Facility Location Planning Under Disruption

Badr Afify

A Thesis
in
The Concordia Institute
for
Information Systems Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy (Information and Systems Engineering) at
Concordia University
Montréal, Québec, Canada

January 2020

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Badr Afify**

Entitled: **Facility Location Planning Under Disruption**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Information and Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________________ Chair
Dr. Luis Amador-Jimenez

______________________________ External Examiner
Dr. Amin Chaaban

______________________________ External to Program
Dr. Akshay Kumar Rathore

______________________________ Examiner
Dr. Mourad Debabbi

______________________________ Examiner
Dr. Chun Wang

______________________________ Thesis Supervisor
Dr. Anjali Awasthi

Approved by ______________________________
Dr. Dr. Mohammad Mannan, Graduate Program Director

March 24, 2020 ______________________________
Dr. Amir Asif, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Facility Location Planning Under Disruption

Badr Afify, Ph.D.

Concordia University, 2020

Facility Location Problems (FLPs) such as the Uncapacitated Facility Location (UFL) and the Capacitated Facility Location (CFL) along with the k-Shortest Path Problem (k-SPP) are important research problems in managing supply chain networks (SCNs) and related operations. In UFL, there is no limit on the facility serving capacity while in CFL such limit is imposed. FLPs aim to find the best facility locations to meet the customer demands within the available capacity with minimized facility establishment and transportation costs. The objective of the (k-SPP) is to find the k minimal length and partial overlapping paths between two nodes in a transport network graph. In the literature, many approaches are proposed to solve these problems. However, most of these approaches assume totally reliable facilities and do not consider the failure probability of the facilities, which can lead to notably higher cost.

In this thesis, we investigate the reliable uncapacitated facility location (RUFL)and the reliable capacitated facility location (RCFL) problems, and the k-SPP where potential facilities are exposed to disruption then propose corresponding solution approaches to efficiently handle these problems. An evolutionary learning technique is elaborated to solve RUFL. Then, a non-linear integer programming model is introduced for the RCFL along with a solution approach involving the linearization of the model and its use as part of an iterative procedure leveraging CPLEX for facility establishment and customer assignment along with a knapsack implementation aiming at deriving the best facility fortification. In

RUFL and RCFL, we assume heterogeneous disruption with respect to the facilities, each customer is assigned to primary and backup facilities and a fixed fortification budget allows to make a subset of the facilities totally reliable. Finally, we propose a hybrid approach based on graph partitioning and modified Dijkstra algorithm to find k partial overlapping shortest paths between two nodes on a transport network that is exposed to heterogeneous connected node failures. The approaches are illustrated via individual case studies along with corresponding key insights. The performance of each approach is assessed using benchmark results. For the k-SPP, the effect of preferred establishment locations is analyzed with respect to disruption scenarios, failure probability, computation time, transport costs, network size and partitioning parameters.

Acknowledgments

I am so grateful to the people who have touched my life, granted me the highest level of support and contributed towards reaching this goal. Through the tough days of my Ph.D program, keeping the motivation was one of the challenges and with the subtle guidelines from those we trust, I was able to find my own path. I will try to mention those who helped me maintain the goal on sight in one way or another, but memory sometimes can play games with our minds.

In particular, I would like to express my sincere gratitude to my advisor Prof. Anjali Awasthi for the continuous support of my Ph.D study and related research, for her patience, motivation, and immense knowledge. Her kind words, scientific curiosity, knowledge, and vast experience have transformed these years into a true learning and life-changing journey. Her guidelines were essential and her guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Mourad Debbabi, Prof. Chun Wang, and Dr. Akshay Kumar Rathore, not only for their insightful comments and encouragement, but also for the hard question which enticed me to widen my research from various perspectives. I am also grateful to all my friends who helped and supported me during my Ph.D study. Thank you Dr. Andrei Soeanu and thank you Dr. Sujoy Ray for your generosity and support.

Moreover, I would like to extend my deepest gratitude to my family, especially my wife

Ahlam Fouda, who deserved my constant gratitude at every step I took as I was standing on their shoulders to reach higher and higher. In times like these, written words are the only means available to show appreciation. I am especially grateful to my parents, Safia Abdulhakim and Mahmoud Selim. Although they both passed away on May 17, 1999 and July 31, 2019 respectively, I still feel their souls have always been there to support me. Through own life example, they have been the inspiration behind my dreams and set my ethical guidelines. Their teachings go beyond any academic program and their endless demonstrations of affection are certainly one of my main motivations in life.

I also wish to thank my brothers and sisters for their support. And finally, to all the people that I have not mentioned by name but who have shown interest in this work and motivated me to get to this point: thank you!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

As stated in [53], "supply chain network (SCN) is defined as a system of facilities, people, activities, information and resources involved in moving a product or service from supplier to customer". From strategic perspective, a SCN should be designed to minimize the cost where most of the cost is determined by the location of facilities and the transport of products. SCN design is also referred to as network modelling due to the fact that a mathematical model can be created to optimize the SCN design [91]. From operation perspective, a SCN should be designed to ensure the efficient flow of products from suppliers to customers.

Facility disruptions can have negative impact on the SCN flow and consequently lead to economic and market-share loss due to customers dissatisfaction as a result of the delay or termination of the deliveries. In the past, at the planning stage, SCN planners assume continuous availability and unlimited capacities of established facilities to guarantee continuous and sustainable flow of the materials over the SCN with no expected delay or termination of the delivery services. However, in reality the SCN infrastructure are often affected by various internal and external disruption factors which may render such network completely or partially disrupted. The aim of this thesis is to explore how such plans can be formed to develop reliable SCN that work efficiently in the normal conditions and minimize the impact of the disruption.

1.2 Motivation

Supply facilities may fail due to internal factors (e.g. poor inventory management, technical issues, work accidents, etc.) as well as external factors (e.g. power outage, road blockages, natural disasters, etc.). Thus, unexpected disruptions can have significant negative impact on the network reliability including its economic activities and lead to higher transportation costs, order delays, inventory shortages and loss of market share.

In [24], the authors investigate a network involving 2610 U.S. domestic airports and 64,204 flight connections. The study mentions that because of a complete disruption of Ted Stevens Anchorage international airport, the average span of flights has changed from 3.192 to 3.566 over the U.S. domestic airport network which is corresponding to an overall 12% extra stops over all flight traversals. In [100], the authors document that the collapse of I-35W bridge over the Mississippi river interrupted the usual routes of about 140,000 daily vehicle trips and caused an additional overall daily loss of $400,000 for travellers and commercial vehicles due to re-routing. In [18], the authors study the impact of Nisqually earthquake in 2001 on SeaTac international airport. The authors mention that the airport operated at reduced capacity for three months during repairs and the major cargo airport in the region suffered from a damage in the runway that led to a significant impact for both the passengers and the cargo delivery.

In [60], the authors state that the Hanshin-Awaji Earthquake caused a direct repair cost of $5.5 billion for the port of Kobe along with an estimated indirect economic loss of $6 billion. In [63], the authors state that the disruption of a flu vaccine manufacturer in Bristol, UK in 2004 resulted in disastrous consequences since the U.K. government stopped production when U.S. regulators inspected a manufacturing plant and found evidence of bacterial contamination problems. This event reduced the U.S.'s supply of the vaccine by nearly 50% during the 2004-2005 flu season. In [49], the authors document that Hurricanes Katrina and Rita in 2005 on the U.S. Gulf Coast destroyed facilities at almost all levels of the supply chain causing a reduction of the production facilities capacity to the minimum level. In [48], the author studies the economic effects of Japan's 2011 Earthquake and Tsunami. The event crippled Japan's nuclear industry since 11 of Japan's 50 nuclear reactors were immediately closed after the disaster and as a result reducing the country's electricity generation by 40%.

According to the aforementioned events, unpredictable and unhandeled disruptions might have serious and negative impact on SCN activities. In this dissertation we focus on eliminating the impact of such disruption on SCN by investigating the following problems:

- Uncapacitated Facility Location (UFL) problem, in which we consider that the SCN facilities are subject to failure. This includes allocating one layer of backup facilities for each customer while fortifying subset of the established facilities within a limited budget. The fortification ensures continuous availability of facilities that are exposed to disruption. The objective of the model is to minimize the total facility establishment and transportation costs for customers to their primary and backup facilities.

- Capacitated Facility Location (CFL) problem, in which we consider that the SCN facilities are subject to failure. Also, we assign each customer to one primary facility and another totally reliable backup facility. Finally, a limited budget is be used to fortify a subset of the established facilities to ensure that the facilities will maintain its capacity in case of disruption. The objective of the model is to minimize the total facility establishment and transportation costs for the customers to their primary and backup facilities.

- *k*-Shortest Path Problem (*k*-SPP), in which we consider partial and complete disruption in facilities over a large size transport network. This includes partitioning the network into p partitions, investigate the effect of facility disruption and find *k* partial overlapping paths from a source node to a destination node such that the generated paths length are minimum.

1.3 Problems Overview

In this thesis, we address the FLP which is considered a strategic decision problem and the *k* which is considered an operational decision problem. More precisely, we investigate the designing of SCN that are exposed to the risk of complete or partial disruption in facilities. We also investigate the *k*-SPP where a subset of nodes (and consequently their connected links) are exposed to the risk of complete or partial disruption. UFL, CFL as well as *k*-SPP are designed over a graph of interconnected nodes. In both UFL and CFL, we consider a subset of nodes to represent the available locations to establish facilities and another subset of nodes to represent customers of known demands that will be assigned to the established facilities. Finally, the set of edges are representing transportation links of known costs.

In UFL, by definition it is assumed that there is no capacity limit on the serving facilities while in CFL the serving facilities have capacity limitations. The objective of UFL and CFL is to establish a number of facilities to serve the customer demands such that the optimal solution minimizes the facility establishment and the transportation costs.

The objective of the shortest path problem is to find a path with minimal length from source to a destination nodes while in the *k*-shortest path problem the objective is to find *k* partial overlapping paths when a subset of the established facilities on the network are disrupted completely or partially.

1.4 Objectives

The aim of this thesis is to design models for the UFL, RCFL and the k-SPP by taking into consideration the existence of complete or partial disruption on the established facilities. We introduce a non-linear integer programming formulation for UFL and CFL and propose solution approaches for both problems. We also propose hybrid approach based on graph partitioning and modified Dijkstra algorithm for the k-shortest path problem. Thus, the objectives of this dissertation are as follows:

- *Modeling the Uncapacitated Facility Location Problem Under Disruption*: Our first objective is to model the uncapacitated facility location problem under disruption, understand problem requirements, complexity, and propose a solution approach.
- *Modeling the Capacitated Facility Location Problem Under Disruption*: Our second objective is to model the capacitated facility location problem under disruption, understand problem requirements, complexity, and propose a solution approach.
- *Modeling Partitioning Based partial Overlapping k-Shortest Path Problem Under Disruption*: Our third objective is to model the shortest path problem under disruption, understand the problem requirements, assumptions, complexity and propose a solution approach.
- *Designing efficient algorithms for the above three mentioned problems*: We develop various techniques and algorithms to solve the problems.
- *Algorithms performance illustration*: We illustrate the proposed approaches using case studies to show the performance of such approaches.
- *Verification and validation of algorithms*: We compare the proposed approaches with respect to existing similar approaches in the literature via experimental results.

1.5 Research Methodology

UFL and CFL are considered as strategic decision problems, in other words they are long-term focused decision problems for SCN. As such, for both of these problems, it is quite important to design models that are taking into account the potential disruption and its impact on the resulting cost.

k-SPP is considered an operational decision for SCN, in other words it is a short-term focused decision problem (day to day activities), and is affected by the SCN design since in some situation it is important to reach a specific site location from different routes in a cost effective manner (time or distance).

In Chapter 3, we address the UFL problem where we consider that the SCN facilities may experience complete disruption, leading the customers to travel to backup facilities. This is reflected as a transportation cost increase since each customer is assigned to primary and backup facilities. We propose a fast evolutionary learning heuristic solution technique to generate near-optimal solutions for reliable p-median problem (RPMP) and RUFL instances. More specifically, we address the uncapacitated facility location problem by combining learning with evolutionary boosting technique. RPMP and RUFL have been addressed before using Lagrangian relaxation (LR) approach in [54]. The difference between our approach and the LR approach is our case, we obtain in a comparatively short time complete solutions in terms of facility establishment, fortification budget allocation and customer assignments while the LR approach takes longer time and provides upper and lower solution cost bounds.

In Chapter 4, we address the CFL where we consider that the SCN facilities are subject to partial disruption which renders such facilities lose portion of its capacity and in this case the customers fulfil their demands from a backup facilities and this will add extra transportation costs to the customers that go to such facilities. We propose a separation-linearization algorithm to generate optimal solutions for the RCFL instances. We address the capacitated facility location problem by separating the initial model into two separate sub-models (location allocation and fortification sub-models). More specifically, we use CPLEX solver for the location allocation sub-model while ue develop an implementation for the knapsack problem to generate the optimal facility fortification strategy. Finally, the approach is flexible enough to hedge against different disruption situations using different fortification budget amount and the ranking (importance) of facilities. In other words, the approach allows partial fortification.

In RPMP, RUFL and RCFL models, the facility fortification budget is defined as a cost that is used to improve the reliability of the established facilities.

Finally, in Chapter 5, we address the *k*-SPP where we consider partial and complete disruption in the connected nodes over a large size transport network (TN). In case of partial disruption, a subset of the connected nodes fail partially and consequently extra cost will be added to their connected links. As a result, all the generated paths will be of higher costs comparing to the normal scenario. In the case of complete disruption, a subset of the connected nodes fail completely and consequently they became unavailable and their connected links as well. Consequently this would lead to the generation of paths with higher costs compared to the normal (undisrupted) situation. To solve the *k*-SPP under disruption, we propose hybrid approach based on graph partitioning and modified Dijkstra algorithm. The approach enable partitioning the TN sub-networks and allows faster solution generation via reducing the query time when computing the shortest path between two nodes for a very large TN. The approach can generate a maximum predefined number of shortest paths between any pairs of nodes on a transportation network unless the destination cluster is isolated.

1.6 Document Organization

The thesis is organized as follows. Chapter 2 discusses the literature related to our problems, namely UFL, CFL and k-SPP and details the gap analysis. Chapter 3 presents the uncapacitated facility location problem under disruption (RPMP and RUFL), details the solution approach, and provides a case study and benchmarks. Chapter 4 presents the capacitated facility location problem under disruption (RCFL) along with the separation-linearization solution generation. It also provides a case study and benchmarks. Chapter 5 details the k-shortest path problem under disruption and presents the solution approach along with a case study and benchmarks. Finally, Chapter 6 concludes the thesis.

Table 1: Abbreviations

Term	Abbreviation
Facility Location Problem	FLP
P-Median Problem	PMP
Uncapacitated Facility Location	UFL
Capacitated Facility Location	CFL
Reliable Uncapacitated Facility Location	RUFL
Reliable P-Median Problem	RPMP
Reliable Capacitated Facility Location	RCFL
k-Shortext Path Problem	*k*-SPP
Lagrangian Relaxation	LR
Branch and Bound	B & B
Tabu Search	TS
Genetic Algorithm	GA
Continuum Approximation	CA
Supply Chain Network	SCN
Public Service Facility Network	PSFN
Reliable Facility Location-Network Design Problem	RFLNDP
Reliable Uncapacitated Facility Location with Service Levels	RUFL-SL
Integer Programming	IP
Integer Linear Programming	ILP
Mixed Integer Non-Linear Programming	MINLP
Mixed Integer Programming	MIP
Transportation Network	TN
Facility Failure	FF
Non-Linear Integer Programming	NIP
Demand Failure	DF
Link Failure	LF
Iterative Metaheuristic	IM
Variable Neighbourhood Search	VN
Capacitated Arc Routing Problems	CARP

Capacitated Warehouse Location Model with Risk Pooling	CLMRP
Benders Decomposition	BD
ϵ-constraint technique	ECT
Capacitated Location-Routing Problem	CLRP
Green CLRP	G-CLRP
Planar Location-Allocation Problem	PLAP
Neighbourhood Search	NS
Multi-Period Capacitated Flow Refuelling Location Problem	MCFRLP
Electric vehicles	EVs
Cross Decomposition	CD
Distribution Centers	DCs
Capacitated Facility Location and Network Design Problem with Multi-type of Links	CFLNDML
Fix-And-Optimize	FAO
Stochastic Energy-Efficient Facility Location Allocation	SEEFLA
Stochastic Simulation	STS
Scatter Search	SCS
Single Source Multiple Target	SSMT
Reliable Shortest Path Problem	RSPP
Constrained Shortest Path Tour Problem	CSPTP
Dijkstra algorithm	DA
Transit Node Routing	TNR
Density-based Clustering for Shortest Path Deliveries	DenCluSPD
k reliable shortest paths	KRSP
Road Network	RN
Markov Decision Process	MDP
Approximate Dynamic Programming	ADP
Analytical Hierarchical Processing	AHP
Ant Colony Optimization	ACO
Single Source Shortest Path Problem	SSSPP
Single Source Single Destination	SSSD
Shortest Path Routing Problem	SPRP

Wireless Sensor Network	WSN
Shortest Path Routing	SPR
Dynamic Navigation Algorithm	DNA
Spatially Dependent Reliable Shortest Path Problem	SD-RSPP
Simulation Based Genetic Algorithm	SBGA
k Shortest Path With Diversity	KSPD
Replacement Paths Algorithm	RPA
Realtime Shortest Path	RSP
Realtime Shortest Path Plus	R-SP+
Recursive Enumeration Algorithm	REA
Dijkstra-Ant Colony	DACO
Particle Swarm Optimization	PSO
Simulation Based Genetic Algorithm	SBGA
R-Interdiction Median Problem with Fortification	S-RIMF
Fortification	fortif.

Chapter 2

State-of-the-Art

2.1 Introduction

Supply chain management and logistics planning involving facility location and shortest path are important problems extensively studied across the scientific communities all around the world. The computation complexity and practical relevance of commodity delivery problems have attracted the researchers for more than half a century. The planning of logistics activities is a particular type of operational plan that ensures the supply of resources in a timely manner at the right locations. In the chapter, we address two types of FLP including UFL, CFL problems as well ask-shortest path problem. Often time, UFL and CFL problems are formulated as non-linear integer programming and we find several approaches that are proposed in the literature to solve such problems. However, the majority of the proposed solution approaches assume that the established facilities are totally reliable.

2.2 Uncapacitated Facility Location Problem (UFL)

A majority of the FLP employ a graph to represent the underlying network in order to model the problem. The objective function of a typical FLP is to minimize total cost of facility establishment and transport from the serving facilities to the demand (customer) nodes. The objective functions of these models aim to improve path coverage as described by [39], minimize establishment and transportation costs as discussed by [32], reduce failure probability as introduced by [56], etc. [31] documented a detailed classification of these location problems. A brief taxonomy of the facility location models including the median and plant location models, and the center and covering models can be found in [72]. [50] provide a critical survey of the SCN design problem under uncertainty and the related proposed optimization models. Moreover, [83] provide a survey of supply chain disruptions models.

The Uncapacitated Facility Location Problem is a subcategory of FLP which imposes no limitation on the serving facility capacity. In the literature, the Uncapacitated FLP in classified in two different categories, namely the p-median problem and the uncapacitated Facility Location(UFL), based on the cost and the number of established facilities. In p-median problem there is no cost of establishing facilities but a maximum bound (p) is placed on the number of established facilities. Thus, the objective of the problem is to reduce overall transportation cost only. However, in UFL problem there is a cost of establishing facilities and there is no maximum bound placed on the number of established facilities so each facility has an associated establishment cost. Thus, the objective of the problem is to reduce overall establishment cost and the overall transportation cost. The UFL problems are handled using greedy algorithms, branch and bound schemes, dual descent heuristic, and other programming techniques as discussed by [45]. [3] present a genetic algorithm (GA) and a scatter search (SCS) approaches to solve p-median problem where some facilities may not be operative during certain periods.

Various approximation algorithms for solving such problems have been surveyed by [32] and [81]. The common solution generation techniques for p-median problems include greedy methods, branch and bound heuristic, primal-dual heuristic, node partitioning and substitution, meta-heuristic techniques, etc. as detailed by [90]. [28] introduce the concept of facility disruptions based on two mathematical models. The first model captures a variant of the p-median problem with known facility failure probability. The second model captures a (p; q)-center problem in which p facilities must be located to minimize the maximum cost when at most q facilities fail. He used heuristic algorithm for the solution of both problems. More recent articles show growing academic interest on the reliability-oriented optimization in distribution networks as well as characterizing different sources of disruptions.

[25] study the RUFL problem and presented a model that optimally determines the facility locations as well as the customer assignments in order to minimize initial setup costs and expected transportation costs in both normal and failure scenarios. They solved mid-sized instances using Lagrangian relaxation (LR) algorithm and for large-scale problems they used continuum approximation (CA). Under independent and location specific facility failure probabilities, [59] analyze an uncapacitated fixed-charge facility location model with two types of facilities: unreliable and reliable. In this formulation, each customer will be assigned to one primary facility and one totally reliable backup facility. The objective of this model is to optimally determine the number and location of both types of facilities and the customer assignment. They developed a Lagrangian relaxation-based solution algorithm to solve the model. [7] formulate a non-linear integer programming model to investigate SCN design problem where some facilities are exposed to random disruptions and as a result may fail to serve the customers. They develop a genetic-algorithm-based solution approach to solve the model.

[67] develope a facility location interdiction model to design a coverage-type service

network that is robust to the worst instances of long-term facility loss. They presented a decomposition algorithm to solve the bilevel program optimally. [57] present the Stochastic R-Interdiction Median Problem with Fortification(S-RIMF). In S-RIMF model, the authors aim to minimize the impact of disruption by allocating defensive resources among facilities. They used heuristic reduction rules to solve the model. [45] study the supply chain network design problem that is exposed to the risk of partial and complete facility disruption by considering different disruption scenarios. They develop two solution methods based on Lagrangian relaxation (LR) and genetic algorithm (GA) to solve the model. [82] develop a reliable facility location-network design problem (RFLNDP) which is suitable to hedge against the impact of facility disruptions by hardening selected facilities while taking into account facility location, link construction, and transportation costs. They linearize the model then solve it using CPLEX.

To handle the sources of uncertainty associated with demand and service, [95] model the congested situations in the system within a queuing framework and propose a reliable location-allocation model where the facilities are subject to the risk of disruptions. The authors enhance a Benders decomposition algorithm using two efficient accelerating methods including valid inequalities and knapsack inequalities to obtain exact solution of the proposed model. [76] compare two mathematical formulations for the reliable uncapacitated facility location problem (RUFL)in which correlated facility disruptions is modelled. The study compared two variant of RUFL models. The first model reduces unsatisfied service by allowing the decision maker to price unsatisfied demand using a penalty cost. The second model with service levels (RUFL-SL) in which a minimum level of satisfied demand for each scenario is guaranteed. The authors use LINGO 17.0 to solve RUFL and RUFL-SL models to optimality.

With the increasing attention given to environmentalism, [99] develop a 0-1 mixed integer bi-objective programming mode for a green closed-loop supply chain network design.

To generate lower bounds efficiently, the authors implement an improved reformulation-linearization technique based on decomposed piecewise McCormick envelopes.

Table 2 covers a number of relevant existing efforts to handle uncertainty over the Uncapacitated FLP.

Table 2: Survey of previous effort on uncertainty over Uncapacitated FLP

Authors	Paper Topic	Uncertainty	Application	Category	Model	Approach
[62]	Max. the profit by facility setup & production allocation	Demand, services & sales	SCN	PMP - limited budget	2-step SP	Dual-based procedure
[28]	Min. weighted distance from customers to facilities	FF; independent	Emergency services	PMP, (p,q)-Cent er Problem	MINLP	Heuristic algo.
[29]	Optimal locations to establish new facilities	Demand change	Facility establishment	PMP-Time dependent demand	Non-convex & ILP	Standard mathematical programming
[13]	Capture market expansion and cannibalization effects	Demand change	SCN	FLP;Spatial IMP Worst-case bound	IP	Specialized B&B and Greedy heuristics
[3]	Min. total transportation cost by finding p facility locations that are inexpensive & reliable	FF	SCN	PMP	LIP	GA-SCS
[84]	Min. the weighted sum of operating cost	FF (indep.)	TN	RPMP & fix.-charge RUFL	MINLP	LR
[14]	Min. average distance from customers to facilities	FF; independent	Hospital localization	RPMP	MINLP	Error-bound Greedy Heuristic and B&B
[25]	Min. facility setup & transport costs	FF;dependent, Random	SCN	Fix. charge RUFL	MILP	Customized LR & continuum approx.
[59]	Min. facility setup, fortification & customer conn. costs	FF; independent	SCN	RUFL & UFL facility hardening	MIP	LR
[57]	Allocate resource to min. worst-case impact of disruption	FF; independent, Random	TN	Stochastic RIMP; Fortif. budget	BLMIP	Heuristic reduction rules.
[55]	Min. facility setup & customer transportation costs	FF; Spatially Correlated	Sensor deploy :surveillance	Fix. charge RUFL	MINLP	Continuum Approximation
[81]	Customers reassignment	FF; Constant, independent	TN	RUFL	2-step SP & MINLP	Approximation algo.
[8]	Min. facility setup, DCs labour working, DCs safety stock, transportation and lost sale costs	FF	SCN	PMP	NIP	GA
[67]	Evaluate max. & min. coverage by p facilities w/o disruption	FF; independent	TN	Loc. covering-Interdiction prob.	MIP & BLMIP	Bi-level decomposition algo.

[23]	Min. facility setup, inventory holding and delivery costs	FF; Scenario	Plan & transport structure	RUFL	MINLP	Custom LR & Poly.-time algo.
[56]	Maximize flow & path coverage	Sensor failure	Sensor Deploy	RUFL	ILP	Custom Greedy & LR
[85]	Min. the weighted distance from customers to facilities	DF & LF	TN	UFLP - limited budget	IP	Decomposition algo.
[45]	Optimal facility location & customers assignments	FF; independent, Scenario	SCN	Integrated SC design models	MINLP	LR & GA
[49]	Generate extreme events & characterize impact	FF & LF; indep., Scenario	SCN	Risk model with multi-hazard	3-phase hazards	Monte-Carlo simulation
[82]	Opt. facility location, customer assignment & facility hardening costs	FF; independent	Integrated facility & SC design	RUFL (facility & link)	MINLP	Linearization
[54]	Min. facility setup, fortification & customer connection costs	FF; independent	TN	RPMP & RUFL - fortif. budget	MINLP	LR
[51]	Max. the expected net revenues	Sales & transport costs	SCN (1-Echelon)	SMLTP	Binary IP	LR, C&W savings, TS & heur.
[8]	Min. facility setup, inventory, transportation & lost sales costs	FF	SCN	Stochastic FLP	NIP	GA
[95]	Min. the fixed setup, transportation, serving & penalty costs	FF	SCN design	RUFL	MIP	BD
[76]	Min. the sum of fixed and transportation costs	FF (Correlated)	SCN	RUFL & RUFL-SL	NLIP	LINGO 17.0
[99]	Min. the total costs and CO2 emissions	Demand and CO2 emission rate	SCN	RUFL	0-1 MI bi-objective programming	Reformulation-Linearization technique

2.3 Capacitated Facility Location Problem (CFL)

Facility Location Problem (FLP) usually employs a graph to represent the underlying transport network in order to model the problem where the objective is to minimize total cost of facility establishment and transport from facilities to demand nodes. Based on the capacity of the serving facilities, FLPs are classified in the literature into capacitated and uncapacitated. In contrast with the uncapacitated FLP, the capacitated FLP impose a limitation on the capacity of the serving facilities. Since capacitated FLPs have more realistic assumptions compared to the uncapacitated FLPs, there is a growing academic interest on considering the capacitated version of FLP under different sources of disruptions. [35] developed a mathematical model for the discrete capacitated FLP where the established facilities are unreliable (unavailable) as a result of disruptions and use the sample average approximation algorithm to solve it. [58] develop a mathematical model for the capacitated FLP with limited budget to fortify a subset of the established facilities. The goal of the model is to optimally protect the network against correlated facility disruptions as well as to optimize protection plans to hedge against large area disruptions. The authors employ adapted tree search (ATS) algorithm to find out which facilities to be protected. [11] propose a model to design supply chain network (SCN) consisting of capacitated production facilities, distribution centers and retailers. The authors assume that the supply chain network is under uncertainties from demand-side and supply-side.

[4] propose a set of two-stage robust optimization models to design reliable p-median facility location networks under the risk of disruption and also include practical cases: facility capacities and demand losses due to disruptions. [69] study the logistics network design problems (LNDP) with facility disruptions and propose a mixed-integer programming model to minimize the nominal cost and reduce the disruption risk using p-robustness criterion. To solve the model, they develop a hybrid metaheuristic algorithm based on genetic algorithm (GA), local improvement (LI) and the shortest augmenting path (SAP) method to solve the model. [86] propose an approach for designing and planning multi-echelon, multi-commodity production distribution networks under deterministic demands. They consider that the decisions related to facilities, supplier and the production flows are dynamic. They also consider that a facility is functioning only between a minimum and

maximum rate of utilization of its installed capacity. To solve the model they use Xpress-MP (MILP solver) based on the branch-and-bound (B&B) approach. A detailed review of FLP including the capacitated version and an overview of the major location problems that have been introduced in the literature can be found in [88].

[44] investigate the single source location problem with the presence of several possible capacities in which the opening fixed cost facility depend on the used capacity and the area where the facility is located. The authors formulate two mathematical models for both the discrete and the continuous cases using the rectilinear and Euclidean distances. To solve the models, they propose two methods, namely an iterative metaheuristic (IM) approach and variable neighbourhood search (VNS) based metaheuristic technique. [92] study the waste collection problem in Denmark which motivated them to develop a fast heuristic solution approach for the large-scale capacitated arc routing problems (CARP) with or without duration constraints. [68] study the capacitated warehouse location model with risk pooling (CLMRP) in which a single plant ships one type of product to a set of retailers, each with an uncertain demand. They formulate the problem as a non-linear integer program and propose a Lagrangian relaxation (LR) solution algorithm to solve it with reasonable computational requirements.

To ensure the desired level of reliability and availability for supply chain facilities, [65] investigate a supply chain problem design that is subject to random failures and consider a stochastic SCN design model that handles facility location and facility unavailability management simultaneously. To solve the model, they use GA based optimization approach to define the optimal supply chain structure and perform two simulation strategies to manage the facilities failure, one by replacing each unavailable facility by the closest facility and the other is to execute reallocation process using GA. [71] propose a mathematical formulation for the capacitated facility location problem under uncertainty and they assume that the uncertainty appears in the demands and costs. They develop an efficient heuristic solution algorithm based on the VNS to solve the problem. [80] introduce a mixed non-linear integer programming model for the distance-constrained mobile hierarchical facility location problem in order to minimize the total system costs. The authors develop a GA based heuristic approach to solve the model.

[34] propose a computational study for the capacitated FLP to address the usefulness of the Benders Decomposition (BD) approach. The study focuses on the particular case where the application of BD yields a single subproblem of the same size as the original problem. To minimize the fuel consumption, they propose a mixed integer linear bi-objective model for the Capacitated Location-Routing Problem (CLRP), named Green CLRP (G-CLRP). The authors solve the proposed mathematical model using the classical Pareto optimisation ϵ-constraint technique (ECT).

[77] present a linear mixed integer formulation for the generalized version of the reliable capacitated facility location problem (RCFLP) under correlated facility disruptions with uncertain joint distribution. The model guarantees a minimum level of service in terms of satisfied demands. They optimally solve the model for several penalty cost values using LINGO 17.0. A survey about deterministic FLP, dynamic FLP, stochastic FLP, and robust FLP can be found in [15]. To handle the sources of uncertainty associated with demand and service, [95] model the congested situations in the system within a queuing framework and propose a reliable location-allocation model where the facilities are subject to the risk of disruptions. The model is solved using an enhanced BD algorithm employing two accelerating methods including valid inequalities and knapsack inequalities.

[76] compare two mathematical formulations for the reliable uncapacitated facility location (RUFL) problem where a correlated facility disruption is modelled. The first model reduces unsatisfied service by allowing the decision makers to price unsatisfied demand using a penalty cost. The second model employs service levels (RUFL-SL) where a minimum

level of satisfied demand for each scenario is guaranteed. The authors use LINGO 17.0 to optimality solve RUFL and RUFL-SL models. [73] introduce a mixed integer non-linear programming model for the RCFLP in which certain facilities are exposed to disruption. The authors develop two heuristic procedures to solve the problem. [6] propose a model for reliable distribution centers (DCs) in case of unexpected disruption including the site-dependent failure and random link disruptions. The authors use three solution approaches to solve the proposed model, including LR, cross decomposition (CD) and a firefly algorithm.

To hedge against random disruptions and capacity limitation of the distribution centers (DCs), [5] use the concept of reliable & unreliable facility and propose a two-stage integer programming model for the Reliable Facility Location Distribution Network (RFLDN) in which the authors consider the location decisions for opening DCs in the first stage while assigning customers to either a reliable or an unreliable facility in the second stage. The authors develop LR approach to solve the model.

To solve the proposed model, the authors develop a fix-and-optimize (FAO) heuristic based on the firefly algorithm. [46] present a new stochastic energy-efficient facility location allocation (SEEFLA) model subject to carbon emission, economical, capacitated, and regional constraints. To solve the model, the authors employ an algorithm that integrates stochastic simulation (STS) and scatter search (SCS). [16] present a mathematical model to support location decisions oriented for the Public Service Facility Network (PSFN). The model aims to minimize the total cost needed to provide remaining facilities with additional service capacities to satisfy demand resulting from the reallocation of users previously assigned to service points that have been closed. [99] develop a 0-1 mixed integer bi-objective programming mode for a green closed-loop supply chain network design. To generate lower bounds efficiently, the authors implement a linearization technique.

The work introduced [58] hedge against disruption by identifying the critical portions of the system whose fortification reduces the impact of the worst possible disruption while our work hedges against disruption by assigning each customer to a primary and a backup facility. The works of [35], [71] and [73] introduce penalty costs for the unmet demands from the disrupted facilities while in our work the customers obtain the portion of unmet demand from the backup facility when the primary facility partially fails. The work introduced in [5] has similar assumptions like our work but this work hedge against disruption by adding an emergency cost for supplying from the reliable facilities in the case of disruption. Thus, the customers obtain emergency services from the backup facility when the primary facility fails while in our work we hedge against disruption in the sense that the customers obtain the portion of unmet demand from the backup facility when the primary facility partially fails. The work of [77] use an effectiveness metric to guarantee a minimum level of satisfied demand to avoid the penalty cost for unmet demand in order to ensure an acceptable service level under disruption while in our work we use a limited budget to fortify a subset of the established facility so that they become available and each customer is assigned to a primary and a backup facility.

Table 3 covers a number of relevant existing efforts to handle uncertainty over the Capacitated FLP.

Table 3: Survey of previous effort on uncertainty over Capacitated FLP

Authors	Paper Topic	Uncertainty	Application	Category	Model	Approach
[30]	Min. cost of holding inventory and demand loss	Demand change	Prod., Capacity Planning	Prod.,Capacity plan (time horizon)	LP, SP & MIP	Specialized algo.
[35]	Optimal facility locations for various disruptions	FF; independent	SCN	Cap. FLP	2-step SP	Sampling of average approximation
[58]	Protecting capacitated median system with limited resources	FF(Correlated)	TN	Cap. RIMP; Fortif. budget	3-step, MIP	Adapted tree-search
[61]	Min. nominal cost & cost bounds under disruption	FF; Scenario	SCN	Cap. P-robust Logistics Net. Design	MIP	GA based Hybrid metaheuristic
[11]	Max. SCN expected cost effectiveness	FF(Demand) & LF(indep.)	Multi-product SCN	Cap. FLP & flow detection	MINLP	Piece-wise linear approx. (Commer. S/W)

[4]	Min. weighted distance from customers to facilities	FF; Simult., Scenario	DN	Cap. RPMP & RUFL	MINLP	Benders decomposition; col.-constraints
[92]	Min. the overall cost including fixed cost of using the depots & vehicles, the dead-heading travel and the total penalty costs	Time	RN	CARP	ILP	Branch & Cut, Clustering based heuristic
[68]	Min. the sum of facility setup, transportation, and inventory carrying costs	Demand	SCN	Cap. FLP	NIP	LR
[65]	Min. the sum of the following costs: facility setup, shipment plus transportation, the total inventory and the DC holding safety stock costs.	FF	SCN	Cap. FLP	NIP	GA
[71]	Opt. location for p number of Cap.facilities in such a way that minimizes the total expected costs of transportation, construction, and penalty of uncovered demands	Demand & Cost	SCN	Cap. PMP	MINLP	NS
[80]	Find the optimal number and locations of launch and recharge stations to minimizing the total establishment, drone procurement and drone usage costs	Demand	Telecomm.	Cap. PMP	MINLP	GA
[77]	Min. total facility setup & transportation costs	FF (Correlated)	SCN	Cap. RUFL	MILP	LINGO 17.0
[73]	Min. total investment and operational costs	FF	SCN	Cap. RUFL	MINLP	Relax & Fix and Relax & Round Heuristics
[6]	Min. Total Facility setup, transportation and improvising costs	FF (Dep.)-LF	SCN	Cap. RUFL	MILP	LR, Cross Decomposition & Firefly
[5]	Min. the facility setup, the deterministic service cost for customers, the expected service cost for customers, and the holding costs.	FF-site-dependent	DN	RFLDN	Two-Stage IP	LR - CPLEX solver
[95]	Min. the fixed setup cost as well as expected transportation, serving & penalty costs	FF	SCN design	RUFL	MIP	BD

[76]	Min. the sum of fixed and transportation costs	FF (Correlated)	SCN	RUFL & RUFL-SL	NLIP	LINGO 17.0
[46]	Min. the total transportation energy consumption of customers	FF	Energy-eficient FLA	Cap. FLP	NLIP	Stochastic Simulation & Scatter Search
[16]	Min. the total cost needed to provide remaining facilities with aditional service capacities	Facility Capacity	Public Services Network	Cap. PSFN	ILP	CPLEX solver
[99]	Min. the total costs and CO2 emissions.	Demand and CO2 emission rate	SCN	RUFL	0-1 MI bi-objective programming	Reformulation-Linearization technique

2.4 Shortest Path Problem

The classical shortest path problem is one of the fundamental problems in graph theory in which the objective is to find a path that minimize the length from a source node to a destination node. In the last few decades, a variety of approach have been introduced to solve such problem. In [27], the author develops a single source multiple target (SSMT) algorithm for the shortest path problem. The authors in [94] propose a pre-computation based approach for both single pair alternative shortest path and all pairs shortest paths processing in spatial network. To handle the variability of travel time, [97] present an algorithm to solve the mean-standard deviation reliable shortest path problem (RSPP) in which they take into account the correlations between the link travel time. [33] propose a mathematical model for the constrained shortest path tour problem (CSPTP) in which the path must cross by a sequence of node in a given order must. The author introduces a Branch & Bound method to solve the model.

[41] formulates a non-linear objective functions for the stochastic shortest path problem using the mean and standard deviation values of the resulting probability distribution and general cost functions. To solve the model, the author applies Lagrangian relaxation (LR) to

find the feasible solution. Considering the travel time uncertainty, [21] extend the classical k-loopless shortest path problem to the stochastic transportation network and introduce the k reliable shortest paths (KRSP). To solve the problem in large-scale networks exactly, the authors propose a deviation path algorithm and to further improve the KRSP finding performance they introduce the $A\star$ technique.

The authors in [22] study the travel time uncertainties in a road network(RN)where the goal is to find the most reliable path that maximizes the probability of on-time arrival. They propose a two-stage solution algorithm that exactly solve the problem. The authors in [79] use the historical information and the real-time information of a transportation network and present the dynamic shortest path problem with stochastic disruptions as a discrete time finite horizon Markov decision process (MDP). They develop a hybrid approach based on approximate dynamic programming (ADP) and clustering using deterministic lookahead policy and value function approximation. In [2], the author provides an experimental analysis on how traversal algorithms behave on social networks. Dijkstra algorithm has been used in [89] to generate single source single destination (SSSD) shortest path in large sparse graphs where the goal is to reduce the response time for online queries. [1] present ASAP system that can extremely quickly generate the shortest path for all pair of nodes on large scale networks.

The dynamic single source shortest path problem (SSSPP) has been addressed in [37] via dynamizing Dijkstra algorithm and generate the solution of that problem with complexity O(nlg m) for the update time. To reach an emergency area in a minimum time, [64] develop a dynamic emergency routing approach based on Dijkstra algorithm and analytical hierarchical processing (AHP). For safe evacuation in a short time from an emergency area, [75] model the evacuation route then they generate the directed graph based on the distance between nodes and coordinate of nodes. To generate the shortest and safest path, they use Dijkstra algorithm and Dijkstra-ACO.

[9] optimize Dijkstra algorithm using a faster queue design based on binary (or d-ary) heap implementation to accelerate the computation of the shortest paths. [43] introduce a new time-dependent shortest path algorithm for SSSD in multimodal transportation network.

Considering the stochastic disruption in a network, the authors in [47] present an emergency evacuation model for the SSP and they propose a real-time dynamic navigation algorithm (DNA) to solve it. [42] study a network that is stochastic on-time arrival and propose a modification for the reliable routing algorithm. The authors modify the shortest path algorithm to decrease its computation time by choosing particular subset of graph nodes and links. Taking into account the travellers concern on travel time reliability in congested RN, [20] formulate the spatially dependent reliable shortest path problem (SD-RSPP) on a RN of correlated link travel times. The authors propose a link-based multi-criteria A* algorithm (ERSPA* algorithm) to solve the model. To find the optimal path in an uncertain transportation network, [19] formulate three model for the shortest path problem typically name expected value, dependent-chance and chance constrained models. The authors develop a simulation based genetic algorithm (SBGA) to solve the former models.

[36] derives the uncertainty distribution of the shortest path length in an uncertain network and investigates solutions to the α-shortest path. The author indicates that there is an equivalence relation between the α-shortest path in an uncertain network and the shortest path in a corresponding deterministic network and as so proposes an approach to find the α-shortest path and the equivalence deterministic shortest path. An approach based on GA for the SPRP has been presented in [12]. The authors in [70] indicate that most of the existing approaches that handle the shortest path problem have two main drawbacks: generating the path before movement and re-processing when node and/or link fails. To handle these shortcomings, they propose two novel algorithms typically name Realtime Shortest Path (RSP) and Realtime Shortest Path Plus (R-SP+).

The authors in [98] present a robust shortest path model to hedge against the risk of uncertainty in travel times. To solve the model, they propose an efficient primal approximation method that has to solve the deterministic shortest path problem and the mean-standard deviation shortest path problem. [96] analyze the dynamic stochastic characteristics and the relationships between the links and nodes of the transportation network. The authors define the probabilistic shortest path concept, propose a mathematical model for the dynamic stochastic KSP and develop a GA to solve the model. A survey on variations of the shortest path problem of different objective functions and existing solution approaches can be found in [87]. Table 4 covers a number of relevant existing efforts to handle the shortest path problem and k-shortest path problem.

2.5 Gap Analysis

In the previous Sections 2.2,2.3 and 2.4, we summarize a number of relevant existing efforts to handle the uncapacitated FLP, capacitated FLP, and shortest path problem under disruption (uncertainty) over SCN. Section 2.2 and Section 2.3 cover a number of relevant existing efforts to handle the uncapacitated FLP and capacitated FLP. The majority of these works consider only complete disruption of facilities with equal probability, which is not a very realistic assumption. Also, they ignore partial facility disruptions in the case where facilities fail partially, which is a more realistic assumption. Moreover, a minority of these works consider partial fortification depending on a ranking of the facilities. Furthermore, most of them do not address correlated facility failure probabilities (site dependence and spatial correlation) since it can trigger the failure of other facilities. Finally, most of the works do not consider the impact of SCN disruption on economic conditions and population especially when the SCN facilites are located on a site of a population that is highly correlated to the SC activities.

Section 2.4 covers a number of relevant existing efforts to handle the shortest path problem and k-shortest path problem. The majority of these works do not take into account the failure probabilities in the connected nodes partially or completely and their consequences on the connected links. Also, the majority of them do not handle the partial overlapping k-SPP. Moreover, most of these works do not consider the case when the correlated failures of the connected nodes. Furthermore, a minority of these models handle k-shortest path problem in multi-modal or global networks.

In the next three chapters of this thesis and based on the foregoing explanation, we are addressing the following gaps:

- Partial and Complete disruption in SCN facilities, where they are subject to heterogeneous failure probabilities,
- Partial and Complete disruption in TN connected nodes, where they are subject to heterogeneous failure probabilities
- Partial fortification of SCN facilities, depending on a ranking of the facilities, and
- Partial overlapping k-SPP.

we are going to address the foregoing gaps as follows:

- Presenting a non-linear integer programming model for RPMP, RUFL problems as well as developing an evolutionary learning heuristic as solution generation approach.
- Capturing the RCFL problem using a non-linear integer programming model and developing a separation-linearization-based solution approach for MILP solver based solution generation.
- Modelling the partial overlapping k-SPP under partial and complete disruption in the TN connected nodes as well as developing a clustering-based solution approach.

Table 4: Survey of previous effort on SPP and k-SPP

Authors	Paper Topic	Uncertainty	Application	Category	Approach
[79]	Shortest path pairs	Link (stochastic)	TN	Shortest Path	Heuristic algoritm
[37]	Dynamic single source shortest Path	Link	TN	SPP	Dynamized DA
[64]	Enhanced routing in disaster management	Link	TN	SPP	Enhanced DA
[75]	Emergency SP	Nodes and Links	High rise building	SPP	DA & DACO
[43]	Generate a virtual path from single source to single target	Travel time	Multimodal TN		Heuristic Algorithm
[47]	Find the shortest path in emergency area	Links (Stochastic)	TN	SPP	DNA
[42]	Decrease the computation time of the KSPP	Nodes & Links (Stochastic)	TN	KSPP	Modified reliable routing algorithm
[20]	Find spatially dependent reliable shortest path in RN	Links (Correlated)	RN	SD-RSPP	ERSPA* algorithm
[19]	Find the optimal path under an uncertain environment	Links (Stochastic)	TN	SPP	simulation-based genetic algorithm
[36]	Find the shortest path in uncertain network	Links	RN	SPP	DA
[12]	Finding the shortest path in variable structure network	Links	Randomly generated network topologies	SPP	GA
[70]	Find the best alternative route	Node/Link	RN	SPP	RSP & R-SP+
[98]	Find the hortest path in uncertain network	Random travel times and their distributions	TN	SPP	PAM
[96]	Find the k shortest transport paths in dynamic stochastic networks	Node/Link	TN	Dynamic stochastic KSP	GA
[41]	Introduce an efficient method to handle previous limitations and improve the processing speed further for the SPP	VLSI circuit design		Constrained SPP	LR

Chapter 3

Uncapacitated Facility Location Planning Under Disruption

3.1 Introduction

The p-median problem (pmp) represents a subclass of facility location problem where a fixed p number of facilities are required to be established and the objective function involves minimizing total transportation cost. In UFL, each node has an associated facility establishment cost and the decision makers are required to establish an optimal number of facilities to minimize overall sum of establishment cost and transportation cost. Thus, in general, facility location problems assume guaranteed availability of facilities to serve customer demands. Conversely, reliable p-median problem (RPMP) and reliable uncapacitated facility location problem (RUFL) consider a predefined probability of failure for each node. Therefore, in this context, every customer is required to be associated with multiple facilities (typically a primary and a backup) to ensure availability of facilities under disruptions.

3.2 Model Assumptions

The following assumptions are typically considered to define RPMP and RUFL:

- Facilities have unlimited capacity (as in [59], [54], [25] and [4]).
- If a facility fails, it becomes totally unavailable.
- Facility failures are independent (as in [59], [54], [25] and [4]).
- Each customer is assigned to a primary facility and a backup facility (as in [54]) unless the primary facility is fortified (i.e. totally reliable) in which case it also serves as a backup for the same customer.
- Probability of a simultaneous failure of primary and backup facilities is negligible for each customer.
- The fortification budget is fixed (an input to the problem).

3.3 Model Formulation

We present next the specific details and formulation of each of these two problems.

3.3.1 Reliable p-Median Problem (RPMP)

In RPMP, each member node of a network of interconnected nodes is exposed to a predefined failure probability. The problem aims to select p locations to serve the customers such that the total transportation cost to satisfy customer demands, by assigning each customer to one primary and one backup facilities, is minimal. Additionally, the problem assumes a fixed budget available to fortify few selected facilities to increase network reliability. RPMP model is considered as detailed by [54].

The formulation considers a set of customers I, a set of potential facility locations J, and a predefined number p of facilities to be established. The set of potential facilities is a subset all customers, $J \subseteq I$. However, they can also represent a different set. Each customer $i \in I$ has a specific demand h_i. Let $d_{ij} > 0$ be the transportation cost of one unit of demand from facility location $j \in J$ to customer $i \in I$. For each facility j there exists a failure probability q_j where $0 \leq q_j \leq 1$ and a fortification cost fc_j which consists of a setup cost s_j and a reliability cost $q_j \times rc_j$. rc_j denotes the cost related to unit reduction in the failure probability. Total fortification budget is B.

The decision variables used in the formulation are as follows:

$$x_j = \begin{cases} 1, & \text{If a facility is established at location } j \\ 0, & \text{otherwise.} \end{cases}$$

$$z_j = \begin{cases} 1, & \text{If the facility at location } j \text{ is fortified} \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{ij0} = \begin{cases} 1, & \text{If customer } i \text{ is served by primary facility } j \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{ij1} = \begin{cases} 1, & \text{If customer } i \text{ is served by backup facility } j \\ 0, & \text{otherwise.} \end{cases}$$

The objective function for optimization is described as follows:

$$Min \sum_{i \in I} \sum_{j \in J} \left[h_i d_{ij} y_{ij0} (1 - q_j(1 - z_j)) + h_i d_{ij} y_{ij1} \sum_{r \in J, r \neq j} q_r y_{ir0} (1 - z_r) \right] \tag{1}$$

Subject to:

$$\sum_{j \in J} y_{ij0} = 1 \ \ \forall i \in I \tag{2}$$

$$\sum_{j \in J} y_{ij1} = 1 \ \ \forall i \in I \tag{3}$$

$$y_{ij0} + y_{ij1} \leq x_j \forall i \in I, j \in J \tag{4}$$

$$\sum_{j \in J} x_j = p \tag{5}$$

$$z_j \leq x_j \tag{6}$$

$$\sum_{j \in J} (s_j + q_j \times rc_j) z_j \leq B \tag{7}$$

$$z_j, x_j \in \{0, 1\} \forall j \in J \tag{8}$$

$$y_{ij0}, y_{ij1} \in \{0, 1\} \forall i \in I, j \in J \tag{9}$$

The objective function (1) of RPMP model is to minimize the total transportation cost for satisfying all customer demands. The term $\sum_{j \in J} \left[h_i d_{ij} y_{ij0} (1 - q_j (1 - z_j)) \right]$ is the expected transportation cost between customer i and its primary facility, where $(1 - q_j (1 - z_j))$ is the probability that the primary facility is available. The term $\sum_{j \in J} \left[h_i d_{ij} y_{ij1} \sum_{r \in J, r \neq j} q_r y_{ir0} (1 - z_r) \right]$ is the expected transportation cost between customer i and its backup facility, where the term $\sum_{r \in J, r \neq j} q_r y_{ir0} (1 - z_r)$ is the probability that the primary facility failed (in this case, the backup facility is assumed to be available). Constraints (2) and (3) ensure that each customer will be assigned only to one primary and one backup facility. Constraint (4) guarantees that the facility must be established before serving any customer. Constraint (5) ensures that only p facilities are opened. We add Constraint (6) to the formulation as a valid inequality ensuring that a facility must be established before being fortified. Constraint (7) is the total fortification budget constraint. Finally, both (8) and (9) represent integrality constraints.

3.3.2 Reliable Uncapacitated Facility Location (RUFL)

We present briefly the RUFL formulation as detailed by [54]. The RUFL problem has similar definition and assumptions as RPMP but the restriction on the total number of opened facilities is dropped. However, In RUFL we have an additional establishment cost f_j required to establish a new facility at node j. The goal of RUFL is to minimize the total facility establishment cost and the total transportation cost for the customer to its primary and backup facilities. Thus, in contrast to RPMP, RUFL has two different terms that will affect the objective function (10): the total number (not predetermined) of opened facilities and the transportation cost. The formulation is similar to that of RPMP in terms of decision variables and constraints, except constraint (5) which is dropped in RUFL formulation. The objective function is as follows:

$$Min \sum_{j\in J} f_j x_j + \sum_{i\in I}\sum_{j\in J}\left[h_i d_{ij} y_{ij0}(1-q_j(1-z_j)) + h_i d_{ij} y_{ij1}\sum_{r\in J, r\neq J} q_r y_{ir0}(1-z_r)\right] \quad (10)$$

The objective function (10) in RUFL model minimizes the total facility establishment cost and the total transportation cost associated with satisfying all customer demands. The term $\sum_{j\in J} f_j x_j$ represents total facility establishment cost while the remainder of the objective function (10) is similar to equation (1).

3.4 Evolutionary Boosting Technique

We propose a fast heuristic solution generation technique to reach near-optimal solutions for RPMP and RUFL instances. As defined, a solution to any of these two problems corresponds to a particular assignment for the decision variables which satisfy the constraints. Each such assignment represents a point in the solution search space and each of these solutions has a corresponding cost. The objective functions of RPMP and RUFL refer to a solution or a set of solutions that have the same minimum cost. In RPMP, this cost involves the total transportation cost whereas in RUFL, it involves the summation of all establishment and transportation costs. Our approach iteratively probes or "navigates" various regions of the solution search space in order to reach a near optimal solution progressively. The search procedure is based on evolutionary learning whereby the characteristics of probed regions help to "navigate" towards near-optimality.

We address uncapacitated facility location problem by combining learning with evolutionary boosting technique. The latter was introduced in [52] to find near-optimal solutions based on a statistical model. The optimal solution generation in combinatorial optimization often renders the solution search procedure intractable for large networks. In contrast, evolutionary boosting constructs a computationally tractable procedure for searching the solution search space. The associated search involves repetitive generations of more competitive solutions through an elitist selection. More precisely, in each iteration, we employ a heuristic technique to generate a predefined number (SZ) of solutions for the problem instance and add them to a solution pool. The boosting mechanism applies a periodic reinforced learning over this evolving population pool. The outcome progressively segregates sub-optimal solutions from potential near-optimal solutions. For an uncapacitated facility location problem instance, the approach relies on learning solution quality (as identified by the cost) from successively generated solutions to improve decision making on segregating suitable customer nodes from possibly unsuitable ones to find appropriate facility locations.

3.4.1 Assumptions

In the current set of competitive solution samples generated during the multi-round evolutionary learning technique, there will be solutions with highest and lowest costs. With each iteration, we aim to search a better lowest cost solution and select a new solution with reduced highest cost using elitist selection. This allows to progressively reach a set of acceptable near optimal solutions by learning from elitist samples. The proposed procedure converges to a set of near-optimal solutions by progressively increasing confidence on determining certain nodes for facility establishment while segregating other nodes as unsuitable for facility establishment. This also produces more and more competitive solutions closer to the lowest cost solution in the selected pool.

We assume that, in a solution, the total number of customer nodes ($|N|$) is significantly higher compared to the facilities (p). Typically, we consider $\frac{p}{|N|} < 0.5$. This makes an inappropriate attribution of a node as a potential facility less likely, especially in the earlier iterations of the evolutionary procedure. Another important assumption in this respect is that the lowest cost solution of the combined solution pool exhibits monotonic decrease in terms of solution cost after every iteration with a potential to reach a set of near optimal solutions.

3.4.2 Evolutionary Learning and Solution Pool Handling

Evolutionary learning is related to the boosting technique as presented in [17]. Boosting composes a series of weak rules/learners into a strong learner which is generally used for classification purposes as discussed in [78]. In this setting, let $x^t = \{x_1^t, x_2^t, \ldots, x_n^t\}$ be a vector where x_i^t is a binary variable which indicates whether node i is suitable to establish a facility at iteration t. Simply, $x_i^t = 0$ means that node i is deemed unsuitable to establish a facility at iteration t. In classification, we call x^t as feature vector. Actually, x_i^t denotes multiple instances for decision variable x_i at each iteration. Now, let there also be a set of explored solutions each of which is denoted as s_u. In s_u, $s_{iu} \in \{0, 1\}$ denotes whether a facility is established on customer i in an observed solution s_u. Then, the boosting can be captured through an additive model:

$$H^{\tau}(s_u) = H^{\tau-1}(s_u) + \alpha^{\tau T} h^{\tau}(s_u) = \sum_{t=1}^{\tau} \alpha^{tT} h^{t}(s_u) \tag{11}$$

where H^{τ} is a boosted classifier generated from τ weak hypotheses identified by h^t from t=1 to τ. At an iteration t, hypothesis h^t is incorporated with weight vector α^{tT} (T denotes transpose) to the classifier H^{t-1}. Hypothesis h^t focuses on assessing solutions that are not well classified by H^{t-1} generated at previous iteration. In research literature, the value of weight α^{tT} and error minimization objective of hypothesis h^t are specific to the boosting technique.

In our setting, we also need additional means to explore only a small subset of competitive solutions during successive iterations. In classification, we train a classifier using a solution pool and corresponding known classes. If there exist two classes, near-optimal and sub-optimal, a trained classifier tries predicting a class for an unknown solution. Thus, a decision maker can classify a new solution using the classifier. Let o_u be a boolean variable that determines a binary class. If $o_u = true$ the solution is near-optimal whereas $o_u = false$ corresponds to a sub-optimal solution. Thus, the classifier performs like a black box with respect to the feature variables and the solution class.

There are two key challenges in solving RPMP and RUFL. First, since the optimal solution is unknown during search, proper $o_u = true$ label generation is difficult. At every next iteration, a currently labelled near-optimal solution can be found sub-optimal with further exploration of newer low-cost solutions. However, at any iteration, $o_u = false$ can be correctly labelled based on the cost of currently best-known solution and a user defined gap. All solutions with the cost between the values of the currently best-known solution and gap are called current competitive solutions. Second, the solution generation procedure can evaluate only a very small subset of solution samples from the solution search space of medium and large scale problems. So, we require additional means to meaningfully reduce the search space in order to sample progressively more competitive solutions. Evolutionary learning addresses the forenamed challenges. Unlike traditional classification, it uses an interpretable function $f_i(x_i^t)$ to determine the effect of decision making such as considering node i for facility establishment. A link function $\zeta(\dots)$, as introduced in [52], represents the relation between the expectation (E) of o_u and the observed values of the decision variables (x^t) over a training sample s_u, as shown in Eq. (12):

$$\zeta(E(o_u|x^t = s_u)) = \gamma_0 + \sum_{i=1}^{n} f_i(s_{iu}) \tag{12}$$

where s_{iu} denotes the value of variable x_i^t on a solution sample s_u. As such, Eq. (12) is a Generalized Additive Model (GAM) where γ_0 is an intercept. The value of $\gamma_0 + \sum_{i=1}^{n} f_i(s_{iu})$ can be approximately computed based on various independent factors (e.g. location, connections, demand, etc.) of a node in SCN for a given problem instance. We use heuristic mechanism to locally generate the response (which represents a solution). This, in turn, helps in segregating a solution sample s_u, at a particular iteration, with label $o_u = true$ (for iteration t) or $o_u = false$ (for iteration t and all subsequent iterations) by marking potentially near-optimal and sub-optimal solutions. Among the potentially near-optimal solutions, if a feature x_i^t is highly biased to a particular value 0 or 1 across the explored competitive solutions, then $f_i(x_i^t = 0)$ and $f_i(x_i^t = 1)$ contribute dominantly in determining $\zeta(E(o_u|x^t = s_u))$. For example, at iteration t, if all the near-optimal solutions in a pool indicate that $s_{iu} = 0$ then it means that customer i should not be considered to establish a facility. On the other hand, if the potentially near-optimal solutions exhibit a mix of $s_{iu} = 0$ and $s_{iu'} = 1$, then node i's impact on the decision making is less conclusive.

As such, a simple voting procedure over the elite solutions can be adopted, at every iteration, to (re)assign conclusive dominances over the decision variables. As the number of customers is typically larger than the number of facilities, nodes receive votes for having customer-only role. So, the nodes with most votes represent candidates out of which one or more nodes will have fixed role in the next iterations. This reduces the solution search space. The search procedure converges as more and more nodes increasingly retain their role in the next iteration after the exploration of newer potentially near-optimal solutions.

A pseudo-random *explore* function is used to search for competitive solution samples. At an iteration t, a sample is determined competitive if its cost is closer (within a defined gap) to the existing lowest cost solution. A better lowest cost solution (min^t) leads to the removal of all solution samples where the solution cost is greater than $min^t \times (1+gap)$. The *explore* function uses a generator template of weight vectors (w^t) for n nodes as represented by $\left(w_1^t, w_2^t, \cdots, w_j^t, \cdots, w_n^t\right)$. At iteration t, w_j^t denotes the bias of node j to establish a facility. If $w_j^t = 1$, then node j is considered to have no inclination for the establishment of a facility at its location during the solution search at iteration t. However, if $w_j^t = 0$, the inclination of node j cannot be confirmed. We update these weights in each iteration using voting and normalization. At iteration t, for all competitive solutions (S^t), if node j appears with $vote_j^t$ ($vote_j^t \leq |S^t|$) times as customer-only node, then its weight can be defined as:

$$w_j^t = \begin{cases} 1, & \text{If } rank(vote_j^t, j) \leq cl \textbf{ or } \{w_j^{t-\Delta} = 1 \textbf{ and } min^{t-\Delta} \leq min^{t-(\Delta+1)}\}; \\ 0, & \text{If } j \in S^{exp}; \\ \frac{\rho_j}{Z^t} \cdot \frac{vote_j^t}{|S^t|}, & \text{otherwise.} \end{cases}$$

where, $\Delta < t$. $t - \Delta$ refers to a previous iteration. ρ_j is problem specific and designed from problem data (Section 3.4.4 discusses ρ_j in details). Z^t is a normalization factor. The updated weight is set as an input to a pseudo-random *explore* function in order to search solutions according to the current weight vector. In relation to evolutionary learning, the template vector w^t serves for a numeric approximation of $\zeta(E(o_u|x^t = s_u))$. Over τ

successive iterations, this weight vector is trained and updated such that the final template $\left(w_1^\tau, w_2^\tau, \cdots, w_j^\tau, \cdots, w_n^\tau\right)$ serves to accurately determine role of all nodes.

In the aforementioned, each decision variable (x_i^t) is considered independent in the assessment of near-optimal and sub-optimal solutions. The variable independence allows every solution s_u to be considered as a point on a space of n orthogonal axes. In this setting, near-optimal solutions can be seen as a subset of points delimited by a series of cutting planes over the same orthogonal axes. Function f_i helps computing the cutting planes over variables x_i^t. Thus, for each sample s_u, the error in boosting technique can be seen as the difference: $|H^\tau(s_u) - \zeta(E(o_u|x^t = s_u))|$. This error may come from wrongly locating a facility due to sampling limitations in each iteration. Thus, our proposed procedure demands for error mitigation strategies in the design. Furthermore, a threshold ($MaxCl$) is used in the algorithm to decide a maximum number of nodes that can be declared as customer-only (unsuitable for facility establishment) in each iteration.

3.4.3 Error Mitigation

At each iteration, a decision is made over previously undecided customer nodes whether a subset of them can be excluded as potential candidates to establish a facility. In this respect, error comes from wrong decision making at any iteration. We employ three error mitigation strategies to handle such errors:

- *Confidence Adjustment*: A $rank(\dots)$ function is developed to strictly order undecided customer nodes based on the last explored competitive solutions. The smaller rank of node i indicates increased presence of node i as customer node in the last explored competitive solutions. In each iteration, new customer nodes are excluded as possible locations to establish facilities if their ranks are less than a variable cl ($cl \leq MaxCl$). If the exclusion is appropriate, there is a statistical increase in the likelihood of finding better solutions, if they exist, with more similarity to current elitist solutions and lower cost. As such, we expect to identify more competitive solutions in the next exploration. This potentially improves the lowest cost bound as well. However, if the lowest cost solution of the current iteration exceeds the previous lowest cost solution, we assume that the last decision making is incorrect. Then, the technique backtracks to the previous decision. It also reduces the value of cl to $max(1, \lfloor \frac{cl}{2} \rfloor)$, Therefore, in the next iteration, it performs an alternative assignment whereby a smaller number of additional nodes are excluded from establishing facilities at their locations. If the current lowest cost solution still exceeds the previous one at the limit of $cl = 1$, we place the respective node in an exception set S^{exp}. Nodes in S^{exp} are not considered for exclusion in subsequent iterations.

- *Node Swap*: At the end of each iteration, we perform role swaps (between customer-only nodes and customer nodes where facility establishment was considered for newly derived competitive solutions) in pursuit of improving the lowest cost solution. This can also mitigate the impact of potential error(s) in selecting the locations of facilities. This often allows to obtain the best possible lowest cost solution of the current iteration based on the generated solution samples. During the swapping, if certain customer node(s) present in S^{exp} are considered as customer-only node while producing a competitive solution, we remove the respective node(s) from S^{exp}.

- *Selective Exhaustive Solution Search*: During the process run, more and more customer nodes are excluded from the choice to establish facilities which also reduces the search space significantly. In this respect, whenever the number of all remaining possible combinations is less or equal to the user-defined sample size, an exhaustive search is performed to identify the final solution.

3.4.4 Weight Bias Determination

Sections 3.4.2 and 3.4.3 present a general problem-solving approach using evolutionary learning. To solve RPMP, we use two sets of different inputs to construct ρ_j for each node j. In RPMP, two sets of decision variables x_j and z_j determine one of three roles for each node j: customer-only (C), unfortified facility (U-F) or fortified facility (F-F). For each node j, we determine two values from the problem data to construct ρ_j. The first one, $\varphi 1_j$ corresponds to an independent bias of node j to establish a facility in its location. The second one, $\varphi 2_j$ represents its bias to fortify the facility. These two values affect node's role determination as well as the solution search. The value of $\varphi 1_j$ is influenced by node's demand and transportation cost. We propose assessing this value as a ratio between weighted transportation costs considering j to host a facility against being customer-only. Considering node j hosting a facility, let $K_1 \frac{(D-h_i)h_j}{d_{ij}^2}$ determine the probability of facility j to serve customer i where K_1 is a constant and D is an integer bigger than highest demand of all nodes. Then, the probabilistic travel cost can be computed as $K_1 \frac{(D-h_i)h_j}{d_{ij}^2}(1-q_j)h_i d_{ij} = K_1 \frac{h_i h_j (D-h_i)(1-q_j)}{d_{ij}}$ where $(1-q_j)$ represents the probability of facility j to be available. Similarly, if node j is customer-only, then it needs to be assigned to a primary facility (i) and a backup facility (k) with transportation cost: $K_2 h_j (D-h_j)\left[\frac{(1-q_i)h_i}{d_{ji}} + \frac{q_i h_k}{d_{jk}}\right]$ (K_2 is a constant). We define $\varphi 1_j$ as the following ratio:

$$\varphi 1_j = K \frac{(1-q_j)\left[\sum\limits_{i \in N \setminus \{j\}} \frac{(D-h_i)h_j}{d_{ij}}\right]}{(D-h_j)\left[\sum\limits_{\{i,k\} \subset N \setminus \{j\}} \left[\frac{(1-q_i)h_i}{d_{ji}} + \frac{q_i h_k}{d_{jk}}\right]\right]}$$

where $K = \frac{K_1}{K_2}$. The $\varphi 1_j$ represents a ratio that indicates an approximate potential benefit if node j is chosen to host a facility instead of being customer-only. A comparative lower value of $\varphi 1_j$ decreases the probability of the customer j to host a facility.

On the other hand, $\varphi 2_j$ is influenced by fortification cost (fc_j), available budget (B)

and failure probability (q_j). Node j can be fortified if and only if $fc_j \leq B$. Also, there is no need to fortify if it is always available ($q_j = 0$). Moreover, higher cost of fortification for node j has larger impact on the available budget if node j is fortified. Finally, node j should be fortified if the gain from fortification is high in terms of reducing the solution cost. This gain can be represented in the form of $K' q_j (D - h_i) \sum_{i \in N \setminus \{j\}} \frac{h_i}{d_{ij}}$ which indicates the difference between the probabilistic transportation cost if node j hosts an unfortified facility instead of fortified one. Thus, we define $\varphi 2_j$ as follows:

$$\varphi 2_j = \begin{cases} 0, & \text{If } B < fc_j \\ K'(B - fc_j) q_j \left[\sum\limits_{i \in N \setminus \{j\}} \frac{(D-h_i)h_i}{d_{ij}} \right] \quad , & \text{otherwise.} \end{cases}$$

To address RPMP, we define ρ_j as $\frac{1}{(a \cdot \varphi 1_j + b \cdot \varphi 2_j)}$. Here, a and b are user chosen constants to represent the impact of $\varphi 1_j$ and $\varphi 2_j$ respectively on weight determination for w_j^t. Higher value of ρ_j indicates node's increased bias not to host a facility. $\varphi 2_j$ determines node j's bias for fortification during new solution sample generation. It should be mentioned here that given the template of weight vector, it is still possible to find solution(s) during exploration where the combined facility fortification cost exceeds budget B. Additional penalty is enforced to the solution cost in order to make those solutions non-competitive.

We approach RUFL by devising a procedure that leverages the RPMP solution approach in a way that allows to appropriately determine the number of facilities p that have to be established in order to reach near-optimal solution. In this pursuit, the procedure is assessing the best trade-off between the overall establishment cost (which increases when increasing p) and total transport cost (which decreases when increasing p).

Figure 1 depicts the RUFL solution finding strategy exemplified on the benchmark instance of 100 nodes and budget of 180 (Figure 11 (a)). The key aspect is the case of RUFL is the determination of p (facilities to be established) given the trade-off between increasing

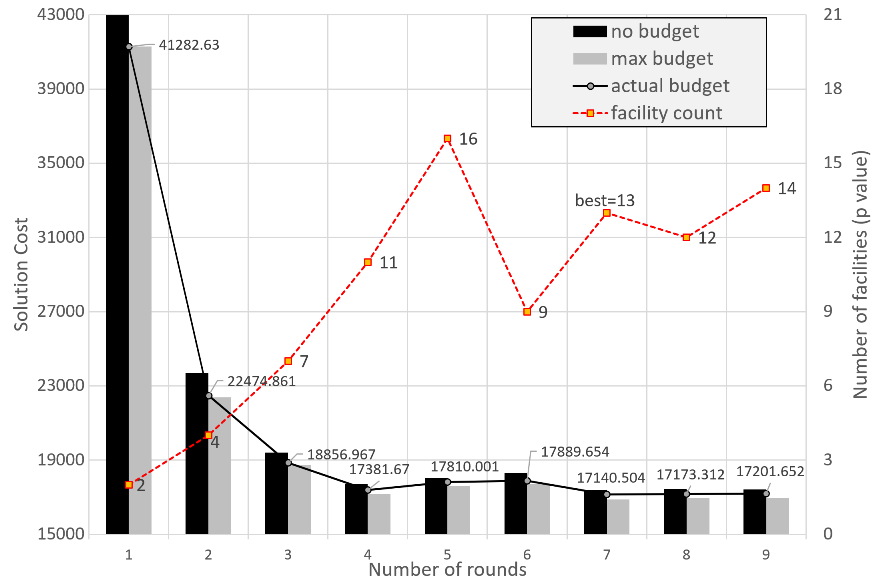


Figure 1: Near-optimal solution finding for a RUFL instance

establishment cost and lowering transportation cost. The procedure involves probing successive p values and the generation of progressively narrow upper and lower bounds until the best p value (yielding the lowest cost) is identified. We illustrate next the corresponding procedure over 9 iterations. The solution costs are shown by two columns and a solid line with respect to the primary axis. Since all cost values are higher than 15000, the primary axis is accordingly bounded. The columns provide the costs obtained with budget 0 (black column) and maximum budget to fortify all facilities (light gray column). The solution cost with budget 180 (black solid line) is in between.

Solving with zero or maximum budget is faster since this allows either total fortification or no fortification at all. Thus, we may effectively use reference estimation from zero and maximum budget solutions (black and light gray) as the procedure involves probing several p values. The corresponding p values are shown through a dashed line graph with respect to the secondary axis. We apply two different series for the p value probing. Initially, we increase p using the following series: $p^{ite+1} = (2 \times p^{ite} - p^{ite-1} + 1)$. This increases p at a higher rate than linear growth but not as much as exponential growth. This allows to avoid unnecessary probing of large p values (which incurs higher computation time). Once a pair of lower and upper bound cost values is determined, we adjust p using binary search.

The probing procedure starts from $p = 2$ and continues in a series expansion of increasing distance between the successive p values as long as the corresponding solution cost is decreasing. Thus, from $p = 2$ to $p = 4$, the distance is 2, then from $p = 4$ to $p = 7$, the distance is 3 and so on until $p = 16$ where the solution cost is increasing to 17810.001 compared to the previous value of $p = 11$ with a cost of 17381.001. Is this case, the procedure switches to a binary search mode and continues the probing with $p = (7 + 11)/2 = 9$. The resulting solution cost 17889.654 is higher than $p = 11$. Thus, the procedure continues probing with $p = \lfloor(11 + 16)/2\rfloor = 13$ which yields a cost of 17140.504. The next probing is then at $p = (11 + 13)/2 = 12$ with a cost of 17173.312 which is higher than 17140.504. Thus, the procedure goes on to probe $p = \lfloor(13 + 16)/2\rfloor = 14$ which gives a cost of 17201.652 which is also higher than 17173.312. The last two p values indicate that the best $p = 13$ since the solution cost values for both $p = 12$ and $p = 14$ are higher than the cost obtained for $p = 13$. Therefore, the procedure stops after probing $p = 14$.

3.4.5 Algorithm Design

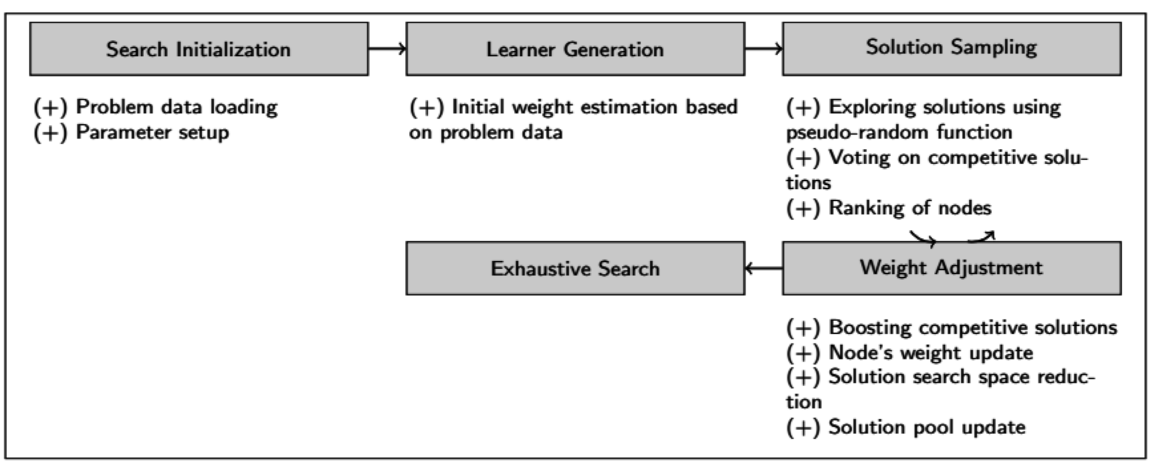


Figure 2: Overview of the Solution Technique

Figure 2 depicts the synopsis of our proposed solution technique. The procedure first loads the problem data and input parameters. The former includes transport network, demand and facility establishment cost (in RUFL) of each node and their probability of failure. The parameters are fortification budget and number of facilities (in RPMP). Then, the initial bias (customer or facility) of the nodes is estimated based on the problem data. The procedure continues by a cycle involving solution sampling and weight adjustment. The solution sampling involves the exploration of solution samples using pseudo random $explore$ function followed by voting on the competitive solutions which allows to generate node ranking. The weight adjustment boosts the competitive solutions leading to progressive solution space reduction and solution pool update. The cycle terminates when the stop condition is satisfied (e.g. maximum number of iterations reached, solution space reduction to levels comparable to the sample count). Once the solution space is sufficiently reduced, an exhaustive search is performed to find the final solution.

The proposed technique relies on successive sampling steps of the solution space whereby the most likely customers are identified with progressively better confidence until the final solution is identified. Each sampling step uses the $explore$ function to generate a defined number (SZ) of solution samples equal to the sample count. However, an elitist selection is performed on the samples such that only the competitive (within a defined gap) samples are retained. The samples are used in a voting process to extract knowledge with respect to the role of the nodes as customers or facilities and subsequently rank the nodes. This allows to segregate customers from facilities such that a progressively larger set of nodes is assumed to represent customers. After the generation of each solution, an improvement is attempted by carrying out customer-facility swaps.Then, we assign votes to the nodes as many times as they appear as customer in the selected set of competitive solutions. Thus, if a node represents a customer in every solution, it receives the maximum number of votes. The received votes allow ranking the nodes in the decreasing order of the votes such that the nodes with more votes (higher rank) represent more likely customers. If the same number of votes are received by more than one node, the nodes are ranked in decreasing order of their demand. The node index is used to assure total ordering if the demands are also same. Thus, for equal votes, the nodes with lower demand are considered as more likely customers. From the candidate customers, cl nodes are assumed as customers based on their ranking from high to low, prior to the next sampling step. Algorithm 1 uses several initial parameters and input variables stated in the beginning. These parameters and variables are subsequently described along with the related logic where they are involved. There is a main loop which can have at most ite iterations. Inside the main loop, we have a secondary loop with SZ iterations whereby the solution sampling takes place for problem instance $\mathcal{P}i$ based on a weighted pseudo-random process. The sampling involves an elitist selection (better than the best $cost$ so far) such that only progressively better samples (out of the maximum SZ) are retained in $\mathcal{S}^{curr}$. Each time a valid solution sample (respecting the budget constraint and the elitist selection) is added to $\mathcal{S}^{curr}$, the corresponding solution value is also used to update the $cost$ variable. After sampling, using $acceptTh \in (0, 1)$ as threshold (typically from 0 to 1%) for allowed tolerance from the current lower bound min^t (needed to handle the case where the best $cost$ so far is nearly missed due to insufficient sampling), the algorithm checks if $\mathcal{S}^{curr}[0]$ (the least cost solution of the current round) is less than or equal to $min^t \times (1+acceptTh)$. If so, all solutions with cost greater than $cost \times (1+gap)$ are discarded (where $gap \in (0, 1)$ represents the allowed cost gap, typically 10% from $cost$). Then, the retained solutions are used to count votes for each of the nodes that appears as client and is not a member of the exception set S^{exp}. Subsequently, these nodes are ranked in a set S' in descending order of their received votes. While not explicitly mentioned in the algorithm, the nodes that receive the same amount of votes can be further ranked using their demand as previously explained. Using the ranking, the algorithm selects from the top at most cl nodes (such that p facilities can still be established) that are not in the exception set S^{exp} and adds them to the set of potential customers S^t. Subsequently, if S^t is empty, the search is pre-emptively terminated. Otherwise, the set of confirmed customers S^{cl} is updated to include S^t; cl is reset to $MaxCl$ (maximum number of customers that can be confirmed in a round) and $\mathcal{S}^{curr}[0]$ is used to update min. Otherwise, if cl is greater than 1, we half it (flooring to the integer part). If the previous condition does not hold, we add $S'[0]$ to the exception set since based on the current sampling it could not be confirmed as a customer and may represents a potential facility. Then, the weights are updated as well as the $cost$ (typically to a value that is slightly higher than the current lower bound). Finally, if the combinatorial search space size (corresponding to the confirmed customers S^{cl}) is less or equal to SZ the algorithm employs an exhaustive search (since it will be equally costly with the sampling procedure but guaranteed to provide the best lower bound) and pre-emptively terminate the main loop.

Algorithm 1 Solution Algorithm ($RPMPSolver$)

Initially: $min \leftarrow \infty$, $cost \leftarrow \infty$, $cl \leftarrow MaxCl$, $S^{exp} \leftarrow \phi$, $\mathcal{S}^{curr} \leftarrow \phi$, $S^{cl} \leftarrow \phi$, $S^{t} \leftarrow \phi$, $vote[0 \dots n-1] \leftarrow \{0, 0 \dots 0\}$, $\begin{pmatrix} w_1^t & \cdots & w_n^t \end{pmatrix} \leftarrow \begin{pmatrix} w_1^0 & \cdots & w_n^0 \end{pmatrix}$;
Input: $\mathcal{P}i$, p, bgt, ite, SZ, gap, $acceptTh$;
Output: S^*;
while $ite > 0$ **do**
  $vote \leftarrow \{0, 0 \dots 0\}$, $ite \leftarrow ite - 1$;
  for $i := 0$ **to** $SZ - 1$ **do**
    explore a new solution S using template $\begin{pmatrix} w_1^t & \cdots & w_n^t \end{pmatrix}$ for $\mathcal{P}i$ (considering $S^{cl} \cup S^t$ as customers) seeking p facilities;
    if $fortifyCost(S) \leq bgt$ **and** $solCost(S) < cost$ **then**
      $\mathcal{S}^{curr} \leftarrow \mathcal{S}^{curr} \cup \{S\}$; $cost \leftarrow solCost(S)$;
    end if
  end for
  if $solCost(\mathcal{S}^{curr}[0]) \leq min \times (1 + acceptTh)$ **then**
    delete each $S \in \mathcal{S}^{curr}$ where $solCost(S) > cost \times (1 + gap)$;
    for $i := 0$ **to** $n - 1$ **do**
      for each $S \in \mathcal{S}^{curr}$ **do**
        if $i \in getClients(S)$ **and** $i \notin S^{exp}$ **then**
          $vote[i] = vote[i] + 1$;
        end if
      end for
    end for
    rank n nodes in descending order of $vote$ and *assign* to $S^{'}$;
    $S^{cl} \leftarrow S^{cl} \cup S^t$, $cl \leftarrow MaxCl$;
    $min \leftarrow$ ***minimumOf***$(min, solCost(\mathcal{S}^{curr}[0]))$;
  else if $cl > 1$ **then**
    $cl \leftarrow \lfloor \frac{cl}{2} \rfloor$;
  else
    $S^{exp} \leftarrow S^{exp} \cup \{S^{'}[0]\}$, $cl \leftarrow MaxCl$;
  end if
  $S^t \leftarrow \phi$, $k \leftarrow n$;
  while $k > 0$ **and** $|S^t| < cl$ **and** $|S^{cl}| + |S^t| \leq n - p$ **and** $|S^{exp}| \leq p$ **do**
    if $S^{'}[n-k] \notin S^{exp}$ **and** $S^{'}[n-k] \notin S^{cl}$ **then**
      $S^t \leftarrow S^t \cup \{S^{'}[n-k]\}$;
    end if
    $k \leftarrow k - 1$;
  end while
  if $|S^t| = 0$ **then**
    break;
  end if
  update $\begin{pmatrix} w_1^t & \dots & w_n^t \end{pmatrix}$ and *cost* to *explore* new solutions;
  if $searchspace(\mathcal{P}i, p, S^{cl}) \leq SZ$ **then**
    run exhaustive solver; break;
  end if
end while
$S^* \leftarrow \mathcal{S}^{curr}[0]$;

3.4.6 Case Study

The case study involves a modified version of a 12 node (p=5) example from [26] as depicted in Figure 3.

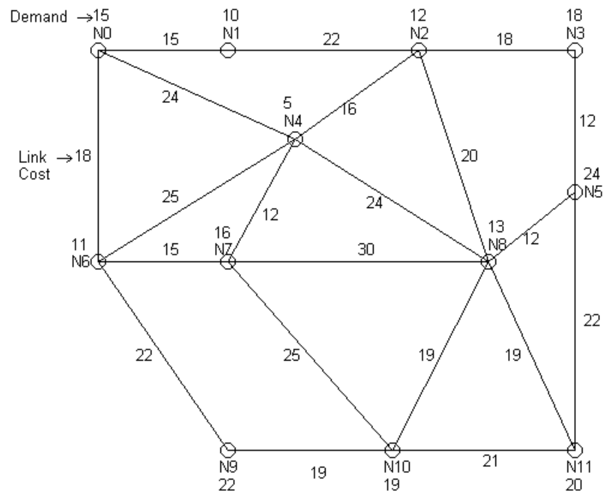


Figure 3: Case Study: A 12 node example p-Median problem

While the cost matrix is the same, the failure probability and fortification cost values have been borrowed from the first 12 nodes of the 30-node benchmark problem. The complete data of the case study problem is presented in Table 5 and in addition, we consider a fortification budget of 180.

Table 5: Case Study: A 12 node example p-Median problem

i/j	N0	N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	Demand	Fail_prob	Fortif_cost
N0	0	15	37	55	24	60	18	33	48	40	58	67	15	0.014	38.69106
N1	15	0	22	40	38	52	33	48	42	55	61	61	10	0.045	50.6559
N2	37	22	0	18	16	30	41	28	20	58	39	39	12	0.015	34.88595
N3	55	40	18	0	34	12	59	46	24	62	43	34	18	0.035	64.265
N4	24	38	16	34	0	36	25	12	24	47	37	43	5	0.026	31.14998
N5	60	52	30	12	36	0	57	42	12	50	31	22	24	0.005	32.73
N6	18	33	41	59	25	57	0	15	45	22	40	61	11	0.042	52.84212
N7	33	48	28	46	12	42	15	0	30	37	25	46	16	0.048	58.95696
N8	48	42	20	24	24	12	45	30	0	38	19	19	13	0.044	61.11592
N9	40	55	58	62	47	50	22	37	38	0	19	40	22	0.017	38.26999
N10	58	61	39	43	37	31	40	25	19	19	0	21	19	0.035	49.8401
N11	67	61	39	34	43	22	61	46	19	40	21	0	20	0.038	37.2979

We detail the solution finding procedure in a stepwise manner by referring to Figure 4. In essence, the procedure involves successive sampling with progressively enhanced ability to segregate the most likely customers until the final solution is identified. The sampling involves generating a SZ number of solutions using weighted randomized facility selection where a progressively larger set of nodes is assumed to represent customers. This rests on the assumption that is more likely to correctly perform an educated guess of a customer node than it is for a facility node since the number of customers is larger than the number of facilities. After each solution generation, an improvement is attempted via customer-facility permutations aiming at lowering the cost. Thus, even though some nodes may be considered as representing customers during the weighted randomized facility selection, the improved solution may contain customer-facility swaps.

We start with an initial sampling step whereby a maximum size SZ solution pool is formed. From this pool, only the solutions with $gap = 10\%$ cost increase or less compared to the current best solution are further considered. Thus, whenever a newly formed solution becomes the current best, all other solutions with cost over the gap are discarded such that the final size of the pool is notably smaller than SZ. Then, we evaluate each solution in the pool and assign votes to the nodes that represent customers. Thus, if a node represents a customer in every solution in the pool, then it receives maximum number of votes. Moreover, if a node is customer only in a subset of solutions, then it receives correspondingly less than maximum votes. Otherwise, if a node never appears as customer, then it receives 0 (no) votes. Consequently, we can rank the customer node candidates in the decreasing order of the votes provided by the solutions in the pool. Thus, the nodes with more votes represent more likely customers. In this setting, the same number of votes may be received by more than one customer. In this case, the nodes can be further ranked in the increasing order of their demand. Thus, nodes with lower demand are considered as more likely customers. This way, a list of potential client candidates is formed from which a maximum of cl nodes are assigned as customers (in their ranking order) before the next sampling step. At the end of each sampling step, the obtained solutions are evaluated against the best solution obtained in the previous sampling steps. If the best solution of the current sampling step is at least as good (same or lower cost) as the previous best solution, then this serves to confirm that the previous client selection was adequate. Conversely, if the best solution of the current sampling step has higher cost than the previous best solution, this indicates an inadequate client selection at the previous step. This leads to backtracking on the previous customer assignment by progressively halving the cl nodes assigned as customers until confirming an adequate client selection or at the limit (in case of only one client candidate) placing the respective candidate in an exception list containing nodes that represent potential facilities. In this respect, the nodes in the exception list receive no votes in subsequent iterations.

iteration, (lowest cost)	nodes(demand)/[votes]												client identification	exceptions
	N0 (15)	N1 (10)	N2 (12)	N3 (18)	N4 (5)	N5 (24)	N6 (11)	N7 (16)	N8 (13)	N9 (22)	N10 (19)	N11 (20)		
1, (1485.681)	0	[9]	6	8	8	1	[9]	1	9	3	5	4	1, 6	
2, (1485.681)	0	—	6	8	8	1	—	1	[9]	3	5	4	1, 6, 8	
3, (1473.876)	0	—	7	[9]	8	1	—	2	—	3	6	4	1, 3, 6, 8	
4, (1473.876)	0	—	7	—	[8]	1	—	2	—	3	6	4	1, 3, 4, 6, 8	
5, (1475.109)†	0	—	[9]	—	—	1	—	2	—	3	8	4	1, 2, 3, 6, 8	4
6, (1473.876)	0	—	—	—	—	1	—	2	—	3	[8]	4	1, 2, 3, 6, 8, 10	4
7, (1473.876)	0	—	—	—	—	1	—	2	—	3	—	[4]	1, 2, 3, 6, 8, 10, 11	4
8, (———)†	0	—	—	—	—	1	—	2	—	[3]	—	—	1, 2, 3, 6, 8, 9, 10	4, 11
9, (———)†	0	—	—	—	—	1	—	[2]	—	—	—	—	1, 2, 3, 6, 7, 8, 10	4, 9, 11
10, (1473.876)	0	—	—	—	—	1	—	—	—	—	—	—	1, 2, 3, 6, 7, 8, 10	4, 9, 11

Figure 4: Solution search trace for Case Study

Figure 4 depicts the successive iterations performed to identify the solution for the case study problem. Since the latter has a small number of nodes and a correspondingly small solution search space, we set $cl = 2$ in order to show in more detail the various steps that can be involved during the generation of the solution. The leftmost column lists the iterations along with the current best solution. The presence of '†' mark at a particular iteration signals that backtracking is performed due to solution quality degradation or lack of any solution within gap. The next columns provide the votes obtained by the candidate client nodes at each iteration. Also, on the top of each node column we can see the node id and its demand in round parentheses. Then, the last two columns list respectively the gradual client list identification and the exception list.

In the first iteration, the initial solution pool is generated and the current best solution has a cost of 1485.68. The majority votes go to nodes 1, 6 and 8, all with 9 votes. For $cl = 2$, only nodes 1 and 6 are selected as customer candidates since node 1 has demand 10, node 6 has demand 11 while node 8 has demand 13. In the second iteration, no improvement is found but the previous best solution appears again. Thus, nodes 1 and 6 are deemed as adequately identified customers. Moreover, in the same iteration, the majority votes go to node 8 (9 votes) which is selected as customer candidate. In the third iteration, a new solution with decreased cost (1473.876) becomes the current best solution and confirms the adequate customer identification of node 8. Also, in the same iteration, the majority votes go to node 3 (9 votes) which is selected as customer candidate. In the fourth iteration, no improvement is found but the previous best solution appears again. Thus, node 3 is deemed as adequately identified among customers. In the same iteration, the majority votes go to node 4 (8 votes) which is selected as customer candidate. In the fifth iteration, the best solution found has increased cost (1475.109) compared to the current best solution. Thus, backtracking is performed and given that node 4 was the only one selected as customer candidate, it goes to the exception list. Furthermore, in the same iteration, the majority votes go to node 2 (9 votes) which is selected as customer candidate. In the sixth iteration, no improvement is found but the previous best solution appears again. Thus, node 2 is deemed as adequately identified among customers. In the same iteration, the majority votes go to node 10 (8 votes) which is selected as customer candidate. In the seventh iteration, no improvement is found but the previous best solution appears again. Thus, node 10 is deemed as adequately identified among customers. Moreover, in the same iteration, the majority votes go to node 11 (4 votes) which is selected as customer candidate. In the eighth iteration, no solution is found within gap such that backtracking is performed and since node 11 was the only one selected as customer candidate, it goes to the exception list. Furthermore, in the same iteration, the majority votes go to node 9 (3 votes) which is selected as customer candidate. In the ninth iteration, no solution is found within gap such that backtracking is performed and since node 9 was the only one selected as customer candidate, it goes to the exception list. Also, in the same iteration, the majority votes go to node 7 (2 votes) which is selected as customer candidate. In the tenth iteration, no improvement is found but the previous best solution appears again. Thus, node 7 is deemed as adequately identified among customers. Moreover, the tenth iteration is also the last since the number of adequately identified clients reaches the maximum value (7) such that the remaining 5 nodes can be only facilities (0,4,5,9 and 11). The final solution with cost of 1473.876 applies fortification on facilities 4, 5, 9 and 11 (*see* Appendix for detailed solution steps).

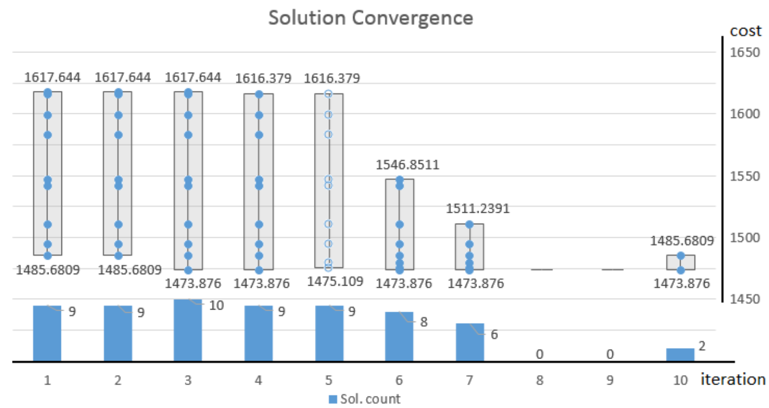


Figure 5: Near-optimal solution finding via evolutionary learning

Figure 5 depicts the execution of the solution generation procedure for the case study. We use a gap of 10% to keep near-optimal solutions (for voting) with respect to the lowest solution cost found at each round. The gray columns in the upper side show the gap between the lowest and highest solution costs as stored in each iteration.

As discussed in Section 3.4, the evolutionary learning helps in reducing the lower bound (as long as no backtracking is involved as in iterations 5, 8 and 9) while improving similarity among the solutions. The blue columns below show the number of solutions stored in each iteration. We note the trend whereby progressively a smaller number of accepted solutions are retained according to the already confirmed customers in each round. The difference between the lowest and highest solutions is also reduced.

3.4.7 Experimental Results

Tables 6 and 7 compare our RPMP solutions obtained by our approach against solutions obtained using the LR-based algorithm published by [54].

Column *Nodes* provides the number of nodes while column *Budget* represents the fortification budget. Column *Facilities* represents the established facilities where an underlined number denotes a fortified facility. Columns *LR:LB* and *LR:UB* provide the lower and respectively the upper bounds provided by the mentioned LR-based algorithm while columns *Best Known* and *Heur. Cost* columns denote the best-known cost (where $*$ indicates optimal value) and respectively the cost obtained by our approach. The *Gap(%)* column indicates the gap of our solution cost with respect to the best known cost calculated as the difference between unity and the ratio of these two costs. Column *Time (s)* presents our computation time in seconds.

Our approach produces complete solutions in terms of established and fortified facilities for each instance. For p=5, we obtain on the 30 and 49 nodes instances 0% Gap with respect to the best known cost. The latter was obtained by means of an exhaustive search for all instance where p=5. Moreover, for 100 and 150 nodes instances, we also obtain in many cases a 0% Gap. For p=8, we also obtain in many cases 0% Gap with respect to the best known cost (given by an exhaustive search for 30 and 49 nodes or the upper bound of the LR-based algorithm for 100 and 150 nodes) and even negative values meaning our solution cost is less than the upper bound of the LR-based algorithm. Actually, for p=8, in all cases of 30, 49, 100 and 150 nodes, we produce better or same results in term of solution cost when compared to the upper bound of the LR-based algorithm. Also, our approach produces these results faster than the LR-based algorithm.

Table 6: Benchmark details on RPMP problem instances for P=5

Nodes	Budget	Facilities	LR:LB	LR:UB	Best Known	Heur. Cost	Gap(%)	Time(s)
30	0	1,2,3,7,21	3694.2	3694.2	3694.2	* 3694.2	0.0	<1
	30	1,2,3,7,21	3694.2	3694.2	3694.2	* 3694.2	0.0	1
	60	1,2,3,7,21	3573.8	3573.8	3573.8	* 3573.8	0.0	1
	120	1,2,3,7,21	3502.5	3502.5	3502.5	* 3502.5	0.0	1
	180	1,2,3,7,21	3366.1	3382.1	3382.1	* 3382.1	0.0	1
	240	1,2,3,7,21	3327.9	3344.4	3344.4	* 3344.4	0.0	1
	300	1,2,3,7,21	3309.7	3309.7	3309.7	* 3309.7	0.0	1
	360	1,2,3,7,21	3283.5	3299.2	3299.2	* 3299.2	0.0	1
49	0	1,3,5,6,11	8826.3	8870.4	8853.2	* 8853.2	0.0	<1
	30	1,3,5,6,11	8826.3	8870.4	8853.2	* 8853.2	0.0	2
	60	1,3,5,6,11	8704.7	8736.3	8736.3	* 8736.3	0.0	2
	120	1,3,5,6,11	8625	8653.5	8653.5	* 8653.5	0.0	3
	180	1,2,3,6,11	8538.4	8538.5	8538.5	* 8538.5	0.0	3
	240	1,2,3,6,11	8417	8459	8459	* 8459	0.0	3
	300	1,3,6,9,11	8360	8401.9	8401.9	* 8401.9	0.0	4
	360	1,3,6,9,11	8325.4	8366.9	8366.9	* 8366.9	0.0	4
100	0	3,35,54,59,89	17 594.4	17682.8	17682.3	* 17682.3	0.0	<1
	30	3,35,54,59,89	17 594.4	17682.8	17682.3	* 17682.3	0.0	4
	60	2,3,35,54,59	17 328.8	17380.3	17380.3	* 17469.6	0.5112	6
	120	2,3,54,83,88	17 086.4	17172.2	17143.3	* 17143.3	0.0	7
	180	1,2,3,59,83	16 977.0	17061.6	17006.9	* 17061.5	0.32	7
	240	1,2,3,59,83	16 897.5	16927.0	16926.9	* 16927.1	0.0012	7
	300	1,2,3,59,83	16 847.2	16847.2	16847.2	* 16847.2	0.0	9
	360	1,2,3,59,83	16 810.0	16886.6	16833.4	* 16833.4	0.0	9
150	0	3,15,59,65,87	20 136.7	20237.0	20214.2	* 20214.2	0.0	1
	30	3,15,59,65,87	20 136.7	20237.0	20214.2	* 20214.2	0.0	9
	60	2,3,15,59,65	19 881.7	19968.5	19968.5	* 19968.5	0.0	9
	120	2,3,15,59,65	19 760.8	19829.1	19808.9	* 19808.9	0.0	20
	180	1,2,3,15,59	19 653.5	19747.2	19669.4	* 19724.4	0.2788	12
	240	1,2,3,15,109	19 494.4	19592.3	19581.9	* 19581.9	0.0	11
	300	1,2,3,15,109	19 451.0	19547.3	19547.2	* 19547.2	0.0	11
	360	1,2,3,8,15	19 512.0	19589.9	19530.9	* 19530.9	0.0	11

Table 7: Benchmark details on RPMP problem instances for P=8

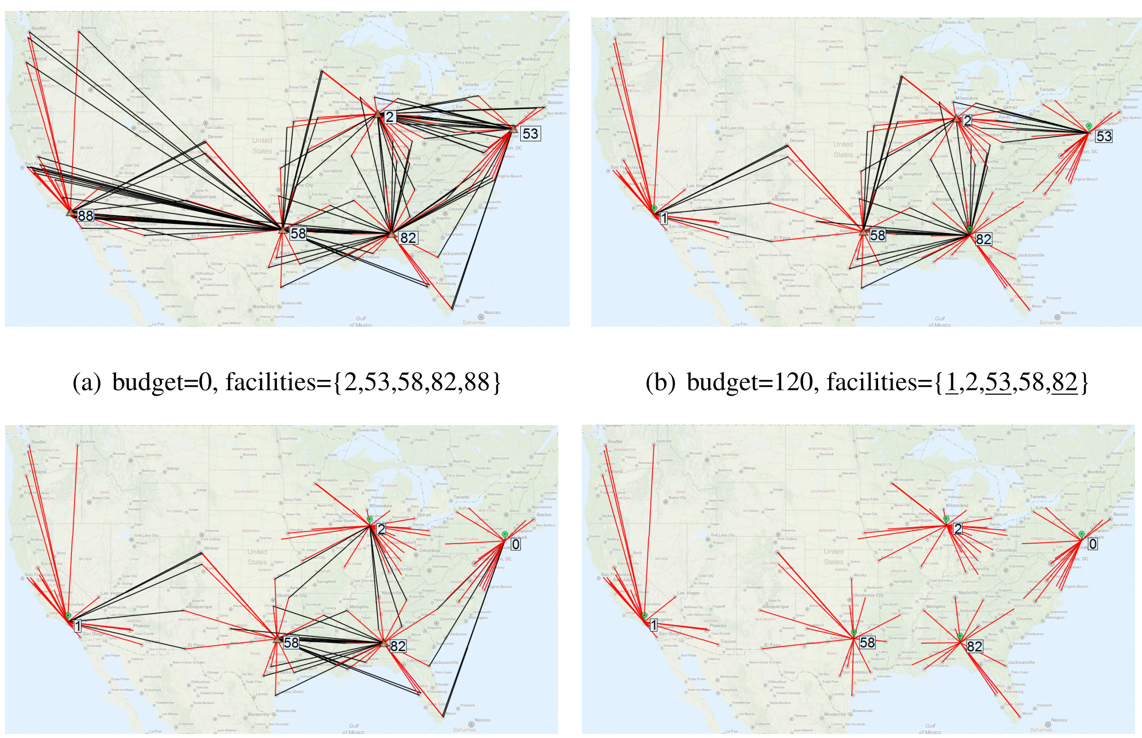
Nodes	Budget	Facilities	LR:LB	LR:UB	Best Known		Heur. Cost	Gap(%)	Time(s)
30	0	1,2,3,9,11,15,16,19	2192.5	2201.0	2200	*	2200	0.0	<1
	30	1,2,3,9,11,15,16,19	2192.5	2201.0	2200	*	2200	0.0	1
	60	1,2,3,9,11,15,16,19	2144.1	2150.1	2150.1	*	2150.1	0.0	1
	120	1,2,3,9,11,15,16,19	2096.8	2102.7	2102.7	*	2102.7	0.0	1
	180	1,2,3,9,11,15,16,19	2044.4	2053.7	2052.8	*	2052.8	0.0	1
	240	1,2,3,9,11,15,16,19	2014.8	2024.7	2024.7	*	2024.7	0.0	2
	300	1,2,3,9,11,15,16,19	1980.5	1990.5	1990.5	*	1990.5	0.0	2
	360	1,2,3,9,11,15,16,19	1981.4	1991.3	1983.3	*	1983.3	0.0	2
49	0	1,2,3,4,6,7,19,26	5874.6	5903.2	5877.6	*	5877.6	0.0	<1
	30	1,2,3,4,6,7,19,26	5874.6	5903.2	5877.6	*	5877.6	0.0	1
	60	1,2,3,4,6,7,19,26	5772.2	5801.1	5777.7	*	5777.7	0.0	1
	120	1,2,3,4,6,7,19,26	5724.6	5752.8	5729.4	*	5729.4	0.0	2
	180	1,2,3,4,6,7,19,26	5678.3	5705.6	5695	*	5695	0.0	2
	240	1,2,3,4,6,7,19,26	5638.9	5647.3	5647.3	*	5647.3	0.0	3
	300	1,2,3,4,6,7,19,26	5625.1	5627.0	5627	*	5627	0.0	4
	360	1,2,3,4,6,7,19,26	5580.9	5608.1	5608.1	*	5608.1	0.0	6
100	0	1,2,3,16,38,51,59,83	12711.3	12775.1	12775.1		12729.8	-0.36	<1
	30	1,2,3,16,38,51,59,83	12711.3	12775.1	12775.1		12729.8	-0.36	4
	60	1,2,3,16,38,51,83,88	12571.7	12593.1	12593.1		12652.2	0.46	5
	120	2,3,16,38,51,54,83,88	12431.5	12491.3	12491.3		12454.4	-0.29	8
	180	1,2,3,16,38,51,83,88	12311.3	12372.8	12372.8		12371.2	-0.01	12
	240	1,2,3,16,38,51,83,88	12283.6	12339.3	12339.3		12311.9	-0.22	21
	300	1,2,3,16,38,51,83,88	12230.6	12252.4	12252.4		12252.4	0.0	27
	360	1,2,3,16,38,51,83,88	12198.9	12198.9	12198.9		12198.9	0.0	35
150	0	2,3,15,16,59,65,71,137	14682.7	14755.9	14755.9		14755.9	0.0	<1
	30	2,3,15,16,59,65,71,137	14682.7	14755.9	14755.9		14755.9	0.0	2
	60	2,3,15,16,59,65,71,137	14659.3	14685.7	14685.7		14650.3	-0.24	3
	120	1,2,3,15,16,71,109,137	14494.1	14566.7	14566.7		14566.6	0.0	5
	180	1,2,3,15,16,71,88,137	14436.3	14508.8	14508.8		14496.9	-0.08	8
	240	1,2,3,15,16,71,88,137	14389.3	14462.3	14462.3		14447.6	-0.1	13
	300	1,2,3,15,16,71,88,137	14385.3	14440.8	14440.8		14413	-0.19	17
	360	1,2,3,15,16,71,88,137	14389.7	14401.5	14401.5		14398.7	-0.29	23

Table 8 compares the results for different RUFL problem instances obtained using our approach against the results published by [54] using LR-based algorithm. For the 30 and 49 nodes instances we obtain the best known cost. The latter was obtained using exhaustive search for the 30 and 49 node instances. In case of 100 and 150 nodes instances our results are showing the same or better cost compared to the best known cost given by the corresponding the upper bound of the LR-based algorithm.

Figure 6 depicts a snapshot of a solution for a 100-node problem instance with five facilities under different fortification budget. Figure 6(a) shows that when the available budget is 0, nodes 2,53,58, 82 and 88 are selected as facilities. However, no facilities are fortified since the budget is 0. Figure 6(b) shows that when the available budget is 120, the selected facilities are 1,2,53,58 and 82. Among these facilities 1,53 and 82 are fortified. Figure 6(c) represents the solution for budget 240. The selected facilities are 0,1,2,58 and 82. Among these facilities, 0,1 and 2 are fortified. We can note that the increased budget value of 240 was better spent by selecting different nodes as facilities compared to the case where the budget is 120. Finally, Figure 6(d) shows the solution for an available budget of 360. The selected facilities are 0,1,2,58 and 82. With this budget, all of these facilities are fortified. It is clear that a higher budget helps to fortify more facilities which lowers the cost (fortified facilities take both primary and backup role) while increasing reliability.

In Figure 7, we provide the performance appraisal of the heuristic technique by taking the 30, 49, 100 and 150 node problem instances where $P = 5$. Thus, Figure 7(a) presents a bar-graph depicting the total search space size for each problem instance and for increasing budget values (0 to 360). In addition, Figure 7(b) depicts a dual axis graph illustrating the same problem instances showing the percentage of the search space that is heuristically visited (on the left side primary axis) and the computation time (on the right-side secondary axis). We also provide the backtrack (BT) count for each heuristic search.

In this context, we note that a higher backtrack count of 8 for the 150 nodes instance with budget 120 leads to a notable increased computation time. We observe in Figure 7(a) that the search space grows exponentially with the number of nodes and budget values, ranging from sub-million to many billions. The heuristic search predominantly visits a small fraction of the total search space as shown in Figure 7(b). However, in order to obtain competitive solutions, a certain critical mass of a few million solutions is needed during each search. Consequently, in the few cases (i.e. the 30-node instance) where the total search space is comparable to the critical mass, the percentage of the search space heuristically visited is high. In contrast for the cases where the search space is vastly greater than the critical mass (i.e. 150 nodes instance), the percentage of the search space heuristically visited is around 0.1%.



(a) budget=0, facilities={2,53,58,82,88}

(b) budget=120, facilities={1,2,53,58,82}

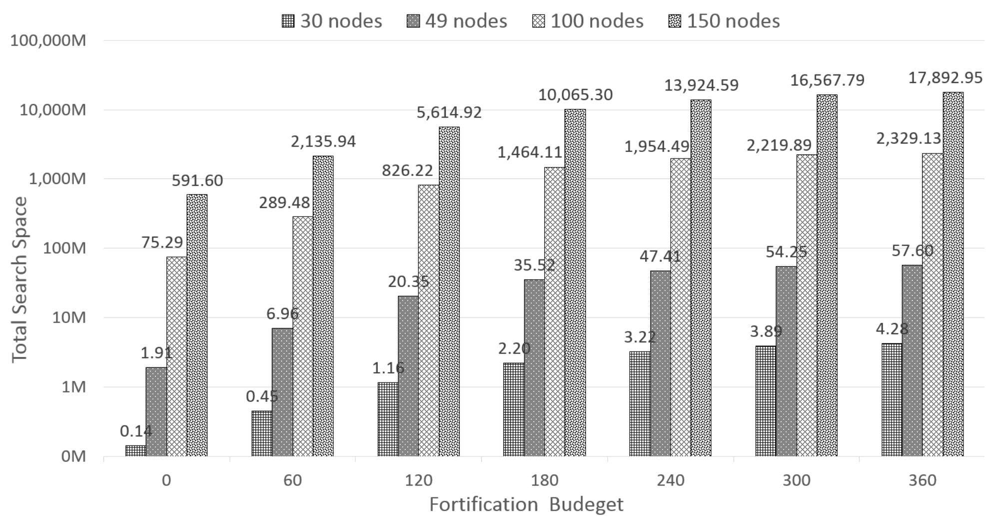
(c) budget=240, facilities={0,1,2,58,82}

(d) budget=360, facilities={0,1,2,58,82}

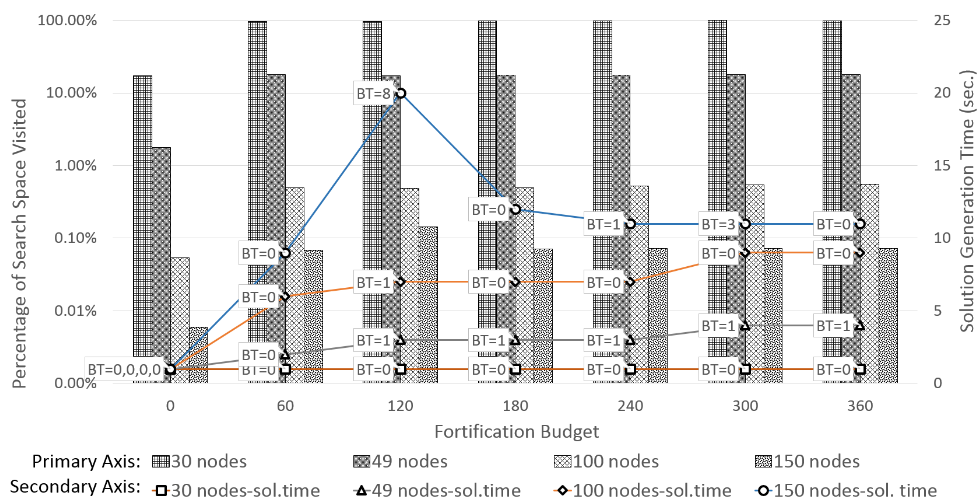
Figure 6: Solution profile for 100-node problem instance ($P = 5$)

Table 8: Benchmark details on RUFL problem instances

Nodes	Budget	Facilities	LR:LB	LR:UB	Best Known		Heur. Cost	Gap(%)	Time(s)
30	0	1,10,12,13	7963.9	8003.9	8003.9	*	8003.9	0.0	<1
	30	1,10,12,13	7963.9	8003.9	8003.9	*	8003.9	0.0	1
	60	1,10,12,13	7854.1	7886.3	7886.3	*	7886.3	0.0	1
	120	1,10,12,13	7751.5	7789.8	7789.8	*	7789.8	0.0	1
	180	1,10,12,13	7713.3	7751.1	7751.1	*	7751.1	0.0	1
	240	1,10,12,13	7701.0	7734.3	7734.3	*	7734.3	0.0	1
	300	1,10,12,13	7697.7	7734.3	7734.3	*	7734.3	0.0	1
	360	1,10,12,13	7696.0	7734.3	7734.3	*	7734.3	0.0	1
49	0	1,3,6,8,12,18,22,27	12090.9	12151.1	12151.1	*	12151.1	0.0	6
	30	1,3,6,8,12,18,22,27	12090.9	12151.1	12151.1	*	12151.1	0.0	6
	60	1,3,6,8,12,18,22,27	11983.0	12042.5	12042.5	*	12042.5	0.0	6
	120	1,3,6,8,12,18,22,27	11933.2	11992.4	11992.4	*	11992.4	0.0	6
	180	1,3,6,8,12,22,27	11899.6	11959.3	11959.3	*	11959.3	0.0	7
	240	3,6,8,12,22,27,39	11884.4	11943.2	11943.2	*	11943.2	0.0	7
	300	3,6,8,12,22,27,39	11846.5	11903.0	11903.0	*	11903.5	0.0	9
	360	1,3,6,8,12,22,27	11837.2	11896.6	11881.3	*	11881.3	0.0	11
100	0	3,9,11,15,19,27,47,50,54,55,58,69,91	17295.3	17295.3	17380.3		17380.6	0.0	25
	30	3,9,11,15,19,27,47,50,54,55,58,69,91	17295.3	17295.3	17380.3		17380.6	0.0	49
	60	3,9,11,15,19,27,47,50,54,55,58,69,91	17185.6	17271.4	17271.4		17271.4	0.0	47
	120	3,9,11,15,19,27,47,50,54,55,58,69,91	17084.9	17178.5	17178.5		17178.5	0.0	60
	180	3,9,11,15,19,27,47,50,54,55,58,69,91	17056.0	17140.5	17140.5		17140.5	0.0	54
	240	3,9,11,15,19,27,47,50,54,55,58,69,91	17031.0	17116.8	17116.8		17116.7	0.0	64
	300	3,9,11,15,19,27,47,50,54,55,58,69,91	16999.8	17082.6	17082.6		17082.5	0.0	96
	360	3,9,11,15,19,27,47,50,54,55,58,69,91	16999.8	17082.6	17082.6		17082.5	0.0	77
150	0	1,8,21,49,56,57,68,75,94,101,111,120,126	18953.7	19117.4	19117.4		19105.3	-0.06	31
	30	1,8,21,49,56,57,68,75,94,101,111,120,126	18953.7	19117.4	19117.4		19105.3	-0.06	39
	60	1,8,21,49,56,68,75,94,101,102,111,120,126	18840.4	19021.6	19021.6		19021.6	0.0	38
	120	1,8,21,49,56,57,68,75,94,101,111,120,126	18838.9	19000.3	19000.3		18966.6	-0.18	62
	180	1,8,21,49,56,57,68,75,94,101,111,120,126	18763.9	18916.9	18916.9		18945.25	0.15	44
	240	1,8,21,49,56,57,68,75,94,101,111,120,126	18586.8	18907.5	18907.5		18870.4	-0.2	45
	300	1,8,21,49,56,68,75,94,101,102,111,120,126	18600.2	18841.3	18841.3		18841.2	0.0	49
	360	1,8,21,49,56,57,68,75,94,101,111,120,126	18545.7	18853.0	18853.0		18786.2	-0.36	346



(a) Total search space size for each problem (in millions of combinations) vs. increasing budget



(b) Percentage of search space heuristically visited (primary) and computing time (secondary)

Figure 7: Heuristic Performance for 30, 49, 100 and 150 node problem instances ($P = 5$)

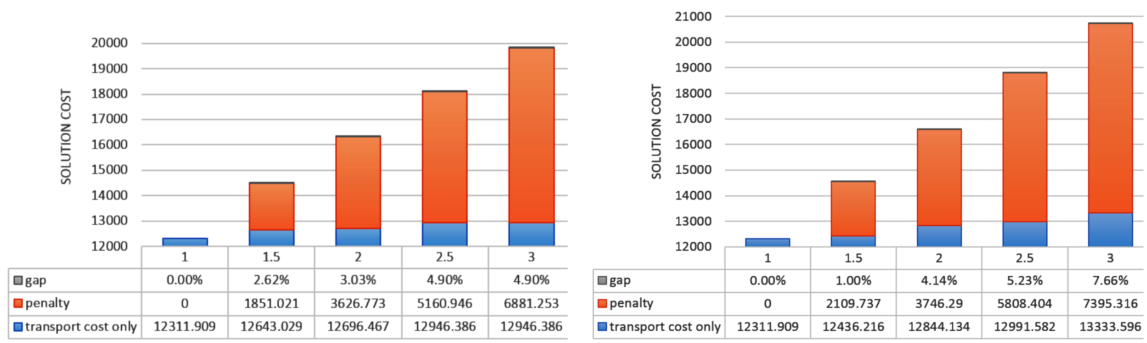
3.4.8 Sensitivity Analysis

In this section, we conduct a sensitivity analysis with respect to increasing demand priorities and fortification costs. In addition, we assess the impact of different distance calculation methods (e.g. Euclidean vs. Geographic) on the generated solutions.

3.4.8.1 Demand Priority & Fortification Cost

We discuss next the effect of preferred establishment locations on the heuristic solution with respect to demand priority and transport cost.This relates to practical situations where certain locations may be considered as having more priority to access their supplying facilities. Some nodes may represent the locations of various critical infrastructure objectives (e.g. power plants, telecommunications relay hubs, etc.) which can have demands that need to be served faster from a strategic perspective. In this setting, we employ a priority factor over the existing demands specific to different number of nodes and assess the impact on the corresponding solutions. We assume the presence of strategic objectives close to locations with larger population since the locations in the used data set represent important urban centers. While this abstraction is somewhat coarse (e.g. conventional power plants are not necessarily adjacent to major urban centers but also not very distant), it serves nevertheless to illustrate the effect on the heuristically obtained solutions.

We use, as example, the 100-node RPMP instance used in the benchmarks. Figure 8 contains two graphs that depict the increase in solution cost with respect to various priority factors.



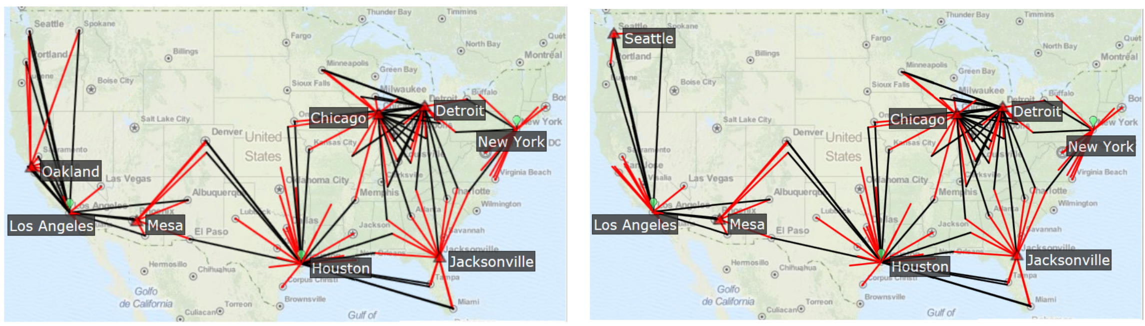
(a) Increasing demand priority

(b) Increasing demand priority and fortif. cost

Figure 8: Cost comparison for 100-node RPMP with fortification budget=240

We increase the priority factor as follows: 1.5, 2, 2.5 and 3 for the top 20 most populated cities in the data set. The original demands are multiplied with the priority factor. The total fortification budget is 240 and $p = 8$. Figure 8(a) shows the effect on the solution cost with increasing demand priority. Figure 8(b) shows the effect when increasing both demand priority and fortification cost by the same factor. The application of the factor naturally results in more costly solutions. However, since the transport (i.e. the distances among nodes) does not change, we reassess the solutions (after obtaining the initial location of the facilities and the corresponding customer assignment) without applying the factor. This way, we employ a form of a penalty method over the cost. The actual solution cost still increases albeit to a lesser extent since the initial solution differs from the optimal/near optimal solution in terms of where the facilities are located (i.e. in locations more favourable for the prioritized nodes). The corresponding transport cost and penalty are separately shown in Figure 8(a) and Figure 8(b). We use the Euclidean metric for cost calculation in order to have the means to assess the gap between the solution obtained for preferred establishment locations and the near-optimal solution (as provided in the benchmark results which employ reference values obtained using the Euclidean metric). The gap can provide an important insight to decision makers with respect to the relative increase in cost (as percentage) compared to the optimal/near-optimal solution.

Figure 9 illustrates the effects of demand priority on a geographic information system. We apply the priority factor on demands and fortification costs over the same 100-node RPMP instance as used in the benchmark results with budget 240 and $P = 8$.



(a) 2 $\times$ Demand Priority

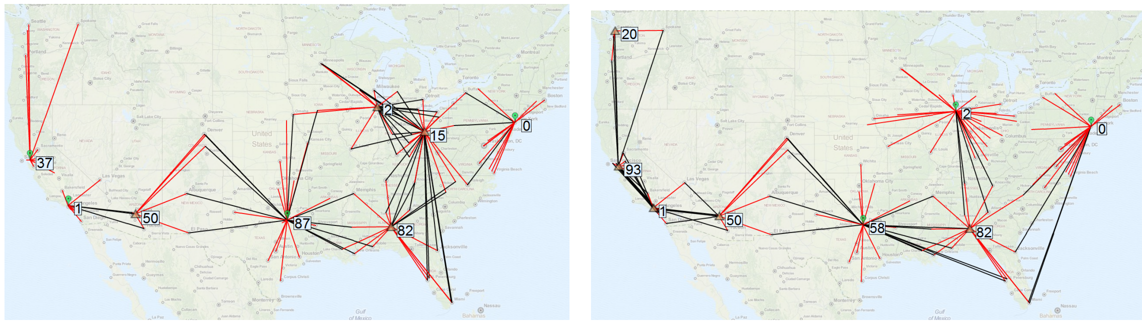
(b) 3 $\times$ Demand Priority

Figure 9: Solutions for 100-node RPMP with budget=240 and demand priority for 20% nodes

Figure 9(a) and Figure 9(b) show the results obtained by prioritizing the demands of 20% nodes with a factor of 2 and respectively 3. We note that compared to the case where no demand priority is considered (as in Figure 10(a) later), the results show depots located at more populated United States cities. Thus, Figure 9 shows depots established respectively at Houston, Jacksonville and Detroit instead of Garland (node 87), Montgomery (node 82) and Dayton (node 15) depicted in Figure 10(a). Also, with increased priority factor from double to triple, Seatle, with more population, is chosen as a facility instead of Oakland.

3.4.8.2 Euclidean vs. Geographic Distance

Figure 10 contrasts the solutions obtained using Euclidean and respectively geographical distances between nodes for the same 100-node RPMP problem instance.



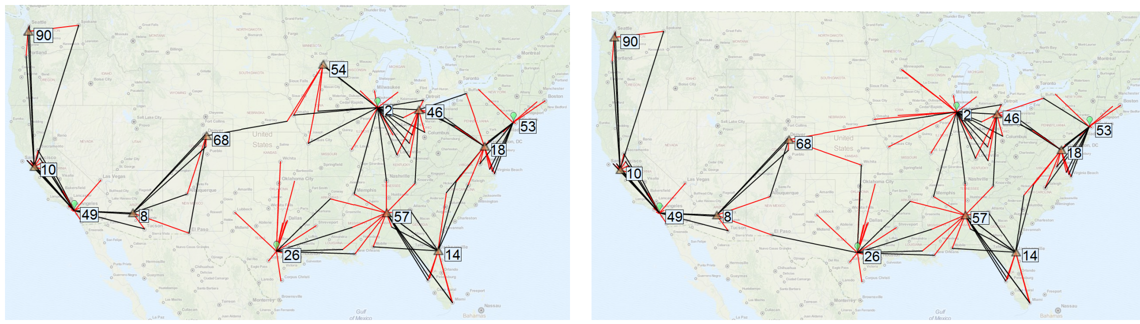
(a) Using Euclidean distance between nodes

(b) Using geographic distance between nodes

Figure 10: Solution comparison for 100-node RPMP instance ($p = 8$ and $budget = 240$)

We assess the solutions for $p = 8$ facilities under a fortification budget of 240. Figure 10(a) shows that when using the Euclidean distance, the selected facilities are 0,1,2,15,37,50,82 and 87. Among these selected facilities, 0,1,37 and 87 are fortified. Figure 10(b) shows that when using the geographical distance the selected facilities become 0,1,2,20,50,58,82 and 93. Among these selected facilities, 0,2 and 58 are fortified. Thus, the solutions are notably different according to the way of measuring the distance between nodes and subsequently the solution cost. In case of using the Euclidean distance calculated based on the latitude and longitude values taken as Cartesian coordinates, as by [54], the solution cost is 12311.9. In contrast, the solution cost is 1228228.4 in case of using the latitude and longitude values to compute the geographical distance, in kilometers. Thus, the Euclidean abstraction used by [54] can lead to different solution both in terms of facility establishment and fortification as well as transport cost. This aspect is even more important in the case of RUFL where the objective function is used to minimize the combined cost of transport and facility establishment.

Figure 11 depicts a solution comparison between the Euclidean and geographical distances for the 100-node RUFL problem instance under a fortification budget of 180. Since the calculated geographical distances are approximately two orders of magnitude greater than in the Euclidean case, we apply a corresponding magnification factor for the facility establishment cost. Without such factor, the best solution would be to establish facilities at every node since the savings in transport cost would cover the establishment cost.



(a) Using Euclidean distance between nodes (b) Using geographic distance between nodes

Figure 11: Solution comparison for 100-node RUFL problem with fortification budget=180

Figure 11(a) shows that when using the Euclidean distance, the solution has 13 established facilities (2,8,10,14,18,26,46,49,53, 54,57,68 and 90). Among the selected facilities 2,26,49 and 53 are fortified. Figure 11(b) shows that when using the geographical distance, (along with multiplying the establishment cost of all facilities by 100), the solution has 12 established facilities (2,8,10,14,18,26,46,49,53,57,68 and 90). Among these selected facilities, 2,26,49 and 53 are fortified. The solution is different according to the way of measuring the distance between nodes. In case of using the Euclidean distance as mentioned by [54], the solution cost is 17140.5. In contrast, the solution cost in case of using the geographical distance, is 1709895.2. Also, in this case, the use of geographical distance is more appropriate since only 12 facilities are required to be established.

3.4.9 Conclusion

In this chapter, we presented an evolutionary learning technique to near-optimally solve reliable facility location problems where potential facilities have individual failure probabilities. We addressed the RPMP and RUFL problems using a generic approach that can also be used to solve similar class of problems (e.g. PMP, UFL, etc.) with simple modifications. The approach is innovative and allows faster solution generation via evolutionary learning whereby the solution search space can be significantly reduced by progressively fixing the role of some nodes as customer only after learning from successive generations of solutions obtained using an evolving solution generator template. Of key significance is the possibility of selecting of a smaller or larger number of nodes to receive fixed roles per iteration, which allows for a trade-off between performance and computing time. This represents a distinctive feature compared to other approaches such as Tabu search. We provided the key highlights of the proposed approach using an illustrative example and demonstrated the performance via benchmark results. Our solution generation is affected by the following key factors: the problem size, the number of facilities, fortification budget and transport network properties. The solution quality is also affected when the facility count is comparatively larger with respect to the total number of nodes since accurately ascertaining the nodes role as customer only turns increasingly difficult with the increase in the number of facilities. Moreover, a limited fortification budget allowing just partial fortification also influences the solution quality as more extensive exploration of the solution space is needed. Conversely, there are situations where the budget is too small to allow any fortification at all or is sufficiently large to fortify all established facilities. In such situations, the solution search space is reduced since there is no need to explore various combination of spending the fortification budget. In general, this leads to obtaining more competitive solutions. The solution quality may also be influenced by the transport network setting, which may correspond to a sparse search space where the exploration of solution requires visiting more local minima (potentially involving backtracking) in order to reach a good quality solution.

Chapter 4

Capacitated Facility Location Planning Under Disruption

4.1 Introduction

CFL problem represents a subclass of FLP where a number of facilities with limited capacity must be established to serve customers with known demands such that the optimal solution minimizes an objective function typically involving establishment and transportation costs. In CFL problem, each node has an associated facility establishment cost and limited capacity so the decision makers have to establish an optimal number of facilities to minimize overall sum of establishment cost and transportation costs to meet all customer demands. Thus, in general, FLP assumes guaranteed availability of and unlimited capacity of the established facilities. Conversely, Reliable Capacitaed Facility Location (RCFL) problem considers a predefined probability of failure and limited capacity for each node. Therefore, in this context, each customer is assigned to multiple facilities (typically a primary and a backup) to ensure service continuity in the existence of such disruptions.

4.2 Model Assumptions

The following assumptions are typically made to define RCFL problem:

- Facilities have limited capacity.
- If a facility fails, it will lose portion of its capacity proportional to the failure probability and if it is fortified it maintains its original capacity.
- Facility failures are independent.
- Each customer is assigned to primary and backup facilities unless the primary facility is fortified (i.e. totally reliable) in which case the customer still has a backup facility but won't be assigned to.
- Probability of a simultaneous failure of primary and backup facilities is negligible for each customer.
- The fortification budget is fixed (represents an input to the problem).

4.3 Model Formulation

We present next the details and initial non-linear formulation of the RCFL problem based on a modification of the model introduced by [54] where we add supplementary valid inequality and capacity limitation constraints. In Section 4.4, we detail how we linearize the initial model after separating it into two sub-models.

RCFL problem is represented over, a complete network of interconnected nodes in which some nodes represent customers with particular demand value while the others represent specific locations associated with predefined failure probability and limited capacity where facilities can be established. The problem aims to serve all customers by establishing a number of facilities such that the total establishment cost and transportation cost to each customer from one primary and one backup facilities, is minimum. Additionally, the problem assumes a fixed budget available to fortify a number of selected facilities to increase network reliability.

We introduce a non-linear integer programming formulation to tackle the capacitated situation of the FLP under the risk of disruption. The formulation involves a set of I customers and a set of J facilities where a subset of J has to be established to serve the set of I customers. Each customer $i \in I$ has a specific demand d_i. Let $c_{ij} >= 0$ be the allocation cost of customer $i \in I$ to facility location $j \in J$. For each facility j there exist additional establishment cost f_j required to establish a new facility at node j and limited capacity cap_j. Also, each facility has a capacity cap_j and is assigned to a failure probability q_j where $0 \leq q_j \leq 1$. Moreover, each facility has a fortification cost fc_j consisting of a setup cost s_j and a reliability cost $q_j \times rc_j$ where rc_j denotes the cost related to unit reduction in the failure probability. Finally, total fortification budget in this model is B. The decision variables used in the formulation are as follows:

$$x_j = \begin{cases} 1, & \text{If a facility is established at location } j \\ 0, & \text{otherwise.} \end{cases}$$

$$z_j = \begin{cases} 1, & \text{If the facility at location } j \text{ is fortified} \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{ij0} = \begin{cases} 1, & \text{If customer } i \text{ is served by primary facility } j \\ 0, & \text{otherwise.} \end{cases}$$

$$y_{ij1} = \begin{cases} 1, & \text{If customer } i \text{ is served by backup facility } j \\ 0, & \text{otherwise.} \end{cases}$$

The objective function for optimization is expressed as follows:

$$Min \sum_{j \in J} f_j x_j + \sum_{i \in N} \sum_{j \in J} \left[h_i d_{ij} y_{ij0} (1 - q_j(1 - z_j)) + h_i d_{ij} y_{ij1} \sum_{r \in J, r \neq j} q_r y_{ir0} (1 - z_r) \right] \quad (13)$$

Subject to:

$$\sum_{j \in J} y_{ij0} = 1 \;\; \forall i \in I \quad (14)$$

$$\sum_{j \in J} y_{ij1} = 1 \;\; \forall i \in I \quad (15)$$

$$y_{ij0} + y_{ij1} \leq x_j \;\; \forall i \in N, j \in J \quad (16)$$

$$\sum_{i \in I} [d_i (1 - q_j) y_{ij0} + d_i q_j y_{ij1}] \leq (1 - q_j) cap_j \;\; \forall j \in J \quad (17)$$

$$z_j \leq x_j \quad (18)$$

$$\sum_{j \in J} (s_j + q_j \times rc_j) z_j \leq B \quad (19)$$

$$z_j, x_j \in \{0, 1\} \;\; \forall j \in J \quad (20)$$

$$y_{ij0}, y_{ij1} \in \{0, 1\} \;\; \forall i \in N, j \in J \quad (21)$$

The objective function Eq. (13) is used to minimize the total facility establishment cost and the total transportation cost together for the customers assigned to their primary and backup facilities. The term $\sum_{j \in J} f_j x_j$ represents total facility establishment cost. The term $\sum_{j \in J} h_i d_{ij} y_{ij0} (1 - q_j(1 - z_j))$ is the expected transportation cost between customer i and its primary facility, where $(1 - q_j(1 - z_j))$ is the probability that the primary facility is available. The term $\sum_{j \in J} h_i d_{ij} y_{ij1} \Big(\sum_{r \in J, r \neq j} q_r y_{ir0}(1 - z_r) \Big)$ is the expected transportation cost between customer i and its backup facility, where the term $\sum_{r \in J, r \neq j} q_r y_{ir0}(1 - z_r)$ is the probability that the primary facility failed (in this case, the backup facility is assumed to be available).

Constraints Eq. (14) and Eq. (15) ensure that each customer will be assigned only to one primary and one backup facility. Constraint Eq. (16) guarantees that the facility must be established before serving any customer. Constraint Eq. (17) ensures that the total demand assigned to a facility will not exceed its capacity. Constraint Eq. (18) is a valid inequality ensuring that a facility must be established before being fortified. Constraint Eq. (19) ensures that the total fortification cost is within the available budget. Finally, both Eq. (20) and Eq. (21) represent integrality constraints.

4.4 Separation-Linearization Solution Approach

Based on the definition of RCFL, Constraint Eq. (18) states that the facility must be established before being fortified. This simply means that the facility establishment, and of course customer allocation, are prior to the facility fortification. Based on the foregoing explanation, we can separate the model introduced in Section 4.3 to two separate sub-models typically name: establishment-allocation and fortification.

In the following two sections 4.4.1 and 4.4.2 we detail the foregoing two sub-models.

4.4.1 Establishment-Allocation

Based on the separation criteria explained above, the objective function of RCFL described in Eq. (13) can be rewritten as follows:

$$Min \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} \left[c_{ij} y_{ij0}(1 - q_j) + c_{ij} y_{ij1} \sum_{r \in J, r \neq j} q_r y_{ir0} \right] \tag{22}$$

Subject to:

$$\sum_{j \in J} y_{ij0} = 1 \;\; \forall i \in I \tag{23}$$

$$\sum_{j \in J} y_{ij1} = 1 \;\; \forall i \in I \tag{24}$$

$$y_{ij0} + y_{ij1} \leq x_j \;\; \forall i \in N, j \in J \tag{25}$$

$$\sum_{i \in I} [d_i(1 - q_j)y_{ij0} + d_i q_j y_{ij1}] \leq (1 - q_j)cap_j \;\; \forall j \in J \tag{26}$$

$$x_j \in \{0, 1\} \;\; \forall j \in J \tag{27}$$

$$y_{ij0}, y_{ij1} \in \{0, 1\} \;\; \forall i \in I, j \in J \tag{28}$$

4.4.1.1 Model Linearization

Theorem 1 *The RCFL is NP*-hard. **Proof.** We prove this by showing that a special case of the RCFL is *NP*-hard. If we consider the fortification budget $B = 0$ and the facility failure probability $q_j = 0, \;\; \forall j \in J$, then RCFL becomes the classical CFL problem which has been proven to be *NP*-hard as in [40].

RCFL is proven to be *NP*-hard and has a non-linear objective function as in Eq. (22) since $c_{ij} y_{ij1} \sum_{r \in J, r \neq j} q_r y_{ir0}$ can be expressed as $c_{ij} \sum_{r \in J, r \neq j} q_r y_{ij1} y_{ir0}$ which is the non-linear part of the objective function. The optimal solution of such problem requires combinatorial optimization that often renders the solution search procedure intractable for large supply networks and one of the possible solutions is to linearize the summation of product of the decision variables y_{ij1} and y_{ir0} model by introducing an auxiliary decision variable to the proposed model as follows. Let $v_{ijr} = y_{ij1} y_{ir0}$ where

$$v_{ijr} = \begin{cases} 1, & \text{If customer } i \text{ is served by primary facility } r \text{ and backup facility } j \\ 0, & \text{otherwise.} \end{cases}$$

Based on the foregoing, the RCFL problem can be rewritten as follows.

$$Min \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} \left[c_{ij} y_{ij0}(1 - q_j) + c_{ij} \sum_{r \in J, r \neq j} q_r v_{ijr} \right] \tag{29}$$

Subject to:

$$\sum_{j \in J} y_{ij0} = 1 \;\; \forall i \in I \tag{30}$$

$$\sum_{j \in J} y_{ij1} = 1 \;\; \forall i \in I \tag{31}$$

$$y_{ij0} + y_{ij1} \leq x_j \;\; \forall i \in N, j \in J \tag{32}$$

$$\sum_{i \in I} [d_i(1 - q_j) y_{ij0} + d_i q_j y_{ij1}] \leq (1 - q_j) cap_j \;\; \forall j \in J \tag{33}$$

$$v_{ijr} \leq y_{ir0} \;\; \forall i \in I, j \in J, r \in J, r \neq j \tag{34}$$

$$v_{ijr} \leq y_{ij1} \;\; \forall i \in I, j \in J, r \in J, r \neq j \tag{35}$$

$$y_{ir0} + y_{ij1} \leq v_{ijr} + 1 \;\; \forall i \in I, j \in J, r \in J, r \neq j \tag{36}$$

$$x_j \in \{0, 1\} \;\; \forall j \in J \tag{37}$$

$$y_{ij0}, y_{ij1}, v_{ijr} \in \{0, 1\} \;\; \forall i \in I, j \in J, r \in J, r \neq j \tag{38}$$

4.4.2 Fortification

In the fortification sub-model, a specific criteria is used to rank the established facilities that have been produced by the establishment-allocation sub-model in the normal scenario nef (recall in this case the facility failure probability is equal to 0) which are considered the most reliable and optimal facilities.

We calculate the rank of each facility f_j as follows:

- If a facility f_j is established in the normal situation (i.e., $f_j \in \ nef$) then $Rank(f_j)$ = $Clients(f_j)$ where $Clients(f_j)$ is the number of clients that are assigned to f_j
- otherwise $Rank(f_j)$ = 0

Thus, the total expected cost reduction from fortifying facility f_j is $Rank_j$ and accordingly our objective is to maximize the utilization of the available fortification budget B over the set of established facilities. Let fc_j be the cost of fortifying a facility z_j, then the problem can be expressed as follows.

$$Max \sum_{j \in J} Rank_j \ z_j \tag{39}$$

Subject to:

$$\sum_{j \in J} fc_j \ z_j \leq B \tag{40}$$

$$z_j \in \{0, 1\} \ \ \forall j \in J \tag{41}$$

4.4.3 Solution Approach Summary

According to section 4.4.1, the first goal is to use the model introduced in section 4.4.1.1 to generate the optimal location for facility establishment and the optimal customer allocation such that the total cost including establishment and transportation costs is minimum. Finally, the second goal is to use the model introduced in section 4.4.2 to optimally fortify subset of the established facilities according to available fortification budget.

Since the model is linearized, we propose an iterative approach based on CPLEX solver to generate exact and optimal solutions for RCFL instances in terms of facility establishment and customer allocation as well as using implementation for the knapsack problem to generate the optimal facility fortification strategy. According to the problem definition, a solution of RCFL problem instance corresponds to a particular assignment for the decision variables which satisfy the constraints and has a corresponding cost. In RCFL, this cost involves the summation of facility establishment and transportation costs. Our approach iteratively solve a problem instance as follows.

The first phase is called the Normal Allocation Phase (NAP) in which we consider that the facility failure probability is equal to 0 and we call CPLEX solver to generate the optimal assignment that minimize the linear objective function in Eq. (29). The output of this phase is the set of established facilities in the normal scenario nef, the customer allocation as well as the corresponding normal optimal solution cost (Nos). The set nef contains the optimal facilities that minimize the total establishment and transportation costs.

The second phase is called the Disrupted Allocation Phase (DAP) in which a failure probability is assigned to each of the available locations where we can establish a facility. We call CPLEX solver to produce the optimal assignment that minimize the linear objective function in Eq. (29). The output of this phase is the set of established facilities in the disruption scenario def, the customer allocation as well as the current optimal solution cost (Cus). In general, a subset of the elements in the set def will be found in the set nef.

The third phase is called the Fortification Phase (FP) in which we solve the sub-model detailed in Section 4.4.2. The output of this phase is the set of fortified facilities (sff) using the available fortification Budget B.

Finally, in the case where the problem is solved for multiple budget values, we iteratively call knapsack solver with facility fortification costs as well as the available fortification budget, then CPLEX solver with multiple input parameters including the matrix of allocation cost, facility establishment costs, facility capacities, facility failure probabilities. In this case, by using the output of the third phase, the proposed approach identifies which facilities will be fortified and then updates the facility failure probabilities vector accordingly. The approach terminates, if facility f_j is selected to be fortified then the corresponding failure probability will set to zero (i.e., $q_j = 0$) and the CPLEX solver will be called again with the modified parameters.

The approach terminates according to the following conditions:

- No more budget to fortify other facilities, or
- $(1 - \frac{Cus}{Nos}) <= 10\%$

where Nos represents the optimal solution cost in the normal scenario while Cus represents the optimal solution cost in the disrupted scenario. Note that the above 10% is user defined value.

4.4.4 Algorithm Design

The synopsis of our proposed iterative solution technique is depicted in Figure 12

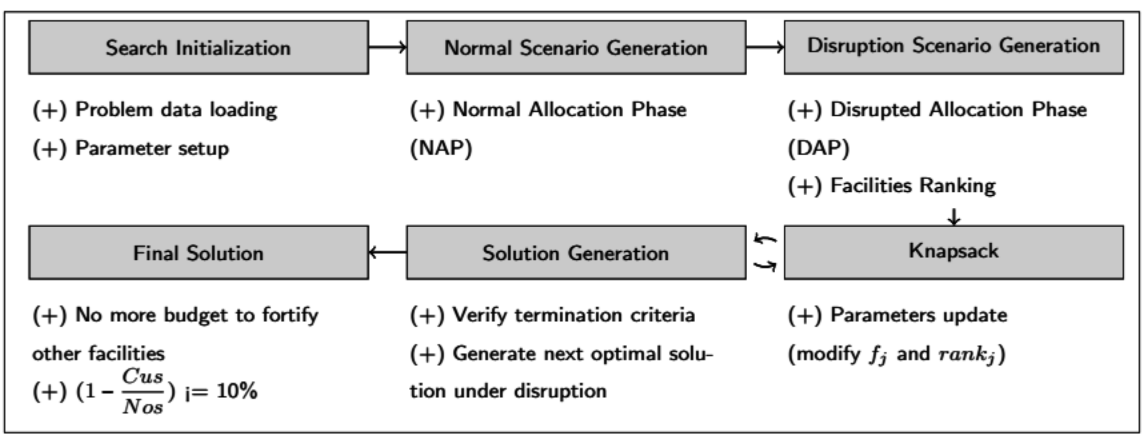


Figure 12: Overview of the Solution Technique

Algorithm 2 summarizes the main steps in the proposed iterative approach as follows.

Algorithm 2 RCFL Solution Algorithm ($RCFLSolver$)

i = Number of customers, j = Number of facilities
c_{ij} = Allocation cost matrix
d_i = Customer demand, cap_j = Facility capacity
f_j = Facility setup cost
$fort_j$ = Facility fortification cost
$rank_j$ = Facility rank
q_j = Failure probabilities
B = Maximum fortification budget
$Cub[k]$ = Current available budget
Nos Solution cost in normal situation
Cus Solution cost in disruption situation
Initialization
Set $q = 0$;
Input i, j, d, cap, c, q
Call CPLEX solver (i, j, d, cap, c)
Get Allocation strategy, Nos
Update facility failure probabilities q
Call CPLEX solver (i, j, d, cap, q, c)
Get Get Customers Allocation, Cus
Calculate facilities rank
Set k=0
Main body of the algorithm
while $Cub[k] \leq B \; or \; (1 - \frac{Cus}{Nos})$ <= 10% **do**
    Call knapsack ($cub[k], Rank, fort$)
    Update facility failure probabilities q
    Call CPLEX solver (i, j, d, cap, q, c)
    Get Customers Allocation, Cus
end while
Return CuS

4.4.5 Case Study

We present next an illustrative example of a RCFL instance, as depicted in Table 9. The example is based on a modified version of a 15 node instance where the number of clients is 10 and the number of candidate locations to establish facilities is at most 5. While the cost matrix is borrowed from [40], the failure probabilities and the fortification cost values have been borrowed from the first 10 nodes of the 30-node problem considered by [54].

Table 9: 15 Nodes Instance Details

Client	Customer Demand	Facility	Capacity	Estab. Cost	Failure Prob.	Fort. Cost
1	12	1	59	329	0.014	18.691
2	18	2	48	144	0.045	30.656
3	18	3	65	408	0.015	14.886
4	19	4	43	202	0.035	44.265
5	26	5	64	369	0.026	11.15
6	21					
7	18					
8	19					
9	18					
10	11					

Table 10 presents the 10 by 5 matrix of transportation costs between customers and facilities where the data is borrowed from [40]. We initially use the procedure to generate the optimal solution Nos in the normal scenario (i.e., facility failure probability q_j is equal 0). The solution provides the exact set of established facilities and customer allocation. Table 11 presents the optimal solution details of the 15 nodes instance in the normal (no disruption) scenario. The set of established facilities in this scenario are considered to be the most reliable.

Table 10: Customer Allocation Cost

i/j	F1	F2	F3	F4	F5
C1	8	85	5	49	42
C2	9	0	4	65	59
C3	35	21	65	94	46
C4	60	80	82	46	16
C5	63	44	67	47	62
C6	96	72	38	51	26
C7	51	36	34	53	71
C8	81	66	17	78	70
C9	10	62	2	26	0
C10	95	61	92	44	54

Table 11: Solution Details of 15 Nodes RCFL Instance in Normal Scenario

Facility	Rank	Assigned Customers	Estab. Cost	Allocation Costs	Total
1	3	1, 2, 3	329	52	381
2	2	7, 8	144	102	246
4	2	5, 10	202	61	263
5	3	4, 6, 9	369	42	411

According to the information presented in Table 11, the established facilities are 1, 2, 4, & 5 and the optimal solution cost in the normal scenario is 1331.

By updating the failure probabilities associated with each facility we use CPLEX solver to generate the optimal solution Cus in the disruption case.

The solution provides the exact set of established facilities, the role of each facility (primary or backup) as well as the customer allocation. The optimal solution of the 15 nodes instance in the disruption scenario (where the facilities are disrupted) is detailed in Table 12. According to Table 12, the selected facilities are: 1, 2, 4, 5 and the optimal solution cost in the disrupted case increases to 1333.9, which reflects the effect of the disruption.

Table 12: Customers Allocation of 15 Nodes RCFL Instance in Disruption Scenario

Facility	Role	Assigned Customers
1	Primary	1, 2, 3
	Backup	7, 9
2	Primary	7, 8
	Backup	2, 3, 5
4	Primary	5, 10
	Backup	4, 6
5	Primary	4, 6, 9
	Backup	1, 8, 10

The established facilities will be fortified according to their rank, which corresponds to the number of their assigned customers (more customers corresponds to higher rank) and our procedure involves progressively enhanced ability to segregate the most likely facilities that will be fortified based on their effect in reducing the solution cost. The output of the previous process are facilities 1, 2, 4, and 5 with ranks 3, 2, 2, and 3 respectively. We use the knapsack part of the proposed approach to sequentially fortify a subset of the established facilities based on the facility rank values and the available fortification budget. We use the information generated from the knapsack process and update the failure probabilities associated with the facilities that have been selected to be fortified (i.e., if f_j is fortified then $q_j = 0$). Then, we call the CPLEX solver to generate the next solution. Table 13 illustrate the output of the previous process where the underlined elements are the facilities that have been selected for fortification.

Table 13: Final Solution Details of 15 Nodes RCFL Instance

Instance	Budget	Facilities	Best Cost	Time(s)
15 Nodes	0	1,2,4,5	1333.9	4
	30	$\underline{1}$,2,4, $\underline{5}$	1332.1	4
	60	$\underline{1}$,2,4, $\underline{5}$	1332.1	4
	120	$\underline{1}$, $\underline{2}$, $\underline{4}$, $\underline{5}$	1311	4

In Table 13, it is observed that the effect of disruption on the solution cost is reduced by using the available budget to fortify some selected facilities. For instance, using budget 30, the cost has been reduced from 1333.9 to 1332.1 with total gain of 1.8. It might be unrealistic to use budget 30 to save 1.8 however from SCN activities and sustainability perspectives, the amount of 1.8 can represent savings on daily, weekly or monthly basis in contrast to a one time saving. On the other hand, the amount of 1.8 includes transportation cost but can also include maintenance operating cost, etc. Thus, it might be beneficial to spend a fixed fortification amount once in order to have repetitive savings on the long term.

Figure 13 depicts a snapshot of the relation between the disruption and fortification on one side and the solution cost on the other side. The figure shows that if the SCN facilities are subject to disruption and in the absence of any countermeasure (no fortification) the solution cost increase from 1331 to 1333.9. The figure also shows that by using the fortification budget, we can hedge against the effect of such disruption. Finally, it is observed that if there exist enough budget to fortify all the established facilities the solution cost is brought back to the normal situation where the SCN is totally reliable.

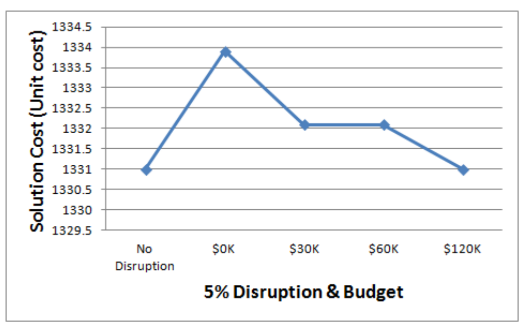


Figure 13: Disruption and Fortification Effect

Figure 14 depicts a snapshot of the relation between the disruption represented by the facility failure probabilities (expressed as percentages) and the solution cost on a 30-node instance borrowed from [40] where the failure probabilities are borrowed from [54]. The relation between the facility failure probability and solution cost is almost linear at the beginning when the failure probability percentage is low. On the other hand, when the failure probability percentage starts to increase, the relation turns to be quadratic which indicates the significant negative effect that disruptions can have of the SCN reliability and shows the importance of hedging against such disruption.

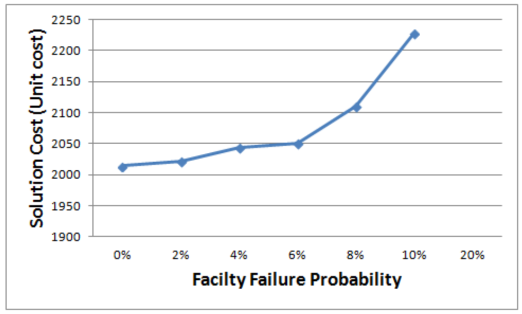


Figure 14: Disruption Effect vs. Solution Cost

4.4.6 Experimental Results

The experiments for RCFL have been conducted using a dataset comprising seven relevant instances used in [40] where the customer allocation costs, the cusomers' demands, the fixed costs for establishing facilities, and the facilities's capacities are available online, while the fortification costs and facilities failure probability are taken from [54]. The dataset contains respectively 30, 45, 60, 70, 60, 105 and 120 nodes problem instances. Each instance involves a set of nodes representing the customers associated with their demands and a set of nodes location representing the serving facilities as well as transport cost between the nodes in both sets. The knapsack sub-model of the proposed approach is implemented using C# and the benchmark results have been obtained using a 64-bit core i7 machine, with 8 GB RAM, running Windows operating system. The following paragraphs discuss the performance assessment of the proposed approach on the reference set of problem instances.

Table 14 partially details the solution generated by our approach in the normal case where the SCN is not exposed to any disruption. In other words, we consider that the facility failure probability is equal to 0. Each solution includes the established facilities, customer assignment, solution cost, fortification strategy as well as the rank of each of the established facilities. We recall that facility rank is the total number of customers served by a facility in the normal scenario. Our solutions is compared with the optimal solutions reported by [40]. We obtain the same results in terms of established facilities, customer assignment, and solution cost meanwhile we do not compare the time since we used a different computing platform.

The information in Table 14 provide another important insight to decision makers with respect to the facility rank since by calculating and sorting based on the rank of each of the established facilities, the decision makers can realize on what facilities they have to focus (which of these facilities they must start to fortify to reduce most of the failure effect).

Table 14: Solution Details of RCFL Problem Instances in Normal Scenario

Instance	Estab. Facilities	Rank	Best Cost	Optimal Cost (†)	Time (s)
30-nodes	2, 3, 4, 5, 7, 8, 9	3, 3, 2, 4, 3, 2, 3	2014	2014	12
45 Nodes	2, 5, 6, 7, 11, 14, 15	2, 1, 6, 7, 5, 6, 3	4366	4366	81
60 Nodes	1, 3, 5, 6, 7, 9, 12, 13, 14, 16, 17, 20	5, 4, 3, 2, 5, 5, 4, 1, 3, 2, 1, 5	15607	15607	259
70 Nodes	1, 2, 3, 4, 5, 6, 8, 11, 14, 16, 19, 20	7, 2, 8, 6, 6, 25, 7, 3, 4, 17, 11, 4	4448	4448	183
90 Nodes	6, 11, 16, 17, 21, 23, 24, 28	12, 5, 4, 7, 10, 10, 5, 7	4701	4701	200
105 Nodes	2, 3, 8, 10, 13, 14, 15, 22, 24	7, 9, 8, 8, 10, 8, 7, 10, 8	7887	7887	551
120 Nodes	13, 14, 17, 18, 19, 20 22,	13, 7, 11, 16, 13, 17, 13	5937	5937	814

Optimal Cost (†): The optimal solution costs reported in [40]

Table 15 partially details the solution generated by the proposed approach in the disruption scenario (i.e., each facility in SCN is subject to failure probability). The generated solution includes the established facilities, customer assignment to primary and back facilities and the solution cost. The obtained solutions are compared to the solutions obtained by the same approach in the normal case. The Avg. Fail(%) column represents the average failure probability for each instance and is calculated as follows: $\frac{1}{J}\sum_{j \in J} q_j$, where J is the total number of facilities in the corresponding instance and q_j represents the failure probability of facility f_j.

The Gap% represents the gap between the solutions obtained in the normal situation (i.e., no facility disruption) and the solution obtained in the disruption situation (i.e., all or a subset of the facilities are partially disrupted) and is calculated as follows: $Gap\% = (\frac{Disruptedsolution}{Normalsolution} - 1) * 100$

In partial disruption situation, the facility loses portion of its capacity proportional to the failure probability associated to the facility. We note that $Gap\%$ is always positive since the solution cost in the disruption scenario is greater or equal (with enough fortification) than the solution cost in the normal scenario as a result of the facility failure. Moreover, the customer in the disruption scenario will be assigned to two layers of services (primary and backup facilities) which adds extra transportation cost for the customer to go to the backup facility while in the normal case the customer will be assigned only to one layer of serving facility (recall that each customer still has a backup facility but would not have to go to).

Table 15: Solution Details of RCFL Problem Instances in disruption Scenario

Instance	Estab. Facilities	Best Cost	Avg. Fail(%)	Solution Gap(%)	Time (m)
30-nodes	1 2 4 5 7 8 9 10	2041.8	2.9	1.4	1.32
45 Nodes	2, 5, 6, 7, 11, 14	4398.2	3.1	0.74	4.1
60 Nodes	1, 3, 5, 6, 7, 9, 12, 13, 14, 16, 17, 20	15632	2.8	0.16	111.17
70 Nodes	1, 2, 3, 4, 5, 6, 8, 11, 14, 16, 19, 20	4499.2	2.8	1.2	405.3
90 Nodes	6, 11, 16, 17, 21, 23, 24, 28	4723.2	2.6	0.47	42.82
105 Nodes	2, 3, 8, 10, 13, 14, 15, 22, 24	7922.3	2.6	0.45	82.36
120 Nodes	9, 10, 13, 14, 18, 20, 22	6037.3	2.6	1.7	349.45

In Tables 14 and 15, there are two key factors that have impact on the solution computation time of each instance: the problem size and the number of established facilities. Finally, for each instance, the computation time in the case of the disruption scenario is always larger than in the case of the normal scenario since the solution search space of the disruption case is always larger than the one of normal case. This due to the fact that in the disruption case, each customer will be assigned to primary and backup facilities and accordingly the search space will contain a larger number of combinations of primary and backup while in the normal case, each customer will be assigned only to a primary facility which corresponds to a search space with a lower number of combinations. Table 16 depicts the solution details of different RCFL instances of different size versus the impact of disruption as well as the impact of fortification using different budget values on the solution cost and established facilities. The table depicts the solutions in terms of established facilities and solution cost of different scenarios. In our experiment, we consider the following scenarios:

- Normal scenario (no disruption),
- disruption scenario using no fortification budget ($0K), and
- disruption scenario using different fortification budget values ($30K, $60K, $120K, $180K, $240K and $300K).

The underlined facilities are the facilities that have been selected to be fortified. We note that the solution cost in the case of the disruption scenario is always larger than in the case of the normal scenario as a result of the impact of disruption. By starting the fortification process, we notice that the solution cost in the disruption scenario starts to be reduced. Finally, Table 16 shows that when fortifying all the established facilities, the solution costs and the established facilities in the both the disrupted and the normal scenarios are the same.

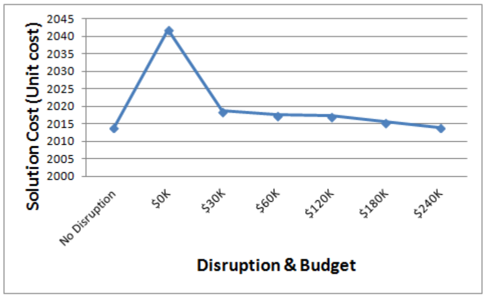
Table 16: Solution Details vs. Disruption/Fortification Effect

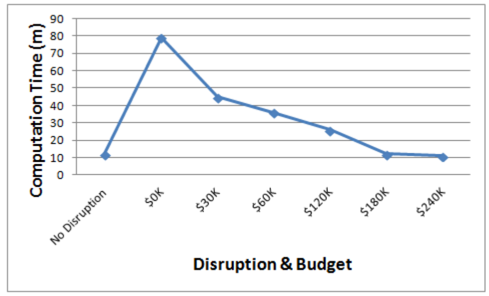
Instance	Solution Details	Normal	Budget 0	Budget \$30K	Budget \$60K	Budget \$120K	Budget \$180K	Budget \$240K	Budget \$300K
30-nodes	Est. Facilities	2 3 4 5 7 8 9	1 2 4 5 7 8 9 10	2 3 4 5 7 8 9	2 3 4 5 7 8 9	2 3 4 5 7 8 9	2 3 4 5 7 8 9	2 3 4 5 7 8 9	
	Solution Cost	2014	2041.8	2018.6	2017.5	2017.2	2015.5	2014	
45 Nodes	Est. Facilities	2 5 6 7 11 14 15	2 5 6 7 9 11 14	2 5 6 7 11 14 15	2 5 6 7 11 14 15	2 5 6 7 11 14 15	2 5 6 7 11 14 15		
	Solution Cost	4366	4398.2	4379	4371.1	4366.3	4366		
60 Nodes	Est. Facilities	1 3 5 6 7 9 12 13 14 16 17 20	1 3 5 6 7 9 12 13 14 16 17 20	1 3 5 6 7 9 12 13 14 16 17 20	1 3 5 6 7 9 12 13 14 16 17 20	1 3 5 6 7 9 12 13 14 16 17 20	1 3 5 6 7 9 12 13 14 16 17 20	1 3 5 6 7 9 12 13 14 16 17 20	
	Solution Cost	15607	15632	15619	15614	15610	15609	15607	
70 Nodes	Est. Facilities	1 2 3 4 5 6 8 11 14 16 19 20	1 2 3 4 5 6 8 11 14 16 19 20	1 2 3 4 5 6 8 11 14 16 19 20	1 2 3 4 5 6 8 11 14 16 19 20	1 2 3 4 5 6 8 11 14 16 19 20	1 2 3 4 5 6 8 11 14 16 19 20	1 2 3 4 5 6 8 11 14 16 19 20	1 2 3 4 5 6 8 11 14 16 19 20
	Solution Cost	4448	4499.2	4454	4453	4451.6	4449.5	4449	4448
90 Nodes	Est. Facilities	6 11 16 17 21 23 24 28	6 11 16 17 21 23 24 28	6 11 16 17 21 23 24 28	6 11 16 17 21 23 24 28	6 11 16 17 21 23 24 28	6 11 16 17 21 23 24 28		
	Solution Cost	4701	4723.2	4721.9	4719.3	4705.9	4701		
105 Nodes	Est. Facilities	2 3 8 10 13 14 15 22 24	2 3 8 10 13 14 15 22 24	2 3 8 10 13 14 15 22 24	2 3 8 10 13 14 15 22 24	2 3 8 10 13 14 15 22 24	2 3 8 10 13 14 15 22 24	2 3 8 10 13 14 15 22 24	
	Solution Cost	7887	7922.3	7913.6	7911.3	7900.1	7891.8	7887	
120 Nodes	Est. Facilities	13 14 17 18 19 20 22	9 10 13 14 18 20 22	13 14 17 18 19 20 22	13 14 17 18 19 20 22	13 14 17 18 19 20 22	13 14 17 18 19 20 22		
	Solution Cost	5937	6037.3	5978.6	5963.4	5941.4	5937		

Figures 15 and 16 depict the effect of disruption and fortification process on the solution cost and computation time for different size RCFL instances. It can be observed that there is a strong relation between the solution cost and the facility failure levels. Recall that by using a suitable fortification budget, the impact of facility failure can be reduced. Thus, the data in Table 16 as well as Figures 15 and 16 provide another important insight to decision makers with respect to the relative increase in solution cost compared to the optimal solution in the normal scenario, by using different fortification levels (strategies) according to the available fortification budget (partial or complete fortification).

Another important insight can be observed in Figures 15 and 16 which show that the maximum reduction on the solution cost mostly happens when using budget 30 and 60 respectively since the highest ranked facilities will be fortified up to this level which also shows the importance of facility ranking as discussed before. Figures 15 and 16 also depict the effect of disruption and fortification on the solution computation time of different size RCFL instances. It can be observed as discussed before that there is a strong relation between the computation time and the disruption level from one side and between the computation time and the problem size on the other side.

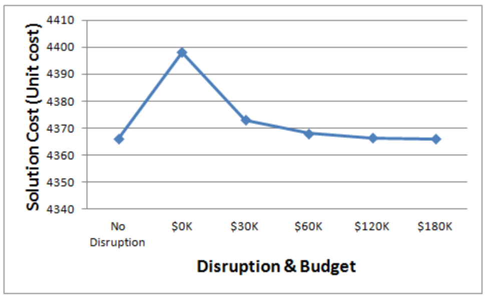
The final detailed solutions of all the instances that have been used in our experimental results can be found in the Appendix A.

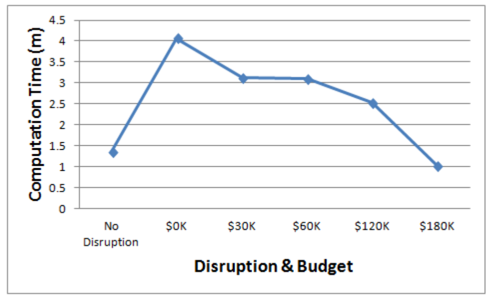




(a) 30-nodes (Cost)

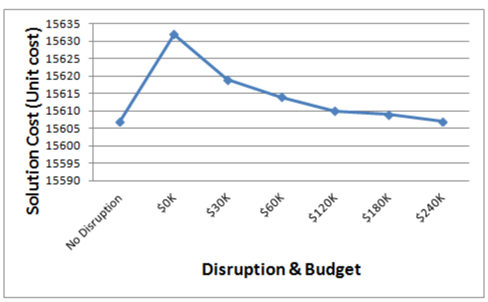
(b) 30-nodes (Time)

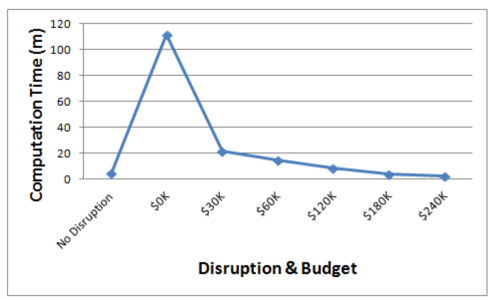




(c) 45 Nodes (Cost)

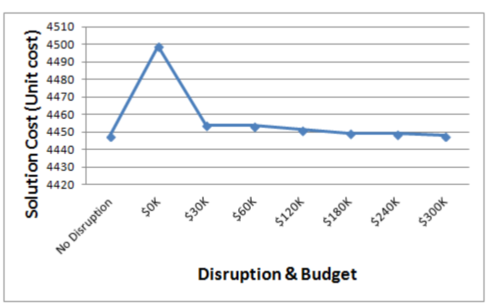
(d) 45 Nodes (Time)

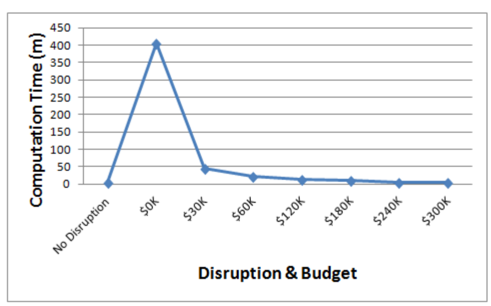




(e) 60 Nodes (Cost)

(f) 60 Nodes (Time)

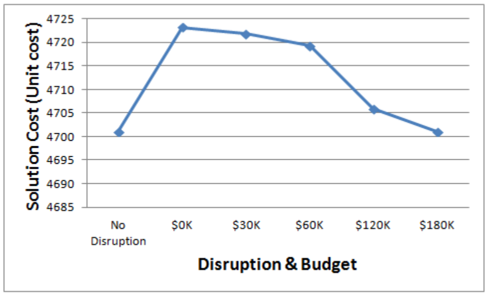




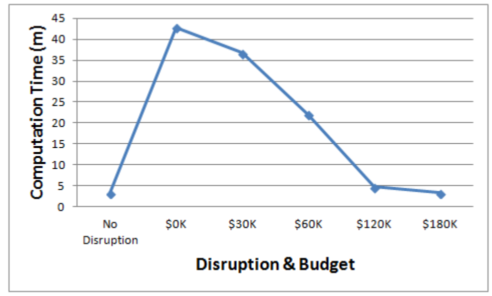
(g) 70 Nodes (Cost)

(h) 70 Nodes (Time)

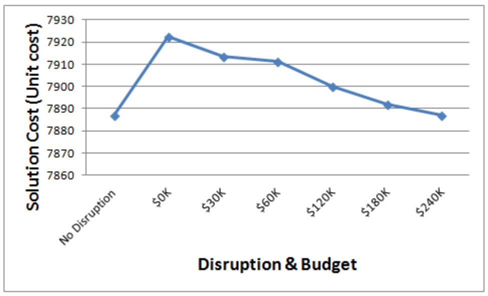
Figure 15: Disruption & Fortification (30 to 70 nodes) vs. Solution Cost & Computation Time



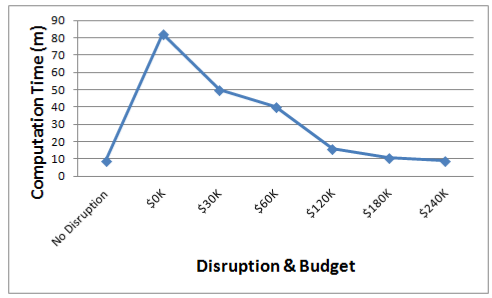
(a) 90 Nodes (Cost)



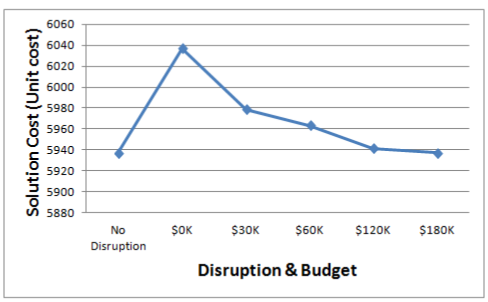
(b) 90 Nodes (Time)



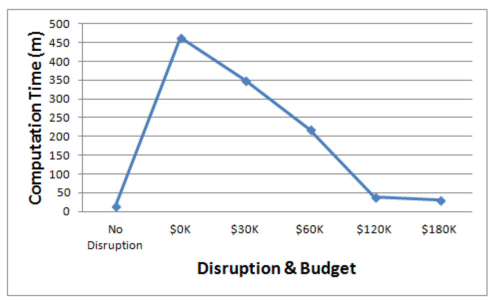
(c) 105 Nodes (Cost)



(d) 105 Nodes (Time)



(e) 120 Nodes (Cost)



(f) 120 Nodes (Time)

Figure 16: Disruption & Fortification (90 to 120 nodes) vs. Solution Cost & Computation Time

4.5 Conclusion

In this chapter, we presented a non-linear integer model for the RCFL problem. The model considers limited facility capacity, heterogeneous facility failure probabilities. Moreover, a limited fortification budget will be used to fortify a subset of the established facilities. The model assumes that if a facility is disrupted, it will lose portion of its capacity and if it is fortified, it maintains its entire capacity. We solve by separating the initial model into two sub-models dealing respectively with establishment-allocation and fortification. We presented a linearization of the establishment-allocation sub-model which we employed as part of an iterative approach leveraging the CPLEX solver and an implementation for the knapsack problem to solve the fortification sub-model. The approach allows generating optimal solutions in terms of of established facilities, customers allocation and the fortification strategy within the available budget. Our approach provides the decision makers with flexibility to hedge against different disruption situations and choose between different level of fortification according to the available budget.

Chapter 5

k-Shortest Path Problem Under Disruption

5.1 Introduction

Shortest path problem (SPP) represents one of the basic and classical problems of graph theory. The importance of such problem is back to its wide range of application such as GIS network analysis. The objective of the classical shortest path problem is to find the minimal length path between source and destination nodes on a transport network graph while the objective of the (k)-shortest path problem is to find k minimal length partial overlapping shortest paths.

5.2 Problem Assumptions

The following assumptions are typically used to define the k-shortest path problem.

- Transportation network is represented as a symmetric directed graph with no negative link length,

- If a node fails partially, all its connected links will incur extra cost (length) proportional to corresponding node failure value,
- If a node fails completely, it becomes unavailable as well as the connected links,
- Nodes and all its connected links are all failure dependent, and
- Nodes heterogeneous failure probabilities.

5.3 Problem Description

The following two sections provide a description for the graph partitioning and the shortest path problems.

5.3.1 Graph Partitioning Problem (GPP)

Graph partitioning problem (GPP) is an optimization problem where the goal is to partition a transportation network into a set of partitions in order to balance the workload and minimize the communication among generated partitions. Let $G = (N, L, C)$ be a directed graph that represents a transportation network where N is a set of nodes, L is a set of links and C is a square cost matrix whose diagonal are equal to zero (i.e. self-loops are not permitted) represents the lengths associated to each link. We define the cardinality of a partition as the number of nodes in such partition. We also define the density of a partition as the ratio of the number of the external edges of a partition to its cardinality. GPP aims to partition the set of N nodes into m mutually disjoint partitions as follows:

- The cardinality of each generated partition is limited to W as the maximum number of nodes in such partition, and
- The density of each generated partition is limited to Q as the maximum density of a cluster

The parameters W and Q are user-defined values and they have to be selected carefully since they may have the highest influence on the quality of the produced partitions.

5.3.2 Shortest Path Problem (SPP)

The objective of the *k*-shortest path problem is to find *k* shortest and partial overlapping paths between source and destination nodes on a graph. Let $G = (N, L, C)$ be a directed graph where N is a set of nodes, L is a set of links and C is a square cost matrix whose diagonal are equal to zero represents the length associated to each link. In general, A path p is defined as a sequence of adjacent nodes in a graph. Let $l_{i,j}$ be the link that connects nodes n_i and n_j and f be a weight function. The shortest path from a source node s to a destination node d is defined as the sequence of nodes (path) $P = (s, n_2, \ldots, d)$ such that over all possible combination of n minimizes $\sum_{i=1}^{n-1} f(l_{i,i+1})$.

5.4 Clustering Based Shortest Path Technique

Computing the *k*-shortest path between two locations on a road network is an important problem in the graph theory. The problem has variety applications in transportation network design. The classical solution approach for shortest path problem is Dijkstra algorithm as discussed in Dijkstra and Edsger, 1959. The author in [93] state that Dijkstra algorithm has deficiency in computing the shortest path between two nodes for a very large transportation network. To address this deficiency, handling node disruption as well as reducing the query time, we propose a hybrid solution approach based on a modified graph partitioning algorithm called Hierarchical Recursive Progression1 (HRP1) as detailed in [10] and modified Dijkstra algorithm for the *k*-shortest path problem where a subset of nodes are exposed to the risk of partial or complete disruption. The approach reduces the search space during the processing of shortest path queries by exploiting the fundamental property of border nodes which state that that any path starting from a source node s inside a cluster Cs to a destination node d in a cluster Cd must pass through one or more of the border nodes as stated in [66].

The following sections we detail our approach. We elaborate the main points of the proposed solution approach in section 5.4.1.

5.4.1 Algorithm Design

The solution approach consists of three separated phases:

- Problem Data Loading
- Graph Partitioning
- *k*-Shortest Path Generation

Figure 2 depicts the synopsis of the proposed solution approach.

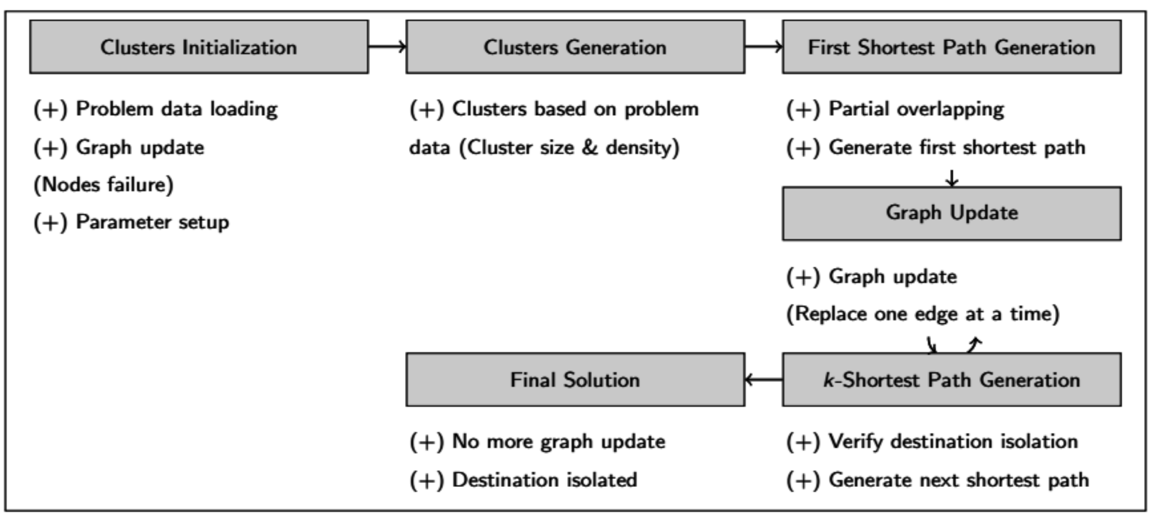


Figure 17: Overview of the Solution Technique

In the problem data loading phase, the problem data and the input parameters are loaded. The former includes the graph that is representing the transport network, node failure probabilities, the disruption scenario (partial/complete nodes failures), cluster size, cluster density, source and destination nodes. Finally, in this phase the graph is updated according to the disruption scenario (partial or complete) as well as node failure probabilities.

In the partitioning phase, we execute the modified HRP1 (known as HRP1+) to partition the graph into a number of clusters. HRP1+ initially consider that the total number of clusters is equal to $\mid N \mid$ (the total number of nodes in the graph) and each cluster contains only one single node. HRP1+ iteratively attempts to find a pair of clusters that can be combined together into one cluster by exploring all the connected pairs with combined cardinality (the total number of internal nodes in a cluster) less than or equal to W (user defined value represents the maximum number of permitted nodes in a cluster) and with individual density greater than or equal to Q (user defined value and is calculated as the ratio of the number of external connections of a cluster divided by its cardinality). A pair of clusters with the lowest density among all generated pairs is selected to be merged together with respect to the value of W which must be greater than 0 and less than the total number of nodes N. The crucial element of the algorithm is the appropriate chosen values of W and Q because as in [10] they have the major effect on the quality of the produced clusters. Q should not be too high or too low since the low-density value results in isolated clusters and the high-density value will generate a large number of inter-connections. HRP1+ terminates when the generated clusters reach has a size and density that satisfies the user defined Q and W constraints. HRP1+ also generates the border nodes (BN), border links (BL), critical nodes (CN) and critical links (CL) that are corresponding to each individual generated cluster. CN is defined as the subset of nodes in a network in which if any of these nodes completely fail that would isolate part of the network. CL is defined as the subset of links in a network in which if any of them completely fail that would isolate part of the network.

Finally, in the *k*-shortest generation phase, we use a modified version of Dijkstra algorithm to generate k partial overlapping shortest paths. In this phase, the total number of clusters (CTN), clusters values (CV), clusters adjacency matrix CAM(Connectivity matrix), border links BL, the source (s) and destination (d) node values are loaded. Using the previous informations, the approach is able to identify the source cluster (C_s) and the destination cluster (C_d) then attempts to find the shortest path from C_s to C_d ($SPCsCd$) by running Dijkstra algorithm in which CAM, (C_s), (C_d) are mimicking the graph, the source node, and the destination node respectively. Once the shortest path from C_s to C_d is generated and using the information generated in the second phase, the approach identifies the shortest border link (SBL) between each pairs of connected clusters located on the aforementioned shortest path $SPCsCd$ from C_s to C_d. For instance, $SBL12$ refers to the shortest border link between cluster 1 and cluster 2. The approach continues identifying the inner shortest paths (ISP) in each cluster individually. For instance, if cluster no. 1 is connected to cluster no. 2 then $ISP1$ refers to the shortest path in cluster 1 starting from the source node s to border node in $SBL12$ and so on. Finally, by adding the $ISPs$ values to the $SBLs$ values sequentially (for instance, $ISP1$ then $SBL1$ and so on) the approach produces the first shortest path from source node s in C_s to destination node d in C_d.

To generate the next *k* partial overlapping shortest paths, the approach iteratively, discards one edge at a time among those edges located on the first generated shortest path as well as from the original graph then repeats the previous steps to generate the next shortest path.

Finally, the termination criteria of this phase are either when the required number of paths are generated or the destination node d in C_d or any other two connected clusters, where the shortest path pass through, are isolated (disconnected). Recall, the network isolation (disconnection) will arise when any 2 nodes, that are representing a link, located on the last generated shortest path are among the subset CN generated in phase 2 and have to be discarded from the graph.

section 5.4.2 and section 5.4.3 elaborate the clustering and *k*-shortest generation phases of the solution approach.

5.4.2 Graph partitioning algorithm

HRP1+ considers a complete and partial disruption in graph nodes. The notations that are used in HRP1+ are as follows. Let N be the total number of nodes of the network, E the total number of edges of the network, H $\{n_1, n_2, \ldots, n_N\}$ group of nodes of the clusters, Q a user defined maximum density for a cluster, W a user defined maximum number of nodes in a cluster, r_k node subset k, where $k = 1, 2, \ldots, K$, and $q(r_k)$ is the density of a cluster r_k where $q(r_k) = \frac{C(rk)}{card(rk)}$. We handle the network failure as detailed in Section 5.2

The main steps of HRP1+ partitioning is presented in in Algorithm 3.

5.4.3 *k*-Shortest Path Generator

k-Shortest Path Generator(KSPG) exploits the Dijkstra algorithm that has been introduced by Dijkstra, 1959.

KSPG starts finding a shortest path from the source node (s) to destination node (d). Once a shortest path is generated, the procedure will discard one of the existing links that are in the generated path from the set $SPCsCd$. The algorithm is repeatedly executed on the pruned $SPCsCd$ to find the next shortest path and the process is continued until C_d is isolated.

The pseudo-code of the Dijkstra algorithm and KSPG algorithms are presented in Algorithm 4 and Algorithm 5 respectively.

Algorithm 3 HRP1 Partitioning($PartitioningSolver$)

Initialization
Update the network w.r.t. the disruption scenario
$K = N$ where K is a variable contains the number of clusters at each instant
for $k := 1$ **to** K **do**
  Compute $r_k = \{k\}$ and $q(r_k) = \frac{C(rk)}{card(r_k)}$
end for
Main body of the algorithm
Set $n = 0$
for $i = 1$ **to** $K - 1$ **do**
  for $j = i + 1$ **to** K **do**
    if $(r_i and r_j$ **are connected and** $(card(r_i) + card(r_j) <= W)$ **and** $(q(r_i) > Q)$ **and** $(q(r_j) > Q)$ **then**
      Set $n = n + 1$
      Set $s_n = r_i \cup r_j$.
      Calculate $q(s_n) = \frac{C(s_n)}{card(s_n)}$
      if $n = 1$ **then**
        Set $s^* = s_n$, $h1^* = i$, $h2^* = j$, $q^* = q(s_n)$
      end if
    end if
    if $(n > 1)$ **and** $q(s_n) < q^*$ **then**
      Set $s^* = s_s$, $h_1^* = i$, $h_2^* = j$, $q^* = q(s_n)$
    end if
    if $n = 0$ **then**
      Stop, Print r_k **for** $k = 1, 2, \ldots, K$, Terminate the algorithm
    end if
    Set $r_{h_1^*}$ = s_n **and** q($r_{h_1^*}$)
    if $h_2^* < k$ **then**
      for $i = h_2^*$ **to** $K - 1$ **do**
        Set $r_i = r_i + 1$ **and** $q(r_i) = q(r_i + 1)$
      end for
    end if
    $K = K - 1$
    go to 8
  end for
end for

Algorithm 4 Dijkstra Algorithm Steps (SP_Solver)

Function Dijkstra(Graph, Source, Destination)
for Each vertex v in Graph **do**
  Set $dist[v]$ = infinity
  Set $previous[v] = undefined$
end for
Set $dist[Source] = 0$
Set GN = The set of all graph nodes
while GN is not empty **do**
  Set u = vertex in GN with smallest $dist[]$
  if $dist[u]$ = infinity **then**
    Break
  end if
  if u = Destination **then**
    Break
  end if
  Remove u from GN
  for Each neighbour v of u **do**
    Set $alt = dist[u] + cost(u, v)$
    if $alt < dist[v]$ **then**
      Set $dist[v] = alt$
      Set $previous[v] = u$
    end if
  end for
end while
Set S = empty sequence
Set $u = Destination$
while $previous[u]$ is defined **do**
  Set Insert u at the beginning of S
  Set $u = previous[u]$
end while
Return S

Algorithm 5 k-Shortest Path($KSP\_Generator$)

s = Source node, C_s = Source cluster, d = Destination node, C_d = Destination cluster
$SPCsCd[i]$ = Shortest path from C_s to C_d, CPL = Length of $SPCsCd$
$CSP_s[i]$ = Source node i in $SPCsCd[i]$, $CSP_d[i]$ = Destination node i in $SPCsCd[i]$
$ISP[i]$ = Shortest path in cluster i
$SBL[i,j]$ = Shortest border link from cluster i to cluster j
$CAM[i,j]$ = Cluster adjacency matrix
$S[i]$ = Shortest path i,
k = number of paths found so far
Initialization
Update Overlapping Status
Set Isolated = $False$
Set k = 1
Set $SPCsCd$ = empty sequence
Set $SPCsCd[k]$ = Dijkstra (CAM, C_s, C_d)
Set S = empty sequence
Main body of the algorithm
while Isolated = $False$ **do**
  Set $S[k]$ = empty sequence
  Set CPL = Length(SPCsCd [k])
  for i = 1 **to** CPL **do**
    Dijkstra ($SPCsCd[i], CSP_s[i], CSP_d[i]$)
    if $(SBL[i, i+1])$ = empty **then**
      Set $S[k] = S[k] + ISP[i] + SBL[i, i+1]$
    else
      $S[k] = S[k] + ISP[i]$
    end if
  end for
  Remove one edge that appears in $S[k]$ from $SPCsCd[k]$
  Update CAM
  for $i = 1 to Length(CAM)$ **do**
    if $CAM[i] = 0$ **then**
      Set Isolated = $True$
      Exit For
    end if
  end for
  if Isolated = $False$ **then**
    Set $k = k + 1$
    Set $SPCsCd$ = empty sequence
    Set $SPCsCd[k]$ = Dijkstra (CAM, C_s, C_d)
  end if
end while

5.4.4 Case Study

Figure 18 depicts a graph of 24 nodes and 32 weighted links as in [10], that is representing the transportation network.

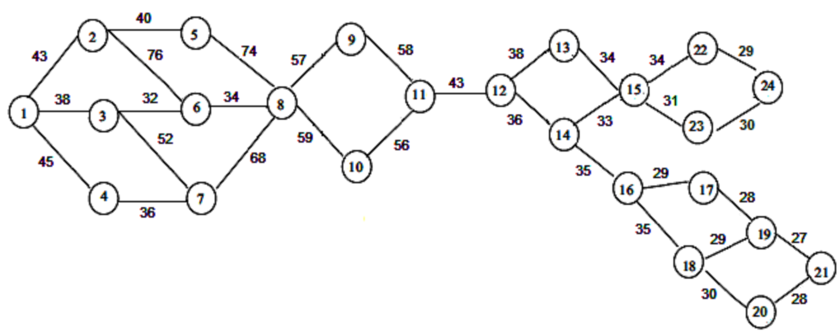


Figure 18: Case Study: A 24 node example Clustering problem

The partitioning phase generated 4 clusters (Typically named C_1, C_2, C_3, C_4), the border nodes, the border links, critical nodes and critical links that are corresponding to each individual generated cluster as presented in Tables 17 and 18. We assume that the maximum number of nodes in each cluster is 8 and the max density of a cluster is 0.125. Finally, we will refer to the cluster that contains the source node by source cluster and the cluster that contains the destination node by destination cluster.

Table 17: Network Partitioning Details

Cluster	Nodes	Border Nodes	Border Links
1	1, 4, 3, 2, 5, 6, 7, 8	8	8-9 , 8-10
2	9, 11, 10	9, 10, 11	9-8 , 10-8 , 11-12
3	12, 14, 16, 17, 18, 20, 21, 19	12, 14	12-11 , 12-13 , 14-15
4	13, 15, 22, 24, 23	13, 15	13-12 , 15-14

Table 18: Critical Nodes & Links

Inner Clusters	Critical Nodes	Critical Links
1 & 2	8	8-9 , 8-10
2 & 1	9, 10	9-8 , 10-8
2 & 3	11	11-12
3 & 2	12	12-11
3 & 4	12, 14	12-13 , 14-15
4 & 3	13, 15	13-12 , 15-14

Figure 19 shows the generated clusters, the sets of BN, BL, CN and CL.

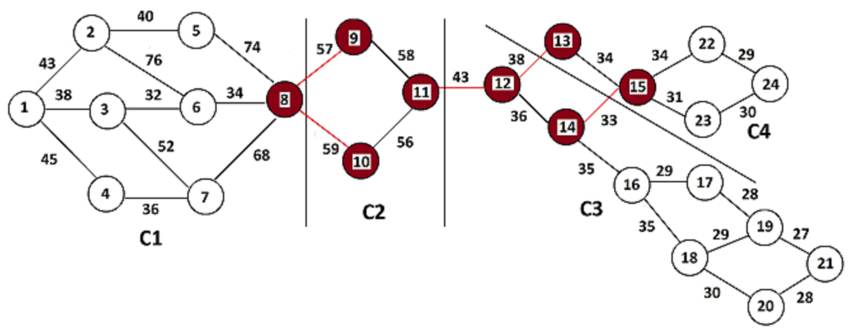


Figure 19: Clustering problem details

Table 19 presents the cluster adjacency matrix (CAM).

Table 19: Cluster Adjacency Matrix

C_i/C_j	C_1	C_2	C_3	C_4
C_1	0	(8-9, 57),(8-10, 59)	0	0
C_2	(8-9, 57),(8-10, 59)	0	(11-12, 43)	0
C_3	0	(11-12, 43)	0	(12-13, 38),(14-15, 33)
C_4	0	0	(12-13, 38),(14-15, 33)	0

In the *k*-shortest path generation phase, we consider that the source node to be node 1 in C_1 and the destination node to be node 24 in C_4. Using the clusters adjacency matrix (CAM), the approach identifies that the shortest path (SP) from C_1 to C_4 (SPC_sC_d) is [C_1,

C_2, C_3, C_4]. The next step is to find the shortest border link between each pair of the aforementioned 4 clusters starting from C_1 to C_4 as shown in Figure 19. In this example, the algorithm uses the information in CAM as well as in SPC_sC_d and identifies that the shortest 3 border links (Typically name $SBL[1,2]$, $SBL[2,3]$, $SBL[3,4]$) are 8-9,11-12 and 14-15 respectively. Now, the algorithm uses the information in CAM and $SPCsCd$ to identify the inner shortest in each cluster individually (Typically name $ISP[1]$, $ISP[2]$, $ISP[3]$, and $ISP[4]$). Finally, the algorithm merge ISP and SBL sequentially (i.e., $ISP[1]$ then $SBL[1,2]$) to generate a full shortest path between nodes 1 and 24. Table 20 summarizes the former steps.

Table 20: Single Shortest Path Solution Generation

Cluster	Inner Shortest Path, (Length)	Shortest Border Link (Clusters)(Length)
C_1	1, 3, 6, 8 (104)	8-9 (C_1-C_2) (57)
C_2	9, 11 (58)	11-12 (C_2-C_3) (43)
C_3	12, 14 (36)	14-15 (C_3-C_4) (33)
C_4	15, 23, 24 (61)	None
SP 1: 1, 3, 6, 8, 9, 11, 12, 14, 15, 23, 24, Length: 392		

To generate the next partial overlapping path, the approach identifies one edge at a time to be discarded from the graph and repeat the previous steps to generate the next shortest path. The procedure terminates when the predefined number of paths are reached or the destination node is isolated. the destination node is isolated when either the source cluster or the destination cluster isolated from the network.

Table 21 summarizes the 7 generated partial overlapping paths.

Table 21: Shortest Paths Details

Path No.	Paths Details	Paths Length
1	1, 3, 6, 8, 9, 11, 12, 14, 15, 23, 24	392
2	1, 4, 7, 8, 9, 11, 12, 14, 15, 23, 24	437
3	1, 4, 7, 8, 10, 11, 12, 14, 15, 23, 24	437
4	1, 4, 7, 8, 10, 11, 12, 13, 15, 23, 24	440
5	1, 4, 7, 8, 10, 11, 12, 13, 15, 22, 24	442
6	1, 2, 6, 8, 10, 11, 12, 13, 15, 22, 24	446
7	1, 2, 5, 8, 10, 11, 12, 13, 15, 22, 24	450

Table 22 details the solution generation of the 7 generated shortest paths shows that the paths pass through different clusters and through the border nodes detailed in Table 18 which confirms the fundamental property of border nodes mentioned in section 5.4.

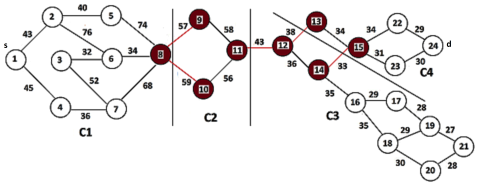
Table 22: k-Shortest Path Solution Generation

Cluster	Inner Shortest Path, (Length)	Shortest Border Link (Clusters)(Length)
C_1	1, 3, 6, 8 (104)	8-9 (C_1-C_2) (57)
C_2	9, 11 (58)	11-12 (C_2-C_3) (43)
C_3	12, 14 (36)	14-15 (C_3-C_4) (33)
C_4	15, 23, 24 (61)	None
SP 1: 1, 3, 6, 8, 9, 11, 12, 14, 15, 23, 24 (Length: 392)-(Discard edge 1-3)		
C_1	1, 4, 7, 8 (149)	8-9 (C_1-C_2) (57)
C_2	9, 11 (58)	11-12 (C_2-C_3) (43)
C_3	12, 14 (36)	14-15 (C_3-C_4) (33)
C_4	15, 23, 24 (61)	None
SP 2: 1, 4, 7, 8, 9, 11, 12, 14, 15, 23, 24 (Length: 437)-(Discard edge 8-9)		
C_1	1, 4, 7, 8 (149)	8-10 (C_1-C_2) (59)
C_2	9, 11 (58)	11-12 (C_2-C_3) (43)
C_3	12, 14 (36)	14-15 (C_3-C_4) (33)
C_4	15, 23, 24 (61)	None
SP 3: 1, 4, 7, 8, 10, 11, 12, 14, 15, 23, 24 (Length: 437)-(Discard edge 12-14)		
C_1	1, 4, 7, 8 (149)	8-10 (C_1-C_2) (59)
C_2	10, 11 (56)	11-12 (C_2-C_3) (43)
C_3	12 (0)	12-13 (C_3-C_4) (38)
C_4	13, 15, 23, 24 (95)	None
SP 4: 1, 4, 7, 8, 10, 11, 12, 13, 15, 23, 24 (Length: 440)-(Discard edge 23-24)		
C_1	1, 4, 7, 8 (149)	8-10 (C_1-C_2) (59)
C_2	10, 11 (56)	11-12 (C_2-C_3) (43)
C_3	12 (0)	12-13 (C_3-C_4) (38)
C_4	13, 15, 22, 24 (97)	None
SP 5: 1, 4, 7, 8, 10, 11, 12, 13, 15, 22, 24 (Length: 442)-(Discard edge 1-4)		
C_1	1, 2, 6, 8 (153)	8-10 (C_1-C_2) (59)
C_2	10, 11 (56)	11-12 (C_2-C_3) (43)
C_3	12 (0)	12-13 (C_3-C_4) (38)
C_4	13, 15, 22, 24 (97)	None
SP 6: 1, 2, 6, 8, 10, 11, 12, 13, 15, 22, 24 (Length: 446)-(Discard edge 2-6)		
C_1	1, 2, 5, 8 (157)	8-10 (C_1-C_2) (59)
C_2	10, 11 (56)	11-12 (C_2-C_3) (43)
C_3	12 (0)	12-13 (C_3-C_4) (38)
C_4	13, 15, 22, 24 (97)	None
SP 7: 1, 2, 5, 8, 10, 11, 12, 13, 15, 22, 24 (Length: 450)-(No more edge discard)		

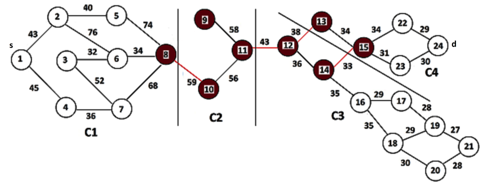
Figure 20 depicts the solution generation of the 7 partially overlapping shortest paths for a 24-node problem instance. The black nodes and the red links represent the border nodes and the border links respectively. It shows also that all the generated shortest paths always pass by the border links and the procedure generates only 7 partially overlapping paths then terminates because node 24 is isolated from the graph. Finally, the figure also shows that at each round the algorithm select only one edge to be discarded to produce the next partial overlap shortest path.

5.4.4.1 Partial Disruption

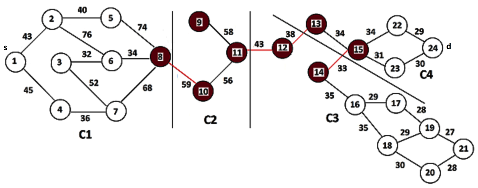
In this case, we randomly consider that 5% of the total number of network nodes are failing. Moreover, we will consider that the node failure probability values are in average 5%. For instance, consider the case of a car accident or road works on some bridges (connected nodes) on a TN. In this case, all the the roads (links) that are directly connected to that bridges (connected nodes) will affected partially by the event until it is fixed. As the assumption in section 5.2 state, the failure will propagate to all the connected links of the disrupted nodes so they will incur 5% extra length. Also, the node failure will not affect the topology of the network and will remain the same as the original topology. Finally, the generated clusters, border node, border links, critical nodes and critical length will also remain the same. According to the foregoing, on the 24 nodes dataset, nodes 5 and 9 are selected randomly to be failed so all the connected links of both nodes will incur 5% extra length. Table 23 present the generated paths in the case of node partial disruption.



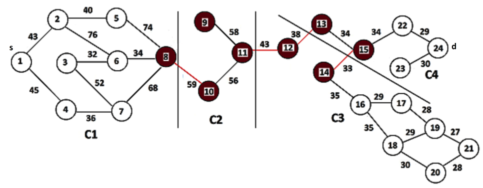
(a) **Path 1: 1, 3, 6, 8, 9, 11, 12, 14, 15, 23, 24 Length: 392, Discard edge 1-3**



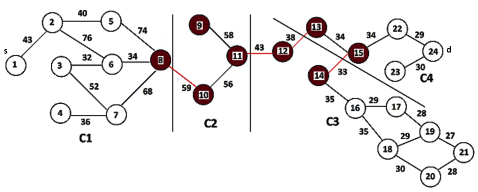
(b) **Path 2: 1, 4, 7, 8, 9, 11, 12, 14, 15, 23, 24 Length: 437, Discard edge 8-9**



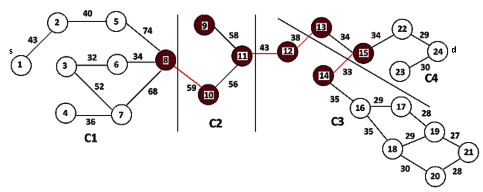
(c) **Path 3: 1, 4, 7, 8, 10, 11, 12, 14, 15, 23, 24, Length: 437, Discard edge (12-14)**



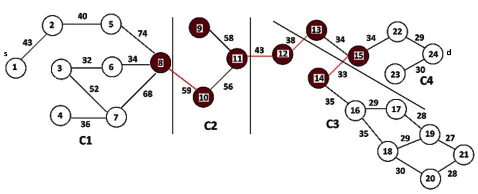
(d) **Path 4: 1, 4, 7, 8, 10, 11, 12, 13, 15, 23, 24, Length: 440, Discard edge (15-23)**



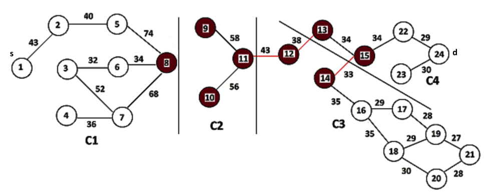
(e) **Path 5: 1, 4, 7, 8, 10, 11, 12, 13, 15, 22, 24, Length: 442, Discard edge (1-4)**



(f) **Path 6: 1, 2, 6, 8, 10, 11, 12, 13, 15, 22, 24, Length: 446, Discard edge (2-6)**



(g) **Path 7: 1, 2, 5, 8, 10, 11, 12, 13, 15, 22, 24, Length: 450, Discard edge (8-10)**



(h) **Path 8: Destination Isolated**

Figure 20: *k*-Shortest path generation details

Table 23: Node Partial Disruption SP Details

Path No.	Paths Details	Paths Length
1	1, 3, 6, 8, 10, 12, 14, 15, 23, 24	392
2	1, 4, 7, 8, 10, 11, 12, 14, 15, 23, 24	441
3	1, 4, 7, 8, 9, 11, 12, 14, 15, 23, 24	447
4	1, 4, 7, 8, 9, 11, 12, 13, 15, 23, 24	450
5	1, 4, 7, 8, 9, 11, 12, 13, 15, 22, 24	452
6	1, 2, 6, 8, 9, 11, 12, 13, 15, 22, 24	452
7	1, 2, 5, 8, 9, 11, 12, 13, 15, 22, 24	456

It is observed that the generated path lengths in the disruption case are greater than those who have been generated in the normal case. The average length in the normal case is 434.86 where in the disrupted case is 441.43 with an increase of 6.57 which reflect the effect of the partial disruption.

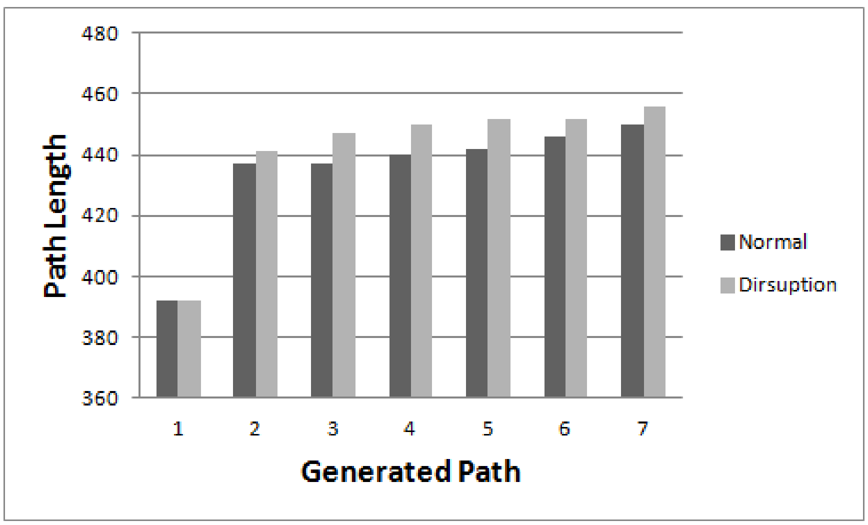


Figure 21: Normal vs. Partial Disruption Paths

Figure 21 compares the generated paths in the normal and partial disruption situations. Series 1 represents the generated paths in normal situation while Series 2 represents the generated paths in the disrupted situation. It is observed that only path 1 in both situations are equal and 6 out of 7 generated paths in the disrupted situation exceed the corresponding paths in the normal situation.

5.4.4.2 Node Complete Disruption

In this case, we also randomly consider that 5% of the total number of network nodes are completely failing (unavailable). As the assumptions in section 5.2 state, the failure will propagate to all the connected links of the disrupted nodes so each disrupted facility and all the corresponding connected links will not be available any more in the network. According to the foregoing, the complete disruption of the nodes will change the topology of the network and we have to repartition the network accordingly to maintain the connectivity among the network. Figure 22 depicts the topology changes of our original case study mentioned in Section 5.4.4.

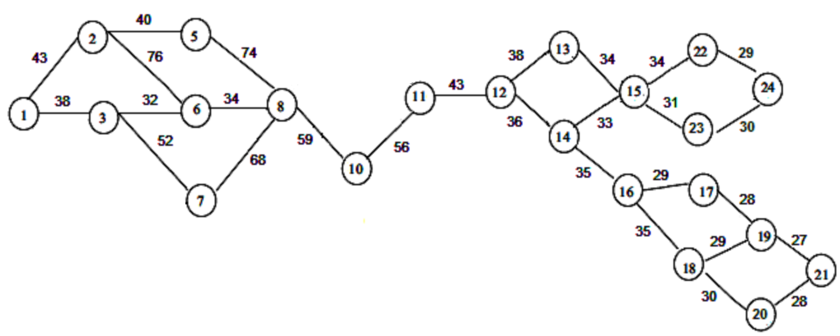


Figure 22: A Disrupted 22 Nodes Case Study

As the network topology change, the generated clusters, border node, border links, critical nodes and critical length will not remain the same as the normal scenario. By applying the previous steps on the 24 nodes dataset, nodes 4 and 9 are selected randomly to be disrupted completely (this explain why the dataset size decrease from 24 to 22 nodes) both nodes as well as the corresponding connected links will not be available any more in the network.

Table 24 present the details of the repartitioning process in which the maximum number of nodes per partition in each cluster should not exceed 8 nodes per cluster and the max density of each partition is 0.125. It is observed that the number of generated clusters increase to 6 as a result of the unavailability of nodes 4 and 9, and their related links.

Table 24: Network Repartitioning Details

Cluster	Nodes	Border Nodes	Border Links
1	1, 2, 3, 5, 6, 7, 8	8	8-10
2	10, 11, 12, 13, 14, 15, 22, 23	10, 14, 22, 23, 15	10-8, 14-16, 22-24, 23-24, 15-23
3	16, 17, 18, 19, 20	16, 19, 20	16-14, 19-21, 20-21
4	21	21	21-19, 21-20
5	24	24	24-22, 24-23, 24-23
6	23	23	23-15, 23-24

Table 25 present the generated paths in case of node complete disruption. It is also observed that the number of generated paths is decrease to 3 which the reflect the effect of topology change as a result of the unavailability of node 4 and 9 and, their related links.

Table 25: Node Complete Disruption SP Details

Path No.	Paths Details	Paths Length
1	1, 3, 6, 8, 10, 12, 14, 15, 23, 24	392
2	1, 2, 6, 8, 10, 11, 12, 14, 15, 23, 24	441
3	1, 2, 5, 8, 10, 11, 12, 14, 15, 23, 24	445

The average length of the first three generated paths in the normal case is 422 where in the disrupted case is 426 with an increase of 4 which reflect the effect of the node complete disruption.

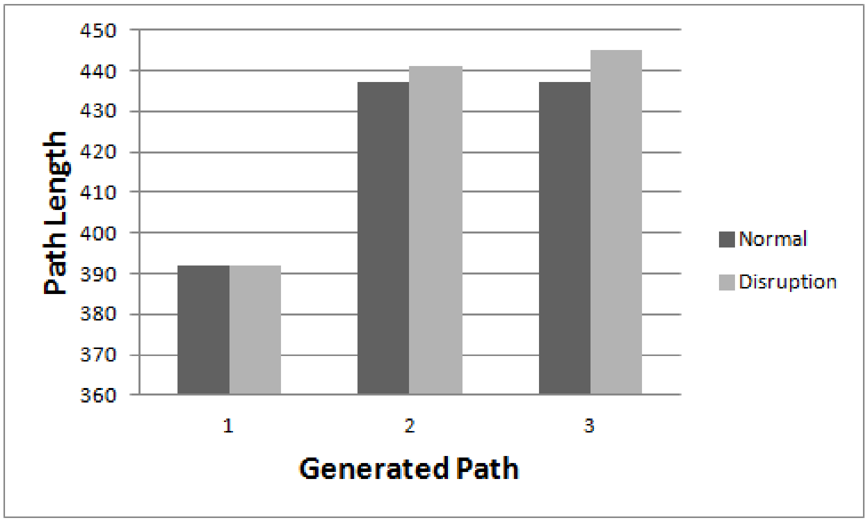


Figure 23: Normal vs. Complete Disrupted Paths

Figure 23 compares the first three generated paths in the normal and disrupted situation. Series 1 represents the generated paths in normal situation while Series 2 represents the generated paths when nodes 4 and 9, and their related links became completely unavailable. It is observed that only path 1 in both scenarios are equal and 2 out of 3 generated paths in the disrupted situation exceed the corresponding paths in the normal situation.

5.4.5 Experimental Results

The experiments for both the graph partitioning and *k*-shortest path generation problem have been conducted using several relevant datasets utilized in [74] and [38] for which the underlying data was obtained upon request. The datasets are ds332, ds662, ds1138, ds3353 which contain respectively 332, 662, 1138 and 3353 nodes, and are represented as a directed and symmetric graph. The cluster parameter W & Q are the maximum number of nodes in each cluster and the maximum value of a cluster density respectively. The proposed approach was implemented in C# and the benchmark results have been obtained using a 64-bit core i7 machine running Windows 7 Ultimate operating system.

Table 26 details the datasets that are used in these experiments, the partitioning parameters and the number of generated clusters.

Table 26: Datasets Details and Partitioning Parameters & Output

Dataset	Nodes	Links	Cluster Parameter	No. of clusters
ds332	332	2100	W = 50, Q = 0.1	8
ds662	662	906	W = 100, Q = 0.1	15
ds1138	1138	1458	W = 150, Q = 0.1	35
ds3353	3353	8870	W = 400, Q = 0.1	30

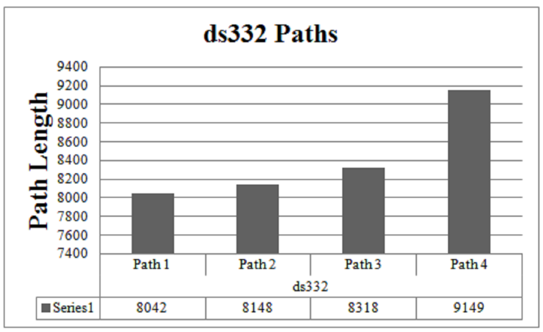
Table 27 summarizes the datasets information as well as the number of generated partial overlapping paths.

Table 27: Datasets and Partial Overlapping Paths Details

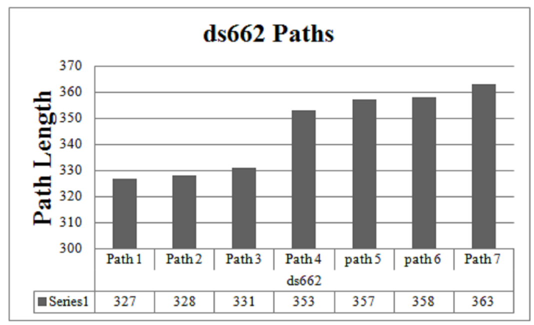
Dataset	Nodes	Links	Directed	Weighted	Symmetric	No. of Paths	Time (s)
ds332	332	2100	Yes	Yes	Yes	4	<1
ds662	662	906	Yes	Yes	Yes	7	<1
ds1138	1138	1458	Yes	Yes	Yes	7	<1
ds3353	3353	8870	Yes	Yes	Yes	4	<1

In this experiment, we assume that the required number of paths is 7 and we consider that the source node s and the destination node d is (1,332), (1,662), (1,1138), and (1,3353) in $ds332$, $ds662$, $ds11383$, and $ds3353$ datasets respectively. The approach generates only 4 partial overlapping paths fr datasets $ds332$ and $ds3353$ because the destination node is isolated in other words the source cluster and the destination cluster are disconnected. Finally, concerning $ds662$ and $ds1138$ datasets, the approach generates 7 partial overlapping paths.

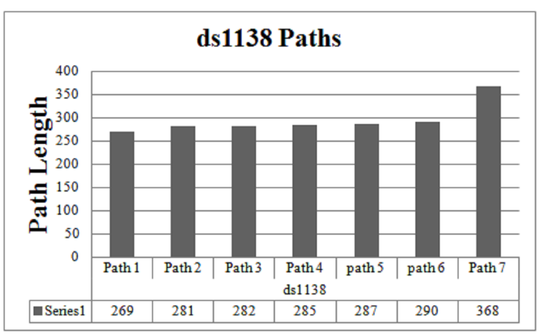
Figure 24 depicts the lengths of the generated partially overlapping paths that are corresponding to each of the 4 datasets. it is observed that the generated paths are varying in their lengths as expected.



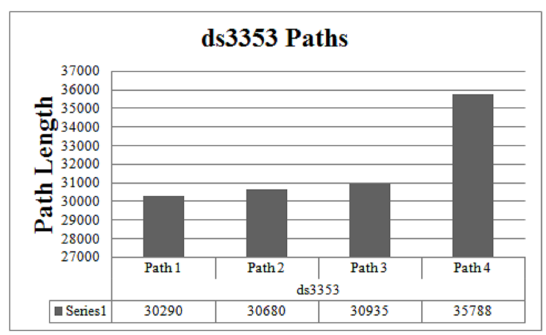
(a) Dataset ds332



(b) Dataset ds662



(c) Dataset ds1138



(d) Dataset ds3353

Figure 24: Shortest paths comparison

Table 28 details the generated partially overlapping paths. Table 29 presents the effect of the proposed approach on reducing the clusters search space. For instance, the original total cluster search space of $ds3353$ is 30 clusters (C_1-C_{30}) and the proposed approach reduced it in the first round to 9 clusters (C_1-C_9) (since the source and the destination nodes are located in C_1 and C_1 respectively) and in the final round reduced it to 4 clusters only (C_1, C_2, C_5 and C_9) which represents 13.33% of the original clusters search space.

Table 30 shows the effect of the clustering process on the search space reduction(SSR). The search space is represented by the total number of links in each dataset. For instance, in $ds3353$ the source node is 1 and the destination node is 3353. According to the obtained clustering data the source node is located in C_1 and the destination node is located in C_9. This reduces the search space from 8870 to 2848 links representing 31.1% of the original search space. We refer to this as the search space reduction in the first round (SSR1) The 2848 links represents the total number of links in the clusters search area [C_1-C_9] in addition to the border links between these clusters. The approach reduces the search space by identifying the shortest path from source cluster C_1 to the destination cluster C_9 and as a result the search space is reduced from 2848 to 1702 links representing 19.2% of the original search space (8870 links). We refer to this as the search space reduction in the second round (SSR2).

Table 29: Heuristics and Clusters Search Space Reduction

Dataset	No. of clusters	Phase 1	Phase 2 Avg.	Reduction (%)
ds332	8	8	2.5	31.25%
ds662	15	13	4.75	31.67%
ds1138	35	21	4	11.43%
ds3353	30	9	4	13.33%

Table 30: Clustering and Search Space Reduction

Dataset	Links	No. of Paths	C_s-C_d	SSR1(†) (%)	SSR2(†) (%)
ds332	2100	4	C_1-C_8	2100 (100%)	115 (5.84%)
ds662	906	7	C_1-C_{13}	813 (89.74%)	415 (45.81%)
ds1138	1458	7	C_1-C_{21}	1202 (82.44%)	459 (31.48%)
ds3353	8870	4	C_1-C_9	2848 (32.11%)	1752 (19.75%)

SSR1(†): Search space reduction in the first round round

SSR2(†): Search space reduction in the second round round

Considering $ds332$, $ds662$, $ds1138$, and $ds3353$ problem instances, Figure 25 shows how the clustering process reduces the cluster search space. Thus, the figure presents a bar-graph depicting the total search space size represented by the number of generated clusters for each dataset. The first phase of the reduced cluster search space is depicted by the initial bar and the subsequent bar depicts the average number of clusters that are heuristically visited to generate the shortest paths. It is observed that there is no reduction in the first phase for $ds332$ since the source node is located in cluster 1 and the destination node is located in cluster 8. In other words, the search space in the first phase of $ds332$ problem instance will be as the original search space while the average number of clusters in the final phase reduction will be 2.5 clusters.

Table 28: Shortest paths solution details

Dataset	Path No.	Paths Details	Clusterss	Cost
ds332	1	1,8,313,329,327,332	C_1,C_8	8042
	2	1,4,8,313,329,327,332	C_1,C_8	8148
	3	1,2,8,313,329,327,332	C_1,C_8	8318
	4	1,4,47,313,329,327,332	C_1,C_3,C_8	9149
ds662	1	1,222,172,272,425,414,413,393,323,456,544,659,658,662	C_1,C_8,C_7,C_{13}	327
	2	1,286,48,34,23,89,293,40,124,11,122,155,128, 192,290,55,206,392,442,441,379,650,655,662	C_1,C_4,C_7,C_{13}	328
	3	1,286,48,34,136,90,13,294,293,40,124,11,122,155, 128,192,290,55,206,392,442,441,379,650,655,662	C_1,C_{11},C_4,C_7,C_{13}	331
	4	1,286,48,34,310,410,414,413,393,323,456,544,659,658,662	C_1,C_8,C_7,C_{13}	353
	5	1,286,48,34,196,85,139,292,143,215,79,294,293,40,124,11, 122,155,128,192,290,55,206,392,442,441,379,650,655,662	C_1,C_2,C_{11},C_4,C_7,C_{13}	357
	6	1,286,48,34,310,410,414,413,408,566,518,456,544,659,658,662	C_1,C_8,C_7,C_{13}	358
	7	1,286,48,34,196,85,139,292,143,65,7,45,234,235,295,294,293,40, 124,11,122,155,128,192,290,55,206,392,442,441,379,650,655,662	C_1,C_2,C_{11},C_4,C_7,C_{13}	363
ds1138	1	1,563,567,566,555,556,579,927,921,527,526,505,578,530,797,805,1138	C_1,C_{16},C_{17},C_{21}	269
	2	1,5,9,104,13,34,553,503,507,508,781,776,797,805,1138	C_1,C_{17},C_{21}	281
	3	1,5,9,10,104,13,34,553,503,507,508,781,776,797,805,1138	C_1,C_{17},C_{21}	282
	4	1,5,9,10,104,34,553,503,507,508,781,776,797,805,1138	C_1,C_{17},C_{21}	285
	5	1,5,9,10,2,563,567,566,555,556,579,927, 921,527,526,505,578,530,797,805,1138	C_1,C_{16},C_{17},C_{21}	287
	6	1,5,9,10,2,563,567,566,555,557,558,998,995, 517,515,507,508,781,776,797,805,1138	C_1,C_{19},C_{17},C_{21}	290
	7	1,5,9,10,2,563,567,566,555,557,558,989,985,987,903, 556,579,927,921,527,526,505,578,530,797,805,1138	C_1,C_{19},C_{16},C_{17},C_{21}	368
ds3353	1	1,22,165,162,167,164,171,190,191,336,338,343,344, 340,347,348,335,515,407,524,582,589,596,597,636,641, 642,632,649,655,1229,1442,1277,1473,1282,1286,1474,1374,3353	C_1,C_2,C_5,C_9	30290
	2	1,2,3,23,166,163,164,171,190,191,336,338,343,344, 340,347,348,335,515,407,524,582,589,596,597,636,641, 642,632,649,655,1229,1442,1277,1473,1282,1286,1474,1374,3353	C_1,C_2,C_5,C_9	30680
	3	1,2,4,21,17,18,19,15,20,160,162,167,164,171,190,191,336,338, 343,344,340,347,348,335,515,407,524,582,589,596,597,636,641,642, 632,649,655,1229,1442,1277,1473,1282,1286,1474,1374,3353	C_1,C_2,C_5,C_9	30935
	4	1,2,4,5,8,12,14,15,20,160,162,167,164,171,190,191,336,338,343, 344,340,347,348,335,515,407,524,582,589,596,597,636,641, 642,632,649,655,1229,1442,1277,1473,1282,1286,1474,1374,3353	C_1,C_2,C_5,C_9	35788

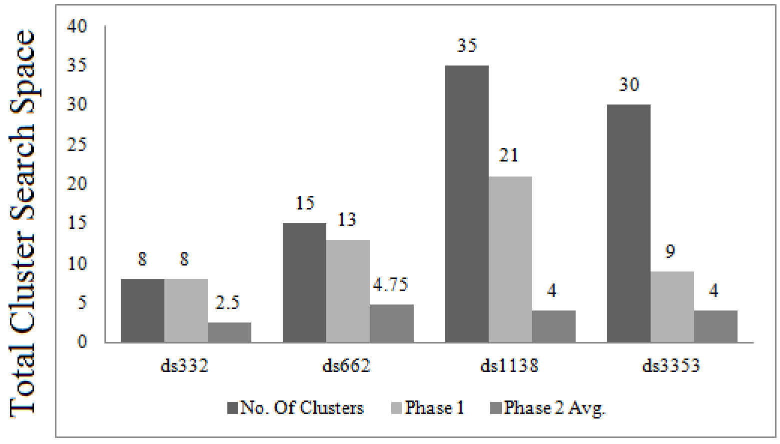


Figure 25: Cluster Search Space Reduction

Considering $ds332$, $ds662$, $ds1138$, and $ds3353$ problem instances, Figure 26 shows the performance appraisal of the heuristic approach. Thus, the figure presents a bar-graph depicting the total search space size represented by the total number of links in each problem instance and two bars depicting the reduced search space that were heuristically visited to generate the shortest paths. It is observed that there is no reduction in the first phase for $ds332$ since the source node is located in cluster 1 and the destination node is located in cluster 8. In other words, the first reduction will be identical to the original search space while the second and final reduction will include only cluster 1 and 8 since they are the shortest path cluster and they are both connected.

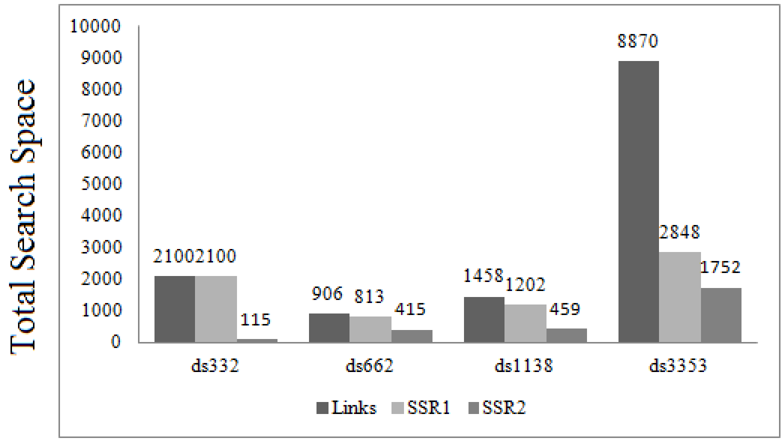


Figure 26: Solution finding via search space reduction

Figure 27 depicts the relationship between the density (multiplies by 1000) of each of the four datasets used in this experiment and the solution search space reduction (SSR) percentages. We notice that there is direct relationship between the density and the search space reduction. SSR1 is comparable to SSR2 for the four datasets except in ds332 since it has the maximum number of clusters (35 clusters) and the highest density value so there is a high chance that most of the 35 clusters interconnect. As expected, cluster 1, where the source node is located, is connected to cluster 8, where the destination node is located, and that explains why the reduced search space drops from 100% to 5.48%.

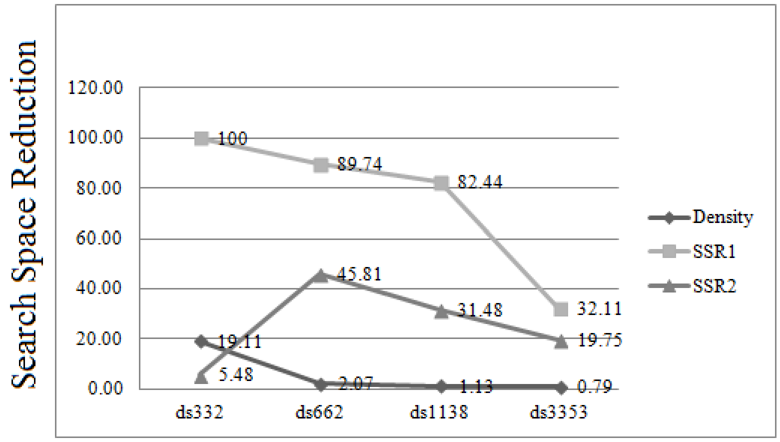


Figure 27: Search Space Reduction and Graph Density

5.4.5.1 Node Partial Disruption

In this section we will discuss the effect of partial disruption on both the network partitioning and the generated shortest paths. We will use the same datasets ds332, ds662, ds1138, ds3353 that have been detailed in Section 5.4.5. We will consider the percentage of disrupted nodes of is 5% (as Sectio 5.2) for each used dataset. Table 31 details the effect of partial disruption on each of the used dataset. The third column of the table represents the total number of disrupted nodes while the fourth column represents the total number of links that will incur 5% (recall this percentage represents the node failure probability) extra cost. The fourth column is Ratio(%) column which is defined as the percentage of failed links per node and is calculated as follows.

$$Ratio(\%) = \frac{Total\ disrupted\ links}{Total\ number\ of\ nodes}$$

For instance, the ratio of ds332 is 30 means that on average 30% of the connected to each node will fail partially. The ratio can provide an important insight to decision makers with respect to the average increase in paths in the disrupted scenario.

Table 31: Dataset Partial Disruption details

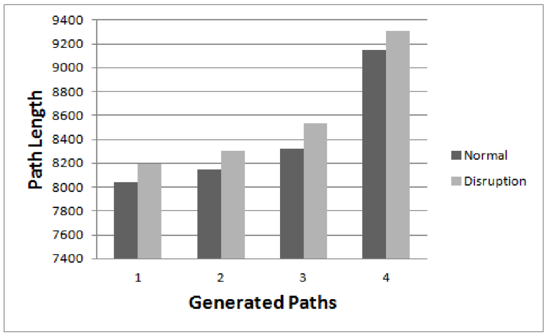
Dataset	Nodes	No. of Dis. Nodes	No. of Dis. Links	Ratio(%)
ds332	332	17	100	30
ds662	662	32	119	18
ds1138	1138	57	180	16
ds3353	3353	168	887	27

Table 32: Partial Disruption Shortest Paths details

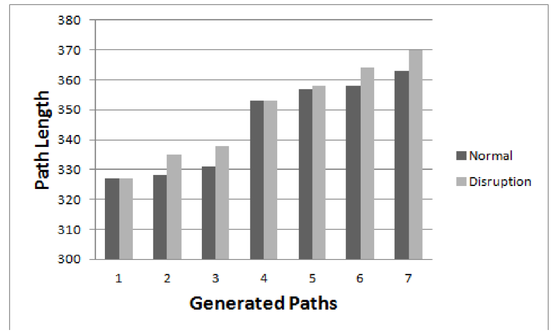
Dataset	Path No.	Normal Length	Disrupted Length	Avg. Normal Length	Avg. Disrupted Length	Gap
ds332	1	8042	8197	8414.3	8585.5	171.2
	2	8148	8303			
	3	8318	8538			
	4	9149	9304			
ds662	1	327	327	345.3	349.3	4
	2	328	335			
	3	331	338			
	4	353	353			
	5	357	358			
	6	358	364			
	7	363	370			
ds1138	1	269	271	294.6	295.4	0.8
	2	281	281			
	3	282	282			
	4	285	285			
	5	287	289			
	6	290	290			
	7	368	370			
ds3353	1	30290	30471	31923.3	32102.8	179.5
	2	30680	30855			
	3	30935	31116			
	4	35788	35969			

As expected, neither the number of generated clusters nor the number of generated paths will change only the paths length is expected to be changed.

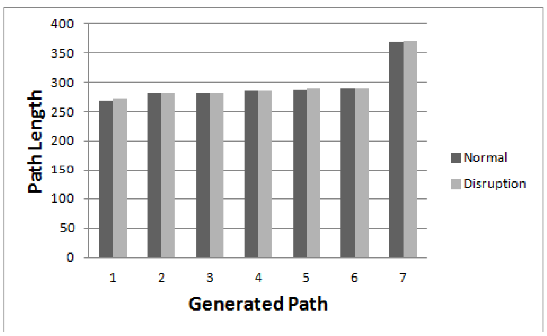
Table 32 compares the length of the generated paths for each dataset in the normal and partial disruption scenario.



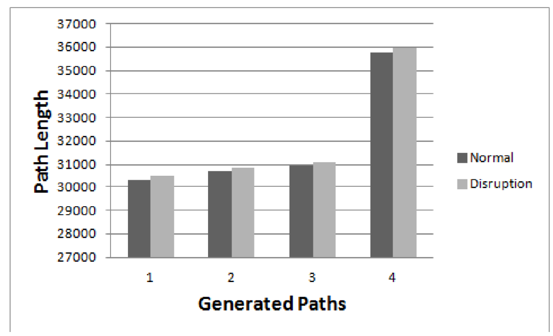
(a) Dataset ds332



(b) Dataset ds662



(c) Dataset ds1138



(d) Dataset ds3353

Figure 28: Partial Disruption Shortest Paths Comparison

Figure 28 depicts the lengths of the generated paths in normal and the partial disruption scenarios. It is observed that in ds331 and ds3353 all the 4 generated shortest paths in the disrupted scenario have higher cost (length) than their counterparts in the normal scenario.

Table 32 also presents the average increase (gap) of paths length in the disrupted scenario compare to the normal scenario. It is observed that ds332 and ds3353 have higher average length than ds662 and ds1138 which reflect the effect of node ration presented in Table 31 as explained above. Finally and with respect to the ratio effect, all the generated paths of ds332 and ds 3353 in the disrupted scenario will have higher cost that their counterparts in the normal scenario where in ds662 and ds1138 all the generated paths in both scenarios are very slightly different.

5.4.5.2 Node Complete Disruption

Again, we will consider the same assumptions in section 5.2 which state that the failure will propagate to all the connected links of the failed nodes. In this case 5% of the total number of each dataset and all their corresponding connected links will not be available any more in the network and as a result the topology of the network will change and we have to repartition the network accordingly to maintain the connectivity among the network. Table 33 details the number of nodes and links in each disrupted dataset as well as repartitioning results with respect to the network topology change.

Table 33: Datasets Repartitioning Results

Dataset	No. of Nodes	No. of Links	No. of Clusters (Normal)
ds332	315	2000	None (8)
ds662	628	787	56 (15)
ds1138	1081	1278	115 (35)
ds3353	3185	7983	232 (30)

In ds332, the repartition process can not generate any clusters since an isolation happened in the network. The reason behind this isolation is that nodes 313 and 329 are disrupted and they are indeed among the subset of critical nodes and the subset *critical sub-path*. The critical nodes are defined as the subset of nodes in a network in which if any of these nodes completely fail that would isolate part of the network. The critical nodes are part of the network partitioning process output in the normal scenario. We also define the *critical sub-path* as the as the subset of nodes the are exist in both the critical nodes and in all the generated path in the normal scenario. For instance, the critical node between cluster 1 and 8 is 313 between cluster 8 and 1 is 329. That is simply means cluster 1 and 8 will isolate from each other if node 313 or 329 fail. The *critical sub-path* of ds332 is 313-329-327-332. As a result, on the foregoing isolation no shortest paths can be generated. Table 34 compares the length of the generated paths for each dataset in the normal and partial disruption scenario.

Table 34: Complete Disruption Shortest Paths details

Dataset	Path No.	Normal Length	Disrupted Length	Avg. Normal Length	Avg. Disrupted Length	Gap
ds332	1	8042	None	8414.3	None	None
	2	8148	None			
	3	8318	None			
	4	9149	None			
ds662	1	327	327	345.3	385.9	40.6
	2	328	353			
	3	331	392			
	4	353	393			
	5	357	406			
	6	358	412			
	7	363	418			
ds1138	1	269	272	294.6	352.1	57.5
	2	281	281			
	3	282	282			
	4	285	285			
	5	287	290			
	6	290	351			
	7	368	705			
ds3353	1	30290	35396	31923.3	36912.8	4989.5
	2	30680	35780			
	3	30935	35811			
	4	35788	40664			

We notice that the average increase of the path length cost (represented by the gap) in the complete disruption scenario is greater than its counterpart in the normal scenario as well the partial disruption scenario. We also notice that ds3353 has the highest gap value as a result of the network topology change.

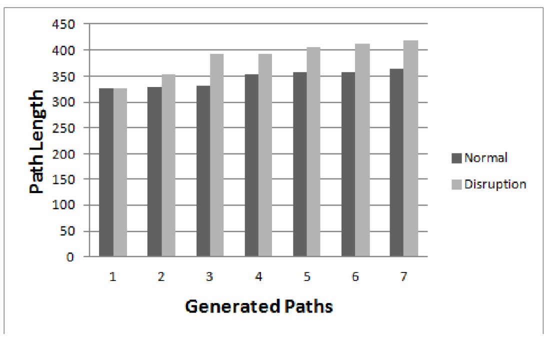
Figure 29 depicts the lengths of the generated paths in normal and the complete disruption scenarios. It is observed that in ds662 and ds1138 a subset of generated shortest paths in the disrupted scenario are almost have the same cost as those in the normal scenario except the last shortest path in ds1138 extremely has higher cost. Finally, in ds3353 all the generated 4 paths have higher cost (length) than their counterparts in the normal scenario.

Table 34 compares the gaps from normal paths length in both partial and complete disruption scenarios where gap1 is used for the partial disruption and gap2 is used for the complete disruption.

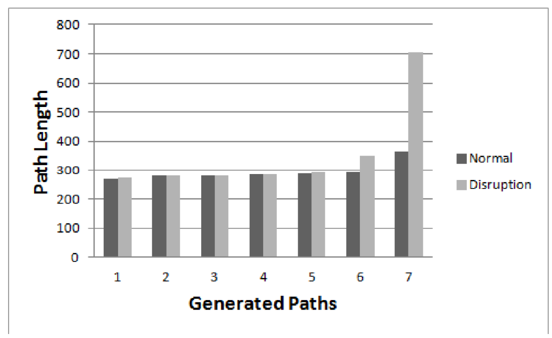
Table 35: Gap Comparison

Dataset	Gap1(†)	Gap2(†)
ds332	172.2	None
ds662	4	40.6
ds1138	0.8	57.5
ds3353	179.5	4989.5

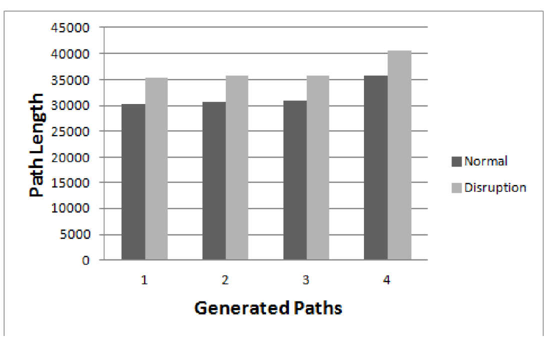
Gap1(†): Partial disruption, Gap2(†): Complete disruption.



(a) Dataset ds662



(b) Dataset ds1138



(c) Dataset ds3353

Figure 29: Complete Disruption Shortest Paths Comparison

5.5 Conclusion

In this chapter, we present a hybrid approach based on the graph partitioning and Dijkstra algorithm for the (k)-SPP in large scale TN where a subset of the connected nodes are subject to pertial or complete disruption. We address the partial overlapping (k)-SPP using a generic approach that can also be used to solve the graph partitioning problem and the shortest path problem with simple modifications. The approach is innovative and allows faster solution generation via partitioning large networks into sub-networks whereby the solution search space can be significantly reduced and as so generating k-paths will be much faster. The approach also reduce the query time when computing the shortest path between two nodes for a very large TN. The approach can generate a maximum predefined number of shortest paths between any pairs of nodes on a transportation network unless the destination cluster is isolated. We also deeply investigate the effect of partial and complete node disruption on the generated shortest paths with respect to the topology change. Our solution generation is affected by three key factors: the disruption situation (partial-complete) the disrupted nodes, failure probability value, the network size and network partitioning properties.

Chapter 6

Conclusion and future work

In this thesis, we first investigated the facility location problem under disruption where potential facilities have individual failure probabilities. We addressed three variants of such problems: reliable p-median problem (RPMP), the reliable uncapacitated facility location problem (RUFL) and the reliable capacitated facility location problem (RCFL). Second, we investigated the k-shortest path problem (k-SPP) under disruption. In k-SPP we considered that the connected nodes are subject to individual failure probabilities partially or completely, which will have an impact on the corresponding connected links in terms of road blockage or travel time delay.

In RPMP we aim to establish p facilities that are minimizing the total transportation cost for satisfying all customer demands. In RUFL we aim to minimize the total facility establishment and transportation costs associated with satisfying all customer demands. In contrast to RPMP, there is no restriction on the total number of established facilities. However, each facility has a corresponding establishment cost. Thus, in RPMP we aim to minimize the transportation cost while in RUFL our aim is to minimize the total establishment and transportation costs.

In RPMP and RUFL, we assume that the facilities have unlimited capacities. Also, if a facility fails, it is considered unavailable and if it is fortified it is considered reliable (available). Facility failures are independent and each customer is assigned to primary and backup facilities unless the primary facility is fortified (i.e., totally reliable) in which case the customer will not have to go to the backup facility. Finally, The probability of a simultaneous failure of primary and backup facilities is negligible.

To solve the RPMP and RUFL problems, we employed an evolutionary learning technique which can be customized to also address similar types of problems (e.g., PMP, UFL). The approach is innovative and allows faster solution generation by efficiently exploring the solution search space using evolutionary learning. This involves a progressive reduction of the search space by fixing the role of some nodes as customer-only after learning from successive generations of solutions obtained from an evolving solution generator template. An important aspect is the ability to select a smaller or larger number of nodes to have fixed roles in each iteration. This allows a trade-off between performance and computing time. This is a distinctive feature compared to other approaches such as Tabu Search.

We illustrated the proposed evolutionary learning technique using an instructive case study. Moreover, we demonstrated the performance via benchmark results. The proposed solution generation approach is affected by the following factors: the problem size, the number of facilities, the fortification budget and the transport network properties. With respect to RUFL instances, the solution quality is also affected when the facility count is comparatively larger relative to the total number of nodes. This stems from the fact that accurately estimating the nodes role as customer-only turns increasingly difficult with the increase in the number of facilities. A limited fortification budget that allows just partial fortification also impacts the solution quality as more extensive exploration of the solution space is needed. Conversely, when the fortification budget is too low to fortify any established facility or large enough to fortify all the established facilities, then the solution search space is reduced since there is no need to consider budget spending allocation. In such problem settings, the obtained solutions can be comparatively more competitive.

The solution quality is also affected by the transport network settings. If the solution space contains distant near-optimal solutions, the exploration potentially requires visiting more local minima in order to obtain a good quality solution. Also, a typical challenge for the proposed approach is related to probing the vast combinatorial solution search spaces of larger problems instances where the knowledge gathered by evolutionary learning may have a limited effectiveness. However, the proposed technique is relevant for problem of practical size and can be useful in situations that require quick deployment of facilities, as in the case of establishing distribution centers in situations of crisis.

For the Reliable Capacitated Facility Location (RCFL), we detailed a novel non-linear integer programming formulation where the aim is to minimize the total facility establishment and the transportation costs associated with satisfying all customer demands while respecting the capacities of the established facilities. The model assumes limited facility capacities, heterogeneous facility failure probabilities and one layer of supplier backup. The facility failure probabilities are assumed to be independent. The model also considers a finite budget for facility fortification. The facility fortification cost is considered to be location specific. We also assumed that if a facility fails, it loses a portion of its capacity and if it is fortified, the facility maintains its original capacity. Moreover, in RCFL each customer is assigned to primary and backup facilities unless the primary facility is fortified (i.e., totally reliable) in which case the customer still has a backup facility but would not need to go to. The probability of a simultaneous failure of primary and backup facilities is considered as negligible.

We presented a linearization of the proposed model and used an iterative approach based on CPLEX solver for facility establishment and customer allocation in conjunction with a C# implementation to solve the knapsack fortification budget problem. The approach optimally solves the linear integer model of RCFL. Also, the approach allows generating optimal solutions in terms of customers allocation and the fortification strategy within the available budget. Finally, the approach provides the decision makers with flexibility to hedge against different disruption scenarios and choose between different level of fortification according to the available budget. We provided key highlights of the proposed approach using an illustrative example and demonstrated the performance via benchmark results. The solution generation is affected by the following factors: the problem size, the number of facilities, the fortification budget and the transport network properties (size and network topology). The time required for generating the optimal solution for problems of small and medium size (up to 90 nodes) is faster in contrast to problems of larger size (100 nodes and more). Also, a limited fortification budget that allows partial fortification influences the solution quality since more extensive exploration of the solution space is needed. This is the reason why the solution generation time is comparatively longer for small budget values.

Finally, for the *k*-shortest path problem (*k*-SPP), we aim to find the minimum cost of *k* partial overlapping paths between any pairs of nodes in a transportation network graph. In *k*-SPP, we assume that the connected nodes and all their corresponding connected links are all failure dependent. Thus, if a connected node fails partially, all its corresponding connected links will incur extra cost (length or time) depending on the connected node failure probability. Moreover, if a connected node fails completely, it becomes unavailable as well as all its corresponding connected links.

To solve *k*-SPP, we introduced a hybrid approach based on graph partitioning and a modified version of Dijkstra algorithm to generate *k* partial overlapping paths between any pairs of nodes. We showed that the approach is scalable and useful in situations requiring quick response in the form of relevant paths to traverse across a large size transportation network where a subset of the connected nodes are subject to disruption. We addressed the partial overlapping *k*-SPP using a generic approach that can be customized to address the graph partitioning and the shortest path problems, both considered under disruption. The approach is innovative and allows faster solution generation via partitioning large size network into a number of clusters whereby the solution search space is significantly reduced. Consequently, the generation of the *k* partial overlapping paths is notably faster.

The approach can overcome the deficiency in Dijkstra algorithm and can reduce the query processing time when computing the shortest path between a pair of nodes in large transportation networks. The approach can generate a predefined number of partial overlapping shortest paths between any pairs of nodes in a transportation network. Moreover, the approach can be customized to generate all the partial overlapping shortest paths between any pairs of nodes in a transportation network. Furthermore, we provided key highlights of the proposed approach using an illustrative example and demonstrated the performance via benchmark results. Also, we analyzed the effect of partial and complete disruptions of the connected nodes on the generated shortest paths with respect to changes in the network topology. The solution generation is affected by the following factors: the disruption case (partial or complete including the disrupted connected nodes and the failure probability values), the network properties (size and topology) and the network partitioning parameters.

As future work, there are a number of interesting directions to explore including the elaboration of suitable heuristic approaches to solve the capacitated and uncapacitated maximum coverage location problems under disruption. We also envision extending the k-SPP approach to tackle the dynamic k-SPP and addressing correlated facility failure probabilities, multiple facility failures and, the duration and the frequency of the facility disruptions. Moreover, it is important to address uncertainty in travel times, amounts of returns in reverse logistics, transportation costs, demand variation and the production lead times. Furthermore, another future work direction involves addressing the multi-period formulation of the dynamic and site-specific demand variations and multi-commodity problems. Finally, other areas to explore relate to integrating the design of supply networks with other SCN problems such as inventory management, capacity expansion and vehicle routing.

Bibliography

[1] R. Agarwal, M. Caesar, P. Godfrey, and B. Y. Zhao. Shortest paths in microseconds. *arXiv preprint arXiv:1309.0874*, 2013.

[2] N. Akhtar. Social network analysis tools. In *2014 Fourth International Conference on Communication Systems and Network Technologies*, pages 388–392. IEEE, 2014.

[3] J. Alcaraz, M. Landete, and J. F. Monge. Design and analysis of hybrid metaheuristics for the reliability p-median problem. *European Journal of Operational Research*, 222(1):54 – 64, 2012.

[4] Y. An, B. Zeng, Y. Zhang, and L. Zhao. Reliable p-median facility location problem: two-stage robust models and algorithms. *Transportation Research Part B: Methodological*, 64:54–72, 2014.

[5] R. Anand and V. Kumar. Reliable back-up facility in distribution network. *Procedia computer science*, 115:312–321, 2017.

[6] R. Anand and V. G. Kumar. *Design and Development of Reliable Capacitated Facility Location for Distribution Network Design*. PhD thesis, 2017.

[7] M.-B. Aryanezhad, S. G. Jalali, and A. Jabbarzadeh. An integrated supply chain design model with random disruptions consideration. *African Journal of Business Management*, 4(12):2393–2401, 2010.

[8] M.-B. Aryanezhad, S. G. Jalali, and A. Jabbarzadeh. An integrated supply chain design model with random disruptions consideration. *African Journal of Business Management*, 4(12):2393–2401, 2010.

[9] N. Aviram and Y. Shavitt. Optimizing dijkstra for real-world performance. *arXiv preprint arXiv:1505.05033*, 2015.

[10] A. Awasthi, S. S. Chauhan, M. Parent, and J.-M. Proth. Algorithms for partitioning of large routing networks. *Journal of the Operational Research Society*, 61(7):1159–1167, 2010.

[11] A. Baghalian, S. Rezapour, and R. Z. Farahani. Robust supply chain network design with service level against disruptions and demand uncertainties: A real-life case. *European Journal of Operational Research*, 227(1):199–215, 2013.

[12] A. Bagheri, M. R. Akbarzadeh Totonchi, et al. Finding shortest path with learning algorithms. *International Journal of Artificial Intelligence*, 1, 2008.

[13] O. Berman and D. Krass. Locating multiple competitive facilities: spatial interaction models with variable expenditures. *Annals of Operations Research*, 111(1-4):197–225, 2002.

[14] O. Berman, D. Krass, and M. B. Menezes. Facility reliability issues in network p-median problems: strategic centralization and co-location effects. *Operations Research*, 55(2):332–350, 2007.

[15] C. Boonmee, M. Arimura, and T. Asada. Facility location optimization model for emergency humanitarian logistics. *International Journal of Disaster Risk Reduction*, 24:485–498, 2017.

[16] G. Bruno, A. Genovese, and C. Piccolo. Capacity management in public service facility networks: a model, computational tests and a case study. *Optimization Letters*, 10(5):975–995, 2016.

[17] P. Bühlmann and T. Hothorn. Boosting algorithms: Regularization, prediction and model fitting. *Statist. Sci.*, 22(4):477–505, 11 2007.

[18] S. E. Chang, D. Ericson, and L. Pearce. Airport closures in natural and human-induced disasters: Business vulnerability and planning. online, `http://publications.gc.ca/collections/Collection/PS4-8-2004E.pdf`, 2003.

[19] A. Chen and Z. Ji. Path finding under uncertainty. *Journal of advanced transportation*, 39(1):19–37, 2005.

[20] B. Y. Chen, W. H. Lam, and Q. Li. Efficient solution algorithm for finding spatially dependent reliable shortest path in road networks. *Journal of Advanced Transportation*, 50(7):1413–1431, 2016.

[21] B. Y. Chen, Q. Li, and W. H. Lam. Finding the k reliable shortest paths under travel time uncertainty. *Transportation Research Part B: Methodological*, 94:189–203, 2016.

[22] B. Y. Chen, C. Shi, J. Zhang, W. H. Lam, Q. Li, and S. Xiang. Most reliable path-finding algorithm for maximizing on-time arrival probability. *Transportmetrica B: Transport Dynamics*, 5(3):248–264, 2017.

[23] Q. Chen, X. Li, and Y. Ouyang. Joint inventory-location problem under the risk of probabilistic facility disruptions. *Transportation Research Part B: Methodological*, 45(7):991–1003, 2011.

[24] E. Conti, S. Cao, and A. Thomas. Disruptions in the US airport network. *arXiv preprint arXiv:1301.2223*, 2013.

[25] T. Cui, Y. Ouyang, and Z.-J. M. Shen. Reliable facility location design under the risk of disruptions. *Operations Research*, 58(4-part-1):998–1011, 2010.

[26] M. S. Daskin. *Network and discrete location: models, algorithms, and applications*. John Wiley & Sons, 2013.

[27] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[28] Z. Drezner. Heuristic solution methods for two location problems with unreliable facilities. *Journal of the Operational Research Society*, pages 509–514, 1987.

[29] Z. Drezner. Dynamic facility location: The progressive p-median problem. *Location Science*, 3(1):1–7, 1995.

[30] L. F. Escudero, P. V. Kamesam, A. J. King, and R. J. Wets. Production planning via scenario modelling. *Annals of Operations Research*, 43(6):309–335, 1993.

[31] R. Z. Farahani, N. Asgari, N. Heidari, M. Hosseininia, and M. Goh. Covering problems in facility location: A review. *Computers & Industrial Engineering*, 62(1):368 – 407, 2012.

[32] R. Z. Farahani, M. Hekmatfar, A. B. Arabani, and E. Nikbakhsh. Hub location problems: A review of models, classification, solution techniques, and applications. *Computers & Industrial Engineering*, 64(4):1096 – 1109, 2013.

[33] D. Ferone, P. Festa, and F. Guerriero. An efficient exact approach for the constrained shortest path tour problem. *Optimization Methods and Software*, pages 1–20, 2019.

[34] M. Fischetti, I. Ljubić, and M. Sinnl. Benders decomposition without separability: A computational study for capacitated facility location problems. *European Journal of Operational Research*, 253(3):557–569, 2016.

[35] D. Gade and E. Pohl. Sample average approximation applied to the capacitated-facilities location problem with unreliable facilities. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 223(4):259–269, 2009.

[36] Y. Gao. Shortest path problem with uncertain arc lengths. *Computers & Mathematics with Applications*, 62(6):2591–2600, 2011.

[37] D. Garg et al. Dynamizing dijkstra: A solution to dynamic shortest path problem through retroactive priority queue. *Journal of King Saud University-Computer and Information Sciences*, 2018.

[38] M. G. Gianni Storchi, Paolo Dell'Olmo. Directed road network of the city of rome. online, `http://users.diag.uniroma1.it/challenge9/data/rome/rome99.gr`, 1999.

[39] A. Hassanzadeh, A. Altaweel, and R. Stoleru. Traffic-and-resource-aware intrusion detection in wireless mesh networks. *Ad Hoc Networks*, 21(0):18 – 41, 2014.

[40] S. C. Ho. An iterated tabu search heuristic for the single source capacitated facility location problem. *Applied Soft Computing*, 27:169–178, 2015.

[41] L. K. Hwang. Stochastic shortest path algorithm based on lagrangian relaxation. 2010.

[42] I. I. Abdulganiev and A. A. Agafonov. Acceleration of the reliable shortest path algorithm in a time-dependent stochastic transport. 2017.

[43] A. Idri, M. Oukarfi, A. Boulmakoul, K. Zeitouni, and A. Masri. A new time-dependent shortest path algorithm for multimodal transportation network. *Procedia Computer Science*, 109:692–697, 2017.

[44] C. A. Irawan, M. Luis, S. Salhi, and A. Imran. The incorporation of fixed cost and multilevel capacities into the discrete and continuous single source capacitated facility location problem. *Annals of Operations Research*, 275(2):367–392, 2019.

[45] A. Jabbarzadeh, S. G. Jalali Naini, H. Davoudpour, and N. Azad. Designing a supply chain network under the risk of disruptions. *Mathematical Problems in Engineering*, 2012, 2012.

[46] H. Jia, Y. Xu, G. Tian, M. Zhou, J. Zhang, and H. Zhang. Random energy-efficient models for sustainable facility location subject to carbon emission, economical, capacitated and regional constraints. *IEEE Access*, 6:72757–72765, 2018.

[47] J. Jianga and L. Wub. A dynamic navigation algorithm considering network disruptions. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 4, 2014.

[48] R. Johnson. Japan's 2011 earthquake and tsunami: Food and agriculture implications. *Current Politics and Economics of Northern and Western Asia*, 20(4):651, 2011.

[49] W. Klibi and A. Martel. Scenario-based supply chain network risk modeling. *European Journal of Operational Research*, 223(3):644–658, 2012.

[50] W. Klibi, A. Martel, and A. Guitouni. The design of robust value-creating supply chain networks: a critical review. *European Journal of Operational Research*, 203(2):283–293, 2010.

[51] W. Klibi, A. Martel, and A. Guitouni. The impact of operations anticipations on the quality of stochastic location-allocation models. *Omega*, 62:19 – 33, 2016.

[52] A. Klose and A. Drexl. Facility location models for distribution system design. *European Journal of Operational Research*, 162(1):4–29, 2005.

[53] I. V. Kozlenkova, G. T. M. Hult, D. J. Lund, J. A. Mena, and P. Kekec. The role of marketing channels in supply chain management. *Journal of Retailing*, 91(4):586–609, 2015.

[54] Q. Li, B. Zeng, and A. Savachkin. Reliable facility location design under disruptions. *Computers & Operations Research*, 40(4):901–909, 2013.

[55] X. Li and Y. Ouyang. A continuum approximation approach to reliable facility location design under correlated probabilistic disruptions. *Transportation research part B: methodological*, 44(4):535–548, 2010.

[56] X. Li and Y. Ouyang. Reliable sensor deployment for network traffic surveillance. *Transportation Research Part B: Methodological*, 45(1):218 – 231, 2011.

[57] F. Liberatore, M. P. Scaparra, and M. S. Daskin. Analysis of facility protection strategies against an uncertain number of attacks: the stochastic r-interdiction median problem with fortification. *Computers & Operations Research*, 38(1):357–366, 2011.

[58] F. Liberatore, M. P. Scaparra, and M. S. Daskin. Hedging against disruptions with ripple effects in location analysis. *Omega*, 40(1):21–30, 2012.

[59] M. Lim, M. S. Daskin, A. Bassamboo, and S. Chopra. A facility reliability problem: formulation, properties, and algorithm. *Naval Research Logistics (NRL)*, 57(1):58–70, 2010.

[60] T. Liu and J. S. L. Lam. Impact of port disruption on transportation network. In *8th Annual Plastics Electronics Conference*, Germany, June 2012.

[61] Z. Liu, S. Guo, L. V. Snyder, A. Lim, and P. Peng. A p-robust capacitated network design model with facility disruptions. In *International Heinz Nixdorf Symposium*, pages 269–280. Springer, 2010.

[62] F. Louveaux. Discrete stochastic location models. *Annals of Operations research*, 6(2):21–34, 1986.

[63] A. Madadi, M. E. Kurz, S. J. Mason, and K. M. Taaffe. Supply chain network design heuristics for capacitated facilities under risk of supply disruptions. *arXiv preprint arXiv:1205.0567*, 2012.

[64] V. Mali, M. Rao, and S. Mantha. Enhanced routing in disaster management based on gis. In *International Conference on Intuitive Systems & Solutions*, pages 5–6, 2012.

[65] F. Maliki, M. Brahami, M. Dahane, and Z. Sari. A location-allocation problem design with unavailabilities management. In *44th inernational conference on comptuers & industrial engineering (CIE44&IMSS?14 proceedings), Istanbul, Turkey*, pages 14–16, 2014.

[66] K. Mouratidis and M. L. Yiu. Shortest path computation with no information leakage. *arXiv preprint arXiv:1204.6076*, 2012.

[67] J. R. O'Hanley and R. L. Church. Designing robust coverage networks to hedge against worst-case facility losses. *European Journal of Operational Research*, 209(1):23–36, 2011.

[68] L. Ozsen, C. R. Coullard, and M. S. Daskin. Capacitated warehouse location model with risk pooling. *Naval Research Logistics (NRL)*, 55(4):295–312, 2008.

[69] P. Peng, L. V. Snyder, A. Lim, and Z. Liu. Reliable logistics networks design with facility disruptions. *Transportation Research Part B: Methodological*, 45(8):1190–1211, 2011.

[70] L. Qi and M. Schneider. Realtime response of shortest path computation. In *Proceedings of the 7th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, pages 41–46. ACM, 2014.

[71] R. Rahmaniani, M. Saidi-Mehrabad, and H. Ashouri. Robust capacitated facility location problem: Optimization model and solution algorithms. *Journal of Uncertain Systems*, 7(1):22–35, 2013.

[72] C. S. Revelle, H. A. Eiselt, and M. S. Daskin. A bibliography for some fundamental problem categories in discrete location science. *European Journal of Operational Research*, 184(3):817–848, 2008.

[73] M. Rohaninejad, A. Amiri, and M. Bashiri. Heuristic methods based on minlp formulation for reliable capacitated facility location problems. *International Journal of Engineering Science (2008-4870)*, 26(3), 2015.

[74] R. A. Rossi and N. K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[75] N. A. M. Sabri, A. S. H. Basari, B. Hussin, and K. A. F. A. Samah. Dijkstra-ant colony optimization algorithm for shortest and safest evacuation in high rise building. *Jurnal Teknologi*, 79(3), 2017.

[76] J. A. Santiváñez, H. Carlo, and A. Miranda. Pricing unsatisfied demand in the reliable uncapacitated facility location problem. In *Proceedings of the 2018 Industrial and Systems Engineering Research Conference (ISERC)*, 2018.

[77] J. A. Santiváñez and H. J. Carlo. Reliable capacitated facility location problem with service levels. *EURO Journal on Transportation and Logistics*, 7(4):315–341, 2018.

[78] R. E. Schapire and Y. Freund. *Boosting: Foundations and Algorithms*. The MIT Press, 2012.

[79] D. Sever, L. Zhao, N. Dellaert, T. Van Woensel, T. de Kok, et al. Dynamic shortest path problem with travel-time-dependent stochastic disruptions: hybrid approximate dynamic programming algorithms with a clustering approach. *Beta Res. School Oper. Manage. Logistics, Eindhoven Univ. Technol., Eindhoven, The Netherlands, Tech. Rep*, 423, 2013.

[80] S. M. Shavarani, M. G. Nejad, F. Rismanchian, and G. Izbirak. Application of hierarchical facility location problem for optimization of a drone delivery system: a case study of amazon prime air in the city of san francisco. *The International Journal of Advanced Manufacturing Technology*, 95(9-12):3141–3153, 2018.

[81] Z.-J. M. Shen, R. L. Zhan, and J. Zhang. The reliable facility location problem: Formulations, heuristics, and approximation algorithms. *INFORMS Journal on Computing*, 23(3):470–482, 2011.

[82] D. Shishebori and M. S. Jabalameli. A new integrated mathematical model for optimizing facility location and network design policies with facility disruptions. *Life Science Journal*, 10(1):1896–1906, 2013.

[83] L. V. Snyder, Z. Atan, P. Peng, Y. Rong, A. J. Schmitt, and B. Sinsoysal. Or/ms models for supply chain disruptions: A review. *IIE Transactions*, pages 1–21, 2015.

[84] L. V. Snyder and M. S. Daskin. Reliability models for facility location: the expected failure cost case. *Transportation Science*, 39(3):400–416, 2005.

[85] A. D. Sonmez and G. J. Lim. A decomposition approach for facility location and relocation problem with uncertain number of future facilities. *European Journal of Operational Research*, 218(2):327–338, 2012.

[86] P. N. Thanh, N. Bostel, and O. Péton. A dynamic model for facility location in the design of complex supply chains. *International Journal of Production Economics*, 113(2):678–693, 2008.

[87] L. Turner. Variants of shortest path problems. *Algorithmic Operations Research*, 6:91–104, 2011.

[88] Z. Ulukan and E. Demircioğlu. A survey of discrete facility location problems. *World Academy of Science, Engineering and Technology, International Journal of Social, Behavioral, Educational, Economic, Business and Industrial Engineering*, 9(7):2450–2455, 2015.

[89] D. Wagner and T. Willhalm. Geometric speed-up techniques for finding shortest paths in large sparse graphs. In *European Symposium on Algorithms*, pages 776–787. Springer, 2003.

[90] Q. Wang, R. Batta, J. Bhadury, and C. M. Rump. Budget constrained location problem with opening and closing of facilities. *Computers & Operations Research*, 30(13):2047 – 2069, 2003.

[91] M. Watson. *Supply chain network design: applying optimization and analytics to the global supply chain*. Pearson Education, 2013.

[92] S. Wøhlk and G. Laporte. A fast heuristic for large-scale capacitated arc routing problems. *Journal of the Operational Research Society*, 69(12):1877–1887, 2018.

[93] L. Wu, X. Xiao, D. Deng, G. Cong, A. D. Zhu, and S. Zhou. Shortest path and distance queries on road networks: An experimental evaluation. *Proceedings of the VLDB Endowment*, 5(5):406–417, 2012.

[94] K. Xie, K. Deng, S. Shang, X. Zhou, and K. Zheng. Finding alternative shortest paths in spatial networks. *ACM Transactions on Database Systems (TODS)*, 37(4):29, 2012.

[95] N. Zarrinpoor, M. S. Fallahnezhad, and M. S. Pishvaee. Reliable location-allocation model for congested systems under disruptions using accelerated benders' decomposition. 2016.

[96] S. Zhang and X. Liu. A new algorithm for finding the k shortest transport paths in dynamic stochastic networks. *Journal of Vibroengineering*, 15(2), 2013.

[97] Y. Zhang and A. Khani. An algorithm for reliable shortest path problem with travel time correlations. *Transportation Research Part B: Methodological*, 121:92–113, 2019.

[98] Y. Zhang, S. Song, Z.-J. M. Shen, and C. Wu. Robust shortest path problem with distributional uncertainty. *IEEE transactions on intelligent transportation systems*, 19(4):1080–1090, 2018.

[99] X. Zhao, X. Xia, L. Wang, and G. Yu. Risk-averse facility location for green closed-loop supply chain networks design under uncertainty. *Sustainability*, 10(11):4072, 2018.

[100] S. Zhu and D. M. Levinson. Disruptions to transportation networks: a review. In *Network Reliability in Practice*, pages 5–20. Springer, 2012.

Appendix A

First Appendix

Table A.1: Detailed Solution of 30-nodes RCFL Problem Dataset

30-nodes	No.(†)	Cost	Bu(†)	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost
		2014	0	2041.8	30	2018.6	60	2.17.5	120	2017.2	180	2015.5	240	2014
Customer	P(†)	B(†)	P	B	P	B	P	B	P	B	P	B	P	B
1	5	3	1	5	5	3	5	2	5	8	5	8	5	3
2	2	3	1	2	2	3	2	3	2	3	2	5	2	3
3	9	3	2	1	9	2	9	2	9	3	9	2	9	3
4	5	8	5	4	5	3	5	3	5	4	5	8	5	8
5	8	7	8	10	8	2	8	2	8	2	8	2	8	7
6	8	4	7	5	8	5	8	5	8	5	8	2	8	4
7	7	2	9	2	7	9	7	9	7	9	7	2	7	2
8	3	4	7	10	3	5	3	2	3	9	3	5	3	4
9	3	4	5	1	3	5	3	2	3	5	3	5	3	4
10	9	8	9	10	9	5	9	5	9	3	9	3	9	8
11	4	9	10	4	4	5	4	5	4	5	4	5	4	9
12	4	2	4	2	4	2	4	2	4	9	4	7	4	2
13	2	8	2	1	2	5	2	3	2	5	2	9	2	8
14	5	4	5	1	5	3	5	3	5	4	5	9	5	4
15	2	7	2	4	2	3	2	5	2	3	2	8	2	7
16	5	9	4	8	5	3	5	2	5	4	5	9	5	9
17	7	8	8	2	7	8	7	8	7	8	7	5	7	8
18	7	2	7	9	7	5	7	5	7	9	7	5	7	2
19	3	4	1	4	3	5	3	2	3	4	3	2	3	4
20	9	8	9	4	9	3	9	3	9	4	9	8	9	8

P(†): Primary Facility, B(†): backup Facility, Bu(†): Fortification Budget, No.(†): Normal Scenario Solution Cost

Table A.2: Detailed Solution of 45 Nodes RCFL Problem Dataset Instances

45 Nodes	No.	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost
		4366	0	4398.2	30	4373.1	60	4368	120	4366.3	180	4366
Customer	P	B	P	B	P	B	P	B	P	B	P	B
1	11	2	9	6	11	14	11	14	11	7	11	2
2	7	11	7	14	7	14	7	14	7	6	7	11
3	14	7	14	2	14	5	14	6	14	5	14	7
4	11	14	2	6	11	14	11	14	11	7	11	14
5	7	15	7	14	7	14	7	14	7	5	7	15
6	6	2	6	14	6	5	6	14	6	7	6	2
7	6	2	9	6	6	5	6	14	6	7	6	2
8	14	7	6	2	14	5	14	6	14	6	14	7
9	15	11	14	6	15	14	15	14	15	11	15	11
10	7	2	7	5	7	5	7	5	7	5	7	2
11	15	14	5	6	15	5	15	5	15	6	15	14
12	15	6	2	5	15	5	15	5	15	6	15	6
13	14	5	9	6	14	5	14	5	14	6	14	5
14	11	2	11	6	11	14	11	14	11	6	11	2
15	7	15	7	6	7	5	7	6	7	11	7	15
16	7	11	7	6	7	14	7	14	7	11	7	11
17	11	15	11	14	11	14	11	14	11	7	11	15
18	7	6	7	6	7	14	7	6	7	15	7	6
19	14	2	14	5	14	5	14	6	14	5	14	2
20	7	15	7	6	7	5	7	5	7	6	7	15
21	14	5	14	6	14	5	14	6	14	7	14	5
22	11	14	9	5	11	5	11	5	11	6	11	14
23	6	2	6	9	6	5	6	14	6	7	6	2
24	6	2	6	2	6	5	6	5	6	15	6	2
25	2	6	14	2	2	14	2	14	2	15	2	6
26	14	6	14	6	14	5	14	6	14	5	14	6
27	5	15	6	5	5	14	5	14	5	15	5	15
28	6	2	9	6	6	14	6	14	6	11	6	2
29	2	6	11	6	2	5	2	6	2	11	2	6
30	6	14	11	6	6	5	6	5	6	5	6	14

Table A.3: Detailed Solution of 60 Nodes RCFL Problem Dataset

60 Nodes	No.	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost
		15607	0	15632	30	15619	60	15614	120	15610	180	15609	240	15607
Customer	P	B	P	B	P	B	P	B	P	B	P	B	P	B
1	14	1	14	6	14	13	14	12	14	12	14	17	14	1
2	1	12	1	13	1	5	1	5	1	16	1	5	1	12
3	9	16	9	16	9	5	9	5	9	5	9	1	9	16
4	20	1	20	17	20	16	20	9	20	9	20	16	20	1
5	9	3	20	16	9	20	9	12	9	13	9	1	9	3
6	20	1	20	6	20	13	20	13	20	13	20	1	20	1
7	3	6	3	13	3	13	3	13	3	1	3	1	3	6
8	17	3	17	16	17	16	17	16	17	16	17	5	17	3
9	12	20	12	6	12	17	12	13	12	1	12	13	12	20
10	1	17	1	6	1	7	1	7	1	5	1	16	1	17
11	12	20	12	16	12	16	12	5	12	1	12	16	12	20
12	16	12	16	13	16	5	16	5	16	1	16	1	16	12
13	9	3	9	16	9	16	9	16	9	5	9	12	9	3
14	20	3	9	6	20	9	20	12	20	12	20	6	20	3
15	6	3	9	1	6	16	6	12	6	1	6	9	6	3
16	5	17	5	1	5	13	5	12	5	12	5	16	5	17
17	9	1	9	16	9	16	9	16	9	5	9	1	9	1
18	6	3	6	13	6	5	6	16	6	20	6	9	6	3
19	12	1	12	13	12	5	12	5	12	9	12	9	12	1
20	14	20	14	13	14	20	14	20	14	9	14	9	14	20

Table A.4: Detailed Solution of 60 Nodes RCFL Problem Dataset (Cont..)

60 Nodes	No.	Cost 15607	Bu 0	Cost 15632	Bu 30	Cost 15619	Bu 60	Cost 15614	Bu 120	Cost 15610	Bu 180	Cost 15609	Bu 240	Cost 15607
Customer	P	B	P	B	P	B	P	B	P	B	P	B	P	B
21	9	3	7	6	9	5	9	5	9	1	9	6	9	3
22	1	20	3	1	1	5	1	5	1	13	1	20	1	20
23	3	5	1	16	3	16	3	12	3	12	3	12	3	5
24	1	5	1	13	1	13	1	12	1	12	1	17	1	5
25	20	3	20	6	20	16	20	16	20	9	20	13	20	3
26	5	1	5	13	5	16	5	16	5	1	5	6	5	1
27	7	5	7	16	7	16	7	12	7	12	7	12	7	5
28	7	1	3	1	7	17	7	17	7	17	7	17	7	1
29	13	20	13	6	13	5	13	16	13	1	13	16	13	20
30	12	1	12	16	12	16	12	5	12	1	12	1	12	1
31	5	3	5	1	5	16	5	12	5	16	5	20	5	3
32	20	1	20	6	20	16	20	12	20	12	20	1	20	1
33	14	1	14	17	14	5	14	17	14	17	14	17	14	1
34	3	7	7	6	3	5	3	16	3	7	3	7	3	7
35	7	6	7	1	7	5	7	12	7	1	7	1	7	6
36	1	5	1	16	1	5	1	5	1	16	1	13	1	5
37	7	1	6	13	7	5	7	5	7	5	7	6	7	1
38	16	5	16	13	16	5	16	5	16	12	16	6	16	5
39	3	16	3	13	3	13	3	13	3	13	3	13	3	16
40	7	1	6	13	7	5	7	5	7	5	7	6	7	1

Table A.5: Detailed Solution of 70 Nodes RCFL Problem Dataset Instances

70 Nodes	No.	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost
		4448	0	4499.2	30	4454	60	4453.8	120	4451.6	180	4449.5	240	4449	300	4448
Customer	P	B	P	B	P	B	P	B	P	B	P	B	P	B	P	B
1	11	1	11	6	11	6	11	5	11	6	11	14	11	1	11	1
2	4	11	4	19	4	3	4	19	4	3	4	19	4	19	4	11
3	20	14	20	6	20	2	20	1	20	1	20	19	20	1	20	14
4	8	19	3	6	8	1	8	1	8	1	8	1	8	3	8	19
5	20	8	16	6	20	2	20	1	20	1	20	1	20	5	20	8
6	14	2	14	6	14	1	14	1	14	1	14	6	14	3	14	2
7	8	1	8	3	8	20	8	20	8	20	8	1	8	2	8	1
8	3	8	8	3	3	1	3	2	3	20	3	20	3	1	3	8
9	20	1	20	19	20	3	20	5	20	3	20	6	20	2	20	1
10	16	3	16	6	16	1	16	19	16	2	16	5	16	2	16	3
11	1	8	4	6	1	3	1	5	1	2	1	19	1	6	1	8
12	16	2	16	6	16	3	16	1	16	3	16	3	16	2	16	2
13	4	2	3	19	4	20	4	20	4	20	4	20	4	20	4	2
14	1	3	3	5	1	3	1	5	1	20	1	3	1	16	1	3
15	5	1	5	16	5	2	5	20	5	3	5	16	5	19	5	1
16	8	4	8	19	8	16	8	19	8	16	8	1	8	19	8	4
17	8	14	5	11	8	11	8	11	8	11	8	1	8	2	8	14
18	2	4	5	16	2	5	2	5	2	1	2	5	2	3	2	4
19	11	5	11	5	11	2	11	2	11	2	11	2	11	1	11	5
20	16	3	1	6	16	6	16	19	16	1	16	3	16	19	16	3
21	3	2	20	6	3	5	3	2	3	2	3	20	3	1	3	2
22	5	3	1	5	5	3	5	1	5	3	5	6	5	20	5	3
23	5	3	16	6	5	3	5	1	5	3	5	8	5	6	5	3
24	19	6	19	6	19	6	19	1	19	6	19	3	19	3	19	6
25	1	2	1	19	1	3	1	19	1	3	1	3	1	3	1	2

Table A.6: Detailed Solution of 70 Nodes RCFL Problem Dataset Instances(Cont..)

70 Nodes	No.	Cost 4448	Bu 0	Cost 4499.2	Bu 30	Cost 4454	Bu 60	Cost 4453.8	Bu 120	Cost 4451.6	Bu 180	Cost 4449.5	Bu 240	Cost 4449	Bu 300	Cost 4448
Customer	P	B	P	B	P	B	P	B	P	B	P	B	P	B	P	B
26	3	6	3	4	3	6	3	20	3	20	3	20	3	1	3	6
27	19	14	6	16	19	6	19	5	19	6	19	8	19	5	19	14
28	16	11	20	16	16	20	16	20	16	1	16	19	16	3	16	11
29	19	3	19	6	19	5	19	1	19	6	19	3	19	2	19	3
30	14	1	14	16	14	16	14	16	14	16	14	3	14	11	14	1
31	1	16	1	6	1	3	1	19	1	3	1	3	1	3	1	16
32	4	2	4	16	4	2	4	2	4	2	4	2	4	2	4	2
33	1	2	1	6	1	3	1	19	1	3	1	3	1	3	1	2
34	1	11	1	6	1	5	1	19	1	3	1	3	1	20	1	11
35	4	2	4	16	4	2	4	2	4	2	4	2	4	2	4	2
36	14	19	14	19	14	6	14	19	14	6	14	5	14	1	14	19
37	1	2	1	6	1	3	1	19	1	3	1	3	1	3	1	2
38	16	4	4	16	16	3	16	5	16	3	16	5	16	5	16	4
39	8	1	8	16	8	20	8	20	8	20	8	3	8	1	8	1
40	8	1	8	3	8	3	8	1	8	3	8	1	8	19	8	1
41	20	14	2	6	20	2	20	1	20	3	20	19	20	1	20	14
42	4	1	16	6	4	16	4	16	4	16	4	11	4	11	4	1
43	8	1	8	6	8	6	8	16	8	6	8	6	8	2	8	1
44	6	16	6	19	6	3	6	19	6	1	6	3	6	1	6	16
45	14	4	14	16	14	16	14	16	14	16	14	6	14	5	14	4
46	3	1	3	16	3	5	3	16	3	1	3	14	3	6	3	1
47	6	1	8	6	6	1	6	20	6	16	6	1	6	8	6	1
48	16	20	16	6	16	3	16	19	16	2	16	3	16	1	16	20
49	8	2	2	19	8	20	8	19	8	20	8	5	8	19	8	2
50	3	4	19	6	3	1	3	19	3	5	3	1	3	2	3	4

Table A.7: Detailed Solution of 90 Nodes RCFL Problem Dataset

90 Nodes	No.	Cost 4701	Bu 0	Cost 4723.2	Bu 30	Cost 4721.9	Bu 60	Cost 4719.3	Bu 120	Cost 4705.9	Bu 180	Cost 4701
Customer	P	B	P	B	P	B	P	B	P	B	P	B
1	23	6	23	24	23	24	23	24	23	21	23	6
2	28	11	28	21	28	21	28	21	28	6	28	11
3	16	6	16	21	16	21	16	24	16	23	16	6
4	23	28	23	16	23	16	23	16	23	6	23	28
5	17	11	17	23	17	23	17	23	17	23	17	11
6	24	21	24	16	24	6	24	21	24	28	24	21
7	6	11	6	24	6	24	6	21	6	21	6	11
8	28	6	28	21	28	16	28	16	28	6	28	6
9	21	6	21	24	21	16	21	6	21	28	21	6
10	17	23	17	16	17	16	17	16	17	16	17	23
11	17	6	17	16	17	16	17	16	17	16	17	6
12	11	6	11	16	11	16	11	16	11	28	11	6
13	16	6	16	6	16	6	16	6	16	6	16	6
14	21	6	21	16	21	16	21	6	21	28	21	6
15	6	21	6	21	6	24	6	21	6	21	6	21
16	24	17	24	6	24	6	24	6	24	6	24	17
17	17	23	17	16	17	16	17	16	17	16	17	23
18	21	6	21	23	21	23	21	16	21	6	21	6
19	21	11	21	16	21	16	21	6	21	6	21	11
20	6	23	6	21	6	24	6	21	6	28	6	23
21	6	11	6	24	6	24	6	21	6	16	6	11
22	24	28	24	21	24	6	24	21	24	21	24	28
23	21	6	21	16	21	16	21	24	21	6	21	6
24	23	11	23	16	23	16	23	16	23	16	23	11
25	23	17	23	24	23	24	23	24	23	6	23	17
26	21	6	21	16	21	16	21	6	21	6	21	6
27	28	11	28	21	28	21	28	21	28	6	28	11
28	16	6	16	24	16	24	16	6	16	6	16	6
29	6	28	6	24	6	24	6	21	6	21	6	28
30	16	23	16	6	16	6	16	24	16	21	16	23

Table A.8: Detailed Solution of 90 Nodes RCFL Problem Dataset(Cont..)

90 Nodes	No.	Cost 4701	Bu 0	Cost 4723.2	Bu 30	Cost 4721.9	Bu 60	Cost 4719.3	Bu 120	Cost 4705.9	Bu 180	Cost 4701
Customer	P	B	P	B	P	B	P	B	P	B	P	B
31	28	21	28	6	28	6	28	6	28	24	28	21
32	6	21	6	21	6	24	6	21	6	28	6	21
33	11	6	11	24	11	24	11	24	11	24	11	6
34	6	11	6	23	6	21	6	21	6	21	6	11
35	24	11	24	16	24	6	24	6	24	28	24	11
36	6	11	6	21	6	24	6	24	6	21	6	11
37	23	11	23	6	23	6	23	6	23	21	23	11
38	6	17	6	21	6	24	6	16	6	23	6	17
39	11	6	11	23	11	23	11	23	11	23	11	6
40	28	6	28	24	28	24	28	24	28	24	28	6
41	21	6	21	23	21	23	21	6	21	28	21	6
42	23	6	23	24	23	16	23	16	23	21	23	6
43	23	11	23	6	23	6	23	6	23	24	23	11
44	17	6	17	6	17	6	17	6	17	6	17	6
45	24	16	24	16	24	6	24	6	24	6	24	16
46	21	6	21	24	21	24	21	6	21	6	21	6
47	28	6	28	24	28	24	28	24	28	24	28	6
48	21	6	21	6	21	6	21	6	21	6	21	6
49	6	17	6	24	6	24	6	21	6	21	6	17
50	23	6	23	6	23	6	23	6	23	28	23	6
51	28	23	28	24	28	24	28	24	28	6	28	23
52	6	17	6	23	6	24	6	24	6	21	6	17
53	23	6	23	6	23	6	23	6	23	28	23	6
54	17	21	17	23	17	23	17	23	17	23	17	21
55	11	21	11	16	11	16	11	16	11	16	11	21
56	11	6	11	24	11	16	11	16	11	16	11	6
57	6	11	6	16	6	24	6	21	6	28	6	11
58	17	23	17	24	17	24	17	24	17	28	17	23
59	23	11	23	21	23	21	23	21	23	6	23	11
60	21	11	21	16	21	16	21	6	21	23	21	11

Table A.9: Detailed Solution of 105 Nodes RCFL Problem Dataset

105 Nodes	No.	Cost 7887	Bu 0	Cost 7922.3	Bu 30	Cost 7913.6	Bu 60	Cost 7911.3	Bu 120	Cost 7900.1	Bu 180	Cost 7891.8	Bu 240	Cost 7887
Customer	P	B	P	B	P	B	P	B	P	B	P	B	P	B
1	2	15	2	24	2	24	2	24	2	10	2	8	2	15
2	15	14	15	2	15	8	15	8	15	8	15	8	15	14
3	13	8	13	3	22	13	22	13	22	3	22	13	13	8
4	3	10	3	2	3	22	3	10	3	2	3	10	3	10
5	15	14	15	24	15	24	15	24	15	24	15	14	15	14
6	10	13	10	13	10	24	10	3	10	3	10	22	10	13
7	22	3	22	13	22	13	22	24	22	2	22	2	22	3
8	13	2	13	10	13	10	13	10	13	2	13	22	13	2
9	15	3	15	24	15	24	15	24	15	2	15	14	15	3
10	13	2	13	10	13	10	13	3	13	2	13	10	13	2
11	15	13	15	13	15	13	13	15	13	10	13	14	15	13
12	13	3	13	3	13	3	13	3	13	2	13	2	13	3
13	3	10	3	24	14	3	14	3	14	3	14	2	3	10
14	13	3	13	3	13	15	13	15	13	2	13	2	13	3
15	14	3	14	10	14	10	14	10	14	2	14	2	14	3
16	22	15	22	13	13	22	22	8	22	24	22	2	22	15
17	24	13	24	13	24	10	24	10	24	13	24	8	24	13
18	10	3	10	3	10	24	10	3	10	2	10	8	10	3
19	24	10	24	10	24	10	24	10	24	10	24	8	24	10
20	3	2	3	24	15	3	15	3	15	3	15	3	3	2
21	14	2	14	3	8	3	8	3	8	3	8	22	14	2
22	13	14	13	3	13	8	13	8	13	24	13	8	13	14
23	10	2	10	22	10	24	10	3	10	2	10	2	10	2
24	2	22	2	10	2	10	2	10	2	3	2	13	2	22
25	10	15	10	3	10	24	10	3	10	13	10	2	10	15

Table A.10: Detailed Solution of 105 Nodes RCFL Problem Dataset(Cont..)

105 Nodes	No.	Cost 7887	Bu 0	Cost 7922.3	Bu 30	Cost 7913.6	Bu 60	Cost 7911.3	Bu 120	Cost 7900.1	Bu 180	Cost 7891.8	Bu 240	Cost 7887
Customer	P	B	P	B	P	B	P	B	P	B	P	B	P	B
26	3	8	3	24	2	3	3	10	3	2	3	10	3	8
27	24	10	24	10	24	10	24	10	24	2	24	10	24	10
28	2	14	2	10	10	24	10	3	10	13	10	13	2	14
29	24	15	24	13	13	24	13	24	13	10	13	2	24	15
30	13	3	13	2	13	10	13	2	13	2	13	2	13	3
31	13	3	13	3	13	3	13	2	13	2	13	2	13	3
32	15	3	15	24	15	24	15	24	15	2	15	2	15	3
33	24	2	24	10	24	10	24	3	24	2	24	13	24	2
34	22	3	22	3	22	3	22	3	22	2	22	2	22	3
35	8	3	8	24	8	15	8	15	8	15	8	3	8	3
36	3	8	3	10	3	10	3	10	3	2	3	14	3	8
37	22	15	22	10	22	8	22	8	22	13	22	10	22	15
38	2	3	2	24	2	15	2	15	2	3	2	14	2	3
39	24	14	24	2	24	10	24	10	24	2	24	14	24	14
40	22	13	22	10	2	10	2	10	2	3	2	3	22	13
41	8	10	8	24	24	10	24	3	24	3	24	10	8	10
42	8	10	8	24	8	24	8	24	8	24	8	22	8	10
43	8	3	8	22	8	15	8	15	8	15	8	22	8	3
44	8	3	8	3	8	3	8	3	8	3	8	3	8	3
45	15	3	15	3	3	15	15	3	15	2	15	2	15	3
46	3	10	3	13	3	13	3	10	3	2	3	2	3	10
47	10	8	10	22	10	24	10	3	10	2	10	2	10	8
48	15	3	15	24	15	24	15	24	15	24	15	8	15	3
49	14	2	14	10	14	10	14	10	14	10	14	10	14	2
50	2	13	2	24	2	8	2	22	2	10	2	10	2	13

Table A.11: Detailed Solution of 105 Nodes RCFL Problem Dataset(Cont..)

105 Nodes	No.	Cost 7887	Bu 0	Cost 7922.3	Bu 30	Cost 7913.6	Bu 60	Cost 7911.3	Bu 120	Cost 7900.1	Bu 180	Cost 7891.8	Bu 240	Cost 7887
Customer	P	B	P	B	P	B	P	B	P	B	P	B	P	B
51	10	3	10	13	10	24	10	3	10	2	10	2	10	3
52	14	22	14	13	14	13	14	8	14	8	14	3	14	22
53	24	13	24	3	24	10	24	10	24	3	24	10	24	13
54	14	15	14	3	14	3	14	3	14	3	14	10	14	15
55	24	15	24	13	24	10	24	3	24	13	24	13	24	15
56	22	3	22	3	22	3	22	3	22	10	22	10	22	3
57	14	3	14	10	14	10	14	10	14	10	14	22	14	3
58	14	3	14	2	14	2	14	2	14	2	14	2	14	3
59	10	3	10	24	10	24	10	3	10	13	10	24	10	3
60	22	3	22	13	22	13	22	13	22	2	22	2	22	3
61	8	3	8	2	8	2	8	2	8	2	8	10	8	3
62	22	3	22	3	22	3	22	3	22	2	22	2	22	3
63	3	8	3	10	3	10	3	10	3	2	3	2	3	8
64	3	2	3	13	3	13	3	10	3	2	3	10	3	2
65	13	2	13	3	3	13	3	10	3	10	3	2	13	2
66	10	3	10	24	24	10	24	3	24	2	24	2	10	3
67	3	10	3	22	3	22	3	10	3	10	3	10	3	10
68	14	3	14	13	14	13	14	10	14	13	14	2	14	3
69	22	3	22	10	22	10	22	10	22	2	22	2	22	3
70	2	22	2	22	2	22	2	22	2	3	2	8	2	22
71	8	2	8	24	8	24	8	24	8	24	8	2	8	2
72	22	3	22	13	22	13	22	13	22	2	22	2	22	3
73	13	3	13	10	13	15	13	15	13	2	13	3	13	3
74	2	22	2	22	22	15	2	22	2	10	2	22	2	22
75	8	3	8	24	8	15	8	15	8	15	8	22	8	3

Table A.12: Detailed Solution of 120 Nodes RCFL Problem Dataset

120 Nodes	No.	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost	Bu	Cost
		5937	0	6037.3	30	5978.6	60	5963.4	120	5941.4	180	5937
Customer	P	B	P	B	P	B	P	B	P	B	P	B
1	22	13	22	18	22	19	22	13	22	13	22	13
2	18	22	18	10	18	19	18	19	18	14	18	22
3	20	13	20	14	20	14	20	13	20	13	20	13
4	20	22	20	14	20	13	20	13	20	13	20	22
5	18	13	18	14	18	19	18	19	18	14	18	13
6	18	19	18	20	18	19	18	20	18	20	18	19
7	13	18	13	18	13	19	13	20	13	20	13	18
8	17	19	13	14	17	13	17	13	17	20	17	19
9	13	22	10	13	13	19	13	20	13	14	13	22
10	19	13	20	13	19	13	19	13	19	13	19	13
11	13	20	13	18	13	19	13	20	13	17	13	20
12	22	18	10	9	22	13	22	20	22	17	22	18
13	13	20	22	13	13	19	13	20	13	22	13	20
14	17	22	13	14	17	13	17	13	17	13	17	22
15	17	22	18	14	17	14	17	14	17	19	17	22
16	13	18	13	9	13	19	13	20	13	20	13	18
17	17	22	20	9	17	19	17	20	17	14	17	22
18	19	17	10	14	19	13	19	13	19	17	19	17
19	13	18	13	14	13	19	13	19	13	22	13	18
20	18	13	18	13	18	13	18	13	18	17	18	13
21	22	13	22	9	22	13	22	13	22	13	22	13
22	18	13	18	9	18	22	18	22	18	22	18	13
23	17	13	13	18	17	13	17	13	17	20	17	13
24	22	20	22	13	22	13	22	13	22	13	22	20
25	20	13	10	13	20	13	20	19	20	17	20	13
26	20	13	20	9	20	19	20	13	20	13	20	13
27	18	13	18	10	18	22	18	22	18	17	18	13
28	18	13	18	14	18	19	18	19	18	19	18	13
29	13	22	13	14	13	19	13	20	13	17	13	22
30	13	17	9	13	13	19	13	19	13	14	13	17

Table A.13: Detailed Solution of 120 Nodes RCFL Problem Dataset (Cont..)

120 Nodes	No.	Cost 5937	Bu 0	Cost 6037.3	Bu 30	Cost 5978.6	Bu 60	Cost 5963.4	Bu 120	Cost 5941.4	Bu 180	Cost 5937
Customer	P	B	P	B	P	B	P	B	P	B	P	B
31	20	22	20	13	20	19	20	13	20	22	20	22
32	14	20	14	13	14	13	14	13	14	13	14	20
33	22	13	22	14	22	19	22	19	22	13	22	13
34	19	17	9	22	19	13	19	13	19	13	19	17
35	13	18	13	18	13	19	13	19	13	17	13	18
36	18	13	18	9	18	14	18	14	18	17	18	13
37	13	18	13	9	13	19	13	20	13	20	13	18
38	19	13	18	9	19	13	19	13	19	17	19	13
39	18	13	18	10	18	13	18	20	18	20	18	13
40	13	18	13	22	13	19	13	19	13	14	13	18
41	22	17	10	22	22	13	22	13	22	13	22	17
42	18	13	18	13	18	13	18	20	18	17	18	13
43	18	13	9	14	18	19	18	19	18	14	18	13
44	18	20	10	18	18	19	18	22	18	22	18	20
45	20	18	9	20	20	19	20	13	20	13	20	18
46	17	14	10	22	17	19	17	19	17	22	17	14
47	17	22	18	14	17	19	17	19	17	13	17	22
48	20	22	20	9	20	13	20	13	20	13	20	22
49	22	13	22	9	22	13	22	13	22	13	22	13
50	20	13	20	14	20	14	20	19	20	13	20	13
51	19	17	22	9	19	13	19	13	19	13	19	17
52	14	19	14	9	14	18	14	18	14	13	14	19
53	18	13	18	14	18	14	18	14	18	14	18	13
54	19	13	22	14	19	13	19	20	19	22	19	13
55	14	17	14	22	14	22	14	20	14	13	14	17
56	19	22	22	13	19	13	19	20	19	13	19	22
57	19	22	9	14	19	13	19	20	19	17	19	22
58	14	17	14	9	14	13	14	13	14	13	14	17
59	17	14	20	13	17	13	17	20	17	20	17	14
60	19	18	9	20	19	13	19	13	19	13	19	18

Table A.14: Detailed Solution of 120 Nodes RCFL Problem Dataset (Cont..)

120 Nodes	No.	Cost 5937	Bu 0	Cost 6037.3	Bu 30	Cost 5978.6	Bu 60	Cost 5963.4	Bu 120	Cost 5941.4	Bu 180	Cost 5937
Customer	P	B	P	B	P	B	P	B	P	B	P	B
61	19	17	10	13	19	13	19	20	19	13	19	17
62	14	13	14	20	14	19	14	20	14	20	14	13
63	18	13	18	20	18	19	18	20	18	17	18	13
64	20	18	20	22	20	19	20	13	20	13	20	18
65	22	17	22	9	22	13	22	13	22	17	22	17
66	22	17	10	18	22	13	22	20	22	17	22	17
67	20	18	9	20	20	19	20	19	20	13	20	18
68	18	22	13	18	18	13	18	13	18	13	18	22
69	22	20	10	22	22	19	22	19	22	19	22	20
70	20	13	20	22	20	19	20	13	20	13	20	13
71	22	14	22	9	22	13	22	13	22	13	22	14
72	14	17	9	14	14	13	14	13	14	13	14	17
73	19	22	10	9	19	13	19	13	19	17	19	22
74	20	22	20	9	20	13	20	13	20	13	20	22
75	13	17	9	13	13	19	13	19	13	17	13	17
76	18	20	14	18	18	14	18	14	18	14	18	20
77	19	13	20	18	19	13	19	13	19	13	19	13
78	20	13	20	14	20	13	20	13	20	13	20	13
79	17	19	9	14	17	13	17	14	17	19	17	19
80	20	19	9	22	20	13	20	19	20	13	20	19
81	22	13	22	13	22	19	22	19	22	20	22	13
82	22	13	22	9	22	13	22	13	22	13	22	13
83	17	22	20	13	17	19	17	20	17	22	17	22
84	20	19	9	20	20	13	20	13	20	22	20	19
85	20	13	20	9	20	13	20	13	20	13	20	13
86	17	13	18	14	17	19	17	14	17	13	17	13
87	13	22	13	20	13	19	13	19	13	14	13	22
88	19	18	9	22	19	13	19	13	19	17	19	18
89	20	22	9	20	20	13	20	13	20	13	20	22
90	14	22	10	9	14	13	14	13	14	13	14	22