a_1 + c_1 - b_1}{m_1^* - m_1}, \frac{m_1^* (m_1 a_1 + c_1) - m_1 b_1}{m_1^* - m_1} \right) \quad (3.53)$$

$$\mathbf{g}_2 = \left(\frac{m_1^* a_1 + c_1 - b_1}{m_1^* - m_1}, m_2 \frac{m_1^* a_1 + c_1 - b_1}{m_1^* - m_1} + c_2 \right) \quad (3.54)$$

$$\mathbf{g}_3 = \left(\frac{\frac{m_1^* (m_1 a_1 + c_1) - m_1 b_1}{m_1^* - m_1} - c_2}{m_2}, \frac{m_1^* (m_1 a_1 + c_1) - m_1 b_1}{m_1^* - m_1} \right) \quad (3.55)$$

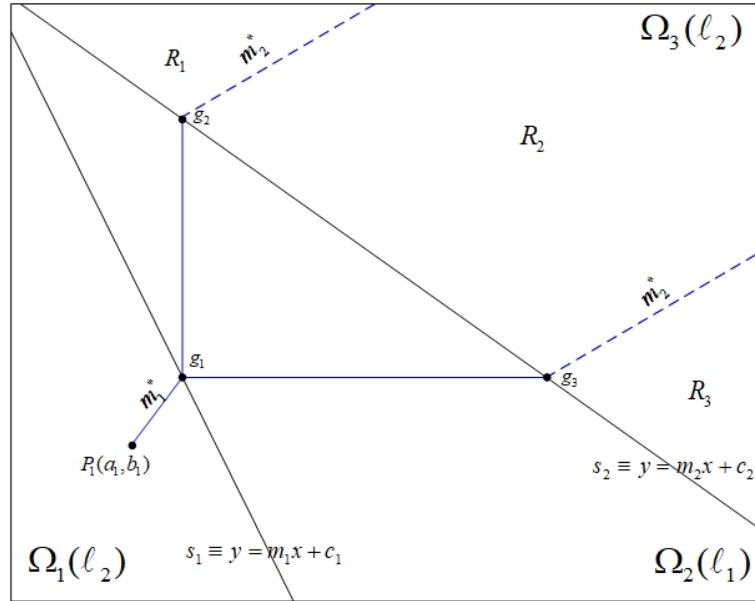


FIGURE 3.6 – An illustrative example of SMDN3- ℓ_2 with three sub-regions of Ω_3 .

More detailed illustration of the shortest paths in SMDN3- ℓ_2 for both cases with two separating lines having slopes $m_1 < m_2 < 0$ and $m_2 < m_1 < 0$ are shown with Figure 3.7a and

3.7b, respectively. Following the instances from Figure 3.7a and Figure 3.7b, for both cases, the coordinates of the gate-points are incorporated in Table 3.5 and the formulation of the unknown gate-point $\mathbf{t}_4$ is calculated as follows.

$$\mathbf{t}_4 = \left(\frac{m_2^* a_3 + c_2 - b_3}{m_2^* - m_2}, \frac{m_2^* (m_2 a_3 + c_2) - m_2 b_3}{m_2^* - m_2} \right) \quad (3.56)$$

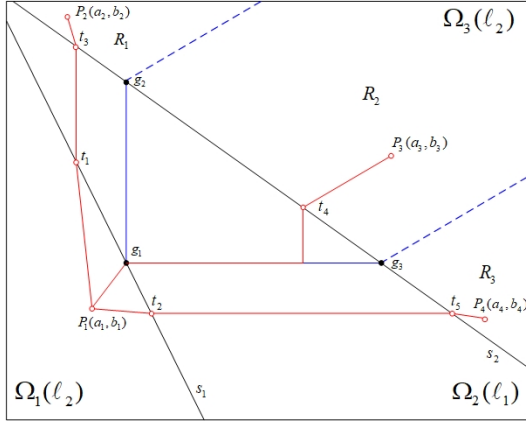
In addition, note that we couldn't find a clear closed form solution of the locations of $\mathbf{t}_1 = (t_{11}, t_{12})$ and $\mathbf{t}_2 = (t_{21}, t_{22})$ which are located within the convex hulls constructed by the points $\mathbf{P}_1$ and $\mathbf{P}_2$ and the points $\mathbf{P}_1$ and $\mathbf{P}_4$, respectively. They can be optimally determined by employing an unconstrained optimization method such as the Golden Section Search algorithm (Bazaraa et al., 2006). Note that we also implement the Bisection and Newton method for this operation. However, according to our observations, the Golden Section Search algorithm arises to be most efficient. Finally, considering the cases of the shortest path in sub-regions R_1 and R_3 , the path between two gate-points in the different separating lines, i.e. $\mathbf{t}_1$ to $\mathbf{t}_3$, and $\mathbf{t}_2$ to $\mathbf{t}_5$ are proven to be vertical and horizontal, respectively, which can also be observed from Figures 3.7a or 3.7b.

TABLE 3.5 – Gate-points of the shortest paths for the SMDN3- ℓ_2 with $m_2 < m_1 < 0$.

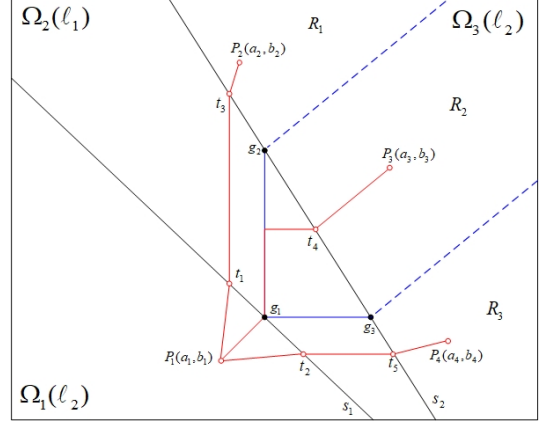
Path between	Gate-points on s_1	Gate points on s_2
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_2 \in R_1$	$\mathbf{t}_1$	$\mathbf{t}_3 = (t_{11}, m_2 t_{11} + c_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_3 \in R_2$	$\mathbf{g}_1$	$\mathbf{t}_4$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_4 \in R_3$	$\mathbf{t}_2$	$\mathbf{t}_5 = (\frac{t_{22} - c_2}{m_2}, t_{22})$

SMDN3- ℓ_2 with $m_1 = m_2 < 0$:

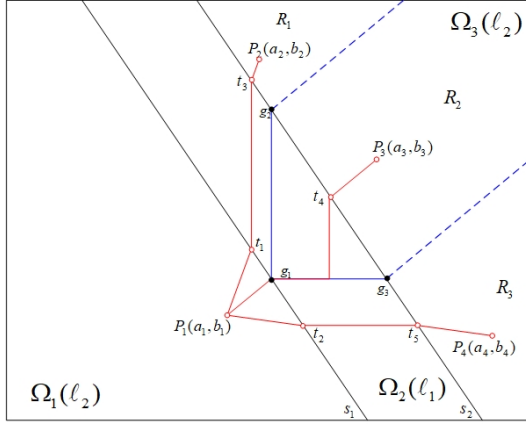
This is a special case of SMDN3- ℓ_2 with $m_1 < m_2 < 0$ and $m_2 < m_1 < 0$ which does not have any sub-case without a closed form gate-point formulation. This case is described with Figure 3.7c. Similar to the previous case, there are three sub-cases of the shortest path between two points located in non-adjacent regions defined by using three sub-regions R_1 , R_2 and R_3 , respectively. These sub-regions are defined by using separating points $\mathbf{g}_1$, $\mathbf{g}_2$, and $\mathbf{g}_3$. The coordinates of the separating point $\mathbf{g}_1$ on the separating line s_1 , and two separating points $\mathbf{g}_2$ and $\mathbf{g}_3$ on the separating line s_2 are formulated as follows.



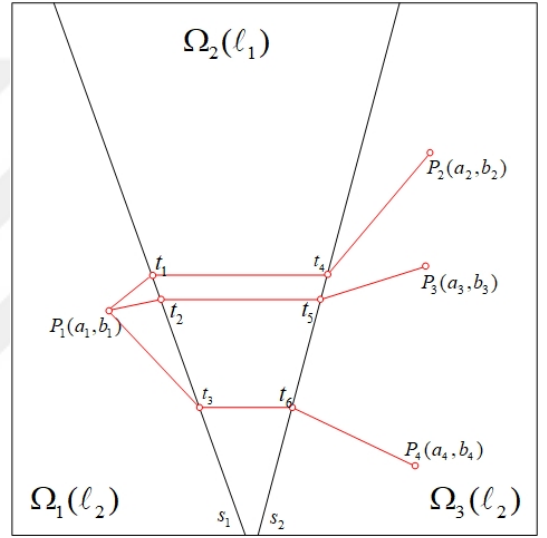
(a) Two separating lines with slopes $m_1 < m_2 < 0$.



(b) Two separating lines with slopes $m_2 < m_1 < 0$.



(c) Two separating lines with slopes $m_1 = m_2 < 0$.



(d) Two separating lines with slopes $m_1 < 0 < m_2$.

FIGURE 3.7 – The shortest paths for the SMDN3- ℓ_2 separated by different line pairs with various slopes.

$$\mathbf{g}_1 = \left(\frac{m_1^* a_1 + c_1 - b_1}{m_1^* - m_1}, \frac{m_1^* (m_1 a_1 + c_1) - m_1 b_1}{m_1^* - m_1} \right) \quad (3.57)$$

$$\mathbf{g}_2 = \left(\frac{m_1^* a_1 + c_1 - b_1}{m_1^* - m_1}, m_2 \frac{m_1^* a_1 + c_1 - b_1}{m_1^* - m_1} + c_2 \right) \quad (3.58)$$

$$\mathbf{g}_3 = \left(\frac{\frac{m_1^* (m_1 a_1 + c_1) - m_1 b_1}{m_1^* - m_1} - c_2}{m_2}, \frac{m_1^* (m_1 a_1 + c_1) - m_1 b_1}{m_1^* - m_1} \right) \quad (3.59)$$

The gate-points are presented with Table 3.6 where $\mathbf{t}_1$ ($\mathbf{t}_2$) can be calculated in a straightforward way since it has been proven from our calculations that the path between $\mathbf{P}_1$ and

$\mathbf{t}_1$ and, $\mathbf{P}_2$ and $\mathbf{t}_3$ ($\mathbf{P}_1$ and $\mathbf{t}_2$ and, $\mathbf{P}_4$ and $\mathbf{t}_5$) have the same slopes. The gate point $\mathbf{t}_4$ is calculated as follows.

$$\mathbf{t}_4 = \left(\frac{m_2^* a_3 + c_2 - b_3}{m_2^* - m_2}, \frac{m_2^* (m_2 a_3 + c_2) - m_2 b_3}{m_2^* - m_2} \right). \quad (3.60)$$

TABLE 3.6 – Gate-points of the shortest paths for the SMDN3- ℓ_2 with $m_1 = m_2 < 0$.

Path between	Gate-points on s_1	Gate-points on s_2
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_2 \in R_1$	$\mathbf{t}_1$	$\mathbf{t}_3 = (t_{11}, m_2 t_{11} + c_2)$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_3 \in R_2$	$\mathbf{g}_1$	$\mathbf{t}_4$
$\mathbf{P}_1 \in \Omega_1$ and $\mathbf{P}_4 \in R_3$	$\mathbf{t}_2$	$\mathbf{t}_5 = (\frac{t_{22} - c_2}{m_2}, t_{22})$

SMDN3- ℓ_2 with $m_1 < 0 < m_2$

From this case it is illustrated that the shortest path between two points that resides on non-adjacent regions travels horizontally through the region Ω_2 using ℓ_1 norm as in Figure 3.7d. Furthermore, the gate-point locations in such paths lack a closed form solution. Therefore, the gate points on both s_1 and s_2 for each shortest path can be easily determined using an unconstrained optimization approach, such as the Golden Section Search algorithm (Bazaraa et al., 2006).

4 SOLUTION METHODS FOR SMDN3

This chapter includes the solution methods tailored for two variants of SMDN3 i.e. SMDN3- ℓ_1 and SMDN3- ℓ_2 , respectively. However, these solution approaches are also viable for other variants of SMDN problems with minor adjustments as well.

First, as an exact solution approach, the Big Square Small Square Algorithm (BSSS) (Hansen et al., 1981) is adapted to solve SMDN as a geometric branch-and-bound algorithm (B&B), since it is relatively practical to apply to SMDN3 provided that the distance functions are still convex. Second, alternative efficient heuristic approaches, i.e, the Discrete Approximation Algorithm (DA) and the Modified Weiszfeld Algorithm (MW), are developed to solve SMDN3.

In addition, two meta-heuristic approaches, namely Simulated Annealing (SA) and Tabu Search (TS) algorithms with their new local search scheme, and one heuristic approach based on BSSS, namely Beam Search (BS), are proposed for efficiently solving large scale instances of the SMDN3.

4.1 An Exact Solution Method

Since commercial solvers using the MINLP model are proven to be unreliable in solving SMDN3, BSSS is developed as a geometric B&B algorithm, which is also applicable to all cases of SMDN provided that the distance functions are convex. While this algorithm also tends to consume significant computation time depending on the size of the instance or its parameters, its average performance is significantly better than commercial solvers and also yields similar efficiency for the SMDN3 cases when it is not possible to explicitly formulate the problem. Recall that the SMDN3 can only be represented using a mathematical programming formulation when both separating lines are vertical (or horizontal).

4.1.1 Big Square Small Square Algorithm

BSSS algorithm is introduced for the obnoxious facility location problem by Hansen et al. (1981). This well-known global optimization method has been taken into consideration by many researchers (Zaferanieh et al., 2008; Drezner et al., 2016; Akyüz et al., 2019) while solving various problems. The structure of the algorithm is stated as follows. The algorithm begins by forming a larger square (or rectangle), which is obtained from the rectangular hull of the demand points. Throughout iterations, the large square is repeatedly divided into four equally sized smaller squares at each iteration. Each square is identified by its lower and upper bound values. Squares with high lower bound values are labeled as non-promising squares and removed from further branching. The best upper bound value is compared to the lower bound value of each square to evaluate if they are worth further branching. The algorithm breaks the plane into smaller squares until a predetermined stopping condition is satisfied. Finally, an ϵ -optimal solution is obtained after reaching the stopping condition. The stopping condition is defined as follows. When the size of the side length of the smallest square is shorter than a pre-defined ϵ value, the BSSS algorithm stops the branching procedure (Hansen et al., 1985). Another alternative stopping criterion is also employed in the literature (Plastria, 1992), which terminates the algorithm when the best upper bound value is within a given percentage of the lowest (best) lower bound value from the last iteration. Taking into account our initial computational tests, we decided to employ the first stopping rule in our implementation.

In our algorithm, we divide the plane using rectangles as convex quadrilaterals, following the footsteps of Plastria (1992). The objective function value where the facility is situated in the center of a rectangle provides the upper bound value (UB) by calculating the objective function value of the SMDN3 for that specific rectangle. In addition, the lower bound value (LB) is calculated in such a way that it gives a lower value compared to all possible feasible solutions by locating the facility within the rectangle. LB is obtained by simply adding and closest distances from every demand point to the rectangle. Equations (4.1) and (4.2) are used to further explain the LB calculation method, while the distances between the rectangle and the customer locations are illustrated with Figure 4.1.

The BSSS algorithm we developed is structured as follows. After the large rectangle is formed considering the rectangular hull, this rectangle is divided into four equally sized rectangles at the beginning of the BSSS, along with the corresponding UB and LB values. Following the iterations, each of the bigger rectangles obtained from the previous iteration

is separated into four new rectangles. However, it is only necessary to further segment the rectangles whose lower bound values are less than the best-known upper bound value. Furthermore, as long as a rectangle's longer side length exceeds the designated tolerance value or ϵ , it is repeatedly divided into four new ones. Finally, whenever a better solution is discovered, the coordinates of the best-known facility location, $\mathbf{x}^*$, and the best-known upper bound, UB^* , are updated. These actions are taken until the BSSS is terminated, at which point the ϵ -optimal location solution for SMDN3 has been obtained.

Algorithm 1 depicts the BSSS's outline. The smallest rectangle that includes all customer locations R_0 , forms the rectangular hull that serves as the starting point of the algorithm. The rectangle list L is then set to $L = \{R_0\}$. The side length value ϕ is the length of the longer side of the rectangle R_0 . This value is reduced by partitioning the rectangles in each iteration.

Algorithm 1 BSSS algorithm for SMDN3.

```

1: Input : Weights of demand points  $w_k$ , coordinates of demand points  $\mathbf{c}_k = (c_{k1}, c_{k2})$  as  $k \in \mathcal{K}$ 
2: Output : Optimal location  $x^*$  and optimal solution  $UB^*$ 
3: Initialization : Generate rectangular hull of demand points and assign as  $R_0$  and define the length of its longer edge as  $\phi$ .
4: Find  $x^*$  and  $UB^*$  from  $R_0$  by locating in the center. Add  $R_0$  into the list of big rectangles as  $L = \{R_0\}$  and define the small rectangles list as  $L' = \emptyset$ 
5: while  $\phi \geq \epsilon$  do
6:   for  $\bar{R}_i \in L$  do
7:     Partition big rectangle  $\bar{R}_i$  into 4 equally sized smaller rectangles as  $R_j$ 
8:     for each  $R_j$  do
9:       Find  $LB_j$  from Equation (4.1)
10:      if  $LB_j < UB^*$  then
11:        Calculate  $UB_j$ 
12:        if  $UB_j < UB^*$  then
13:           $UB^* = UB_j$  and  $x^* = x_j$ 
14:        end if
15:        Add rectangle  $R_j$  to the small rectangles list as  $L' = L' \cup \{R_j\}$ 
16:      end if
17:    end for
18:     $L = L \setminus \{\bar{R}_i\}$ 
19:  end for
20:  Set  $L = L \cup L'$ ,  $L' = \emptyset$  and  $\phi = \phi/2$ 
21: end while

```

Here is how the branching process is carried out. Following selection, a large rectangle $\bar{R}_i \in \mathcal{L}$ is divided into four equal little rectangles, R_j for $j = 1, \dots, 4$. Each of which with UB_j and LB_j , i.e. the upper bound and the lower bound values, respectively. First, $L' = \emptyset$ is used to initialize a temporary list of these small rectangles. Subsequently, we examine every rectangle R_j for each $j = 1, \dots, 4$. L' is updated as $L' = L' \cup \{R_j\}$ as long as $LB_j < UB^*$ is

satisfied. Afterward, the rectangle list is updated as $L = L \cup L'$ at the end of every iteration, removing the old large rectangles. When the longer side lengths of every rectangle in L fall below the threshold value ϵ , the algorithm ends.

Now we describe the details of the lower bound calculation process. According to Hansen et al. (1985), the lower bound value of a rectangle, or LB , is determined as the total weighted distance between each customer and their respective closest point to that rectangle. By using this method, a lower bound is produced which consists of the sum of the shortest distance between each demand point and the rectangle. Additionally, since the facility is assumed to be positioned inside the rectangle, the demand points inside the rectangle are ignored. Since the distance functions of SMDN are convex, the lower bound scheme of BSSS is useful for any variation of SMDN.

In short, the lower-bound computations take into account the distance between a demand point and the facility located inside the rectangle in three different scenarios. When the demand point is inside the rectangle, the first scenario occurs. Therefore, for the sake of the lower bound computations, the related customers' distance would be assumed to be zero, and they can be disregarded. In the second scenario, the demand point is located outside of the rectangle, and one of the four corner points of the rectangle is the closest point that connects the demand point's location to the rectangle. In this instance, finding the rectangle's closest corner points and figuring out the shortest paths between this corner point and the demand point are essential. Finally, the third scenario occurs when the closest point is on one of the sides of the rectangle while the demand point is outside of the rectangle. The horizontal or vertical distance between the demand point and its closest point on the side with the same ordinate or abscissa as the position of the customer is the shortest distance in this scenario.

Let $\mathbf{p}_j$ represent the point on the rectangle R that is closest to the demand point $\mathbf{c}_j$ for $l = 1, 2, 3$. then, for each $j \in J_l$ and $l = 1, 2, 3$, $d^l(\mathbf{c}_j, \mathbf{p}_j) \leq d^l(\mathbf{c}_j, \mathbf{x}^R)$ holds for the distance functions. When the facility needs to be opened inside the rectangle R , let $\mathbf{x}^R$ be its ideal location. Formulation of LB can be expressed as follows.

$$LB = \sum_{l=1}^3 \sum_{j \in J_l} d^l(\mathbf{c}_j, \mathbf{p}_j) \leq \sum_{l=1}^3 \sum_{j \in J_l} d^l(\mathbf{c}_j, \mathbf{x}^R). \quad (4.1)$$

The lower bound is computed based on the three above-mentioned scenarios, as illustrated with Figure 4.1, which also identifies the closest points in the rectangle that are nearest to each of the demand points. Following Figure 4.1, it can be observed that there are demand

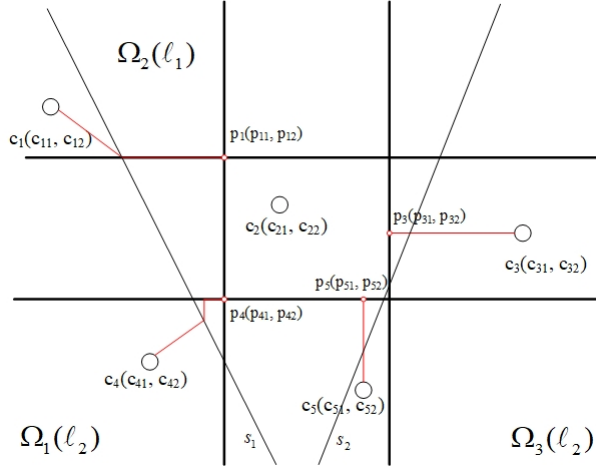


FIGURE 4.1 – An illustration of separate distances taken for lower bound.

points c_1 , c_2 , c_3 , c_4 , and c_5 , and the closest points on the rectangle to each customer are denoted by p_1 , p_2 , p_3 , p_4 , and p_5 , respectively. It can also be seen that p_2 and c_2 coincide, thus $p_2 \equiv c_2$. Consequently, the distance $d^2(c_2, p_2) = 0$ holds, which falls into the first scenario. In the second scenario, the lower bound distances are computed as the shortest path distances $d^1(c_1, p_1)$ and $d^1(c_4, p_4)$, respectively, for customers c_1 and c_4 , which are located in region Ω_1 . In the final scenario, the demand points c_3 and c_5 are located in the region Ω_3 . The shortest paths $d^3(c_3, p_3)$ and $d^3(c_5, p_5)$ correspond to the horizontal and vertical paths from c_3 and c_5 to the points p_3 and p_5 on the right and bottom sides of the rectangle, respectively. We can calculate the lower bound as follows.

$$LB = d^1(c_1, p_1) + |p_{31} - c_{31}| + d^1(c_4, p_4) + |p_{52} - c_{52}|. \quad (4.2)$$

Similarly, let $x^d(x_1^d, x_1^d)$ be the intersection point of the diagonals of a rectangle, then the upper-bound value (UB) of this rectangle is calculated as follows.

$$UB = \sum_{l=1}^3 \sum_{j \in J_l} d^l(c_j, x^d). \quad (4.3)$$

4.2 Heuristic Methods

In this section, we introduce approximate solution procedures for the two variants of SMDN3 i.e. SMDN3- ℓ_1 and SMDN3- ℓ_2 , respectively. In our first work, two straightforward approxi-

mate methods are considered to solve SMDN3 with significant efficiency. The first one, i.e., the Modified Weiszfeld's Algorithm (MW), is a highly efficient algorithm that yields low accuracy, while the second one, is the Discrete Approximation (DA), which is a less efficient method with better accuracy.

Later, for solving larger instances of SMDN3 we consider meta-heuristic approaches. We propose two tailor-made meta-heuristic algorithms, i.e., Tabu Search (TS) and Simulated Annealing (SA) schemes, and an accurate B&B-based heuristic algorithm, namely Beam Search (BS), when dealing with larger-scale instances of SMDN3.

4.2.1 Modified Weiszfeld Algorithm

Considering Modified Weiszfeld Algorithm, which is based on the widely used Weiszfeld algorithm (Weiszfeld, 1937). For SMDN2, the MW was initially suggested by Parlar (1994). To fit the SMDN3, we expanded and modified the MW. The algorithm is structured as follows. The algorithm starts with locating the facility on the center of gravity point the customer coordinates in our implementation. After that, the MW proceeds similar to the well-known Weiszfeld method until no further advances are possible using Equations 4.3 and 4.4. It should be noted that in our problem, the distance function is determined by taking into account the lengths of the shortest routes connecting the various regions with various norms. The following equations are the recursive equations used by the MW to update the position of the facility at each iteration.

$$x_1^{(t+1)} = \frac{\sum_{j=1}^J c_{j1} w_j / d(\mathbf{x}^{(t)}, \mathbf{c}_j)}{\sum_{j=1}^J w_j / d(\mathbf{x}^{(t)}, \mathbf{c}_j)} \quad (4.4)$$

$$x_2^{(t+1)} = \frac{\sum_{j=1}^J c_{j2} w_j / d(\mathbf{x}^{(t)}, \mathbf{c}_j)}{\sum_{j=1}^J w_j / d(\mathbf{x}^{(t)}, \mathbf{c}_j)} \quad (4.5)$$

The iteration number is denoted by t , and the shortest path distance between the current facility $\mathbf{x}^{(t)}$ and the demand point $\mathbf{c}_j$ is defined by $d(\mathbf{x}^{(t)}, \mathbf{c}_j)$ with the demand of w_j . The facility's coordinates at $\mathbf{x}^{(t+1)}$ are updated using the coordinates from the previous iteration, $\mathbf{x}^{(t)}$. Iterations proceed until there is no discernible difference between two successive iterations in the objective function value, which is determined using the current position of the facility.

Since the distance function $d(\mathbf{x}^{(t)}, \mathbf{c}_k)$ produces a non-convex objective function (Parlar, 1994) even though it is convex for the original WP, there is no guarantee that the MW iterations will converge to the optimality and they may become trapped at a local optimum solution. According to our experiments, MW tends to converge within the early iterations. Therefore, we additionally impose an iteration limit of 100 to ensure preventing unnecessary iterations.

4.2.2 Discrete Approximation Algorithm

The discrete approximation is a widely used method to simplify problems on the continuous plane at the cost of minor accuracy loss. Hansen et al. (1998) propose the discrete approximation approach to discover approximate solutions to the multi-source WP, based on the resolution of a formulation of the p-median problem. In their work, the use of customer locations as the median sites yields extremely promising results. Also, Aras et al. (2007) develop approximate solution methods for the Capacitated Multi-facility Weber Problem (CMWP) using the ℓ_p norm as $1 \leq p \leq 2$ following the work of Hansen et al. (1998). Considering the ℓ_1 norm distance approximation, the authors used a Mixed-Integer Linear Programming (MILP) formulation to optimally solve the CMWP. Their MILP formulation is based on the following result. The optimal facility location of the WP with rectilinear distance lies in one of the intersection points that are obtained by drawing vertical and horizontal lines on the customer locations (Wendell and Hurter, 1973). These approximation ideas are then adopted for the multi-commodity CMWP with ℓ_1 and ℓ_2 norms by Akyüz et al. (2012), for the CMWP with polyhedral barriers and ℓ_2 norm by Akyüz, (2017) and the CMWP with ℓ_r norms as $1 \leq r \leq \infty$ by Akyüz et al. (2014). The following result forms the basis for their MILP formulation. By drawing vertical and horizontal lines using the customer locations, one can determine the optimal facility location of the WP with rectilinear distance (Wendell and Hurter, 1973). The multi-commodity CMWP with ℓ_1 and ℓ_2 norms (Akyüz et al., 2012), the CMWP with polyhedral barriers and ℓ_2 norm (Akyüz, 2017), and the CMWP with ℓ_r norms as $1 \leq r \leq \infty$ (Akyüz et al., 2014) are all adoptions of approximation ideas.

In this work, the Discrete Approximation Algorithm (DA) uses a straightforward but effective modification of the single-facility WP approximation method to the SMDN3. Using the findings of Wendell and Hurter (1973), the DA creates a grid by drawing horizontal and vertical lines using the coordinates of each customer. To open the facility, a set of candidate

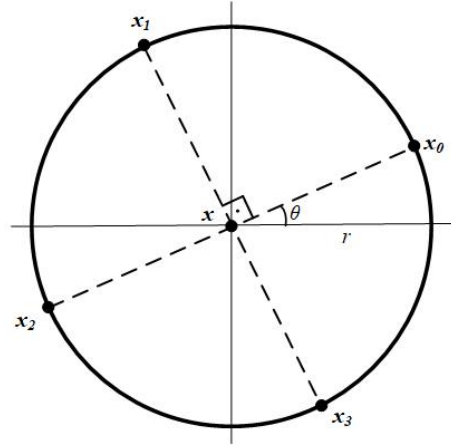
points is selected from the intersection points of the grid. Since there is only one facility to open, it is not necessary to solve a MILP model for DA. Instead, we place the facility in the best candidate site using an enumeration method that finds the candidate site with the lowest value of the objective function. Note that it is prohibitive to include intersection points in every grid generated by using the locations of all customers as candidate facility sites. To address this issue, we limit the total number of candidate sites to be examined to 100,000. First, all of the customer locations are considered as candidate locations. Second, until the maximum number of candidate sites is achieved, the remaining grid points are added to the collection of candidate sites, taking into account the construction order.

4.2.3 Meta-heuristic Solution Approaches

In this section, two efficient meta-heuristic algorithms, i.e., Tabu Search (TS) and Simulated Annealing (SA), are introduced with two local search schemes, i.e., Circular Neighborhood (CN) and Focused Neighborhood (FN) search methods. SA employs both local search algorithms, while TS uses only the CN since the latter does not provide significant improvement to the performance of TS.

4.2.3.1 Circular Neighborhood

The structure of Circular Neighborhood Search (CN) is described as follows. Starting from a point $\mathbf{x}$ located in the plane, a circle is constructed centered on $\mathbf{x}$ with radius r , and an angle θ is randomly selected from the interval $[0, \pi/2]$. From the starting point, using the angles θ , $\theta + \pi/2$, $\theta + \pi$ and $\theta + 3\pi/2$ to four new neighbor solutions $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ are generated on the edge of the circle. Note that these neighbor solutions indicate a possible new location of the facility and among four neighbor solutions, i.e. $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$, the one which yields the minimum objective function is selected. Since both r and θ are input parameters, we denote CN by $N_C(\mathbf{x}, r, \theta)$. An illustrative example of $N_C(\mathbf{x}, r, \theta)$ is depicted with Figure 4.2. Note that given an angle θ , we first fix the location of $\mathbf{x}_0$ and then the locations of the other neighbor solutions $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ are selected considering the angles $\theta + \pi/2$, $\theta + \pi$ and $\theta + 3\pi/2$.

FIGURE 4.2 – Circular Neighborhood $N_C(\mathbf{x}, r, \theta)$

4.2.3.2 Focused Neighborhood

Focused Neighborhood Search (FN) is a local search method that intensifies the search starting from the results of CN. Whenever CN ($N_C(\mathbf{x}, r, \theta)$) is applied, it is possible to explore other neighbor solutions by means of the FN scheme that can be considered a successive application of circular neighborhood in the direction of improvement. To this end, among the four neighbor solutions $\mathbf{x}_0$, $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ assessed by $N_C(\mathbf{x}, r, \theta)$, we focus on two neighbor solutions with the lowest values of the objective function. Let us denote the solution with the smallest objective value and the solution with the second smallest objective value as $\mathbf{x}_k$ and $\mathbf{x}_l$, with angles θ_k and θ_l , respectively. Then we take into account two neighbor points $\mathbf{v}_k$ and $\mathbf{v}_l$, on a circle centered at $\mathbf{x}_k$ (with the smallest objective), considering the directions of the angles θ_k and θ_l , respectively, as depicted with Figure 4.3. Note that the distances from $\mathbf{x}_k$ to the points $\mathbf{v}_k$ and $\mathbf{v}_l$ is equal to the radius of the circle centered at $\mathbf{x}_k$ denoted with $r_0 = \varrho \times r$ where ϱ is a parameter selected within the interval $(0, 1)$. Therefore, r_0 is smaller than r . As a result, the search space is narrowed every time an improvement occurs. Then, we consider the circular arc between $\mathbf{v}_k$ and $\mathbf{v}_l$ and partition this arc into four equidistant arcs by choosing three candidate points, i.e. $\mathbf{v}_0$, $\mathbf{v}_1$ and $\mathbf{v}_2$. Then we evaluate the objective function values of five points, i.e. $\mathbf{v}_k$, $\mathbf{v}_0$, $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_l$. In case none of them yields a better objective function value than $\mathbf{x}_k$ does, we stop the algorithm and continue with the circular neighborhood scheme starting with point $\mathbf{x}_k$, namely $N_C(\mathbf{x}_k, r, \theta)$. Otherwise, we carry on with the point that yields the minimum objective function value and further explore the neighboring points using the FN scheme. For the sake of conciseness, the FN is also represented by $N_F(\mathbf{x})$ in the sequel.

FIGURE 4.4 – Successive Implementation of the Focused Neighborhood

Note that we can successively apply the FN as long as improved solutions are obtained. Figure 4.4 presents an illustrative example in which the FN is repeated three times. In the first run, suppose that $\mathbf{x}_0$ and $\mathbf{x}_3$ are the best and the second best solutions obtained with $N_C(\mathbf{x}, r, \theta)$, respectively. Then FN starts by constructing a circle centered at $\mathbf{x}_0$ and with radius r_0 . Assume that candidate solutions $\mathbf{v}_0$ and $\mathbf{v}_1$ are the best and second best solutions among candidate points on the arc between $\mathbf{v}_k$ and $\mathbf{v}_l$, in the second run, the algorithm focuses on generating new neighbor points i.e. $\mathbf{u}_k$ and $\mathbf{u}_l$ which are on the circle centered at $\mathbf{v}_0$ with radius r_1 such that $r_1 = \varrho \times r_0$ holds where $\varrho \in (0, 1)$. Here, the directions from $\mathbf{x}_0$ towards $\mathbf{v}_0$ and $\mathbf{v}_1$ are considered to determine the directions from $\mathbf{v}_0$ towards candidate points $\mathbf{u}_k$ and $\mathbf{u}_l$. Finally, in the third run of the FN, assuming the best objective function is obtained with $\mathbf{u}_l$, we take into account candidate solutions in the circular arc between $\mathbf{w}_k$ and $\mathbf{w}_l$ which are on the circle centered at $\mathbf{u}_l$ with radius $r_2 = \varrho \times r_1$ such that $\varrho \in (0, 1)$. Note that we address the directions from $\mathbf{v}_0$ towards two candidates points, with the smallest objective function values, on the circle centered at $\mathbf{v}_0$ to determine respectively the directions

of $\mathbf{w}_k$ and $\mathbf{w}_l$ which are on the circle centered at $\mathbf{u}_l$. In case the algorithm could not find a better solution than $\mathbf{u}_l$ among the candidate points on the circular arc between $\mathbf{w}_k$ and $\mathbf{w}_l$, it selects the point with the minimum objective function value, say $\mathbf{u}_l$. Finally, the local search procedure continues by shifting back into $N_C(\mathbf{u}_l, r, \theta)$ starting from $\mathbf{u}_l$.

4.2.3.3 Tabu Search

Tabu search (TS) is a popular meta-heuristic algorithm that keeps previous solutions in short-term memory to diversify the local search process (Glover and Laguna, 1998). In the TS, diversification is ensured by the so-called *tabu list*, which prohibits a certain search space from consideration for a predefined number of iterations. It has been underlined (Boussaïd et al., 2013) that the TS algorithm is mainly applied to combinatorial optimization problems. Since we focus on a continuous optimization problem, i.e. SMDN3, we suggest some modifications to the generic TS framework by Glover and Laguna (1998). In our implementation, we use the circular neighborhood $N_C(\mathbf{x}, r, \theta)$ where the radius r is updated at every Γ iterations to diversify the search space. Parameters of CN are selected as follows. The initial radius $\bar{r}$ of the circular neighborhood is selected as $\bar{r} = (H + L)/20$ where H and L are the height and length of the rectangular hull of customer set.

TABLE 4.1 – Notation used for the TS

ξ	iteration counter
Ξ	iteration limit
η	iteration counter for intensification
χ	iteration limit for intensification
r_t	radius for intensification
μ	radius update parameter for intensification
ν	iteration counter to update radius r
Γ	iteration limit to update radius r
Θ	tabu tenure
$\mathcal{L}$	tabu list
ω	size of the tabu list
Z^*	the best known objective function value
Z'	the objective function of the candidate solution
$\mathbf{x}^*$	the best known solution (facility location)
$\mathbf{x}'$	the candidate solution

Table 4.1 incorporates the notation used for the proposed TS. Throughout the execution of the TS algorithm, we maintain a tabu list $\mathcal{L}$ that contains solutions that we have previously encountered. Each element of $\mathcal{L}$ is a vector $(\mathbf{x}, r)$ that forbids the circular neighborhood

of the previous solution $\mathbf{x}$ within the radius of r , which we call the “tabu area”. In TS, the initial location of the facility is randomly selected within the rectangular hull of the customer locations. Unlike a tabu solution in a combinatorial problem, the tabu list in this study avoids all feasible solutions within the circle defined by $(\mathbf{x}, r)$ and considers them forbidden. This requires checking whether a neighbor solution is within a tabu area or not. When the number of elements in the tabu list $\mathcal{L}$ reaches the tabu tenure limit Θ , the oldest solution is deleted from the list. In addition, a promising candidate solution $\mathbf{x}'$ with an objective function smaller than the value of the objective function of the best-known solution, that is, $Z' < Z^*$, is detected by the circular neighborhood $N_C(\mathbf{x}, r, \theta)$, then this solution is accepted and further neighborhood search operations are applied starting with $\mathbf{x}'$. If such a case occurs for a solution from the tabu area, then the tabu area is ignored as part of the intensification procedure and continue to employ the circular neighborhood $N_C(\mathbf{x}', r_t, \theta)$ for additional χ iterations during which the radius of the neighborhood r_t is initialized as $r_t = \mu \times r$ and updated as $r_t \leftarrow \mu \times r_t$ after each iteration as $\mu \in (0, 1)$. When the TS algorithm is terminated, the best solution is reported. The steps of TS are introduced in Algorithm 2.

Algorithm 2 Tabu Search (TS).

```

1: Input :  $\bar{r}, \Theta, \Xi, \Gamma, \chi$ 
2: Initialization :  $\nu \leftarrow 0, r \leftarrow \bar{r}$ , and  $\mathcal{L} \leftarrow \emptyset$ 
3: Randomly select a location  $\mathbf{x}$ , set  $Z \leftarrow f(\mathbf{x})$ ,  $\mathbf{x}^* \leftarrow \mathbf{x}$  and  $Z^* \leftarrow Z$ 
4: while  $\xi < \Xi$  do                                     \* Stop when the iteration limit is reached
   \*
5:    $\mathbf{x}' \leftarrow N_C(\mathbf{x}, r, \theta)$ ,  $Z' \leftarrow f(\mathbf{x}')$  \* Neighbor solutions  $\mathbf{x}_c^n \in \mathcal{L}$  satisfying  $Z_c^n > Z^*$  are
   ignored \*
6:   if  $\mathbf{x}' \neq \emptyset$  then
7:     Add vector  $(\mathbf{x}, r)$  to  $\mathcal{L}$ 
8:     if  $\mathbf{x}' \in \mathcal{L}$  and  $Z' < Z^*$  then                       \* If  $\mathbf{x}'$  satisfies aspiration criterion
       \*
9:        $\eta \leftarrow 0, r_t \leftarrow \mu \times r$ 
10:      while  $\eta < \chi$  do
11:         $\mathbf{x}' \leftarrow N_C(\mathbf{x}', r_t, \theta)$  and  $Z' \leftarrow f(\mathbf{x}')$  \* Intensification of the search
       \*
12:       $\eta \leftarrow \eta + 1, r_t \leftarrow \mu \times r_t$ 
13:    end while
14:  end if
15:   $\mathbf{x} \leftarrow \mathbf{x}'$  and  $Z \leftarrow Z'$ 
16:  if  $Z < Z^*$  then
17:     $\mathbf{x}^* \leftarrow \mathbf{x}, Z^* \leftarrow Z$  \* Update the best solution \*
18:  end if
19: end if
20: if  $\omega > \Theta$  then \* Remove old elements from the list  $\mathcal{L}$ 
   \*
21:   Remove the first vector of  $\mathcal{L}$ 
22: end if
23: if  $\nu > \Gamma$  then
24:    $\nu \leftarrow 0, r \leftarrow r \times \alpha$ 
25: end if
26:  $\xi \leftarrow \xi + 1$  and  $\nu \leftarrow \nu + 1$ ,
27: end while
28: Output :  $\mathbf{x}^*$  and  $Z^*$ 

```

TABLE 4.2 – Notation used for the SA

β	parameter for geometric cooling parameter
θ	parameter (angle) used within N_C
$\bar{r}$	parameter for initial radius
$\bar{T}$	parameter for initial temperature
T	current temperature value
J	number of regions
κ	the limit on the number of non-improving iterations
Z^*	the best known objective function value
$\tilde{Z}$	the objective function of the candidate solution
$\mathbf{x}^*$	the best known solution (facility location)
$\tilde{\mathbf{x}}$	the candidate solution
$\Psi(\mathbf{x})$	the region where location $\mathbf{x}$ lies
v_j	counter for iterations searching region Ω_j for $j \in J$
σ	the maximum number of iterations to search a region
q	switch for neighborhood search scheme ($q = 0$ for N_C , $q = 1$ for N_F)
τ	counter for iterations without improving the best-known solution
p_a	acceptance probability
ϵ	minimum cooling temperature
Δ	difference between the candidate and previous objective function values
γ	a random number within $[0, 1]$

4.2.3.4 Simulated Annealing

Simulated Annealing (SA) algorithm is a famous meta-heuristic algorithm that dates back to the 1980s (Kirkpatrick et al., 1983). The SA algorithm is recognized for its high performance in many problems (Nikolaev and Jacobson, 2010; Franzin and Stützle, 2019) and for its ease of implementation. Table 4.2 incorporates the notation used for the SA. Throughout the algorithm's execution, both promising and unpromising solutions are accepted with a probability p_a determined by the temperature parameter T and the difference in the objective function Δ . In the initial stages of the SA algorithm, the temperature parameter T is set high, which means the algorithm is more inclined to accept less promising solutions. The temperature parameter T is systematically reduced during the run of the SA algorithm according to a cooling procedure. Thus, the algorithm is less likely to accept worse solutions towards the end. To this end, the temperature parameter T is updated following a geometric cooling rule formulated as $T = T \times \beta$ where $\beta \in (0, 1)$.

SA employs both the neighborhood schemes $N_C(\mathbf{x}, r, \theta)$ and $N_F(\mathbf{x})$ within the local search procedure. Whenever a candidate solution is reached, the SA algorithm accepts it according to an acceptance probability p_a that plays a critical role in the overall performance of the

algorithm. The acceptance probability p_a is defined as follows.

$$p_a = \begin{cases} 1, & \text{for } \tilde{Z} < Z \\ e^{-\frac{\Delta}{T}} & o/w \end{cases} \quad (4.6)$$

As mentioned above, Δ stands for the difference between the candidate and previous objective function values. Δ is calculated as follows $\Delta = (\tilde{Z} - Z)/Z$ where $\tilde{Z}$ and Z are the objective function values of the candidate solution $\tilde{\mathbf{x}}$ and the previous solution $\mathbf{x}$, respectively. Notice that whenever the candidate solution $\tilde{\mathbf{x}}$ is better than $\mathbf{x}$, the candidate solution is accepted. However, if the candidate solution has a worse objective function value, then $\tilde{\mathbf{x}}$ is only accepted with acceptance probability p_a . A formal outline of the SA is presented with Algorithm 3.

Similar to TS, the initial radius of the circular neighborhood is set to be $\bar{r} = (H + L)/20$. Throughout SA iterations, the radius r is updated with a parameter $\alpha \in (0, 1)$, that is, $r = r \times \alpha$, whenever a neighbor solution is accepted. SA algorithm takes into account all regions starting with a random solution $\mathbf{x}$ which can be located in any region. As the algorithm proceeds, when an interregional move occurs, returning to previous region with local search is forbidden for σ iterations for generate diversity. Finally, the algorithm terminates when the cooling procedure is complete, and it reports the best solution found.

4.2.4 Beam Search Algorithm

Beam Search (BS) is an efficient B&B-based heuristic method that focuses on the most promising active nodes of the B&B tree (Akyüz et al., 2013). BS conducts a breadth-first search like strategy while focusing only a given number of promising nodes on each layer of the branching tree unlike a complete breadth-first search strategy of a B&B algorithm that covers every node of the branching tree for evaluation and for further branching. The set of most promising candidates is named as "beam" while the size of this set is called "beam-width", which is denoted by $\mathcal{W}$. Nodes belong to the beam are the nodes that are chosen for further branching. By ignoring even a single node on a branching tree without proving if that node is redundant, the optimality of this algorithm is not guaranteed. However, depending on the branching procedure and the problem, the performance of BS can still yield very accurate results in practice.

The structure of the BS is based on the BSSS algorithm of Altay et al. (2023), which starts

Algorithm 3 Simulated Annealing (SA).

```

1: Input :  $\bar{r}, \bar{T}, \kappa, \alpha, \beta, \sigma$ 
2: Initialization :  $T \leftarrow \bar{T}, r \leftarrow \bar{r}, q \leftarrow 0$  and  $v_j \leftarrow 0$  for all  $j \in \{1, \dots, J\}$ .
3: Randomly select a location  $\mathbf{x}$ , set  $j_{\mathbf{x}} \leftarrow \Psi(\mathbf{x})$ 
4: Compute  $Z \leftarrow f(\mathbf{x}), \mathbf{x}^* \leftarrow \mathbf{x}, Z^* \leftarrow Z$ 
5: while  $T \geq \epsilon$  do                                     \* Stop after cooling *\
6:   if  $q = 0$  then
7:      $\tilde{\mathbf{x}} \leftarrow N_C(\mathbf{x}, r, \theta)$ 
8:   else
9:      $\tilde{\mathbf{x}} \leftarrow N_F(\mathbf{x})$ 
10:  end if
11:  Set  $j_{\tilde{\mathbf{x}}} \leftarrow \Psi(\tilde{\mathbf{x}})$  and  $\tilde{Z} \leftarrow f(\tilde{\mathbf{x}})$           \* Determine the region and objective value of
     $\tilde{\mathbf{x}}$  *\
12:   $p_a = \begin{cases} 1, & \text{for } \tilde{Z} < Z \\ e^{-\frac{\tilde{Z} - Z}{T}}, & \text{o/w} \end{cases}$           \* Acceptance probability
    calculation *\
13:  if  $\tilde{Z} < Z$  then
14:    if  $j_{\mathbf{x}} \neq j_{\tilde{\mathbf{x}}}$  then                                \* Check when an inter-regional move occurs
    *\
15:     $v_{j_{\mathbf{x}}} \leftarrow \sigma, j_{\mathbf{x}} \leftarrow j_{\tilde{\mathbf{x}}}$           \* Moves towards region  $j_{\mathbf{x}}$  are forbidden for  $\sigma$ 
    iterations *\
16:  end if
17:   $q \leftarrow 1, \mathbf{x} \leftarrow \tilde{\mathbf{x}}, Z \leftarrow \tilde{Z}, r \leftarrow r \times \alpha$           \* If  $p_a = 1$ ;  $\tilde{\mathbf{x}}$  is still accepted when  $\gamma \leq p_a$ 
    *\
18:  else
19:     $q \leftarrow 0$ , generate a random number  $\gamma \in [0, 1]$ 
20:    if  $\gamma < p_a$  then
21:      if  $j_{\mathbf{x}} \neq j_{\tilde{\mathbf{x}}}$  then                                \* Check if there is an inter-regional
    move *\
22:       $v_{j_{\mathbf{x}}} \leftarrow \sigma, j_{\mathbf{x}} \leftarrow j_{\tilde{\mathbf{x}}}$ 
23:    end if
24:     $\mathbf{x} \leftarrow \tilde{\mathbf{x}}, Z \leftarrow \tilde{Z}$                                 \* When  $p_a < 1$ ;  $\tilde{\mathbf{x}}$  is still accepted
    *\
25:  end if
26:  end if
27:  if  $\tilde{Z} < Z^*$  then
28:     $\mathbf{x}^* \leftarrow \tilde{\mathbf{x}}, Z^* \leftarrow \tilde{Z}$                                 \* Update the best known solution
    *\
29:  end if
30:  for  $j \in \{1, \dots, J\}$  do
31:    if  $v_j > 0$  then
32:       $v_j \leftarrow v_j - 1$                                 \* Update the iteration counters for each region
     $\Omega_j$  *\
33:  end if
34:  end for
35:  Update  $T \leftarrow T \times \beta$                                 \* Geometric Cooling *\
36: end while
37: Output :  $\mathbf{x}^*$  and  $Z^*$ 

```

from the rectangular hull of the customer locations as a big rectangle and partitions it into four equally sized-rectangles. This procedure is performed at each iteration so that each remaining rectangle is continuously divided into smaller rectangles until the sizes of the rectangles drop below a tolerance value. Each rectangle corresponds to a B&B node as in the BSSS procedure. Upper bound value of a rectangle, i.e. Z_{UB} , is calculated assuming that the facility is located at the intersection point $\mathbf{x}$ of the diagonals of the rectangle. The initial upper bound value is calculated considering the center of gravity of customer locations as the facility location. For the lower bound value of a rectangle Z_{LB} , we consider the sum

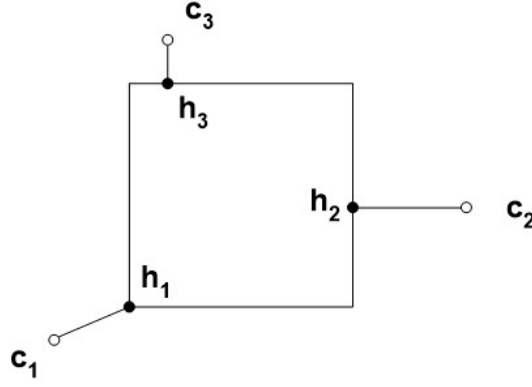


FIGURE 4.5 – The shortest path distances between customer locations and the borders of the rectangle

of the shortest distances between customers and their projection points on the borders of the rectangle. Figure 4.5 illustrates an example in which three customer locations, i.e. c_1 , c_2 , and c_3 , are situated outside of the rectangle and their projection points on the borders of the rectangles are denoted with h_1 , h_2 , and h_3 , respectively. The lower bound is equal to the sum of the distance between customers c_1 , c_2 , and c_3 and their respective projection points h_1 , h_2 , and h_3 . Note that if a customer is located inside the rectangle, the shortest distance to its projection point is assumed to be zero. In addition, for each B&B node, we calculate an evaluation function (Croce et al., 2004) as $\bar{Z} = \psi Z_{UB} + (1 - \psi) Z_{LB}$ to evaluate the usefulness of a node at each iteration while parameter value of ψ is chosen between $0 \leq \psi \leq 1$. At $\mathcal{W}$ nodes with the smallest evaluation function value ($\bar{Z}$) are chosen for further branching and the remaining ones are pruned for each layer of the branching tree. Finally, the solution that yields the best upper bound constitutes the output of the BS.

5 LOCATION ROUTING PROBLEM CONSIDERING MULTIPLE REGIONS WITH DIFFERENT NORMS

In this chapter, we formulate the Location Routing Problem with Multiple Regions with Different Norms (LRP-M). Without loss of generality, we address the case of two regions Ω_1 and Ω_2 with distance norms ℓ_1 and ℓ_2 , respectively. This study focused on two different cases of LRP-M with two regions depending with different slopes the separating line. The first case assumes that customer regions are separated by a single vertical separating line, while the second case considers the case when this separating line is oblique. Initially, both versions are formulated as MINLP. Then, we propose discrete approximation schemes for both versions of the LRP-M.

Considering discrete approximations, for the LRP-M with vertical separating lines, the discrete approximation scheme only consists of considering all of the customer locations as possible candidate facility locations since solving the original location routing problem is already time consuming. On the other hand, for the oblique separating line case, two different discrete approximation approaches are applied while first addressing discrete gate-point yielding a simpler MINLP formulation and second addressing discrete facility locations considering customer locations as the candidate facility locations.

Finally, note that for the sake of conciseness, we confined ourselves to the case of single facility location and single route, which can be extended into more general cases such as multi-facility and multiple routes.

5.1 LRP-M with Vertical Separating Line

This section focuses on the LRP-M with two regions having different norms. Since the separating line is vertical, the solution of finding a gate-point location is straightforward because that gate-point is located at the horizontal projection of a customer or a facility which is located in Ω_1 .

First of all, we introduce a continuous LRP-M model where the unknown facility location is

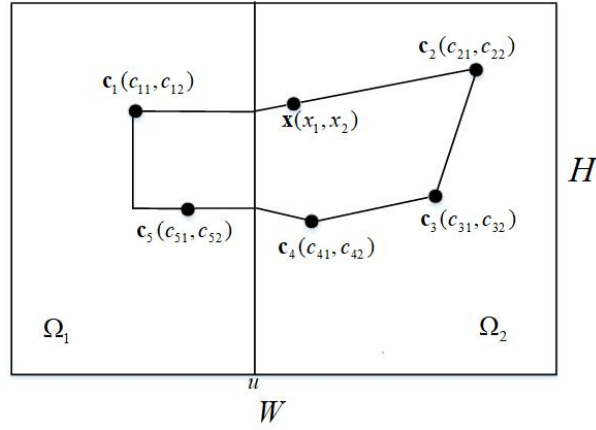


FIGURE 5.1 – LRP-M with vertical separating line

denoted with $\mathbf{x}(x_1, x_2)$. Then, we propose a discrete LRP model using m candidate facility locations as $m = n$ where n denotes the number of customer locations. In that case, since the candidate facility locations are known and are the same as the customer locations, the shortest path distances between customer locations are denoted with d_{ij} . The notations used for both formulations of LRP-M considering two regions with vertical separating line are included in Table 5.1.

TABLE 5.1 – Notation for LRP-M with vertical separating line

Sets	
V	set of vertices
A	set of arcs
A_K	set of arcs between customers
K	customer set, $K = K_1 \cup K_2$
P	set of candidate facility locations
Parameters	
$\mathbf{c}_i(c_{i1}, c_{i2})$	coordinates of customer i
d_{ij}	distance between vertex i and j
H, W	height and width of rectangular hull of customer locations
m	number of candidate facility locations
n	number of customer locations
u	abscissa of vertical separating line s
Variables	
$\mathbf{x}(x_1, x_2)$	facility coordinates
y_{ij}	binary variable that determines if arc $i \rightarrow j$ is used on a route
λ_i^1	distance variable between facility and customer i while $i \in K_1$
λ_i^2	distance variable between facility and customer i while $i \in K_2$
β	facility region variable, $\beta = 1$ if $\mathbf{x} \in \Omega_2$, $\beta = 0$ o/w
g_i	distance from facility to customer i that is used on a route
p_i	binary variable that determines whether candidate facility location i is used
f_{ij}	single commodity flow variable

5.1.1 A MINLP Model for LRP-M with Vertical Separating Line

MINLP formulation of LRP-M considering two regions with a vertical borderline is the simplest version of continuous LRP-M problem. Let V denote the set of vertices including the n customer locations and the facility location $\mathbf{x}$. Let $A_{\mathcal{K}}$ be the set of arcs between customer locations and $A_{\mathcal{K}} \subset A$ while A be the set of arcs that also includes the facility location $\mathbf{x}$. d_{ij} denotes the distance between customer i and j as $i, j \in A_{\mathcal{K}}$. In addition, customer set $\mathcal{K}$ is separated into two subsets $\mathcal{K}_1$ and $\mathcal{K}_2$ according to their regions Ω_1 and Ω_2 , respectively. Observing Figure 5.1, u represents the abscissa of vertical borderline and H and W are respectively the height and width of rectangular hull of customers and $\mathbf{c}_i(c_{i1}, c_{i2})$ denotes the location of customer i .

Variables of this model are as follows. $\mathbf{x}(x_1, x_2)$ denotes the location of the facility, y_{ij} is a binary variable that determines if the arc between i and j is chosen for the route while $i, j \in A$. In addition, y_{1i} and y_{i1} concerns the arcs from facility $\mathbf{x}$ to customer i and from customer i to facility $\mathbf{x}$ as $i \in \mathcal{K}$. f_{ij} is a continuous flow variable for the amount of single commodity from i to j as $i, j \in A$. Similarly, f_{1i} and f_{i1} also consider the arcs using the facility location $\mathbf{x}$. λ_i^1 and λ_i^2 denote the unknown distances between facility and the customer i residing in region Ω_1 and Ω_2 , respectively. In addition, g_i represents the distance between facility and customer i which is used on the route. Note that if an arc between facility and customer i is not used, then $g_i = 0$. Finally, β denotes the region that facility will be located. In summary, $\beta = 0$ if $\mathbf{x} \in \Omega_1$ and $\beta = 1$ if $\mathbf{x} \in \Omega_2$. Henceforth, the MINLP formulation of LRP-M with vertical separating line, namely "MINLP-V" is given as follows.

MINLP-V :

$$\min \sum_{i,j \in A_{\mathcal{K}}} d_{ij} y_{ij} + \sum_{i \in \mathcal{K}} g_i \quad (5.1)$$

Subject to

$$\lambda_i^1 \geq (1 - \beta) (|x_1 - c_{i1}| + |x_2 - c_{i2}|) \quad \forall i \in \mathcal{K}_1 \quad (5.2)$$

$$\lambda_i^1 \geq \beta \left(|u - c_{i1}| + \sqrt{(x_1 - u)^2 + (x_2 - c_{i2})^2} \right) \quad \forall i \in \mathcal{K}_1 \quad (5.3)$$

$$\lambda_i^2 \geq (1 - \beta) \left(|u - x_1| + \sqrt{(c_{i1} - u)^2 + (c_{i2} - x_2)^2} \right) \quad \forall i \in \mathcal{K}_2 \quad (5.4)$$

$$\lambda_i^2 \geq \beta \sqrt{(x_1 - c_{i1})^2 + (x_2 - c_{i2})^2} \quad \forall i \in \mathcal{K}_2 \quad (5.5)$$

$$\sum_{i \in V, i \neq j} y_{ij} = 1 \quad \forall j \in V \quad (5.6)$$

$$\sum_{j \in V, i \neq j} y_{ij} = 1 \quad \forall i \in V \quad (5.7)$$

$$\sum_{j \in \mathcal{K}} f_{1j} = n \quad (5.8)$$

$$f_{ij} \leq ny_{ij} \quad \forall i, j \in A \quad (5.9)$$

$$\sum_{i \in V, i \neq j} f_{ij} - \sum_{t \in V, j \neq t} f_{jt} = 1 \quad \forall j \in \mathcal{K} \quad (5.10)$$

$$x_1 - u \leq \beta W \quad (5.11)$$

$$u - x_1 \leq (1 - \beta)W \quad (5.12)$$

$$g_i + (H + W)(1 - y_{1i} - y_{i1}) \geq \lambda_i^1 \quad \forall i \in \mathcal{K}_1 \quad (5.13)$$

$$g_i + (H + W)(1 - y_{1i} - y_{i1}) \geq \lambda_i^2 \quad \forall i \in \mathcal{K}_2 \quad (5.14)$$

$$x_1, x_2 \geq 0 \quad (5.15)$$

$$g_i \geq 0 \quad \forall i \in \mathcal{K} \quad (5.16)$$

$$f_{ij} \geq 0 \quad \forall i, j \in V \quad (5.17)$$

$$\lambda_i^1 \geq 0 \quad \forall i \in \mathcal{K}_1 \quad (5.18)$$

$$\lambda_i^2 \geq 0 \quad \forall i \in \mathcal{K}_2 \quad (5.19)$$

$$y_{ij} \in \{0, 1\} \quad \forall i, j \in V \quad (5.20)$$

The objective function (5.1) consists of total distance between customer locations and total distance between facility and customer locations. Constraints (5.2) and (5.3) are for the distance constraints that define the distance between facility and customers in Ω_1 while

constraints (5.4) and (5.5) define the distance between facility and customers in Ω_2 . Constraints (5.6) and (5.7) ensure that each location is visited only once. In addition, constraints (5.8)-(5.10) are for the sub-tour elimination (Gavish and Graves, 1978). Constraints (5.11) and (5.12) define the relation between β and facility coordinate $\mathbf{x}$. Constraints (5.13) and (5.14) define the distance covered from each chosen arc between $\mathbf{x}$ and customer i while $H + W$ represents a sufficiently large number. Following constraint (5.13) (and constraint (5.14)), when $y_{i1} = 1$ or $y_{1i} = 1$, an arc between the facility location and customer i is used on the route, hence $g_i = \lambda_i^1$ (and $g_i = \lambda_i^2$). Note that both of these constraints become redundant when both $y_{i1} = 0$ and $y_{1i} = 0$. Finally, constraints (5.15)-(5.20) are for the domain definition.

5.1.2 A MILP Model for the Discrete Approximation of the LRP-M with Vertical Separating Line

Discrete approximation for LRP-M considering two regions with vertical separating lines is obtained by discretizing the facility location as the customer locations. Hence a MILP model of LRP is obtained by discrete approximation of the original model (5.1)-(5.20). This model have $2n$ vertices covering both n candidate facility locations and n customer locations. Let d_{ij} stand for the distance between vertex i and j and y_{ij} is 1 if j is visited immediately after i . Finally, p_i determines whether facility $i \in P$ is opened or not. MILP formulation that uses discrete approximation of facility locations which is named as "MILP- F " is stated below.

MILP- F :

$$\min \sum_{i,j \in V} d_{ij} y_{ij} \quad (5.21)$$

Subject to

$$\sum_{i \in V, i \neq j} y_{ij} = 1 \quad \forall j \in \mathcal{K} \quad (5.22)$$

$$\sum_{j \in V, i \neq j} y_{ij} = 1 \quad \forall i \in \mathcal{K} \quad (5.23)$$

$$\sum_{i \in \mathcal{K}} y_{ij} = p_j \quad \forall j \in P \quad (5.24)$$

$$\sum_{j \in \mathcal{K}} y_{ij} = p_i \quad \forall i \in P \quad (5.25)$$

$$\sum_{i \in P} p_i = 1 \quad (5.26)$$

$$\sum_{j \in \mathcal{K}} f_{ij} = np_i \quad \forall i \in P \quad (5.27)$$

$$f_{ij} \leq ny_{ij} \quad \forall i, j \in A \quad (5.28)$$

$$\sum_{i \in V, i \neq j} f_{ij} - \sum_{t \in V, j \neq t} f_{jt} = 1 \quad \forall j \in \mathcal{K} \quad (5.29)$$

$$f_{ij} \geq 0 \quad \forall i, j \in A \quad (5.30)$$

$$y_{ij} \in \{0, 1\} \quad \forall i, j \in A \quad (5.31)$$

$$p_i \in \{0, 1\} \quad \forall i \in P \quad (5.32)$$

The objective function (5.21) consists of total distance between customer locations, constraints (5.22), (5.23) ensure that each customer location is visited only once. Since there is a single route, following constraints (5.24) and (5.25), sum of incoming and outgoing arcs from a candidate facility location is equal to 1 if that location is opened and 0 otherwise. Constraint (5.26) makes sure that only one of the candidate location is used. Constraints (5.27)-(5.29) obviate sub-tours (Gavish and Graves, 1978). The route covers $n + 1$ vertices including the chosen facility location. Following these constraints, f_{ij} tracks the arcs of this route starting from taking the value of $n + 1 - 1$ and drops by 1 for each next arc. Following constraint (5.28), if an arc isn't used then f_{ij} starts from zero and prevents the usage of any path forward. Finally, constraints (5.30)-(5.32) are for domain definition.

5.2 LRP-M with Oblique Separating Line

In this section, we concentrate on the LRP-M considering two regions with an oblique separating line $s : x_2 = ax_1 + b$. We already know that closed form solution of gate-point locations are known (Franco et al., 2012, 2018). Still, finding gate points on the oblique line while solving LRP-M is still an arduous task. Distance calculations within this model are performed following the footsteps of Parlar (1994). Figure 5.2 Illustrates the LRP-M in two regions with oblique line for a single facility and a single route.

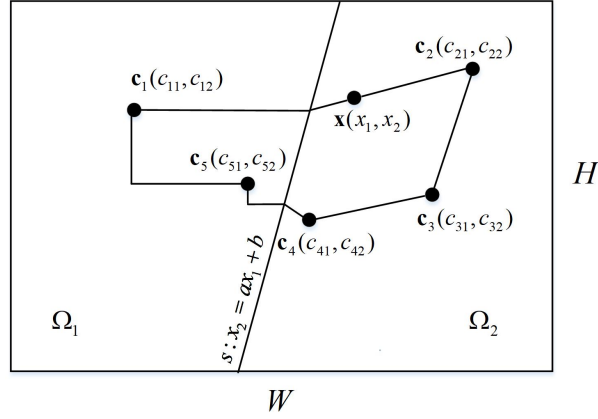


FIGURE 5.2 – LRP-M with an Oblique Separating Line

To solve this problem, first, we propose a MINLP formulation for solving LRP-M with an oblique separating line which also determines the gate-point locations for each customer. Furthermore, for the sake of simplicity, we do not take into account the sub-case with three-link shortest paths (Plastria, 2019) since it isn't practical for application point of view. Secondly, an alternative MINLP formulation that employs a discrete approximation approach for the separating line. That is to say, in this formulation we try to fix the best gate-point from a set of candidate gate-point locations. Lastly, we propose a MILP formulation where the discrete approximation scheme consists of selecting the unknown facility location from a set of candidate locations which are actually customer locations.

The notation used for the formulations of LRP-M with oblique separating line is presented in Table 5.2.

5.2.1 A MINLP Model for the Continuous LRP-M with Oblique Separating Line

This is the most general modelling approach inspired from Parlar (1994). Here, the gate-point locations between facility and customers are determined with the solution of the model. To this end, a new continuous variable v_i is defined for determining abscissa of a gate-point between facility and customer i . Hence, distance constraints (5.2)-(5.5) and binary region constraints (5.11)-(5.12) from vertical separating line problem are revised accordingly while the other constraints are employed as they are.

Similar to LRP-M with vertical separating line, the proposed MINLP model is defined on a graph $G(V, A)$ consisting of $n + 1$ vertices including the location of the first vertex (depot) is unknown. Let A_K be the arcs between customer locations and $A_K \subset A$, then d_{ij} represents the distance between location i and j . In addition, customer set K is separated into two

TABLE 5.2 – Notation used for LRP-M with oblique separating line

Sets	
V	set of vertices, $V = P \cup \mathcal{K}$
A	set of arcs
$A_{\mathcal{K}}$	set of arcs between customers
$\mathcal{K}$	customer set, $\mathcal{K} = \mathcal{K}_1 \cup \mathcal{K}_2$
G	set of candidate gate-points
P	set candidate facility locations
Parameters	
$\mathbf{c}_i(c_{i1}, c_{i2})$	coordinates of customer i
d_{ij}	distance between vertex i and j
H, W	height and width of defined area
n	number of customers
$\mathbf{w}_l(w_{l1}, w_{l2})$	coordinates of gate-point l
Variables	
$\mathbf{x}(x_1, x_2)$	facility coordinates
a, b	parameters of separating line $s : x_2 = ax_1 + b$
y_{ij}	binary variable that determines whether j is visited immediately after i .
λ_i^1	distance variable between facility and customer i for $i \in \mathcal{K}_1$
λ_i^2	distance variable between facility and customer i for $i \in \mathcal{K}_2$
β	facility region variable, $\beta = 1$ if $\mathbf{x} \in \Omega_2$, $\beta = 0$ o/w
v_i	abscissa of a potential gate-point between facility and customer i
g_i	distance from facility to customer i that is used on a route
p_i	binary variable that determines if candidate facility location i is used
σ_{il}	binary variable that shows if gate-point l is used between facility and customer i
f_{ij}	continuous single commodity flow variable

subsets $\mathcal{K}_1$ and $\mathcal{K}_2$ according to their regions Ω_1 and Ω_2 , respectively.

Other parameters of this problem are as follows. Figure 5.2 depicts an instance of LRP-M with a oblique separating line while a and b are parameters of line s and H and W stand for respectively height and width of rectangular hull of customers and $\mathbf{c}_i(c_{i1}, c_{i2})$ denotes the location of customer i .

Let $\mathbf{x}(x_1, x_2)$ denote the location of the facility, y_{ij} determines whether the arc between i and j is chosen or not for the route, f_{ij} is a continuous flow variable for the single commodity. λ_i^1 and λ_i^2 represent the distances between facility and the customer i residing in region Ω_1 and Ω_2 , respectively. In addition, g_i stands for the distance from facility and customer i . Finally, β denotes the region that facility will be located. $\beta = 1$ if $\mathbf{x} \in \Omega_2$ and $\beta = 0$ otherwise. Hence a MINLP formulation is as follows for LRP-M with oblique lines, "MINLP-O" is given as follows.

MINLP-O :

$$\min \sum_{i,j \in A_{\mathcal{K}}} d_{ij} y_{ij} + \sum_{i \in \mathcal{K}} g_i \quad (5.33)$$

Subject to

Constraints (5.6)-(5.10) and (5.13)-(5.20)

$$\lambda_i^1 \geq (1 - \beta) (|x_1 - c_{i1}| + |x_2 - c_{i2}|) \quad \forall i \in \mathcal{K}_1 \quad (5.34)$$

$$\lambda_i^1 \geq \beta (|v_i - c_{i1}| + |(av_i + b) - c_{i1}|) \quad (5.35)$$

$$+ \sqrt{(x_1 - v_i)^2 + (x_2 - (av_i + b))^2} \quad \forall i \in \mathcal{K}_1$$

$$\lambda_i^2 \geq (1 - \beta) (|v_i - x_1| + |(av_i + b) - x_2|) \quad (5.36)$$

$$+ \sqrt{(c_{i1} - v_i)^2 + (c_{i2} - (av_i + b))^2} \quad \forall i \in \mathcal{K}_2$$

$$\lambda_i^2 \geq \beta \sqrt{(x_1 - c_{i1})^2 + (x_2 - c_{i2})^2} \quad \forall i \in \mathcal{K}_2 \quad (5.37)$$

$$x_2 - ax_1 - b \leq (1 - \beta)(H + W) \quad (5.38)$$

$$ax_1 + b - x_2 \leq \beta(H + W) \quad (5.39)$$

$$v_i \geq 0 \quad \forall i \in \mathcal{K} \quad (5.40)$$

Equation (5.33) is the objective function consists of total distance between customer locations and total distance between facility and customer locations. Constraints (5.34) and (5.35) are distance constraints that define the distance between facility and customers in Ω_1 . In constraint (5.35) and (5.36) for each customer in Ω_1 and Ω_2 , its respective gate-point is defined using v_i . Constraints (5.36), (5.37) are distance constraints that define the distance between facility and customers in Ω_2 . Constraints (5.38) and (5.39) define the relation between β and facility coordinate $\mathbf{x}$. Given separating line s has positive slope $a > 0$, $\beta = 1$ if $\mathbf{x} \in \Omega_2$ and $\beta = 0$ if $\mathbf{x} \in \Omega_1$. Constraint (5.40) is a boundary equation for v_i . Finally, flow conservation and sub-tour elimination constraints (5.6)-(5.10) and other constraints (5.13)-(5.20) are taken from vertical separating line problem.

5.2.2 Discrete Approximation of LRP-M with Oblique Separating Line

This section includes two different discrete approximation schemes for LRP-M with oblique separating lines. First method is MINLP formulation by discrete approximation of the gate-points. In this case, the shortest path passes through one of the candidate gate-points. Second method is the MILP formulation by discretization of facility locations, which follows the same idea of discretization of LRP-M with vertical separating lines.

5.2.2.1 A MINLP model of the LRP-M with oblique separating line using discrete gate-points

In this section, a MINLP formulation of continuous LRP-M with discrete gate-points is introduced. In this formulation, since optimal gate-points for single separating line has closed form solution, distance matrix between customer locations d_{ij} is pre-calculated. The shortest path between facility x and each customer location that are located in different regions uses a gate point w_l among set of candidate gate-points G .

This formulation alternates using continuous gate-point variable v_i from the model with finding best gate-point from a given set of candidate gate-point locations using a binary variable σ_{il} as $\sigma_{il} = 1$ if gate-point l is chosen between facility x and customer i . This formulation replaces the distance constraints (5.35) and (5.36) with new discrete gate-point constraints. The only disadvantage of this formulation is the number of candidate gate-points which increases the number of constraints in the model. The proposed MINLP formulation using discrete gate points which is named as "MINLP- G " is given below :

MINLP- G :

$$\min \sum_{i,j \in A_K} d_{ij} y_{ij} + \sum_{i \in K} g_i \quad (5.41)$$

Subject to

Constraints (5.6)-(5.10) and (5.13)-(5.20)

$$\lambda_i^1 \geq (1 - \beta) (|x_1 - c_{i1}| + |x_2 - c_{i2}|) \quad \forall i \in \mathcal{K}_1 \quad (5.42)$$

$$\begin{aligned} \lambda_i^1 \geq & \beta \left(|w_{l1} - c_{i1}| + |w_{l2} - c_{i1}| + \sqrt{(x_1 - w_{l1})^2 + (x_2 - w_{l2})^2} \right) \\ & - 2(H + W)(1 - \sigma_{il}) \quad \forall i \in \mathcal{K}_1, \forall l \in G \end{aligned} \quad (5.43)$$

$$\begin{aligned} \lambda_i^2 \geq & (1 - \beta) \left(|w_{l1} - x_1| + |w_{l2} - x_2| + \sqrt{(c_{i1} - w_{l1})^2 + (c_{i2} - w_{l2})^2} \right) \\ & - 2(H + W)(1 - \sigma_{il}) \quad \forall i \in \mathcal{K}_2, \forall l \in G \end{aligned} \quad (5.44)$$

$$\lambda_i^2 \geq \beta \sqrt{(x_1 - c_{i1})^2 + (x_2 - c_{i2})^2} \quad \forall i \in \mathcal{K}_2 \quad (5.45)$$

$$x_2 - ax_1 - b \leq (1 - \beta)(H + W) \quad (5.46)$$

$$ax_1 + b - x_2 \leq \beta(H + W) \quad (5.47)$$

$$\sum_{l \in G} \sigma_{il} = 1 \quad \forall i \in \mathcal{K} \quad (5.48)$$

$$\sigma_{il} \in \{0, 1\} \quad \forall i \in \mathcal{K}, \forall l \in G \quad (5.49)$$

Objective function (5.41) consists of total distance between customer locations and total distance between facility and customer locations. Constraints (5.42) and (5.43) are distance constraints that define the distance between facility and customers in Ω_1 . Constraints (5.43) and (5.44) define the cases where a gate-point is chosen among the candidate gate-point locations by using a binary σ_{il} as $l \in G$ to calculate the shortest path. If $\sigma_{il} = 0$, then the gate-point l is not chosen thus makes the constraint redundant. Constraints (5.44) and (5.45) define the distance between facility and customers in Ω_2 . Constraints (5.46) and (5.47) give the relation between β and facility coordinate $\mathbf{x}$. Given separating line s has positive slope $a > 0$, $\beta = 1$ if $x \in \Omega_2$, 0 otherwise. Constraint (5.48) ensures that only one candidate gate-point location is chosen for each customer. Finally, constraint (5.49) is for the integrality restriction for σ_{il} . In addition, commodity flow conservation and sub-tour elimination constraints (5.6)-(5.10) and other constraints (5.13)-(5.20) are taken from vertical separating line problem.

5.2.2.2 A MILP model of the LRP-M with oblique separating line using discrete facility locations

MILP Model of LRP-M considering two regions with oblique separating line follows the same principle as the discretization of facility locations for LRP-M with vertical separating lines. Including n candidate locations and n customers there are $2n$ vertices within the graph $G(V,A)$ of this problem. d_{ij} gives the shortest distance between vertex i and j and y_{ij} is 1 if arc between location i and j is used. Finally, p_i determines whether facility $i \in P$ is opened or not.

While, MILP model of discrete LRP-M with oblique separating line is exactly the same as the MILP- F formulation given in Section 5.1.2, the only difference is the distance functions that are used when pre-calculating distances d_{ij} .

6 COMPUTATIONAL STUDIES

This chapter includes the computational results of various studies in the thesis. The first set of computational experiments compares the performance of BSSS with the commercial MINLP solvers and two efficient heuristic approaches, i.e. Discrete Approximation (DA) and Modified Weiszfeld (MW) algorithms for various cases of SMDN3. The second set of computational experiments focuses on solving SMDN3 on larger-scale instances using specially designed metaheuristic methods, i.e., Tabu Search (TS), Simulated Annealing (SA), and Beam Search (BS) algorithms and compares their performance to BSSS. Finally, the third set of computational experiments compares the performance of different formulations utilizing the discrete approximation of the facility or gate-point locations using the commercial solvers on two distinct cases of LRP-M.

6.1 Computational Experiments for SMDN3 Using Commercial Solvers, BSSS, MW and DA

In our computational experiments on SMDN3, MW and DA are coded with C++ and Windows Server 2016 environment with 128 GB RAM and 2.60 GHz processors is used as a workplace. Initially, the test bed and the generation of the instances are explained. Then, the computational results are shared with our remarks.

6.1.1 Test Bed

This test bed consists of two different classes of instances. Class-I is based on sets of randomly generated instances, while Class-II includes the test instances that are adapted from the literature. Both sets of instances are used in each test covering both variants of the SMDN3, namely the SMDN3- ℓ_1 and the SMDN3- ℓ_2 for various slopes of the separating lines.

Considering Class-I data set, 10 different instances are generated with different sizes from 50 to 10000 demand points. The cardinality of the customer set $|J|$ is chosen among $\{50, 100,$

200, 400, 800, 1000, 2500, 5000, 8000, 10000}. The Class I instance naming procedure is presented as “ I_- ” and the cardinality of the demand set $|J|$ for the various sizes of test instances. In Class-I data set, the coordinates of the demand points $\mathbf{c}_i = (c_{j1}, c_{j2})$ are uniformly distributed on an area of 1000×1000 , which means each demand point $\mathbf{c}_j$, the coordinates $c_{j1}, c_{j2} \in [0, 1000]$ for all $j \in J$.

However, if we consider the Class II data set from Brimberg et al. (2000), the test instances are illustrated with for the “ B_- ” and $|J|$ as $|J|$ is from $\{50, 287, 654, 1060\}$. In these instances, the size of the region changes depending on the instance.

Considering SMDN3 with vertical borderlines, we decided on the layout of the separating lines as illustrated in Figure 3.3. The locations of the borderlines s_1 and s_2 are defined through their abscissa coordinates which are a_1 and a_2 , while $a_1 < a_2$ is satisfied. Let c'_1 and c''_1 be the minimum and maximum abscissa values of the customer locations in an instance, respectively, and the coordinates of the separating lines can be determined as follows. $a_1 = 0.7c'_1 + 0.3c''_1$ and $a_2 = 0.35c'_1 + 0.65c''_1$, respectively. Using these separating lines, the plane will be divided into three regions consisting of 30%, 35%, and 35% of the region, respectively.

As for the general case of SMDN3, considering two oblique separating lines $s_1 \equiv y = m_1x + n_1$ and $s_2 \equiv y = m_2x + n_2$. Various cases of SMDN3 with different gate-point properties are experimented with different combinations of slopes, that is, m_1 and m_2 . These cases include the negative slope cases $m_2 < m_1 < 0$ and $m_1 < m_2 < 0$, the parallel slope case $m_1 = m_2 < 0$ and the triangle shaped regions cases, i.e. $m_1 < 0 < m_2$. These cases and their respective shortest path cases are illustrated with Figures 3.5 and 3.7. In our experiments, we use $m_1 = -6, m_2 = -8$ and $m_1 = -8, m_2 = -3$ to address negative slope cases, $m_1 = m_2 = -5$ to define the case of parallel regions and $m_1 = -8, m_2 = 6$ to illustrate the cases of triangle shaped regions.

6.1.2 Computational Results

In this section, we include our computational results and discussions of the suggested BSSS algorithm with two heuristic methods DA and MW, and commercial NLP/MINLP solvers KNITRO 10.2.0 and BARON 17.4.1. We use the GAMS environment to employ these commercial solvers. In our experiments, use both commercial servers in their respective default settings. We used these solvers to provide benchmark solutions to assess the performance

of the suggested algorithms, i.e. BSSS, MW, and DA on SMDN3 with vertical separating lines. Lastly, the stopping condition of each algorithm is decided as 3600 seconds since it is observed that a further search is unnecessary.

In our computations the percentage gap from the best solution, i.e. $\text{gap}\%$ is calculated using the equation below, where UB denotes the best solution of the given algorithm while UB^* denotes the optimal solution.

$$100 \times \frac{(UB - UB^*)}{UB^*}, \quad (6.1)$$

6.1.2.1 SMDN3 with vertical separating lines

Considering SMDN3 with vertical separating lines, Tables 6.1 and Table 6.3 illustrate the results obtained from the solution of BSSS, DA and MW algorithms and KNITRO and BARON solvers following the MINLP model. Tables 6.1 and 6.3 report the results for SMDN3- ℓ_1 and SMDN3- ℓ_2 , respectively, considering the cases of vertical separating lines including both Class-I and Class-II datasets.

Furthermore, Tables 6.2 and 6.4 include the results of extensions of exact solution methods for both SMDN3- ℓ_1 and SMDN3- ℓ_2 with vertical separating lines, all of which solve the problem for each region separately. To this end, MINLP formulation given by (3.7)-(3.26) is separated into three NLP formulations (NLP-1, NLP-2, and NLP-3), each of which is defined for one of the three regions when the binary variables α and β are fixed to suitable values. As an example, following the formulation of NLP-1 can be observed from Equations (3.27)-(3.35), the commercial solver finds the best result for SMDN3 for only one region. Using the NLP formulations, the summation of the CPU time of each of the three computations is reported using KNITRO and BARON as NLP solvers. Similarly, an efficient extension of BSSS can solve the facility location problem separately for each region. This is mostly an advantageous approach since it simplifies the distance functions and if another region starts with a worse lower bound than the best upper bound, then any further unnecessary search on this region is prevented. In short, the extensions of BSSS, KNITRO, and BARON, which are named BSSS', KNITRO', and BARON', solve SMDN separately for each region. KNITRO' and BARON' solve NLP formulations of SMND3 with vertical separating lines for each region separately the NLP formulation considering one region is illustrated with equations (3.27)-(3.35). The summation of the cpu time of three separate NLP model is

reported for each instance. BSSS' runs BSSS algorithm separately for the customers at each region separately while pruning unnecessary rectangles.

TABLE 6.1 – Computational results for SMDN3- ℓ_1 with vertical borderlines.

		BSSS		DA		MW		KNITRO		BARON	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu(s.)	gap%	cpu (s.)	gap%	cpu(s.)	gap%
Class-I	<i>I</i> _50	20127	0.12	0.01	0.01	0.01	0.08	0.98	0.00	1.08	0.00
	<i>I</i> _100	39864.5	0.13	0.01	0.00	0.01	0.01	6.66	0.00	5.68	0.00
	<i>I</i> _200	83934.46	0.51	0.07	0.02	0.01	0.10	6.62	0.00	46.52	0.00
	<i>I</i> _400	162358.54	1.00	0.31	0.00	0.01	0.00	53.27	0.00	1404.67	0.00
	<i>I</i> _800	319251.30	2.35	0.71	0.00	0.04	0.01	91.07	0.00	2644.27	0.00
	<i>I</i> _1000	413279.50	3.57	0.92	0.01	0.01	0.01	377.27	0.00	1572.38	0.00
	<i>I</i> _2500	1009410.82	10.64	2.74	0.00	0.01	0.00	3604.83	0.00	3600.89	0.00
	<i>I</i> _5000	2054161.86	44.78	5.90	0.01	0.02	0.01	3743.53	0.00	3633.09	0.00
	<i>I</i> _8000	3273846.41	72.49	9.61	0.03	0.03	0.02	4564.2	8.05	3632.25	0.00
	<i>I</i> _10000	4079671.98	90.36	12.01	0.00	0.03	0.02	6676.9	7.75	3665.33	0.00
	Aver.		22.60	3.23	0.01	0.02	0.03	1912.53	1.58	2020.62	0.00
Class-II	<i>B</i> _50	195.71	0.01	0.01	0.17	0.01	0.24	1.27	0.00	1.53	0.00
	<i>B</i> _287	17772.47	0.02	0.18	0.04	0.01	0.00	0.26	0.00	25.42	0.00
	<i>B</i> _654	1786155.23	2.28	0.46	0.06	0.01	0.40	128.3	0.00	54.43	0.00
	<i>B</i> _1060	5192436.51	5.26	0.81	0.00	0.01	0.22	210.12	0.00	1840.01	0.00
	Aver.		1.89	0.37	0.07	0.01	0.21	84.99	0.00	480.35	0.00
	Ov. aver.		16.68	2.41	0.03	0.02	0.08	1390.38	1.13	1580.54	0.00

According to the computations from Tables 6.1 and 6.3 it can be seen that MINLP solvers occasionally cannot reach the optimal result before the CPU time limit runs out. This case either occurs either when the solvers can not find any feasible solution within the time limit or when they return the best upper bound value as the time limit is reached. Comparing the MINLP solvers, although KNITRO seems to perform more efficiently compared to BARON on average, KNITRO performs in a less stable manner in larger-scale instances. When we compare the suggested methods, we can easily observe that MW is the fastest method while suffering from the worst percentage gap. DA achieves better results in comparison to MW at the expense of CPU time. In comparison, the BSSS algorithm performs much slower than these two while still yielding acceptable CPU times for an exact method. We can observe from Tables 6.1 and 6.3 that the BSSS algorithm performs quite decently while yielding the overall average CPU time of 16.68 and 28.58 seconds. If we compare its performance with that of both MINLP solvers, which suffer from more than 1000 seconds of average CPU time while not even assuring an optimal solution, the difference is clear.

However, following Tables 6.2 and 6.4, when we separate the MINLP formulation into three different NLP formulations, then the performance of the commercial solvers KNITRO and BARON become significantly better. The performance of the BSSS also improves when treating the regions separately. As a result, among the improved algorithms, we can see that KNITRO' performs as the fastest exact approach, producing CPU times of 0.36 and

TABLE 6.2 – Computational results of improved exact solution methods for SMDN3- ℓ_1 with vertical borderlines.

		BSSS'		KNITRO'		BARON'	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu(s.)	gap%
Class-I	I_{50}	20127	0.04	0.21	0.00	0.27	0.00
	I_{100}	39864.5	0.08	0.2	0.00	0.29	0.00
	I_{200}	83934.46	0.16	0.23	0.00	0.29	0.00
	I_{400}	162358.54	0.3	0.23	0.00	0.58	0.00
	I_{800}	319251.3	0.58	0.24	0.00	0.89	0.00
	I_{1000}	413279.50	1.71	0.28	0.00	1.98	0.00
	I_{2500}	1009410.83	6.9	0.37	0.00	7.15	0.00
	I_{5000}	2054161.86	16.17	0.54	0.00	14.17	0.00
	I_{8000}	3273846.41	27.88	0.77	0.00	49.71	0.00
	I_{10000}	4079671.98	34.54	1.05	0.00	86.27	0.00
	Aver.		8.84	0.41	0.00	16.16	0.00
Class-II	B_{50}	195.71	0.01	0.22	0.00	0.26	0.00
	B_{287}	17772.47	0.1	0.22	0.00	0.21	0.00
	B_{654}	1786155.23	2.12	0.24	0.00	0.38	0.00
	B_{1060}	5192436.51	2.46	0.24	0.00	0.58	0.00
	Aver.		1.17	0.23	0.00	0.36	0.00
	Ov. aver.		6.65	0.36	0.00	11.65	0.00

TABLE 6.3 – Computational results for SMDN3- ℓ_2 with vertical borderlines.

		BSSS		DA		MW		KNITRO		BARON	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu(s.)	gap%	cpu (s.)	gap%	cpu(s.)	gap%
Class-I	I_{50}	21696.25	0.04	0.01	0.00	0.01	0.89	1.97	0.00	22.77	3.33
	I_{100}	45167.46	0.21	0.02	0.00	0.00	0.25	0.17	0.00	28.42	0.00
	I_{200}	89527.19	0.59	0.11	0.00	0.06	0.27	1.83	0.00	27.19	0.00
	I_{400}	175978.54	1.34	0.38	0.00	0.00	0.04	13.76	0.00	294.61	0.00
	I_{800}	347523.61	5.04	0.83	0.00	0.01	0.00	14.09	0.00	33.32	0.00
	I_{1000}	434912.12	6.68	1.05	0.04	0.01	0.03	186.39	0.00	2473.97	0.00
	I_{2500}	1101125.74	16.44	2.95	0.02	0.01	0.01	3620.04	0.00	1904.54	0.00
	I_{5000}	2196241.05	76.51	6.13	0.02	0.01	0.00	3852.21	0.00	3611.52	0.00
	I_{8000}	3509415.45	124.44	9.79	0.00	0.02	0.02	4062.3	5.47	3607.17	0.00
	I_{10000}	4377574.79	157.52	12.53	0.01	0.03	0.01	7435.77	5.76	3602.61	0.99
	Aver.		38.88	3.38	0.01	0.02	0.15	1918.85	1.12	1560.61	0.33
Class-II	B_{50}	208.84	0.02	0.01	0.02	0.01	0.16	0.11	0.00	2.63	0.00
	B_{287}	21538.00	0.02	0.20	0.00	0.01	1.00	31.12	172.17	10.58	0.00
	B_{654}	175982.91	1.33	0.64	0.77	0.01	24.09	5.39	0.37	2946.36	0.37
	B_{1060}	5591975.36	9.95	1.09	0.03	0.01	0.17	9.75	0.00	169.23	0.00
	Aver.		2.83	0.49	0.21	0.01	6.36	11.59	43.13	782.2	0.09
	Ov. aver.		28.58	2.55	0.06	0.01	1.92	1373.92	4.52	1338.21	0.26

0.37 seconds on average. BSSS' still gives better results compared to BARON' providing an average performance of 6.65 and 10.41 seconds for both problems. Although solving this problem for each region separately is worth the effort for certain cases of SMDN3, this scheme can only be applied for special cases like the vertical borderline cases.

TABLE 6.4 – Computational results of improved exact solutions for SMDN3- ℓ_2 with vertical borderlines.

	instances	BSSS'		KNITRO'		BARON'	
		obj. value	cpu (s.)	cpu (s.)	gap%	cpu(s.)	gap%
Class-I	I_{50}	21696.25	0.02	0.22	0.00	0.35	0.00
	I_{100}	45167.46	0.12	0.22	0.00	0.28	0.00
	I_{200}	89527.19	0.17	0.22	0.00	0.37	0.00
	I_{400}	175978.56	0.62	0.24	0.00	1.34	0.00
	I_{800}	347523.63	0.95	0.27	0.00	2.41	0.00
	I_{1000}	434912.11	2.75	0.28	0.00	4.0	0.00
	I_{2500}	1101125.71	10.11	0.43	0.00	12.60	0.00
	I_{5000}	2196241.1	25.86	0.57	0.00	34.54	0.00
	I_{8000}	3509415.44	41.98	0.83	0.00	85.0	0.00
	I_{10000}	4377574.8	54.18	0.98	0.00	139.63	0.00
	Aver.		13.68	0.43	0.00	28.05	0.00
Class-II	B_{50}	208.84	0.02	0.22	0.00	0.28	0.00
	B_{287}	21538.00	0.02	0.27	0.00	1.65	0.00
	B_{654}	175982.91	4.84	0.23	0.00	0.64	0.00
	B_{1060}	5591975.36	4.14	0.26	0.00	0.60	0.00
	Aver.		2.26	0.25	0.00	0.79	0.00
	Ov. aver.		10.41	0.37	0.00	20.26	0.00

6.1.2.2 SMDN3 with oblique separating lines

Considering the cases of SMDN3 divided by oblique separating lines, MINLP formulation similar to (3.7)-(3.26) is not possible for addressing the cases of SMDN3 because of the lack of closed-form solutions for various cases. Furthermore, we already observed that the performance of KNITRO and BARON are inferior in comparison to the other methods as far as the MINLP formulation is concerned. Not to mention, applying the exact solution methods for each region separately is not practical when the oblique borderlines are considered. Therefore, only BSSS is included as an exact solution method, while DA and MW are applied as alternative heuristic approaches to the problems of SMDN3 with oblique separating lines. Tables 6.5-6.12 report the computational results related to various cases of SMDN3 with oblique separating lines using BSSS, MW and DA algorithms.

The average performances of each algorithm for each case of SMDN3 with oblique lines is reported in Table 6.5. Although, BSSS performs the best in the vertical borderline cases since the distance function will be much easier to handle, it can be observed that BSSS also performs efficiently for the cases of SMDN3 with oblique separating lines, given that it is an exact solution approach. Furthermore, MW is the most efficient heuristic approach for each case of the SMDN3 with oblique lines. In addition, DA yields more accurate results in comparison to MW while CPU time of DA is significantly less than BSSS on the average. Furthermore, comparing the difficult of the cases of the cases of SMDN3, the cases

of SMDN3- ℓ_2 requires more CPU time on average than the the cases of SMDN3- ℓ_1 for each algorithm since there are cases with no closed-form gate-point formulations.

TABLE 6.5 – Average results of BSSS, DA and MW for each case of SMDN3 with oblique lines.

	BSSS	DA		MW	
Case	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
SMDN3- ℓ_1 - $m_1 < m_2 < 0$	30.86	4.05	0.01	0.01	0.06
SMDN3- ℓ_2 - $m_1 < m_2 < 0$	43.54	15.45	0.02	0.02	0.29
SMDN3- ℓ_1 - $m_2 < m_1 < 0$	26.81	3.71	0.02	0.01	0.03
SMDN3- ℓ_2 - $m_2 < m_1 < 0$	37.96	10.43	0.05	0.1	0.17
SMDN3- ℓ_1 - $m_1 < 0 < m_2$	29.22	3.81	0.02	0.01	0.14
SMDN3- ℓ_2 - $m_1 < 0 < m_2$	40.05	18.48	0.01	0.01	0.33
SMDN3- ℓ_2 - $m_1 = m_2 < 0$	44.00	4.04	0.01	0.02	0.35
Overall Average	36.06	8.57	0.02	0.03	0.20

Tables 6.6 and 6.7 include the results of the BSSS, DA and MW for the problems of SMDN3- ℓ_1 and SMDN3- ℓ_2 in which the slopes of the separating lines follow the condition of $0 > m_2 > m_1$. Similarly, Tables 6.8 and 6.9 provide the computational results for SMDN3- ℓ_1 and SMDN3- ℓ_2 when the slopes of the separating lines satisfy the condition of $0 > m_1 > m_2$. Furthermore, Tables 6.10 and 6.11 include the computational experiments addressing the triangle-shaped regions case of SMDN3- ℓ_1 and SMDN3- ℓ_2 problems, since the slopes of the separating lines satisfy $m_2 > 0 > m_1$. Finally, Table 6.12 reports the computational results that address the case of parallel separating lines for SMDN3- ℓ_2 as the slopes of the separating lines fulfill the condition of $0 > m_1 = m_2$. It should be added as a remark that we did not share the results on the case of parallel separating lines for SMDN3- ℓ_1 because it does not have any particular property in terms of gate-point formulations, unlike the SMDN3- ℓ_2 case.

TABLE 6.6 – Computational results for the SMDN3- ℓ_1 with oblique borderlines having slopes of $0 > m_2 > m_1$.

		BSSS		DA		MW	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
Class-I	I_{50}	19677.17	0.13	0.01	0.02	0.01	0.05
	I_{100}	39767.24	0.21	0.02	0.00	0.01	0.00
	I_{200}	82618.29	0.87	0.13	0.02	0.01	0.04
	I_{400}	160396.7	1.85	0.64	0.00	0.01	0.02
	I_{800}	315701.58	4.32	1.43	0.00	0.01	0.02
	I_{1000}	408655.77	6.68	1.87	0.00	0.01	0.01
	I_{2500}	999211.74	17.69	4.76	0.01	0.01	0.02
	I_{5000}	2029819.39	76.33	9.79	0.01	0.02	0.02
	I_{8000}	3235330.43	122.95	15.67	0.02	0.03	0.02
	I_{10000}	4030599.42	153.19	19.87	0.01	0.04	0.01
	Aver.		38.42	5.42	0.01	0.02	0.02
Class-II	B_{50}	191.86	0.01	0.01	0.04	0.01	0.14
	B_{287}	17759.26	0.04	0.25	0.04	0.01	0.00
	B_{654}	1772739.7	15.91	0.9	0.00	0.01	0.46
	B_{1060}	5047809.57	31.85	1.3	0.02	0.01	0.01
	Aver.		11.95	0.62	0.02	0.01	0.15
	Ov. aver.		30.86	4.05	0.01	0.01	0.06

TABLE 6.7 – Computational results for the SMDN3- ℓ_2 with oblique borderlines having slopes of $0 > m_2 > m_1$.

		BSSS		DA		MW	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
Class-I	I_{50}	21960.89	0.17	0.01	0.00	0.01	0.24
	I_{100}	45607.17	0.34	0.1	0.00	0.01	0.33
	I_{200}	90612.36	0.73	0.72	0.00	0.01	0.38
	I_{400}	178719.61	3.29	3.16	0.00	0.01	0.07
	I_{800}	352652.68	6.59	6.72	0.00	0.01	0.07
	I_{1000}	441280.85	10.54	8.1	0.03	0.01	0.02
	I_{2500}	1116883.47	28.46	21.15	0.01	0.01	0.06
	I_{5000}	2229238.54	102.29	26.47	0.03	0.02	0.04
	I_{8000}	3560311.5	171.53	54.08	0.01	0.03	0.07
	I_{10000}	4443874.18	234.87	70.97	0.00	0.04	0.05
	Aver.		55.88	19.15	0.01	0.02	0.13
Class-II	B_{50}	209.15	0.01	0.01	0.02	0.01	1.29
	B_{287}	21585.65	0.03	0.33	0.00	0.01	1.02
	B_{654}	1827405.71	20.03	10.35	0.12	0.02	0.46
	B_{1060}	5798022.28	30.71	14.18	0.02	0.01	0.04
	Aver.		12.70	6.22	0.04	0.01	0.70
	Ov. aver.		43.54	15.45	0.02	0.02	0.29

TABLE 6.8 – Computational results for the SMDN3- ℓ_1 with oblique borderlines having slopes of $0 > m_1 > m_2$.

		BSSS		DA		MW	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
Class-I	I_{50}	19311.00	0.10	0.03	0.02	0.01	0.06
	I_{100}	38998.25	0.20	0.02	0.01	0.01	0.02
	I_{200}	81242.16	0.78	0.12	0.02	0.01	0.07
	I_{400}	157402.42	1.49	0.58	0.00	0.01	0.02
	I_{800}	309732.49	3.39	1.31	0.00	0.01	0.03
	I_{1000}	400554.82	5.42	1.69	0.00	0.01	0.03
	I_{2500}	979358.05	15.46	4.37	0.00	0.02	0.02
	I_{5000}	1989963.84	66.16	9.39	0.01	0.02	0.03
	I_{8000}	3174252.79	108.10	14.05	0.02	0.03	0.03
	I_{10000}	3957632.99	135.73	18.52	0.02	0.03	0.03
	Aver.		33.68	5.01	0.01	0.02	0.04
Class-II	B_{50}	189.26	0.02	0.01	0.16	0.01	0.01
	B_{287}	17741.57	0.24	0.22	0.04	0.01	0.01
	B_{654}	1722112.33	10.23	0.68	0.00	0.01	0.02
	B_{1060}	5069667.73	28.06	0.99	0.05	0.01	0.01
	Aver.		9.64	0.48	0.06	0.01	0.01
	Ov. aver.		26.81	3.71	0.02	0.01	0.03

TABLE 6.9 – Computational results for the SMDN3- ℓ_2 with oblique borderlines having slopes of $0 > m_1 > m_2$.

		BSSS		DA		MW	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
Class-I	I_{50}	22642.43	0.06	0.01	0.00	0.01	1.11
	I_{100}	47147.33	0.32	0.06	0.00	0.01	0.37
	I_{200}	94119.95	0.71	0.44	0.00	0.38	0.41
	I_{400}	184030.16	2.12	2.45	0.00	0.01	0.11
	I_{800}	364114.59	7.09	5.04	0.00	0.02	0.04
	I_{1000}	456089.89	9.82	5.63	0.04	0.39	0.01
	I_{2500}	1153222.11	22.3	17.49	0.01	0.03	0.06
	I_{5000}	2301083.51	95.69	19.71	0.02	0.4	0.04
	I_{8000}	3674718.82	158.64	32.59	0.00	0.04	0.08
	I_{10000}	4585899.45	209.05	58.03	0.01	0.03	0.06
	Aver.		50.58	14.15	0.01	0.13	0.23
Class-II	B_{50}	218.74	0.03	0.01	0.00	0.01	0.01
	B_{287}	21588.02	0.03	0.24	0.39	0.01	0.01
	B_{654}	1868199.99	4.2	1.91	0.14	0.01	0.02
	B_{1060}	5830311.85	21.44	2.38	0.02	0.01	0.01
	Aver.		6.43	1.14	0.14	0.01	0.01
	Ov. aver.		37.96	10.43	0.05	0.10	0.17

TABLE 6.10 – Computational results for the SMDN3- ℓ_1 with oblique borderlines having slopes of $m_2 > 0 > m_1$.

		BSSS		DA		MW	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
Class-I	I_{50}	19911.29	0.1	0.01	0.00	0.01	0.24
	I_{100}	39339.13	0.19	0.02	0.00	0.01	0.09
	I_{200}	82695.37	0.77	0.13	0.00	0.01	0.10
	I_{400}	160393.59	1.82	0.62	0.00	0.01	0.06
	I_{800}	314793.63	4.26	1.34	0.00	0.01	0.05
	I_{1000}	408053.68	6.33	1.76	0.01	0.02	0.08
	I_{2500}	996032.85	17.34	4.55	0.00	0.01	0.05
	I_{5000}	2026292.13	71.99	9.27	0.00	0.02	0.08
	I_{8000}	3230128.02	116.67	14.79	0.00	0.03	0.08
	I_{10000}	4029239.36	148.02	18.66	0.00	0.03	0.09
	Aver.		36.75	5.12	0.00	0.02	0.09
Class-II	B_{50}	193.75	0.01	0.01	0.10	0.01	0.05
	B_{287}	17747.14	0.04	0.22	0.04	0.01	0.00
	B_{654}	1737673.58	14.4	0.7	0.08	0.01	0.52
	B_{1060}	5249942.78	27.1	1.21	0.01	0.01	0.45
	Aver.		10.39	0.54	0.06	0.01	0.25
	Ov. aver.		29.22	3.81	0.02	0.01	0.14

TABLE 6.11 – Computational results for the SMDN3- ℓ_2 with oblique borderlines having slopes of $m_2 > 0 > m_1$.

		BSSS		DA		MW	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
Class-I	I_{50}	21958.16	0.08	0.02	0.00	0.01	1.06
	I_{100}	45918.0	0.22	0.11	0.00	0.01	0.50
	I_{200}	91536.71	0.57	0.75	0.00	0.01	0.32
	I_{400}	178790.14	2.04	3.66	0.00	0.01	0.13
	I_{800}	353364.8	5.72	7.60	0.00	0.01	0.04
	I_{1000}	442291.85	7.87	9.02	0.04	0.01	0.10
	I_{2500}	1120121.82	21.87	23.63	0.01	0.01	0.06
	I_{5000}	2234346.58	110.18	30.93	0.02	0.02	0.06
	I_{8000}	3570534.04	169.43	66.97	0.00	0.03	0.08
	I_{10000}	4451337.29	212.4	85.62	0.00	0.04	0.07
	Aver.		53.04	22.83	0.01	0.02	0.24
Class-II	B_{50}	212.10	0.03	0.02	0.02	0.01	0.29
	B_{287}	21624.37	0.02	0.28	0.00	0.01	1.01
	B_{654}	1848563.76	28.93	10.09	0.00	0.01	0.46
	B_{1060}	5545891.38	1.27	20.01	0.04	0.01	0.44
	Aver.		7.56	7.60	0.01	0.01	0.55
	Ov. aver.		40.05	18.48	0.01	0.01	0.33

TABLE 6.12 – Computational results for the SMDN3- ℓ_2 with parallel oblique borderlines having slopes of $0 > m_1 = m_2$.

		BSSS		DA		MW	
	instances	obj. value	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%
Class-I	I_{50}	21513.1	0.13	0.01	0.01	0.01	0.46
	I_{100}	45076.77	0.3	0.02	0.00	0.01	0.54
	I_{200}	90395.17	0.7	0.13	0.00	0.01	0.38
	I_{400}	177389.73	2.12	0.64	0.00	0.01	0.23
	I_{800}	349540.75	6.7	1.41	0.00	0.01	0.09
	I_{1000}	438103.13	12.37	1.85	0.02	0.01	0.00
	I_{2500}	1107469.56	32.48	4.92	0.00	0.01	0.06
	I_{5000}	2213832.02	102.79	9.94	0.02	0.02	0.12
	I_{8000}	3533309.03	171.27	15.93	0.00	0.04	0.11
	I_{10000}	4409598.69	225.26	19.42	0.01	0.04	0.10
	Aver.		55.41	5.43	0.01	0.02	0.208
Class-II	B_{50}	203.42	0.01	0.01	0.02	0.01	0.63
	B_{287}	21588.02	0.03	0.26	0.00	0.01	1.04
	B_{654}	1770698.15	7.09	0.75	0.00	0.01	0.99
	B_{1060}	5659017.05	54.73	1.31	0.02	0.01	0.22
	Aver.		15.47	0.58	0.01	0.01	0.72
	Ov. aver.		44.00	4.04	0.01	0.02	0.35

6.2 Computational Experiments for SMDN3 Using BSSS, BS and Meta-heuristic Algorithms

This section presents the findings from our computational studies using the BS algorithm and two effective meta-heuristic techniques TS and SA in comparison the BSSS algorithm. The suggested algorithms are evaluated on a workstation running Windows Server 2016 with a 2.60 GHz Intel processor and 128 GB of RAM, and all algorithms are written in C++. Finally, all methods are subject to a 3600 second CPU time limit.

6.2.1 Test Bed

Our test bed consists of 14 instances, of which 5 are randomly generated, and the remaining 9 are the standard test instances originally considered for the Traveling Salesman Problem (TSP) and several location problems. The number of customers of the standard (random) test instances varies between 50 and 85,900 (100,000 and 250,000). Random test instances are generated in order to analyze the performance of the proposed algorithms in even larger-scaled instances. Among the 9 standard test instances, three of them are WP instances taken from Brimberg et al. (2000), 5 instances are from Farham et al. (2018) originally developed for a variant of the SFLP, and a TSP instance is inherited from Fredman et al. (1995). The test instances from the literature are named using the first two letters indicating their reference and the number of customers. For example, the test instance “Br-50 ” is adopted from

Brimberg et al. (2000) which consists of 50 customers. On the other hand, a random test instance is represented with “R” and its size. For instances, “R-100000” stands for a random test instance with 100,000 customers.

Given an instance with known customer coordinates, we define a rectangular hull of customers. Let H and L be the height and width of the rectangular hull of the customer locations, respectively. Let $(\underline{x}, \underline{y})$ and $(\bar{x}, \bar{y})$ represent the coordinates of the left-most bottom corner and right-most top corner points of the rectangular hull, respectively. We consider three different cases based on the slopes of two separating lines $s_1 \equiv y_1 = m_1x + n_1$ and $s_2 \equiv y_2 = m_2x + n_2$. Namely, we obtain three partitioning schemes that divide the rectangular hull of customer locations into three regions with (i) vertical lines, (ii) two separating lines with slopes of $m_1 < m_2 < 0$, and (iii) two separating lines with slopes of $m_1 < 0 < m_2$. To be specific, in the partitioning scheme (i), we consider two vertical separating lines such that $x = \underline{x} + 0.3L$ and $x = \underline{x} + 0.65L$ hold. For the partitioning scheme (ii), we take into account two separating lines drawn on the points $(\underline{x} + 0.2L, \bar{y})$ and $(\underline{x} + 0.4L, \underline{y})$ and $(\underline{x} + 0.5L, \bar{y})$ and $(\underline{x} + 0.8L, \underline{y})$. The coefficients of the separating lines are given as follows. $m_1 = -5H/L$, $n_1 = \bar{y} - m_1(\underline{x} + 0.2L)$ and $m_2 = -10H/3L$ and $n_2 = \bar{y} - m_2(\underline{x} + 0.5L)$. Finally, in the partitioning scheme (iii), we address two borderlines which pass through the points $(\underline{x}, \bar{y})$ and $(\underline{x} + 0.4L, \underline{y})$ and $(\bar{x}, \bar{y})$ and $(\underline{x} + 0.4L, \underline{y})$. In case (iii), the coefficient of the separating lines are as follows : $m_1 = -2.5H/L$, $n_1 = \underline{y} - m_1(\underline{x} + 0.4L)$ and $m_2 = 5H/3L$ and $n_2 = \underline{y} - m_2(\underline{x} + 0.4L)$. For the sake of clarity, the rectangular hull and three partitioning schemes are illustrated with Figure 6.1.

From partitioning the rectangular hull by drawing two separating lines, we obtain three adjacent regions Ω_1 , Ω_2 , and Ω_3 for which we consider two different combinations of distance norms ℓ_1, ℓ_2, ℓ_1 and ℓ_2, ℓ_1, ℓ_2 . Hence, for every instance, we consider three partitioning schemes as described, each having two different distance norm settings, that is, the norms ℓ_1, ℓ_2, ℓ_1 or the norms ℓ_2, ℓ_1, ℓ_2 . Therefore, this study experiments on $14 \times 3 \times 2 = 84$ test instances in total.

6.2.2 Parameter Calibration

The performance of the meta-heuristics and BS depends on several parameters. Here, we summarize how these parameters are chosen for the proposed approximate solution procedures, as shown in Table 6.12. The values stated in bold indicate that they are used for the results reported in the remaining.

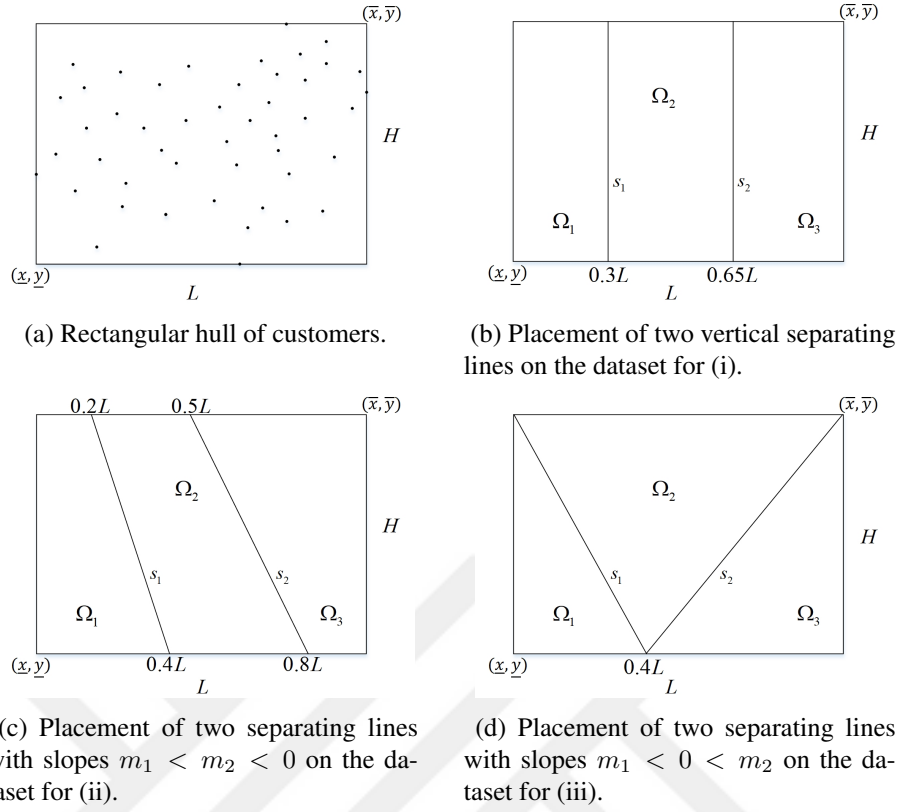


FIGURE 6.1 – Rectangular hull of customers and placement of separating lines in SMDN3

TABLE 6.13 – Parametric calibration

	Parameter	Value
TS	Θ	3, 5, 10
	Ξ	500
	Γ	3, 5, 10
	μ	0.5
	χ	5
SA	β	0.9
	σ	5
	$\overline{T}$	5, 10, 20
BS	$\mathcal{W}$	10
	ψ	0.5

The adjustments to the TS parameters are detailed below. First, tabu tenure Θ is chosen as 5 among the values $\{3, 5, 10\}$. The iteration limit Ξ is fixed to 500. Similarly, Γ is chosen as 5, considering values 3, 5, and 10 to properly control the updates of the radius r . Finally, to balance efficiency and accuracy performance, we have set the radius update parameter for intensification as $\mu = 0.5$ and the iteration limit for intensification as $\chi = 5$.

For the SA algorithm, the initial temperature value is fixed at $\overline{T} = 10$ among the tested values of 5, 10 and 20. Radius updating parameter α and cooling parameter β are both chosen as

0.9. Additionally, the maximum number of iterations to search a region σ is set to 5 in order to allow returning to the previously considered region without spending excessive CPU time.

Finally, the parameters of BS are determined as follows. The beam width $\mathcal{W}$ is chosen as 10, striking a balance between the CPU time requirement and the accuracy of the BS. In addition, ψ is fixed to 0.5 which is determined through our preliminary experiments.

6.2.3 Computational Results

The detailed experimental results are included in Tables 6.14-6.20. The average performance of case of SMDN3 is reported in Table 6.14. Table 6.15 and Table 6.16 include the computations on the test instances where the rectangular hull covering all customers is partitioned by two vertical separating lines following the norms ℓ_1, ℓ_2, ℓ_1 and ℓ_2, ℓ_1, ℓ_2 , respectively. Similarly, computational results obtained with the SMDN3 instances considering separating lines with slopes of $m_1 < m_2 < 0$ ($m_1 < 0 < m_2$), and taking into account the norms ℓ_1, ℓ_2, ℓ_1 and ℓ_2, ℓ_1, ℓ_2 are reported in Table 6.17 and Table 6.18 (Table 6.19 and Table 6.20), respectively.

The structure of each table is given as follows. The first column includes the names of the test instances, and the second column presents for the CPU time of the BSSS algorithm. Columns 3 to 8 incorporate the CPU time in seconds under “cpu(s.)” and the percentage gap of the results obtained with TS, SA, and BS in comparison to the BSSS, denoted with “gap%”, respectively. The percentage gap is calculated as $gap\% = 100 \times (Z - Z^*) / Z^*$ where Z^* and Z are the values of the objective function obtained with the BSSS algorithm and the corresponding heuristic algorithm, respectively. We should note that the values reported in columns 3 to 6 are the averages of the results obtained with 10 different replications of TS and SA in each instance starting from a random initial facility location within the convex hull of the customer locations. Finally, the last rows of Tables 6.15-6.20 present the average values of the corresponding columns where the best results are also marked with a “*” character.

The overall averages of the results presented in Tables 6.15-6.20 are compiled in Table 6.14. Particularly, we observe that the BS yields the best overall percentage gap of 0.00% and is also capable of finding the best-known solution for most instances. On the other hand, the SA is the most efficient solution procedure with an overall CPU time of 0.66 seconds. The slowest performance is yielded by the TS, which results in an overall average CPU time of 8.62 seconds.

TABLE 6.14 – Average of the computational results of heuristic methods and BSSS for SMDN3.

	BSSS	TS		SA		BS	
Case	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%
SMDN3- ℓ_1 -vertical	1494.60	5.06	0.38	0.44	0.17	2.00	0.00
SMDN3- ℓ_2 -vertical	1542.94	6.18	0.00	0.49	0.03	1.82	0.00
SMDN3- ℓ_1 - $m_1 < m_2 < 0$	1599.96	8.79	0.14	0.67	0.09	2.21	0.00
SMDN3- ℓ_2 - $m_1 < m_2 < 0$	1687.70	11.17	0.28	0.89	0.51	2.92	0.00
SMDN3- ℓ_1 - $m_1 < 0 < m_2$	1601.70	8.66	0.65	0.63	0.06	2.04	0.00
SMDN3- ℓ_2 - $m_1 < 0 < m_2$	1759.92	11.89	1.29	0.81	0.14	2.74	0.00
Overall Average	1614.47	8.62	0.46	0.66*	0.17	2.29	0.00*

In Tables 6.15 and 6.16, the results are reported for SMDN3- ℓ_1 and SMDN3- ℓ_2 with vertical separating lines, respectively. We observe that the BS yields the best average accuracy, i.e. percentage gap, of 0.00% for both cases, while the SA algorithm outputs the second-best average accuracy with 0.17% and 0.03%, respectively. Here, the average accuracy of the TS is 0.00% for 12 and 13 out of 14 instances, and SA emerges to be the most efficient approach with an average CPU time of 0.44 and 0.49 secs, respectively.

The results for SMDN3- ℓ_1 and SMDN3- ℓ_2 considering two separating lines with slopes $m_1 < m_2 < 0$ are given with Table 6.17 and Table 6.18, respectively. Note that the computational effort required is significantly higher than that of SMDN3 with vertical separating line cases, for which the results are reported in Tables 6.15 and 6.16. This difference can be attributed to the complexity of the distance functions, which significantly affects the computation times. In Table 6.17, we observe that BS (SA) is the algorithm that performs the best (second-best) with an average precision of 0.00% (0.09%). On the other hand, the TS produces an average precision of 0.00% for 13 and 9 out of 14 instances for SMDN3- ℓ_1 and SMDN3- ℓ_2 , respectively. From Tables 6.17 and 6.18, we notice that the SA yields the shortest CPU times with 0.67 secs. and 0.89 secs.

Table 6.19 and Table 6.20 respectively incorporate the results for SMDN3- ℓ_1 and SMDN3- ℓ_2 considering two separating lines with slopes $m_1 < 0 < m_2$. Similarly, in these tables, BS (SA) produces the most accurate (second most accurate) results with average percentage gaps of 0.00% (0.06%) and 0.00% (0.14%), respectively. We should note that, although the TS achieves a lower average accuracy of 0.65% and 1.29%, it demonstrates a superior average accuracy of 0.00% for 11 of the 14 instances in Table 6.19 and for 9 of the 14 instances in Table 6.20 compared to SA. Regarding efficiency, SA surpasses other algorithms in average CPU times of 0.63 seconds and 0.81 seconds from Table 6.19 and Table 6.20, respectively.

TABLE 6.15 – Computational results of heuristic methods for the SMDN3- ℓ_1 with vertical borderlines.

	BSSS	TS		SA		BS	
test instance	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%
Br-50	0.15	0.01	0.00	0.01	0.00	0.01	0.00
Br-654	16.45	0.04	0.00	0.01	0.05	0.01	0.00
Br-1060	23.69	0.08	0.00	0.02	0.03	0.02	0.00
Fa-2319	43.61	0.14	0.10	0.04	0.13	0.04	0.00
Fa-7397	188.58	0.39	5.15	0.08	1.56	0.11	0.00
Fa-11849	554.23	1.18	0.00	0.07	0.03	0.36	0.00
Fa-13509	271.20	1.09	0.00	0.14	0.06	0.23	0.00
Fa-33810	535.81	1.72	0.00	0.36	0.02	0.41	0.00
Fr-85900	1289.38	5.48	0.00	0.66	0.05	1.01	0.00
R-100000	3600.12	6.55	0.00	0.85	0.09	3.05	0.00
R-130000	3600.16	10.17	0.00	1.04	0.08	4.09	0.00
R-160000	3600.22	11.29	0.00	1.28	0.08	4.87	0.00
R-200000	3600.25	13.99	0.00	1.61	0.11	6.11	0.00
R-250000	3600.59	18.67	0.00	0.01	0.07	7.67	0.00
Average	1494.60	5.06	0.38	0.44*	0.17	2.00	0.00*

TABLE 6.16 – Computational results of heuristic methods for the SMDN3- ℓ_2 with vertical borderlines.

	BSSS	TS		SA		BS	
test instance	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%
Br-50	0.29	0.01	0.03	0.01	0.00	0.01	0.00
Br-654	30.22	0.05	0.00	0.01	0.01	0.02	0.00
Br-1060	37.95	0.07	0.00	0.01	0.01	0.02	0.00
Fa-2319	108.15	0.14	0.00	0.02	0.08	0.03	0.00
Fa-7397	57.83	0.43	0.00	0.04	0.02	0.10	0.00
Fa-11849	833.24	1.41	0.00	0.09	0.02	0.33	0.00
Fa-13509	278.39	1.25	0.00	0.07	0.06	0.23	0.00
Fa-33810	581.27	1.78	0.00	0.14	0.02	0.40	0.00
Fr-85900	1672.83	5.30	0.00	0.33	0.02	0.95	0.00
R-100000	3600.12	8.35	0.00	0.75	0.05	2.77	0.00
R-130000	3600.17	12.39	0.00	0.97	0.03	3.62	0.00
R-160000	3600.19	14.11	0.00	1.18	0.04	4.44	0.00
R-200000	3600.26	17.26	0.00	1.47	0.07	5.57	0.00
R-250000	3600.30	23.95	0.00	1.84	0.06	6.97	0.00
Average	1542.94	6.18	0.00*	0.49*	0.03	1.82	0.00*

TABLE 6.17 – Computational results of heuristic methods for the SMDN3- ℓ_1 with oblique borderlines as $0 > m_2 > m_1$.

	BSSS	TS		SA		BS	
test instance	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%
Br-50	0.27	0.01	0.00	0.01	0.00	0.01	0.00
Br-654	34.59	0.07	0.00	0.01	0.05	0.02	0.00
Br-1060	37.82	0.11	2.00	0.01	0.20	0.03	0.00
Fa-2319	75.15	0.22	0.00	0.02	0.07	0.05	0.00
Fa-7397	392.53	0.68	0.00	0.06	0.01	0.16	0.00
Fa-11849	647.74	1.92	0.00	0.12	0.21	0.37	0.00
Fa-13509	393.53	2.05	0.00	0.11	0.09	0.30	0.00
Fa-33810	855.30	2.50	0.00	0.20	0.03	0.59	0.00
Fr-85900	1961.40	7.70	0.00	0.48	0.02	1.43	0.00
R-100000	3600.13	12.39	0.00	0.99	0.14	3.46	0.00
R-130000	3600.17	17.90	0.00	1.33	0.14	4.28	0.00
R-160000	3600.19	19.28	0.00	1.61	0.10	5.33	0.00
R-200000	3600.25	24.61	0.00	2.01	0.09	6.61	0.00
R-250000	3600.32	33.60	0.00	2.45	0.16	8.24	0.00
Average	1599.96	8.79	0.14	0.67*	0.09	2.21	0.00*

TABLE 6.18 – Computational results of heuristic methods for the SMDN3- ℓ_2 with oblique borderlines as $0 > m_2 > m_1$.

	BSSS	TS		SA		BS	
test instance	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%
Br-50	0.12	0.01	0.01	0.01	0.00	0.01	0.00
Br-654	41.95	0.52	0.93	0.01	0.02	0.03	0.00
Br-1060	51.10	0.19	2.01	0.01	0.08	0.03	0.00
Fa-2319	146.12	0.25	0.00	0.03	0.04	0.07	0.00
Fa-7397	69.58	4.03	0.62	1.27	6.25	0.35	0.06
Fa-11849	1452.36	2.67	0.39	0.14	0.14	0.50	0.00
Fa-13509	382.53	2.38	0.00	0.17	0.07	0.42	0.00
Fa-33810	902.13	2.98	0.00	0.31	0.02	0.90	0.00
Fr-85900	2580.86	9.46	0.00	0.70	0.03	1.92	0.00
R-100000	3600.13	14.55	0.00	1.20	0.06	4.36	0.00
R-130000	3600.18	22.46	0.00	1.52	0.09	5.73	0.00
R-160000	3600.19	25.92	0.00	1.90	0.08	6.95	0.00
R-200000	3600.27	33.63	0.00	2.30	0.13	8.74	0.00
R-250000	3600.32	37.36	0.00	2.93	0.12	10.82	0.00
Average	1687.70	11.17	0.28	0.89*	0.51	2.92	0.00*

TABLE 6.19 – Computational results of heuristic methods for the SMDN3- ℓ_1 with oblique borderlines as $m_2 > 0 > m_1$.

	BSSS	TS		SA		BS	
test instance	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%
Br-50	0.24	0.01	0.00	0.01	0.00	0.01	0.00
Br-654	32.40	0.07	1.96	0.01	0.03	0.02	0.00
Br-1060	37.55	0.12	6.79	0.01	0.12	0.03	0.00
Fa-2319	75.31	0.20	0.00	0.02	0.15	0.06	0.00
Fa-7397	498.22	0.69	0.00	0.05	0.01	0.15	0.00
Fa-11849	662.00	2.00	0.42	0.11	0.12	0.34	0.00
Fa-13509	329.32	2.15	0.00	0.11	0.07	0.30	0.00
Fa-33810	836.64	2.46	0.00	0.17	0.03	0.52	0.00
Fr-85900	1951.03	7.50	0.00	0.44	0.03	1.30	0.00
R-100000	3600.13	11.81	0.00	0.94	0.06	3.06	0.00
R-130000	3600.16	17.34	0.00	1.22	0.08	4.06	0.00
R-160000	3600.19	19.53	0.00	1.48	0.09	4.97	0.00
R-200000	3600.26	24.80	0.00	1.87	0.07	6.15	0.00
R-250000	3600.34	32.58	0.00	2.33	0.06	7.64	0.00
Average	1601.70	8.66	0.65	0.63*	0.06	2.04	0.00*

TABLE 6.20 – Computational results of heuristic methods for the SMDN3- ℓ_2 with oblique borderlines as $m_2 > 0 > m_1$.

	BSSS	TS		SA		BS	
test instance	cpu (s.)	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%
Br-50	0.11	0.01	0.01	0.01	0.00	0.01	0.00
Br-654	45.57	0.33	2.35	0.02	1.08	0.03	0.00
Br-1060	49.46	0.18	5.94	0.01	0.06	0.03	0.00
Fa-2319	52.34	0.25	0.00	0.02	0.14	0.07	0.00
Fa-7397	541.50	19.41	8.59	0.14	0.01	0.25	0.00
Fa-11849	1103.52	2.62	1.20	0.15	0.07	0.47	0.00
Fa-13509	51.53	3.29	0.00	0.25	0.13	0.49	0.00
Fa-33810	1193.69	3.18	0.00	0.33	0.02	0.78	0.00
Fr-85900	3600.09	9.51	0.00	0.81	0.02	1.81	0.00
R-100000	3600.15	14.16	0.00	1.16	0.09	4.11	0.00
R-130000	3600.18	20.90	0.00	1.55	0.04	5.37	0.00
R-160000	3600.20	24.29	0.00	1.79	0.10	6.53	0.00
R-200000	3600.27	29.11	0.00	2.25	0.15	8.13	0.00
R-250000	3600.32	39.26	0.00	2.91	0.07	10.25	0.00
Average	1759.92	11.89	1.29	0.81*	0.14	2.74	0.00*

We further analyze the performance of the two meta-heuristic algorithms following the footsteps of Aiex et al. (2007) and Ribeiro and Rosseti (2015). To this end, we assume that the time-to-target values, i.e. the elapsed time to reach a specified upper bound value of the meta-heuristics fit into the two-parameter shifted exponential distribution (Hoose and Stützle, 1998). Figures 6.2a and 6.3a present the time-to-target (TTT) plots for TS and SA, respectively. Similarly, Figure 6.2b and Figure 6.3b depict quantile-quantile (QQ) plots of the TS and the SA. In the TTT plots, the abscissa axis represents the TTT value, and the ordinate axis shows the probability that the algorithm reaches the target solution. On the other hand, in the QQ plots, the abscissa axis indicates exponential quantiles of a two-parameter shifted exponential distribution, and the ordinate axis points the CPU times measured. These plots are prepared using the instance with 250,000 customers for SMDN3- ℓ_2 considering two separating lines with slopes of $m_1 < m_2 < 0$. This instance requires the longest computation times among the suggested solution approaches, i.e., BS, TS, and SA. Each of the proposed TS and SA is executed repeatedly 50 times for this instance. In our experiments, the target value, namely the upper bound value, is chosen to be %1 higher than the best known solution. Then the CPU times of 50 runs required by TS and SA to reach this upper bound value are collected separately. As it can be observed from Figure 6.2a and Figure 6.3a, the probability of reaching the given target value by TS and SA is at most 0.54 and 0.64 seconds (1.71 and 1.08 seconds) is approximately 40% (80%). Therefore, in general, we can claim that the SA achieves more accurate solutions with better efficiency than, or at least comparable efficiency to, TS.

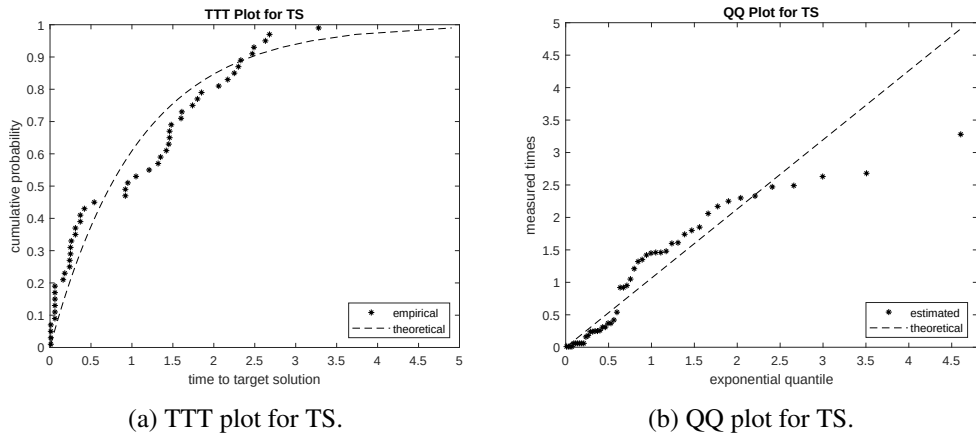


FIGURE 6.2 – TTT and QQ plot of TS for the SMDN3- ℓ_2 instance with 250,000 customers considering two separating lines with slopes $m_1 < m_2 < 0$.

In addition, the average percentage gap values per iteration observed for TS and SA are depicted in Figures 6.4a and 6.4b, respectively. These figures are prepared based on the

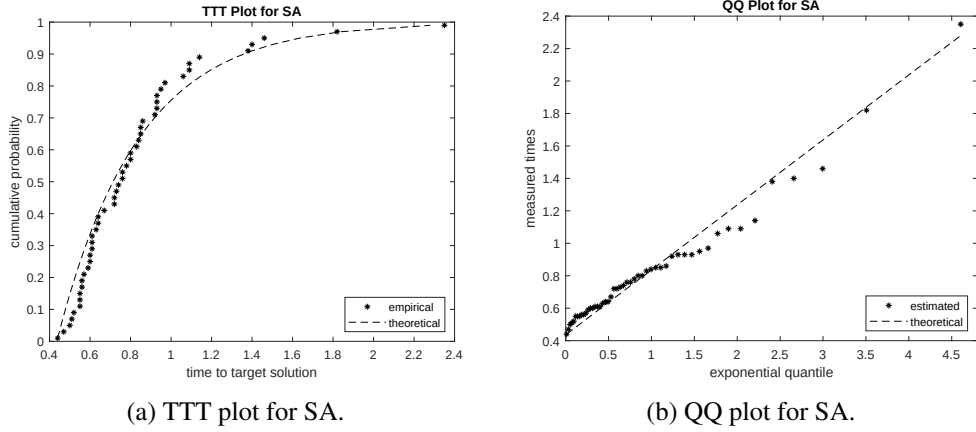


FIGURE 6.3 – TTT and QQ plot of SA for the SMDN3- ℓ_2 instance with 250,000 customers considering two separating lines with slopes $m_1 < m_2 < 0$.

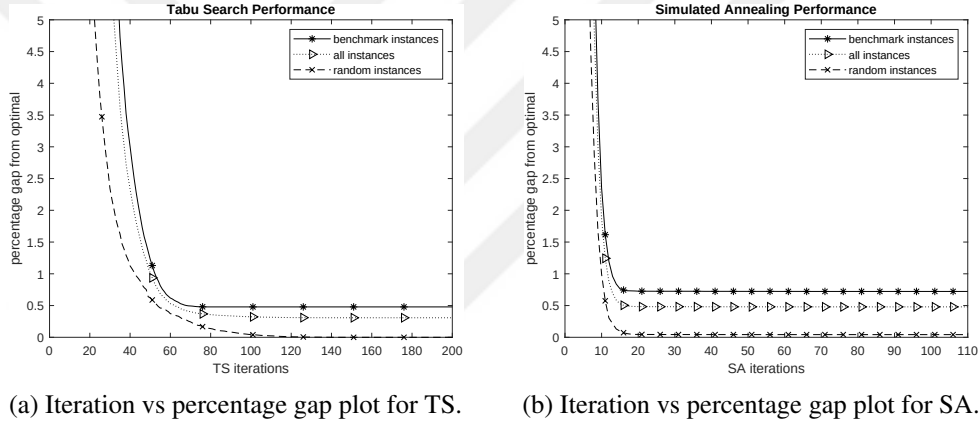


FIGURE 6.4 – Number of iterations vs average percentage gap plots for the SMDN3- ℓ_2 instance with 250,000 customers considering two separating lines with slopes $m_1 < m_2 < 0$.

outcomes of 10 runs for all instances of the SMDN3- ℓ_2 considering two separating lines with slopes $m_1 < m_2 < 0$, which totally makes 140 separate runs for each meta-heuristic. From Figure 6.4a (Figure 6.4b), we find that the average percent gap of TS (SA) drops below 1% in the first 51 (11) iterations of 200 (110).

6.3 Computational Experiments for LRP-M with Two Regions

This section outlines the computational experiments carried out on randomly generated test instances for two different variants of LRP-M. The suggested algorithms are implemented in C++ and evaluated on a 64-bit Windows Server 2016 workstation with Intel CPUs clocked at 2.60 GHz and 128 GB of RAM. Note that all methods are subject to a 7200 second CPU time constraint.

We use GAMS 24.8 environment while solving the various cases of LRP-M. BARON is used as a commercial solver for MINLP formulations of LRP-M. The other commercial MINLP solvers are not reported since they produce unstable results when attempting to solve MINLP models. To solve the MILP models, both CPLEX and GUROBI are employed. Note that, other than CPU time restriction, all of the solvers are employed using their default settings.

6.3.1 Test Bed

Computational experiments are conducted considering randomly generated 10 test instances for different variants of the LRP-M, such as LRP-M with a vertical separating line and LRP-M with an oblique separating line. The number of customers n varies from 5 to 100 customers. In the discrete approximation of the facility locations, the customer locations are used as candidate facility locations. In addition, considering the discrete gate point MINLP model of LRP-M with oblique separating lines, the number of candidate gate-points is chosen as 5 to prevent a loss of efficiency in the computations.

6.3.2 Computational Results

Our experimental results are reported in Tables 6.21 and 6.22, addressing LRP-M with a vertical separating line and LRP-M with an oblique separating line, respectively. The first row of each table denotes the name of the formulation used by commercial solvers. Throughout the computations, the optimality gap is computed as $gap\% = 100 \times (Z - Z^*)/Z^*$ where Z^* and Z are the best objective function value obtained and the objective function value of the current algorithm, respectively.

Since LRP-M is limited to a single route (TSP), the optimal location of the facility is in one of the customer locations, making the optimal objective function values of the MILP and MINLP formulations comparable.

Considering LRP-M with a vertical separation line, performance of the two different formulations namely MINLP- V and MILP- F are tested. The performance of commercial solvers using MILP- F and MINLP- V formulations are reported in Table 6.21. Both CPLEX and GUROBI using MILP- F formulation outperforms BARON using the MINLP- V formulation in terms of efficiency while BARON yields the best average accuracy only by outperforming GUROBI by a small margin. Comparing the performance of both CPLEX and GUROBI using MILP- F , it is clear that GUROBI outperforms CPLEX in every way.

TABLE 6.21 – Experiments with the models for LRP-M with vertical separating line.

sample size	MILP- F				MINLP- V	
	GUROBI		CPLEX		BARON	
	gap%	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)
5	0.00	0.3	0.00	0.69	0.00	1.35
8	0.00	0.37	0.00	1.04	0.00	2.51
10	0.00	0.45	0.00	2.89	0.00	14.31
15	0.00	0.54	0.00	0.64	0.00	5.16
25	0.00	6.73	2.21	2.89	0.22	23.89
40	5.02	26.87	0.06	179.8	0.00	53.18
50	0.00	40.03	0.31	1304.9	1.67	136.23
65	0.00	196.33	1.91	339.27	0.17	5509.33
80	2.89	576.02	2.68	269.23	0.00	4545.66
100	0.00	6229.02	13.87	7201.09	4.84	7218.79
Average	0.79	707.67	2.10	930.24	0.69	1751.04

TABLE 6.22 – Experiments with the models for LRP-M with oblique separating line

sample size	MILP- F				MINLP- O		MINLP- G	
	GUROBI		CPLEX		BARON		BARON	
	gap%	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)	gap%	cpu (s.)
5	1.85	0.34	1.85	0.35	0.00	25.06	1.67	9.19
8	0.74	0.4	0.74	0.7	0.00	8.85	0.74	28.2
10	0.98	0.42	0.98	2.04	0.00	949.71	0.17	1.22
15	1.10	1.21	1.11	0.6	0.00	10.17	0.00	3.97
25	0.98	4.58	1.19	4.64	2.01	7202.1	0.00	458.24
40	0.34	42.81	2.6	103.3	0.06	879.26	0.00	4140.37
50	5.49	35.75	5.32	244.5	0.00	79.44	5.95	225.91
65	1.31	161.88	0.00	1549.76	1.59	7216.4	1.79	527.0
80	0.95	431.71	0.71	5206.59	0.00	6487.53	27.2	565.36
100	0.00	5603.61	19.68	7201.18	16.86	7258.31	6.52	7249.68
Average	1.37	628.27	3.42	1431.34	2.05	3011.68	4.4	1320.91

Following the computational results of our experiments on LRP-M with an oblique separating line from Table 6.22, considering the MILP- F formulation, GUROBI outperforms CPLEX with a significant margin. Furthermore, we can see that solving MINLP- O formulation of the continuous location problem takes significantly more computational effort compared to the MINLP- V formulation for the LRP-M with a vertical separating line. In addition, BARON performs more efficiently using MINLP- G formulation with discrete gate-points at the cost of accuracy comparison to the performance of BARON in MINLP- O .

7 CONCLUSION

The continuous facility location problem with multiple regions and different regions is difficult to solve even when considering a single facility. Considering SMDN with more than two regions with different norms, even finding the location of the gate-points that are used for each shortest path is proven to require significant computational effort. In this study, each significant case of SMDN3 with ℓ_1 and ℓ_2 norms is extensively studied with the goal of taking a significant step towards generalizing the shortest path and gate-point formulations of SMDN3. In the SMDN3, it is observed that there are multiple cases where a closed-form solution of a gate-point location is not possible. This significantly influences the performance of each algorithm developed to solve the SMDN3. Furthermore, when considering four or more regions having different norms, the number of cases without closed-form solutions of the gate-point locations increases significantly.

Considering the performance of the exact solution methods of SMDN3, the BSSS algorithm provides ϵ -optimal location solutions within a reasonable time for each case of SMDN3. Note that depending on the definition of ϵ , the CPU time BSSS algorithm can increase significantly without a significant improvement on the objective function value. Alternatively, considering a specific case of SMDN3 with vertical separating lines, the commercial solvers KNITRO and BARON are also viable in solving the MINLP model of SMDN3 with vertical separating lines. However MINLP solvers yield poor performance compared to BSSS, especially in larger-scale instances. In addition, by decomposing the MINLP model into three NLP models, the performance of KNITRO becomes comparable to the performance of BSSS when considering a special case of SMDN3 with vertical separating lines. Note that it is not possible to model SMDN3 as MINLP in a way that commercial solvers can handle when considering the cases of SMDN3 with oblique separating lines. This leaves the BSSS as the only viable and versatile exact solution approach.

As observed from the first set of experiments from 50 to 10,000 customers, the heuristic methods, i.e., MW and DA, perform more efficiently in comparison to the BSSS while both preserve reasonable accuracy. MW is the most efficient method, while DA yields more accurate results at the cost of efficiency.

Furthermore, from the second set of experiments from 50 to 250, 000, more efficient heuristic algorithms are employed. Two meta-heuristic methods, i.e., Tabu Search and Simulated Annealing, and one BSSS-based heuristic, i.e., Beam Search provide good efficiency when solving SMDN3, especially for larger-scale instances where the efficiency of BSSS starts to fall behind. According to the results, the SA yields the most efficient results due to its simplicity, while the BS provides the most accurate (sometimes the ϵ -optimal) solutions.

Lastly, considering the location routing problem with two regions having different norms, two different cases of LRP-M using a single route, i.e., vertical and oblique separating line cases, are studied. As long as LRP-M has a single-route, solving the MILP model using the discrete approximation of the facility locations can yield the optimal solution.

For both cases of LRP-M, alternative MILP models are presented using the discrete approximation of the facility locations. Additionally, an alternative MINLP model using the discrete approximation of the gate-point locations is also considered for LRP-M with the oblique separating line case, since an alternative formulation using discrete gate-points is not necessary for LRP-M with vertical separating line. Following the computations, the efficiency of BARON using the MINLP model yields the slowest performance compared to GUROBI and CPLEX using the MILP formulation in both cases. GUROBI produces the most efficient results while preserving a decent accuracy on average for both cases. Lastly, for the LRP-M with oblique lines, the performance of BARON on the alternative MINLP model using the discrete gate-points is more efficient compared to original model while yielding less accurate results.

The further research topics are as follows. Considering SMDN, finding the closed-form solution to the gate-point formulations in SMDN with more than three regions is remains to be found. This study only covers the cases where the separating lines are straight lines. SMDN using more complex-shaped separating lines, such as curved lines, is another topic of research. In addition, distance norms are restrictive and may not yield very accurate distance approximations in some domain-specific instances. Therefore, further studies can consider combinations of different distance metrics that generate better distance approximations for some applications.

Considering the solution approaches for SMDN, it has been proven that the discrete approximation approaches are both efficient and practical in real-life applications. Discrete approximation scheme can also be applied to various other continuous location problems using different distance metrics such as LRP-M or Multiple-Facility Location Problem with

two regions having different distance metrics to provide an efficient result. Furthermore, rather than solving the model of the continuous location problem, geometric heuristic search approaches or geometric branch-and-bound algorithms such as BSSS should also be considered.



REFERENCES

- Aiex, R. M., Resende, M. G., and Ribeiro, C. C. (2007). TTT plots : A perl program to create time-to-target plots. *Optimization Letters*, 1 : 355–366.
- Akyüz, M. H. (2017). The capacitated multi-facility Weber problem with polyhedral barriers : efficient heuristic methods. *Computers & Industrial Engineering*, 113 : 221–240.
- Akyüz, M. H., Altınel, İ. K., and Öncan, T. (2014). Location and allocation based branch and bound algorithms for the capacitated multi-facility Weber problem. *Annals of Operations Research*, 222(1) : 45–71.
- Akyüz, M. H., Öncan, T., and Altınel, İ. K. (2012). Efficient approximate solution methods for the multi-commodity capacitated multi-facility Weber problem. *Computers & Operations Research*, 39(2) : 225–237.
- Akyüz, M. H., Öncan, T., and Altınel, İ. K. (2013). Beam search heuristics for the single and multi-commodity capacitated multi-facility Weber problems. *Computers & Operations Research*, 40(12) : 3056–3068.
- Akyüz, M. H., Öncan, T., and Altınel, İ. K. (2019). Branch and bound algorithms for solving the multi-commodity capacitated multi-facility Weber problem. *Annals of Operations Research*, 279(1) : 1–42.
- Almouhanna, A., Quintero-Araujo, C. L., Panadero, J., Juan, A. A., Khosravi, B., and Ouelhadj, D. (2020). The location routing problem using electric vehicles with constrained distance. *Computers & Operations Research*, 115 : 104864.
- Altay, G., Akyüz, M. H., and Öncan, T. (2023). Solving a minisum single facility location problem in three regions with different norms. *Annals of Operations Research*, 321 : 1–37.
- Aras, N., Altınel, İ. K., and Orbay, M. (2007). New heuristic methods for the capacitated multi-facility Weber problem. *Naval Research Logistics*, 54(1) : 21–32.

- Bazaraa, M. S., Sherali, H. D., and Shetty, C. M. (2006). *Nonlinear programming : Theory and algorithms*, third edn, John Wiley and Sons, NJ, USA.
- Boussaïd, I., Lepagnot, J., and Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences*, 237 : 82–117.
- Brimberg, J., Drezner, Z., Mladenović, N., and Salhi, S. (2014). A new local search for continuous location problems. *European Journal of Operational Research*, 232(2) : 256–265.
- Brimberg, J., Hansen, P., Mladenovic, N., and Salhi, S. (2008). A survey of solution methods for the continuous location-allocation problem. *International Journal of Operations Research*, 5(1) : 1–12.
- Brimberg, J., Hansen, P., Mladenović, N., and Taillard, E. D. (2000). Improvements and comparison of heuristics for solving the uncapacitated multisource Weber problem. *Operations Research*, 48(3) : 444–460.
- Brimberg, J., Kakhki, H. T., and Wesolowsky, G. O. (2003). Location among regions with varying norms. *Annals of Operations Research*, 122 : 87–102.
- Brimberg, J., Kakhki, H. T., and Wesolowsky, G. O. (2005). Locating a single facility in the plane in the presence of a bounded region and different norms. *Journal of the Operations Research Society of Japan*, 48(2) : 135–147.
- Brimberg, J., Love, R., and Mladenović, N. (2009). Extension of the Weiszfeld procedure to a single facility minisum location model with mixed ℓ_p -norms. *Mathematical Methods of Operations Research*, 70(2) : 269–283.
- Brimberg, J., Walker, J. H., and Love, R. F. (2007). Estimation of travel distances with the weighted ℓ_p -norm : some empirical results. *Journal of Transport Geography*, 15(1) : 62–72.
- Cacchiani, V., Contreras-Bolton, C., and Toth, P. (2020). Models and algorithms for the traveling salesman problem with time-dependent service times. *European Journal of Operational Research*, 283(3) : 825–843.
- Croce, F. D., Ghirardi, M., and Tadei, R. (2004). Recovering beam search : Enhancing the beam search approach for combinatorial optimization problems. *Journal of Heuristics*, 10(1) : 89–104.
- Drexler, M. and Schneider, M. (2015). A survey of variants and extensions of the location-routing problem. *European Journal of Operational Research*, 241(2) : 283–308.

- Drezner, Z., Brimberg, J., Mladenović, N., and Salhi, S. (2016). New local searches for solving the multi-source Weber problem. *Annals of Operations Research*, 246(1-2) : 181–203.
- Estève, B., Aubijoux, C., Chartier, A., and T'kindt, V. (2006). A recovering beam search algorithm for the single machine just-in-time scheduling problem. *European Journal of Operational Research*, 172(3) : 798–813.
- Farahani, R. Z. and Hekmatfar, M. (2009). Facility location : Concepts, models, algorithms and case studies, first edn, Springer-Verlag, Berlin.
- Farham, M. S., Süral, H., and İyigun, C. (2018). Generalization of the restricted planar location problems : Unified metaheuristic algorithms. *Computers & Operations Research*, 99 : 48–66.
- Francis, R. L., McGinnis, L. F., and White, J. A. (1992). Facility layout and location : An analytical approach, second edn. Prentice-Hall, NJ.
- Franco, L., Velasco, F., and Gonzalez-Abril, L. (2012). Gate points in continuous location between regions with different ℓ_p norms. *European Journal of Operational Research*, 218(3) : 648–655.
- Franco, L., Velasco, F., Gonzalez-Abril, L., and Mesa, J. A. (2018). Single-facility location problems in two regions with 1-and q-norms separated by a straight line. *European Journal of Operational Research*, 269(2) : 577–589.
- Franzin, A. and Stützle, T. (2019). Revisiting simulated annealing : A component-based analysis. *Computers & Operations Research*, 104 : 191–206.
- Fredman, M. L., Johnson, D. S., McGeoch, L. A., and Ostheimer, G. (1995). Data structures for traveling salesmen. *Journal of Algorithms*, 18(3) : 432–479.
- Gavish, B. and Graves, S. C. (1978). The travelling salesman problem and related problems. Working Paper OR-078-78, Operations Research Center, Massachusetts Institute of Technology.
- Gendreau, M., Potvin, J.-Y., et al. (2010). Handbook of Metaheuristics, second edn. Springer New York, NY.
- Glover, F. and Laguna, M. (1998). Tabu search. In Pardalos, P. and Du, D. Z. (Eds.), *Handbook of Combinatorial Optimization*. Springer New York, NY, pp. 2093–2229.
- Gottfried, B. S. and Weisman, J. (1973). Introduction to Optimization Theory, first edn. Prentice-Hall, NJ, USA.

- Hansen, P., Mladenović, N., and Taillard, E. (1998). Heuristic solution of the multi-source Weber problem as a p-median problem. *Operations Research Letters*, 22(2-3) : 55–62.
- Hansen, P., Peeters, D., Richard, D., and Thisse, J.-F. (1985). The minisum and minimax location problems revisited. *Operations Research*, 33(6) : 1251–1265.
- Hansen, P., Perreux, J., and Thisse, J.-F. (1981). On the location of an obnoxious facility. *Sistemi Urbani*, 3 : 299–317.
- Hernández-Pérez, H. and Salazar-González, J.-J. (2022). A branch-and-cut algorithm for the split-demand one-commodity pickup-and-delivery travelling salesman problem. *European Journal of Operational Research*, 297(2) : 467–483.
- Hooe, H. H. and Stützle, T. (1998). Evaluation Las Vegas—pitfalls and remedies. *Proceedings of the 14th conference on uncertainty in artificial intelligence*, pp. 238–245.
- Irawan, C. A., Salhi, S., Luis, M., and Azizi, N. (2017). The continuous single source location problem with capacity and zone-dependent fixed cost : Models and solution approaches. *European Journal of Operational Research*, 263(1) : 94–107.
- Kirkpatrick, S., Gelatt Jr, C. D., and Vecchi, M. P. (1983). *Optimization by simulated annealing*. *Science*, 220(4598) : 671–680.
- Kloster, K., Moeini, M., Vigo, D., and Wendt, O. (2023). The multiple traveling salesman problem in presence of drone-and robot-supported packet stations. *European Journal of Operational Research*, 305(2) : 630–643.
- Love, R. F., Morris, J. G., and O, W. G. (1988). *Facilities Location : Models and Methods*. North Holland, New York.
- Michelot, C. and Lefebvre, O. (1987). A primal-dual algorithm for the Fermat-Weber problem involving mixed gauges. *Mathematical Programming*, 39(3) : 319–335.
- Mitchell, J. and Papadimitriou, C. (1987). The weighted region problem. *Proceedings of the Third Annual Symposium on Computational Geometry*, pp. 30–38.
- Moradi, E. and Bidkhori, M. (2009). Single facility location problem. *Ch. 3 of Facility Location*, Springer, pp. 37–68.
- Nagy, G. and Salhi, S. (2007). Location-routing : Issues, models and methods. *European Journal of Operational Research*, 177(2) : 649–672.
- Nikolaev, A. G. and Jacobson, S. H. (2010). Simulated annealing. In Gendreau, M. and Potvin, J. Y. (Eds.), *Handbook of Metaheuristics*, Springer New York, NY, pp 1–39.

- Parlar, M. (1994). Single facility location problem with region-dependent distance metrics. *International Journal of Systems Science*, 25(3) : 513–525.
- Planchart, A. and Hurter, Jr, A. P. (1975). An efficient algorithm for the solution of the Weber problem with mixed norms. *SIAM Journal on Control*, 13(3) : 650–665.
- Plastria, F. (1992). GBSSS : The generalized big square small square method for planar single-facility location. *European Journal of Operational Research*, 62(2) : 163–174.
- Plastria, F. (2019). Pasting gauges i : Shortest paths across a hyperplane. *Discrete Applied Mathematics*, 256 : 105–137.
- Prodhon, C. and Prins, C. (2014). A survey of recent research on location-routing problems. *European Journal of Operational Research*, 238(1) : 1–17.
- Ribeiro, C. C. and Rosseti, I. (2015). TTTplots-compare : A perl program to compare time-to-target plots or general runtime distributions of randomized algorithms. *Optimization Letters*, 9 : 601–614.
- Salhi, S. and Nagy, G. (2009). Local improvement in planar facility location using vehicle routing. *Annals of Operations Research*, 167(1) : 287–296.
- Schneider, M. and Drexler, M. (2017). A survey of the standard location-routing problem. *Annals of Operations Research*, 259 : 389–414.
- Schwardt, M. and Dethloff, J. (2005). Solving a continuous location-routing problem by use of a self-organizing map. *International Journal of Physical Distribution & Logistics Management*, 35(6) : 390–408.
- Wang, M., Zhang, C., Bell, M. G., and Miao, L. (2022). A branch-and-price algorithm for location-routing problems with pick-up stations in the last-mile distribution system. *European Journal of Operational Research*, 303(3) : 1258–1276.
- Weiszfeld, E. (1937). Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43 : 355–386.
- Wendell, R. E. and Hurter, J. A. P. (1973). Location theory, dominance, and convexity. *Operations Research*, 21(1) : 314–320.
- Zaferanieh, M., Kakhki, H. T., Brimberg, J., and Wesolowsky, G. O. (2008). A BSSS algorithm for the single facility location problem in two regions with different norms. *European Journal of Operational Research*, 190(1) : p. 79–89.

BIOGRAPHICAL SKETCH

Gökhan ALTAY graduated from Haydarpaşa High School in 2008. He completed his undergraduate studies in the Industrial Engineering Department from Yıldız Technical University in 2012. He gained MSc degree in Industrial Engineering at Boğaziçi University in 2018.

PUBLICATIONS

- Altay, G., Akyüz, M. H. and Öncan, T. (2023). Solving a minisum single facility location problem in three regions with different norms, *Annals of Operations Research* 321 : p. 1–37.