Logistics Optimization in Mobile Aid Provision to En Route Refugees

by

O. Baturhan Bayraktar

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Industrial Engineering

KOÇ ÜNİVERSİTESİ

July 24, 2020

Logistics Optimization in Mobile Aid Provision to En Route Refugees

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

O. Baturhan Bayraktar

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by the final

examining committee have been made.

Committee Members:

____

Prof. F. Sibel Salman

____

Prof. Güvenç Şahin

____

Prof. Ceyda Oğuz

____

Assist. Prof. Dilek Günneç

____

Assist. Prof. Eda Yücel

Date: ______________________________

To my family

ABSTRACT

Logistics Optimization in Mobile Aid Provision to En Route Refugees
O. Baturhan Bayraktar
Master of Science in Industrial Engineering
July 24, 2020

As one of the harshest humanitarian problems in the world, displacement has caused more than 79.5 million people to migrate as of June 2020 according to UNHCR. In this thesis, the aim is to provide an efficient, dynamic, and effective solution approach for the optimization of a logistics problem related to the mobile aid provision for en route refugees during migration.

The underlying optimization problem is defined on a time-space network concerning a multi-period planning horizon. Transiting refugee groups move from a node to an adjacent node in each period according to pre-fixed paths on the network. To be served, a refugee group must be at the same node with at least one mobile service facility in the same period. The problem is to find the number of mobile facilities that need to be utilized, in addition to their locations and the routes over time, while providing continual service opportunity with predetermined frequencies. The objective is to minimize fixed and variable transportation costs. We define and study this problem for the first time and prove its $\mathcal{NP}$-hardness.

We develop an exact mixed-integer linear programming (MILP) model. However, due to the model's impracticality in larger instances, a faster approach is needed, and hence we develop a problem-specific Adaptive Large Neighborhood Search (ALNS) algorithm. The performances of both solution methodologies are compared using test data that we generated from the 2018 Honduras Migration Crisis. For small-sized instances, both ALNS and MILP find optimal solutions at a similar pace. In medium-sized instances, ALNS finds optimal solutions in shorter durations, whereas the MILP model is not able to find optimal solutions in all instances in a limited time. In large-sized instances, the better performance of the ALNS algorithm is observed, as it finds better solutions in shorter times compared to the MILP model. We further analyze properties of the proposed solution methods and obtained some insights through computational tests.

ÖZETÇE

Hareket Halindeki Mülteciler için Mobil Yardım Sağlanmasında Lojistik Eniyileme
O. Baturhan Bayraktar
Endüstri Mühendisliği, Yüksek Lisans
24 Temmuz 2020

Dünyanın en zorlu problemlerinden biri olan yerinden edilme sebebi ile, Birleşmiş Milletler Mülteciler Yüksek Komiserliği'nin Haziran 2020 verilerine göre 79.5 milyon insan yaşadıkları yerden göçmek zorunda bırakılmıştır. Bu tezin amacı, göç esnasında hareket halindeki mültecilere yönelik mobil yardım hizmetleri ile servis götürülmesi sırasında ortaya çıkan bir lojistik probleminin optimizasyonu için verimli, dinamik ve etkili bir çözüm yaklaşımı sağlamaktır.

Temelde yer alan optimizasyon problemi, bir zaman-uzay ağında çok-periyotlu bir planlama ufkuna göre tanımlanmaktadır. Hareket halindeki mülteci grupları, ağda önceden belirlenmiş yollara göre her periyotta bir düğümden bitişik bir düğüme geçmektedir. Bir mülteci grubunun bir düğümde servis alabilmesi için en az bir mobil servis aracının aynı periyotta aynı düğümde olması gerekmektedir. Problem, grupların önceden belirlenmiş bir sıklık seviyesine göre servis alabilmeleri için mobil servis araçlarının zamana göre yerlerine, dolayısıyla rotalarına ve bununla birlikte kaç araç kullanılması gerektiğine karar vermektedir. Problemin amacı, mobil araçların sabit ve değişken maliyetlerini enazlamaktır. Bu çalışma ile problem, literatürde ilk kez tanımlanmaktadır ve $\mathcal{NP}$-zor bir problem olduğu ispatlanmıştır.

Problemin çözümü için karışık-tamsayılı doğrusal programlama (MILP) modeli geliştirilmiştir. Ancak modelin büyük ölçekli örneklerde pratik olmaması sebebi ile, problem-spesifik bir Adaptif Geniş Komşuluk Arama (ALNS) algoritması geliştirilmiştir. İki çözüm metodolojisinin performansı, bu çalışma ile üretilen 2018 Honduras Göç Krizi verileri aracılığı ile test edilmiştir. Küçük ölçekli örneklerde, iki yöntem de benzer hızlarla en iyi çözümleri elde etmiştir. Orta ölçekli örneklerde ALNS en iyi çözümleri kısa sürelerde bulabilirken, MILP bazı örneklerde en iyi çözümü sınırlı zamanda bulamamıştır. Büyük ölçekli örneklerde, ALNS'nin yine MILP'den daha iyi performans gösterdiği gözlemlenmiştir. Son olarak, iki yöntemin özellikleri analiz edilmiş ve elde edilen çözümlere göre çeşitli içgörüler sunulmuştur.

ACKNOWLEDGMENTS

First, I would like to express my deepest gratitude to my advisor Prof. Sibel Salman for the endless support on my master's adventure for her guidance, patience, and immense knowledge. She encouraged me every time I needed and I could not have imagined a better mentor for my master's study.

I would like to thank Assist. Prof. Eda Yücel and Assist. Prof. Dilek Günneç for their never-ending support both on our project and my thesis. This thesis would not be written without their precious contributions.

I also thank my other jury members, Prof. Güvenç Şahin and Prof. Ceyda Oğuz, for their valuable comments and support on this thesis.

I would like to acknowledge TUBİTAK for financially supporting our project (grant number 119M229) and this thesis on the last year.

In the last two years, I was very lucky to have Sude with me all the time. We shared all kinds of happy and bitter moments together. I would not be able to finish this thesis without her.

I would like to thank my buddy, Onur, with whom I had the same fate without knowing. We are still on the same path and I hope, we can work together on new research topics in the future.

I thank my friends Hakan, Begüm, Abdullah, Feray, and Şeyma for the times we spent together and supported each other. I also thank my friends Gökçe, Kerim, Çelen, Buse, Karim, Umut, and Ege Mert whose determinations have always inspired me. I am grateful to have my international flatmates Munam, Ayesha, and Pouya who helped me to discover the different perspectives of life. I would like to thank our undergraduate researchers, Tutku and Hümeyra, for their help on our project.

I would like to express my gratitude to my undergraduate friends, Beyza (my sidekick), Üsame, Fatih, Emir, Onur Alp, Oğuzhan, Hasan, Furkan, Seher, Güliz and my highschool friends, Işıkdoğan, Abay, and Serhat who are assisting me every time I need.

Lastly, my biggest thanks to my father, my mother, and my brothers Mehmet Ömer and Orhan Murat for the culmination of two years of study, support, and patience. I would not be able to continue my research journey without having their love and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

ALNS	Adaptive Large Neighborhood Search
CH	Constructive Heuristic
CTP	Covering Tour Problem
FIFLP	Flow Intercepting Facility Location Problem
FLP	Facility Location Problem
IDP	Internally Displaced People
k-PVC	k-Path Vertex Cover Problem
LCSI	Least Cost Subpath Insertion
LRP	Location Relocation Problem
LS	Local Search
MCP	Maximal Covering Problem
MFLP	Mobile Facility Location Problem
MILP	Mixed Integer Linear Programming
MSF	Mobile Service Facility
N-MVC	N-distance Minimal Vertex Cover Problem
PCP	Path Covering Problem
RI-n	n-Regret Insertion
RMFR	Random MSF Removal
RSAR	Random Service Act Removal
RSI	Random Subpath Insertion
SCP	Set Covering Problem
SR	Shaw Removal
WI	Worst Insertion

Chapter 1

INTRODUCTION

Nowadays, the displacement of the people, which is defined as being forced to leave their living place due to several reasons such as violence and persecution, is one of the greatest and most urgent problems of the world.

Displaced people who do not leave their country are categorized as "internally displaced people (IDPs)", whereas people who do not stay are defined as "refugee". Among refugees, people who request sanctuary from the country that they flee to, are defined as "asylum seekers". As of June 2020, more than 79.5 million people are displaced in the world [UN Refugee Agency, 2020]. More than 45.7 million of the displaced people are IDPs. The numbers of refugees, asylum seekers and Venezuelans displaced abroad are more than 26 million, 4.2 million and 3.6 million, respectively. Note that 40% of the refugees are under the age of 18. Globally, 68% of the refugees are coming from the same 5 countries with 6.6 million people from Syria, 3.7 million from Venezuela, 2.7 million people from Afghanistan, 2.2 million South Sudan, and 1.1 million from Myanmar. Nearly 73% of the refugees are hosted in the neighboring countries of the refugees' country of origin. Therefore, the five refugee-hosting countries that receive the most refugees are Turkey, Colombia, Pakistan, Uganda, and Germany with 3.6 million, 1.8 million, 1.4 million, 1.4 million, and 1.1 million refugees, respectively.

Naturally, the displaced individuals are exposed to rough events and these challenges lead them to suffer from various problems including difficulties in accessing basic life needs, such as food, clothing, and housing, according to Article 25 of the Universal Declaration of Human Rights proposed in General Assembly of United Nations in 1948 [UN General Assembly, 1948]. Consequently, without basic needs, the health of refugees deteriorates dramatically, and they face numerous medical problems that can be categorized under physical and mental health. While infectious diseases, injuries, and problems arising from malnutrition can be counted in physical health issues [Worley et al., 2000], post-traumatic stress disorder, depression, and anxiety are mental health problems that are seen among the refugees [Comellas et al., 2015]. Therefore, getting a medical service emerges as an essential aid, that may have a positive impact on uncovered basic needs, for the refugees.

As stated in the first report of the World Health Organization, the health risks carried by displaced people who reach to Europe are more than the citizens of the countries they fled [WHO, 2019]. As in line with this argument, infectious diseases, such as Poliomyelitis, Measles, Hepatitis A and Typhoid Fever, can spread easily among the refugees in a group and the people that are in contact with the group [Sharara and Kanj, 2014]. Thus, governments need to keep track of the moving and settled populations in terms of medical care and other services.

Nevertheless, both getting a medical service or other types of services to satisfy the basic needs of refugees and tracking them by governments cause other challenges in the long and exhausting process of migration. Cultural differences such as language barrier, discordant health practice expectations, and narrow anticipation on the part of the provider are a part of these issues which may cause the refugees to be misdiagnosed and leave them confused, as well [Burgess, 2004]. Moreover, they may lose their confidence in the medical systems of the host countries. On the other hand, even with current technologies, it is still a complicated process for governments to track the refugees. To support governments in this practice, refugee movement simulation models can be utilized, however, there are few studies in the literature on refugee movement simulations [Groen, 2016, Groen, 2018].

Provision of health and basic need services such as food and water supply are essential activities carried out by NGOs and non-profit organizations. On the global scale, governance of migration is executed by United Nations Network on Migration established in 2019 superseding Global Migration Group [Bastia and Skeldon, 2020, UN Network on Migration, 2020]. International Federation of Red Cross and Red Crescent Societies (IFRC) is also working on humanitarian relief aid operations in more than 190 countries and 13 million volunteers around the world [IFRC, 2020]. UN Network on Migration, IFRC, and other international and local humanitarian organizations working on refugee aid support the refugees across different categories such as food, clothing, and healthcare as mentioned as basic needs.

One of the hardest issues in helping refugees is supporting the people on the road or in other words en route refugees. Planning to satisfy the needs of moving groups is a difficult problem, especially due to the scarcity of resources. In most cases, newly displaced people travel on the roads with different transportation modes. Considering the displacement is made by forcing, most of these people do not have private cars or enough money to transport via another vehicle and they start to walk from their origin places to find a safer destination with better opportunities. While on the road, tracking the groups becomes harder and the refugee groups struggle to reach basic life supports on the way. Therefore, the support for these people must have the ability of easy-mobilization [Shortall et al., 2017].

Among the most convenient and economic methods, mobile service facilities

(MSF) come forward as a practical solution for serving the en-route refugees in terms of basic needs [MRS, 2019]. In practice, some of the humanitarian organizations that want to support en route refugees have implemented mobile service facilities as a solution. In such an instance, Türk Kızılayı (Red Crescent of Turkey) started a project named Mobil Mutfak where several trucks are transformed into mobile kitchens and these kitchens, which can serve up to 2000 people for a meal, has been utilized in serving the displaced people in various locations of Syria [Türk Kızılayı, 2020]. In another example resembling the Mobil Mutfak is applied by the partnership of Directorate General of Migration Management and UN Refugee Agency in which, again, multiple trucks are used as mobile service units [Alkan Group, 2020]. These units served basic needs in the border of Turkey and Syria for refugees in this area. Medecins Sans Frontieres (Doctors Without Borders) launched a campaign in which they used a mobile clinic and served nearly 100 refugees in the border of Serbia and Hungary where approximately 2000 people pass everyday [Gulland, 2015]. In 2015, Doctors of World started Refugee Ferry Project. According to this project, primary and mental healthcare is provided for refugees on a commercial ferry [Shortall et al., 2017]. In a different concept, where the refugees are on sea and need to be rescued, Open Arms provides lifeguarding support for the refugees traveling on boats in Mediterranean Sea [Open Arms, 2020]. Due to the mobility of the refugees, it is a hard process for this organization to follow up with them. Since mostly these refugees are tried to be smuggled into Europe illegally, they are also aware of where they may be caught and prepared according to this information. As a result, tracking these people is an important task for governments and humanitarian organizations to save lives. Since the qualitative studies for the en-route refugees are scarce finding effective and sustainable solutions becomes harder according to The Office of the High Commissioner for Human Rights [OHCHR, 2016].

(a) (b)

Figure 1.1: Mobile service facility examples (a) Partnership of UNHCR and Directorate General of Migration Management, (b) Mobil Mutfak of Türk Kızılayı.

Providing an effective solution to this complex problem can benefit from analytical techniques, mathematical modeling, and optimization algorithms. A recent research sub-field in humanitarian logistics having an increasing interest is related to the logistics planning of mobile services targeting migrants on their path to a new country and destination. In this thesis, the operational problem of mobile service provision for en route refugees is undertaken. Specifically, the challenge arises in determining the number and the routes of MSFs for aid provision to displaced people (demand) who enter the network from various origin locations and proceeds to destination locations. Different refugee groups can enter the network at different periods. Since MSFs are located according to the transiting refugee groups' paths, this problem is a variant of Mobile Facility Location Problems (MFLPs). Another challenge is provoked due to the obligation of tracking and serving refugee groups in a continuous manner such that every group must be encountered repeatedly until they leave the network. Therefore, the operational problem covers two main necessities. The first one is serving the transiting displaced people from the humanitarian point of view and the second is tracking refugee groups to assist the monitoring operations of the governments.

From the demand perspective, since refugee groups can enter the network at different periods and they move on the network, the problem is defined on a multi-period time horizon. The network does not change between periods, however, nodes that are labeled as demand points may change because the refugee groups are moving from node to node at each period. The movement can be only made to adjacent nodes and the routes of the refugee groups are predetermined. According to this, the location of each refugee group is assumed to be known at each period after they enter the network. One of the most novel characteristics of the problem compared to other variants of FLPs is that each refugee group must be served at least once in every consecutive prefixed number of periods. In other words, the groups must be tracked and encountered frequently.

On the service systems perspective, this problem is defined for multiple MSFs. Each MSF is located at one and only one node at each period and can move between adjacent nodes between periods. An MSF may stay at a node, as well. It is also assumed that to serve a refugee group, at least one MSF and the group must be at the same node in the same period. The objective of the problem is to minimize total costs incurred by the fixed cost of operating and variable costs of traveling of MSFs while satisfying the service requirement.

Due to these properties, we call this problem as Multi-period Mobile Facility Location Problem with Mobile Demands (MM-FLP-MD).

In this thesis, the MM-FLP-MD problem is defined for the first time in literature. The requirement of demand coverage differs this problem from the related problems which generally maximize the fulfilled demand or set a threshold for the amount of demand to fulfill. In this problem, however, the demand must be satisfied at least once in every consecutive predetermined number of periods. Therefore, in MM-FLP-MD, the demand coverage is a requirement rather than an objective. In MM-FLP-MD, the locations of both demand and supply are changing from period to period. In contrast to the proposed problem, covering problems consider the locations of either supply or demand as static. The implemented metaheuristic solution method, an Adaptive Large Neighborhood Search algorithm, is also a problem-specific framework and it utilizes special structures that are developed for the first time with this study. The generated data based on the 2018 Honduras Migration Crisis is another contribution of this study.

The thesis is organized as follows. The next chapter examines the related literature and identifies the differences in the problem from the well-known problems. Chapter 3 defines the problem in detail and provides an explanatory example. Chapter 4 presents a mathematical programming formulation and Chapter 5 proposes an adaptive large neighborhood search algorithm to address the problem. Chapter 6 provides a numerical analysis and managerial insights to decision-makers on a real-life case, the 2018 Honduras Migration Crisis. Finally, Chapter 7 presents concluding remarks.

Chapter 2

LITERATURE REVIEW

In this chapter, first, related to the facility location problems, location-relocation problems, mobile facility location problems, and flow interception facility location problems are reviewed. The second section reviews covering problems several network optimization problems with coverage aspects. Evacuation problems and problems involving moving moving population are reviewed, as well, due to the movement of demand in both types of problems that are close to the moving-refugee aspect of the proposed problem. Lastly, contributions to the literature are summarized.

2.1 *Facility Location Problem (FLP)*

A vast amount of research has been conducted on facility location problems, which constitutes a a research field in location science. FLP copes with deciding the finest locations of serving facilities, to serve a set of customer points [Laporte et al., 2019]. Generally, the goal is to minimize the total cost resulting from a fixed cost of opening a facility and variable cost from serving to customers with opened facilities for private enterprises [Melo et al., 2009]. Additionally, due to its direct influence on response time, FLP finds place in humanitarian operations [Haghani and Oh, 1996, Balcik and Beamon, 2008].

FLP has been studied in different forms by location science scholars. The roots of this problem family go back to 17^{th} century, Fermat's problem, and Evangelista Torricelli's solution [Laporte et al., 2019]. Weber's problem and its extensions such as p-median problem and p-center problem have created location science as a new research area [Laporte et al., 2019]. In 1965, by Balinski, the first mixed-integer linear programming formulation for uncapacitated facility location problem was developed [Balinski, 1965, Laporte et al., 2019].

In a more general perspective, FLPs have two elements that cover the decision dimensions of FLPs: time and space [Arabani and Farahani, 2012]. Variants of FLPs are obtained from these elements.

In terms of time, FLP can be divided into two categories: static and dynamic FLP. In static FLPs, the inputs of the system are considered to be unchanging with time [Nickel and Saldanha-da Gama, 2019], i.e. a static problem is solved for one time and solution is not considered to be changing over time or the solution is valid until an input changes. Even though they are rare infinite planning horizon cases or single period cases can be classified as static FLPs [Nickel and Saldanha-da Gama, 2019]. In dynamic FLPs, the changes in the input set are considered to be changing over time [Nickel and Saldanha-da Gama, 2019]. Dynamic FLPs allow decision-makers to adapt the needs for changing demand and facility extension and relocations over time [Daskin et al., 1992]. Dynamic FLPs can also categorized into two groups: single-period (continuous) and multi-period (discrete) FLPs [Arabani and Farahani, 2012]. Single-period, i.e. continuous, FLPs are mostly addressed in optimal control [Nickel and Saldanha-da Gama, 2019]. Multi-period FLPs consider time as discrete, therefore, the planning horizon is split to several moments, and the decision is applied in these moments [Nickel and Saldanha-da Gama, 2019].

In space perspective, FLP is divided into three categories: continuous, discrete, and network spaces [Hamacher and Nickel, 1998]. In continuous space, facilities can be located at any point in the designated area. However, in discrete space, there are certain points in the area that the facilities can be placed [Arabani and Farahani, 2012]. The network space is comprised of nodes and links (edges or arcs) and both facilities and demand can be located on nodes or links [Arabani and Farahani, 2012].

Additionally, if the parameters are certain and known a priori, these types of problems are classified as deterministic. Otherwise, if there is any uncertainty in at least one parameter, these types of problems are called stochastic [Farahani et al., 2009a].

The MM-FLP-MD is defined on a multi-period planning horizon, therefore, it is a dynamic FLP time perspective, whereas, in the space dimension, the problem is defined in a network space. Since the parameters of the problem are known beforehand, it is a deterministic problem.

There are several variants of FLP that possess similar properties with the MM-FLP-MD. In the following subsections, these problem variants, their identical and distinct features are compared with the MM-FLP-MD. Since this thesis considers a deterministic problem, stochastic variants of the problem are not reviewed in this literature review.

2.1.1 Location Relocation Problem (LRP)

Location Relocation Problem (LRP) is one of the extensions of FLP where the located facilities are allowed to be relocated after a certain amount of time. Nevertheless, relocation comes with a price. Most of the time, this price shows itself as a fixed cost or traveling cost for the facility in the objective function. Min and Melachrinoudis listed several factors that affect the decision for relocation such as site characteristics, cost, traffic access, market opportunity, quality of living, local incentives, and labor-management relations [Min and Melachrinoudis, 1999].

MM-FLP-MD resembles LRP in several aspects. First, in both problems, the locations of the facilities can change. Secondly, in most of LRP studies, the objective is to reduce the total cost incurred by usage and relocation of facilities as in the objective of MM-FLP-MD. Finally, MM-FLP-MD is defined on a multi-period time horizon as in some of the LRP variants.

The relocation concept was introduced to the literature by Ballou with the article on relocating the inventories due to the changes in economic conditions [Ballou, 1968]. Since the problem has been defined on a multi-period planning horizon, this article approaches the problem with a dynamic programming methodology. Also, this study defends the opinion in which the relocation decision should be made with a dynamic fashion since the dynamic models are claimed to be superior to the static models in these types of situations. In another earlier work on LRP was done by Wesolowsky and it tries to generalize Ballou's study [Wesolowsky, 1973]. The dynamic programming approach in this study has been identified as incomplete dynamic programming since node 0 in this approach cannot be evaluated at the stage it is found. Sweeney and Tatham's study shows that Ballou's work produces a suboptimal solution and they improve Ballou's dynamic programming method with a procedure that can obtain an optimal solution by R_t best rank order on eliminating considered alternatives [Sweeney and Tatham, 1976]. Nonetheless, in Ballous's, and Sweeney and Tatham cases, the cost of relocation does not appear in the objective function. In Campbell's work in 1990, freight carrier terminals are located and can be relocated if the transportation costs for freight carrier services increase [Campbell, 1990]. This work provides near-optimal solutions with myopic approaches for the problem of when to transport and when to relocate. Opening and closing facilities in different periods can also be categorized in the LRP, and the study of Wang et al. the goal is to minimize total traveling distance of the customers to the supply facilities with opening and closing the facilities at different times periods [Wang et al., 2003]. In supply chain-related works, the time-dependent and fluctuating demand requires facilities to gradually relocate or change capacities and in such a work, a supply chain model with a mathematical model is developed by Melo et al. [Melo et al., 2006].

Among later works, Farahani et al.'s study considers a single facility location problem with multiple relocations where weights of demand points are time-dependent and known [Farahani et al., 2009b]. They consider a dynamic programming model and a binary integer programming model (BIP) to tackle this LRP variant. A general approach for both deterministic and stochastic cases of LRP has been proposed in her dissertation by Durukan Sonmez [Durukan Sonmez, 2012]. Again with opening, extending, and closing several facilities concepts, Jena et al. propose a mathematical model and a heuristic for a multi-commodity multi-period LRP [Jena et al., 2016]. Jena et al. propose a Lagrangian relaxation heuristic for the first time for a large-scale dynamic FLP with generalized modular capacities and multiple commodities [Jena et al., 2017]. A case study for service facility relocation is done to show issues and challenges faced due to facility relocation [Srivastava et al., 2016]. In this work, Srivastava and Ghagare develop a mathematical model and find that the relocations affect the service level and cost. In one of the most recent works, Wang et al. develop a framework for the relocation of the facilities to minimize the total traveling cost for serving the moving customers [Wang et al., 2020].

The mentioned articles in this subsection generally concern a single facility or multiple facilities such that relocating them incurs relatively higher costs as in relocating shops and factories. Thus, a relatively small number of relocations are made in these problems. As a result, comparing the defined MSF of the MM-FLP-MD with these facilities, the cost of relocation and traveling is less for MSF and it is easier for MSF to relocate more often. It can also be stated that these decision problems discuss the relocation at the strategical level and in the long run for decision-makers. In the MM-FLP-MD, the decisions must be made weekly, daily, or maybe even every hour, and for MSF, this problem turns out to be an operational level problem. Additionally, earlier articles take continuous location space into account in the problem definition whereas the MM-FLP-MD addresses a network space.

In health care systems, ambulance redeployment problems of Emergency Medical Services (EMSs) carry similar properties with LRP [Arabani and Farahani, 2012]. Therefore, it is also beneficial to mention the studies with LRP as well. The ambulance redeployment processes and decisions are applied typically for the allocation of ambulances and reassignment of ambulances in short periods. Hence, it can be said that these decisions are operational level decisions [Güneş et al., 2019]. However, redeployment models are mainly facilitated by real-time applications and are solved with dynamic programming. Due to the complexity of the solution space, these problems are mainly solved with approximate dynamic programming approaches or heuristics. As an example to such a work, the readers can refer to the study of [Maxwell et al., 2010]. The survey prepared by Ahmadi-Javid can be an important reference point for further studies for EMS type relocation problems [Ahmadi-Javid et al., 2017].

Dynamic programming can be seen as the most common solution method for LRPs as a consequence of its multi-period nature and time-dependent parameters.

2.1.2 Mobile Facility Location Problem (MFLP)

Mobile Facility Location Problem (MFLP) addresses the optimization problem of locating and moving some existing facilities to cover the demand located on the vertices of a network. The objective of MFLP is to minimize the cost of using and relocating the facilities. Customers' traveling costs can be considered in the objective of MFLP, as well. By definition, MFLP and MM-FLP-MD are closely related, since both problems facilitate the location of mobile supply facilities to cover the demand requirement. However, differences occur in two aspects: the coverage of demand and consideration of cost incurred to customers due to their assignment to facilities.

MFLP was first defined by Demaine et al. as a type of movement problems [Demaine et al., 2009]. In their study, they give several approximation algorithms for the problem where they define the problem as the configuration of pebbles on a table. As in this and later studies, the aim is generally to find theoretical bounds and the approximation algorithm of the problem [Friggstad and Salavatipour, 2011, Armon et al., 2014, Anari et al., 2016].

Halper et al. provide an IP formulation and a local search heuristic for MFLP [Halper et al., 2015]. The local search heuristic is proven to be an approximation algorithm by Ahmadian et al. [Ahmadian et al., 2013]. In recent work, an extension of MFLP, capacitated mobile facility location problem (CMFLP), is defined by Raghavan et al. [Raghavan et al., 2019]. In this problem, mobile facilities are considered to have capacity constraints on the number of customers that they can serve.

MFLP differs from the MM-FLP-MD in terms of time horizon. MFLP is generally defined on a single period time-space network; however, the MM-FLP-MD is a multi-period problem. Therefore, the MM-FLP-MD is a generalization of MFLP.

2.1.3 Flow Interception Facility Location Problem (FIFLP)

Flow Interception Facility Location Problem (FIFLP) is an extension of FLP in which the demand is not considered as demand points, but rather flows. In this setting, the customers are not eager to find a supply point but rather they are willing to get a service if there is a supply point on their deterministic paths [Boccia et al., 2009]. Therefore, the facility locators seek to find intersection points or areas of the paths of flows and try to cover as much flow as possible.

FIFLP was first introduced to the literature by Hodgson and Berman in approximately similar times. Hodgson, in 1990, provided a mathematical formulation and a new flow definition, compared to earlier versions of the problem in business-related articles [Hodgson, 1990]. This article also proposes a greedy heuristic. In 1992, Berman et al. provided a similar problem definition as Hodgson's and proposed a greedy algorithm and a branch and bound algorithm to tackle the problem [Berman et al., 1992]. In another work, Berman generalized FIFLP with some definitions [Berman et al., 1995]. In the next study, Hodgson et al. define a preventive FIFLP in which the traffic inspection facilities are located to intercept the hazardous vehicle flow [Hodgson et al., 1996]. In his work in 1997, Berman calls flow-covering instead of interception and combines FIFLP with other location problems [Berman, 1997]. In another work of Berman, FIFLP is combined with spatial interaction problems [Berman and Krass, 1998]. They define a new mathematical model and develop a greedy heuristic which is reported to lack robustness. Gendreau et al. discuss the preventive FIFLP to maximize total risk reduction [Gendreau et al., 2000]. They claim that the tabu search method produces optimal or near-optimal solutions. Placement of inspection station vehicles is studied by Rosenkrantz et al. who defined two categories of problems with objectives such as minimizing maximum gap between two inspection stations and expected penalty cost of failure along the paths[Rosenkrantz et al., 2000]. They propose also some approximation algorithms. This problem can be considered in flow-capture problems due to the definition of problem.

Wu and Lin consider the competitiveness of the facilities in the flow-capture problem [Wu and Lin, 2003]. To achieve this, they propose a greedy heuristic, several variants of ascent search heuristic, and a tabu search algorithm. Kuby and Lim generalize the flow-capturing concept for alternative fuel stations. They propose a MILP and a greedy algorithm [Kuby and Lim, 2005]. Unlike the greedy metaheuristic algorithms of earlier works, Gzara and Erkut develop a Lagrangian relaxation and cutting plane algorithm which finds upper bound and a feasible solution [Gzara and Erkut, 2009]. Boccia et al. review the articles and present comparisons on different algorithmic approaches [Boccia et al., 2009]. They show also related studies on applications in areas such as communication and inspection, hence, the readers can refer to this work for further studies. In later works, Selmic et al. work on inspection facilities with a single objective and multi-objective cases [Šelmić et al., 2010] by developing a bee colony optimization algorithm. Furthermore, Selmic et al. develop a Genetic Algorithm for inspection stations of weigh-in-motion vehicles [Šelmić et al., 2011]. In vehicle-inspection station location decision problems, several works have been done more recently [Tian and Liu, 2014, Tian and Zhou, 2015, Tian et al., 2016, Konstantinou et al., 2020].

Ares et al. study locating roadside clinics for preventing various diseases among truck drivers flow in Africa [Ares et al., 2016]. They model the problem with a novel set-partitioning type of formulation and develop a column generation algorithm that deals with pricing problems by treating them as shortest path problems. Recently, this work has been extended in 2020, and in their newer article, de Vriejs et al. discuss and investigate some managerial insights such as the objectives of the effectiveness of healthcare service delivery and patient volume [de Vries et al., 2020].

The works covered in FIFLP are similar to the MM-FLP-MD especially in terms of the definitions of paths, flows, and type of covering the flows. However, the regular covering of the same demand and tracking them is rather not considered as in the MM-FLP-MD.

2.2 Covering Problems

Let us consider the special case of the MM-FLP-MD where demand does not relocate over time but a limited number of facilities may relocate to satisfy the service requirement which imposes that a demand node should be able to get service (service can be taken when a facility arrives to the demand node) every tau periods (for instance every three days). As described by Taymaz in her thesis and by Farahani et al. in their article, we can partition the covering problems into two as *Set Covering Problem* (SCP) and *Maximal Covering Problem* (MCP) [Farahani et al., 2012, Taymaz and Dellaert, 2013]. SCP necessitates coverage, on the other hand, MCP aims to optimize the coverage. Since the MM-FLP-MD intends to cover all the moving demand and minimize the total cost incurred by traveling and using cost, SCP is reviewed further in this thesis.

In general SCP, the demand points are considered as covered if the distance between the chosen supply point is less than a certain distance. The objective is to minimize the total cost incurred by choosing supply points while satisfying all the demand [Farahani et al., 2012]. An extension of SCP is called *Covering Tour Problem* (CTP) in which a minimum length Hamiltonian tour is searched by satisfying the criteria that every vertex must be less far than a predetermined distance from the cycle [Gendreau et al., 1997]. *Path Covering Problem* (PCP) is closely related to the MM-FLP-MD as well. As in CTP, PCP selects some nodes to cover all the vertices in a network, but this time the selected nodes from a path [Boffey and Narula, 1998]. PCP is extended for multiple paths in which more than one path is formed.

Similarly, path covering concept emerges in more recent works. In *k-Path Vertex Cover Problem* (k-PVC), the problem detects the nodes that they cover every path with the length of k [Brešar et al., 2011]. The roots of this problem reach back to the Canvas Scheme where the number of expensive protected sensor devices should be minimized to defend a Wireless Sensor Network from attackers [Novotnỳ, 2010]. The protected sensor devices should be placed in a way that there should be at least one protected device in the $k-$hop neighborhood of every vertex on the network. k-PVC is proven to be $\mathcal{NP}$-hard and when $k = 2$, the problem simply turns to a well-known vertex cover problem. Likewise, in $N-$*distance Minimal Vertex Cover*

Problem (N-MVC), a node subset must be detected such that the size of the subset must be minimized and every node in the network must be less than $N-$ hops far away [Yadav et al., 2016].

Both k-PVC and N-MVC resemble the MM-FLP-MD, as all three problems cover demand by placing supply to critical points. However, in the MM-FLP-MD the nodes to be covered depends on the period since the demand is assumed to be moving on predetermined paths. Therefore, the multi-period definition of the MM-FLP-MD differs from k-PVC and N-MVC. It can also be stated that if there is a refugee group entrance to each path at each period, the problem turns into N-MVC. Since each refugee group must be covered at least once in every consecutive predetermined number of periods, at least one of every consecutive the same predetermined number of nodes must be covered in each path. This property turns demand coverage definition of MM-FLP-MD as the same definition of N-hops in N-MVC. So, the MM-FLP-MD is an extension of N-MVC.

2.3 Evacuation and Moving Population Problems

Similar to the MM-FLP-MD, evacuation problems consider mobile demand. In these problems, the guidance of trapped people that escape from buildings, hospitals, or areas that are risky should be considered together with traffic congestion or other special conditions such as an emergency. These problems are mainly divided into two categories based on the problem scale [Bayram, 2016]. Macroscopic models are large-scale models that aim to obtain the minimum evacuation time from risky areas. Microscopic models are small-scale models that focus on traffic congestion considering every single vehicle. In both models, the evacuees are directed to some paths to minimize the evacuation time, the transportation time, the number of shelters required, or the amount of risk that may cause threats. Different from evacuation problems, in the MM-FLP-MD, refugees determine their paths, that is, we have no control over demand movement.

In the MM-FLP-MD, a migration act is considered, hence refugees are mobile. Several studies are in the literature about the paths and behaviors of refugees. In the study prepared by Nagurney, human migration is modeled with an equilibrium model according to human and location types [Nagurney, 1989]. In another article, Nagurney presents the migration equilibrium as a network model with adding up the movement costs [Nagurney, 1990]. Nagurney et al. examine the multi-class human migration network equilibrium model with inter-location costs [Nagurney, 1990]. Pan and Nagurney develop a multi-stage network balance model [Pan and Nagurney, 1994]. Yet, in this model, the refugees are considered to move from one location to another on a chain basis compared to other models. As a result, they show that the steady-state population distribution can be estimated under some assumptions. Kim and Song model the human movements with continuous Markov chains [Kim and Song, 2012]. However, these studies do not address locating facilities and serving refugees.

2.4 Contributions to Literature

Alleviating the problems of moving populations is an important issue today. The problems faced by the refugees and their movements, and providing relief aid to them are mostly studied in the context of social sciences. On the other hand, we address one of the problems faced by refugees on the move from Operations Research perspective. We identify the need for providing aid to refugees on the move by mobile facilities and propose to optimize its logistics, namely the decision on where to locate the facilities over time. The facilities may stay at some locations for some durations and relocate to new locations on the paths of the refugee groups that may join and leave the marching groups on different paths. Therefore, as demand moves and changes over time, facilities should provide continual services such as health care and delivery of essential supplies. The optimization problem makes sure that the refugee groups can access service intermittently to fulfill their needs and minimizes the operational costs. Operations research techniques used and the case study in this thesis make this work an analytic and quantitative one in the area of relief aid operation where such kind of studies are sparse.

By definition, covering the demand points by locating the supply points classifies this problem in the family of FLP. In our problem, both the demand points and the facilities relocate over time and they must meet at the same locations in order for the service act to take place. The closest subbranch of FLP to out problem can be stated as FIFLP where demand and supply are both moving. In this subject, most studies have the objective of maximizing the total demand covered. Nevertheless, in the MM-FLP-MD, demand coverage is defined as a requirement and the objective is to minimize the total cost incurred by satisfying the demand. Additionally, periodic demand coverage is a new concept in the literature.

In theoretical studies, covering problems have a similar nature to the MM-FLP-MD. However, the dynamic form of both supply and demand has not been covered in these works. Additionally, problems in these studies do not carry a multi-period nature. The network does not change from period to period in the MM-FLP-MD, however, the nodes to be covered change from period to period. Hence, the facilities may need to be located at different nodes in different periods. This feature extends the well-known vertex covering problems, including those with path coverage.

The formulation of this problem and the mathematical model developed in this thesis is first in the literature. Moreover, an Adaptive Large Neighborhood Search Algorithm is developed for the MM-FLP-MD. Some specific structures for the paths and refugee groups are investigated for the first time in this problem definition.

A case study based on the 2018 Honduras Migration Crisis is examined quantitatively and analytically in this thesis. The data is generated to create instances of differing sizes and properties. Insights are driven in the computational results. The generated data is shared with scholars for further use.

Chapter 3

PROBLEM DEFINITION

In this chapter, we provide a formal description of the MM-FLP-MD together with an illustrative example and prove its computational complexity. To facilitate the problem description, below we provide explanations of some important problem features.

3.1 Problem Description

Network: The MM-FLP-MD is defined on a directed network $G = (V, A)$, where two groups are mobile on the network G, refugee groups, and MSFs that form two subnetworks. The network of refugee groups is represented by $G_\rho = (V, A_\rho)$, where the node sets of G_ρ and G are the same, and V corresponds to the set of nodes on the refugee paths that correspond to the stopover points of refugee groups. Each refugee group follows a path starting from a source node and ending at a destination node. Each arc on the path of a refugee group is assumed to take one period by the group's transportation mode, i.e. by walk or by their vehicle. The arcs of the paths of all refugee groups form the set of arcs of the network G_ρ, i.e. A_ρ. The network of the MSFs, denoted by $G_\nu(V_\nu, A_\nu)$, differs from G_ρ, as the MSFs follow different routes and it may not be possible to locate a facility at each node of V. V_ν represents the set of nodes where the MSFs can be located and it is a subset of V, i.e. $V_\nu \subseteq V$. For node $i \in V$, the binary parameter a_i indicates if node $i \in V_\nu$. Similar to the network of the refugee groups, each arc in G_ν is assumed to be taken by MSFs in one period. Therefore, the arc set of G_ν, i.e. A_ν, comprises of the arcs of paths between the nodes in V_ν that can be taken by a MSF in at most one period.

Paths: The refugees that follow the same path and that enter the network during the same period are assumed to form one refugee group. This is in line with practice because refugees prefer to move in groups and generally follow a path, which is determined at the beginning of the trip. Therefore, we assume that refugee paths are known a priori. P denotes the set of paths, where $|P|$ is finite. For each $p \in P$, l_p represents the number of nodes on path p, and n_{pk} represents the k^{th} node on path p. Thus, $n_{p,1}$ shows the origin node of path $p \in P$ and n_{p,l_p} depicts the destination node.

Refugee Entry: A refugee group entering the network at a period is called a refugee entry. Demand in each period, the location of a refugee group is known since the group's path and entrance time to the network is known. Note that more than one refugee group may follow the same path but may enter the network in different periods. In this case, the number of entries would be more than the number of paths.

Planning Horizon: Each refugee group follows a path over periods. The planning horizon is determined so that all refugee groups in the network have enough time to leave the network. The total number of periods is denoted by T.

Mobile Service Facilities (MSFs): The set of homogeneous MSFs is denoted by M. The departure point of a MSF $m \in M$ is not a depot but the first selected node in its route. Traversing an arc $(i, j) \in A_\nu$ incurs a cost of c_{ij}. MSFs can be at exactly one location (node) at a period.

Service Requirement: To receive and provide a service, a refugee group and a MSF, respectively, must be at the same node in the same period. Every refugee group must be served at least once in every consecutive τ periods.

Objective Function: The objective function comprises of two terms: (i) the fixed cost of preparing the MSFs for service, (ii) total traveling cost of MSFs.

Node-Time Pair: The state of being at node n at period t (either by a MSF or by a refugee group) is referred to as a *node-time pair* and is represented by $\sigma = (n, t)$. The node and period information of the node-time pair σ are provided by $n(\sigma)$ and $t(\sigma)$, respectively.

Service Act: The state of providing service to a refugee group by a MSF in node $n \in V$, at period $t \in T$ is referred to as a *service act*. By definition, a service act is represented by the corresponding node-time pair (n, t).

3.2 An Illustrative Example

Table 3.1 provides an example of MM-FLP-MD with $|P| = 3$ paths followed by 4 different refugee groups. The network $G_\rho(V, A_\rho)$ is illustrated in Figure 3.1. In this example, the network consists of $|V| = 5$ nodes. Since the last refugee group, which enters path p_2 at $t = 3$, completes its trip at $t = 6$, the planning horizon comprises of six periods, i.e. $T = 6$.

In this example, MSFs can be located at each node of the network G_ρ but node 4, i.e. $V_\nu = V \setminus \{4\}$. The MSF network, G_ν, is complete, meaning that all nodes are accessible from each node through a MSF within one period, and the transportation cost of an arc (i, j), $i, j \in V$ is assumed to be 10 times the absolute difference of the node indices, i.e. $c_{ij} = 10|i - j|$, $\forall (i, j) \in A_\nu$. The fixed cost of a MSF is given as $f = 50$. There are $|M| = 2$ MSFs. A refugee group should be served at least once in every consecutive $\tau = 3$ periods.

Table 3.1: Data for the illustrative example.

Path	**Node sequence**	**Refugee Entry at period t**		
		t=1	t=2	t=3
p_1	1-2-3-5	+		
p_2	1-2-4-5	+		+
p_3	1-2-4-3-5	+		

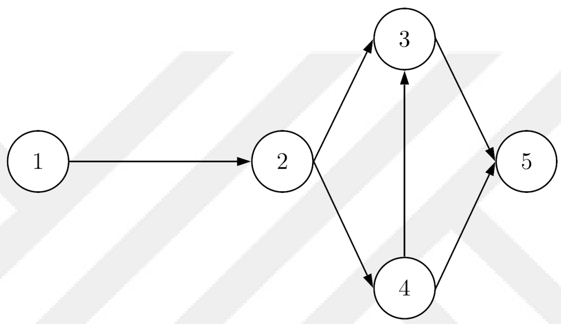


Figure 3.1: Network of refugee groups, $G_\rho(V, A_\rho)$, for the illustrative example.

The optimal solution for the example, which utilizes only one of the two vehicles, is shown in Figure 3.2, where columns and rows represent the periods and nodes, respectively. Refugee group movements are represented with the arrows. The arrows are labeled by (p, t) where p represents the path in which the refugee group is moving and t represents the refugee group's entering time to network.

The timely position of the MSF used is shown with a hashed arrow. Thick bordered circles represent the nodes and periods that the facility gives service, i.e. service acts. The MSF moves only once from node 2 to node 5 at the end of the period 4. The objective function value is the sum of the fixed cost of a single facility, i.e. 50, and the traveling cost from node 2 to node 5, i.e. $c_{2,5} = 30$, which makes 80.

In the optimal solution, the refugee groups that enter to path p_1 at period 1 and path p_2 at period 1 are served just once at node 2 at period 2. After this service, they leave the network in less than $\tau = 3$ number of periods. However, the refugee group that enters path p_3 at period 1 is served first at node 2 at period 2 and second at node 5 at period 5. The refugee group that enters path p_2 at period 3 is served just once at node 2 at period 4.

In the optimal solution, $\sigma_1 = (2, 2)$, $\sigma_2 = (2, 4)$, and $\sigma_3 = (5, 5)$ correspond to the 3 service acts that are provided by the single facility used. The service act $\sigma_1 = (2, 2)$ is provided to the refugee groups entering at period 1, and $\sigma_2 = (2, 4)$ and $\sigma_3 = (5, 5)$ are provided to refugee groups entering to path 2 at period 3 and to path 3 at period 1, respectively. For service act σ_2, $n(\sigma_2) = 2$ and $t(\sigma_2) = 4$ provide the node and period information, respectively.

Let us consider another solution where again a single vehicle used but it is not relocated. In this solution, the total cost includes just the fixed cost of a single vehicle, i.e. 50, however, the refugee group entering to the path p_3 at period 1 is not served in periods 3 to 5, leading to infeasibility. To make the solution feasible, an additional MSF can be used that provides service at node 5 at period 5, leading to a larger total cost that comprises of the fixed costs of two facilities, i.e. $50 + 50 = 100$, and so, suboptimality.

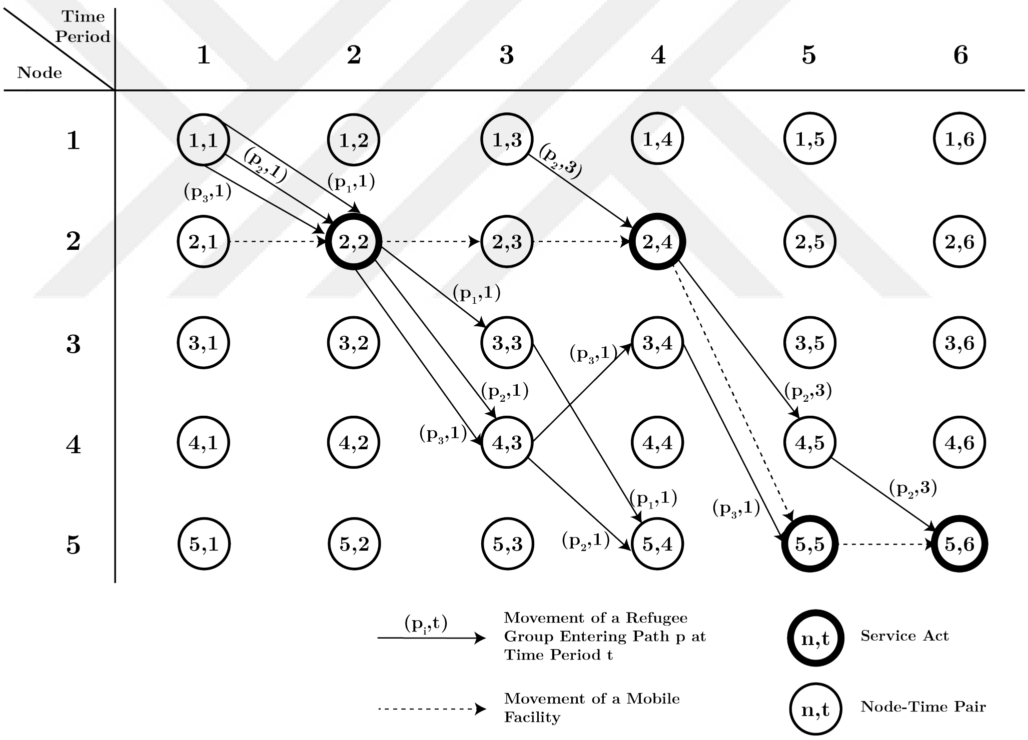


Figure 3.2: The optimal solution of the illustrative example.

3.3 Problem Complexity

Theorem 3.3.1. *The MM-FLP-MD problem is $\mathcal{NP}-hard$.*

Proof. (By reduction from the Hamiltonian Path problem) Given an instance of I_1 of the Hamiltonian Path (HP) problem defined on a directed graph $G(N, A)$, we construct an instance I_2 of MM-FLP-MD defined on a directed graph $H(N, A)$, where for each arc (i, j) in A, there exist two arcs defined in opposite directions between nodes i and j in A. We also let $N^{(s)} = N^{(d)} = N$ and P is defined as the set of all paths in G. Suppose that in I_2, the length of the planning horizon equals the number of nodes, i.e., $T = |N|$ and each refugee group entering the network at any period to use any path should be served at least once within $\tau = |N|$ periods. It is straightforward to see that if I_1 has a Hamiltonian path, an optimal solution to I_2 has a positive number of MSFs; otherwise, the optimal number of MSFs for I_2 is zero. Conversely, if an optimal solution of I_2 has a positive number of MSFs, then I_1 has a Hamiltonian path. Otherwise, I_1 does not have a Hamiltonian path. □

3.4 Mathematical Model

We propose the following mathematical model to solve the MM-FLP-MD problem.

Sets:

P	Set of refugee paths
V	Set of nodes
V_ν	Set of nodes that MSFs can visit, where $V_\nu \subseteq V$
A	Set of arcs
A_ρ	Arc set of refugee network, where $A_\rho \subseteq A$
A_ν	Arc set of MSF network, where $A_\nu \subseteq A$
M	Set of MSFs
T	Set of time periods

Parameters:

d_{pt}	$\begin{cases} 1, & \text{if a refugee group enters path } p \text{ at period } t \\ 0, & \text{otherwise} \end{cases}$	$\forall p \in P, t \in T$
a_i	$\begin{cases} 1, & \text{if a MSF can be located at node } i\text{, i.e. if } i \in V_\nu \\ 0, & \text{otherwise} \end{cases}$	$\forall i \in V$
l_p	length of path p, i.e. number of nodes on path p	$\forall p \in P$
n_{pk}	k^{th} node on path p	$\forall p \in P,\ k = 1, ..., l_p$
f_m	fixed cost of using MSF m	$\forall m \in M$
c_{ij}	traveling cost of a MSF from node i to node j	$\forall (i, j) \in A_\nu$

τ service requirement level

Decision Variables:

$$Y_{imt} \begin{cases} 1, & \text{if MSF } m \text{ is located} \\ & \text{at node } i \text{ at period } t \\ 0, & \text{otherwise} \end{cases} \qquad \forall i \in V_\nu, m \in M, t \in T$$

$$X_{ijmt} \begin{cases} 1, & \text{if MSF } m \text{ travels from} \\ & \text{ node } i \text{ to node } j \\ & \text{ at the end of period } t \\ 0, & \text{otherwise} \end{cases} \qquad \forall (i,j) \in A_\nu, m \in M, t = 1, ..., |T| - 1$$

$$Z_m \begin{cases} 1, & \text{if MSF } m \text{ is utilized} \\ 0, & \text{otherwise} \end{cases} \qquad \forall m \in M$$

Mathematical Model:

$$\min \quad \sum_{m \in M} f_m Z_m + \sum_{(i,j) \in A_\nu} \sum_{m \in M} \sum_{t \in T} c_{ij} X_{ijmt} \tag{3.1}$$

subject to

$$\sum_{i \in V_\nu} Y_{imt} = Z_m \qquad \forall m \in M, t \in T \tag{3.2}$$

$$\sum_{m \in M} \sum_{t'=0}^{\tau-1} Y_{(n_{p,k+t'+1}),m,t+k+t'} \geq d_{pt} \qquad \forall p \in P, 0 \leq k \leq l_p - \tau, t = 1, ..., |T| - l_p + 1 \tag{3.3}$$

$$Y_{imt} \leq a_i Z_m \qquad \forall i \in V, m \in M, t \in T \tag{3.4}$$

$$1 + X_{ijmt} \geq Y_{imt} + Y_{j,m,t+1} \qquad \forall (i,j) \in A_\nu, m \in M, t = 1, ..., |T| - 1 \tag{3.5}$$

$$Y_{imt} \geq X_{ijmt} \qquad \forall i,j \in V, m \in M, t = 1, ..., |T| - 1 \tag{3.6}$$

$$Y_{j,m,t+1} \geq X_{ijmt} \qquad \forall i,j \in V, m \in M, t = 1, ..., |T|-1 \quad (3.7)$$

$$\sum_{j \in V_\nu : (i,j) \in A_\nu} X_{i,j,m,t+1} = \sum_{j \in V_\nu : (j,i) \in A_\nu} X_{jimt} \qquad \forall i \in V, m \in M, t = 1, ..., |T|-2 \quad (3.8)$$

$$Y_{imt} \in \{0,1\} \qquad \forall i \in V, m \in M, t = 1, ..., T \quad (3.9)$$

$$X_{ijmt} \in \{0,1\} \qquad \forall (i,j) \in A_\nu, m \in M, t = 1, ..., |T|-1 \quad (3.10)$$

$$Z_m \in \{0,1\} \qquad \forall m \in M \quad (3.11)$$

The objective function (3.1) minimizes the total cost of MSFs, including fixed costs and traveling costs. Constraints (3.2) guarantee that if a facility is located to provide service at a node at any time period, then it is utilized. Constraints (3.3) correspond to the periodic service requirement of refugee groups and ensures that each refugee group is served at least once in every consecutive τ periods. Table 3.5 provides the open form of constraints (3.3) for the optimal solution provided in Table 3.2 of the illustrative example in Section 3.2. Constraints (3.4) ensure that a MSF cannot be located to deliver service at an unavailable node. Constraints (3.5)-(3.7) define the relation between X_{ijmt} and Y_{imt} variables and ensure that if MSF m is at node i at time period t and travels from node i to node j at the end of time period t to be at node j at time period $t+1$, X_{ijmt} takes the value of 1, otherwise it is 0. Constraints (3.8) are the conservation of flow constraints for each MSF and each node at each time period. Constraints (3.9)- (3.11) define the domains of the variables.

The following symmetry breaking constraints for MSFs are also used to strengthen the mathematical model.

$$Z_{m+1} \leq Z_m \qquad \forall m = 1, ..., |M|-1 \qquad (3.12)$$

Proposition 3.4.1. *The integrality constraints for X and Z variables can be relaxed without loosing the feasibility and optimality.*

Path p	**Period** t	**Node order** k	LHS $\sum_{m\in M}\sum_{t'=0}^{\tau-1} Y_{(n_{p,k+t'+1}),m,t+k+t'}$	RHS d_{pt}
1	**1**	0	$Y_{111} + Y_{212} + Y_{313}$ (0) (1) (0)	1
		1	$Y_{212} + Y_{313} + Y_{514}$ (1) (0) (0)	
1	**2**	0	$Y_{112} + Y_{213} + Y_{314}$ (0) (1) (0)	0
		1	$Y_{213} + Y_{314} + Y_{515}$ (1) (0) (0)	

Table 3.5: The open form of constraints (3.3) for the optimal solution provided in Table 3.2 of the illustrative example in Section 3.2 where $\tau = 3$

Proof. In addition to the integrality constraints (3.10), in the original formulation, the value of X_{ijmt} for arc (i, j), MSF m, and period t is calculated through constraints (3.5)-(3.8) and variables Y_{imt} and $Y_{j,m,t+1}$. Suppose that we introduce $X_{ijmt} \geq 0$ and $X_{ijmt} \leq 1$ constraints $\forall (i, j) \in A_\nu, m \in M, t = 1, ..., |T| - 1$ instead of these integrality constraints. If at least one of Y_{imt} and $Y_{j,m,t+1}$ is zero, X_{ijmt} should be 0 by constraints (3.7) and (3.8) and the new constraint $X_{ijmt} \geq 0$. Otherwise (that is if both Y_{imt} and $Y_{j,m,t+1}$ are 1), then X_{ijmt} should be 1 by constraints (3.5) and the new constraint $X_{ijmt} \leq 1$. This completes the proof of the proposition for X_{ijmt} variables.

In addition to the integrality constraints (3.11), in the original formulation, the value of any Z_m for a MSF m is calculated through constraints (3.2) and (3.4). Suppose that we introduce $Z_m \geq 0$ and $Z_m \leq 1$ constraints $\forall m \in M$ instead of these integrality constraints. Constraints (3.2) guarantee that Z_m cannot take a non-integer value for any m since Y_{imt} is binary $\forall i \in V_\nu, m \in M, t \in T$. As a result, Z_m can only be 0 or 1 through the new constraints $Z_m \geq 0$ and $Z_m \leq 1$.

Let us consider the values of X_{ijmt} case by case according to the possible values of Y_{imt} and $Y_{jm(t+1)}$ (0 and 1). *Case 1: At least one of Y_{imt} and $Y_{jm(t+1)}$ is 0:* According to constraints (3.6) and (3.7), X_{ijmt} must be 0, as X_{ijmt} cannot take a value less than 0. *Case 2: Both Y_{imt} and $Y_{jm(t+1)}$ are 1:* Constraint (3.5) turns out to be $1 + X_{ijmt} \geq 1 + 1$. As a result, X_{ijmt} must be 1, as X_{ijmt} cannot take a value greater than 1. Finally, there is no other case for these constraints and X_{ijmt} can be 0 or 1.

The value of any Z_m is determined by constraints (3.2) and (3.4). Constraint (3.2) guarantees that Z_m variable cannot take a fractional value since Y_{imt} is binary. Additionally, only the lower and upper bounds of Z_m has been relaxed. Therefore, Z_m still has an upper bound and it is 1. As a result, Z_m can only be 0 or 1. $\square$

Chapter 4

SOLUTION METHOD: AN ADAPTIVE LARGE NEIGHBORHOOD SEARCH IMPLEMENTATION

The MM-FLP-MD rapidly gets intractable as the number of refugee groups entering the network and the length of paths increase. We develop a problem-specific *Adaptive Large Neighbourhood Search* (ALNS) algorithm for this problem.

4.1 ALNS Algorithm

The *Large Neighbourhood Search (LNS)* meta-heuristic algorithm, proposed by [Shaw, 1998], is often used by researchers for tackling VRP problems in the literature. The LNS framework is a continuous application of a destroy and a repair heuristic, where destroy heuristics eliminate a part of the current solution (relaxation) and repair heuristics make the destroyed current solution feasible again (re-optimization). The ALNS, introduced by [Ropke and Pisinger, 2006], is an extension of the LNS heuristic and does not employ only one destroy and repair heuristic. Instead, at each iteration, it chooses a destroy and a repair heuristic from a pool of heuristics based on their past performance. According to [Pisinger and Ropke, 2019], ALNS framework generally performs well on routing problems such as VRP.

The proposed ALNS starts with an initial solution, s_0, constructed through a constructive heuristic described in Section 4.6. The current solution, s, and the best found solution, s_{best} are initialized as s_0. The algorithm uses a set of destroy heuristics, D, and a set of repair heuristics, R, described in Sections 4.3 and 4.4, respectively. In an iteration, a destroy heuristic $h_{destroy}$ and a repair heuristic h_{repair} is chosen through a roulette wheel selection procedure, explained in Section 4.5, based on the weights of the heuristics in the corresponding set. Initially, equal weights are assigned to the destroy/repair heuristic. A new solution, s', is obtained through the sequential application of $h_{destroy}$ and h_{repair} on s. The current solution s is set to the new solution s', if s' has a better total cost value than that of s or if s' is accepted based on the acceptance criteria of the Simulated Annealing, described in Section 4.8. The best found solution s_{best} is set to s', if s' improves s_{best}. The ALNS framework is strengthened by using a Local Search procedure that is applied once in every ζ iterations and is explained in Section 4.7. The weight adjustment of the destroy/repair heuristics is performed once in every η iterations based on their past performance as described in Section 4.5. The algorithm ends when the stopping criteria, described in Section 4.8, is met and returns s_{best} after the application of the local search on s_{best}. The pseudo-code of the proposed ALNS is provided in Algorithm 1.

Next, we provide preliminaries to facilitate the description of the details of the proposed ALNS.

4.2 Preliminaries

The proposed ALNS uses the following components to represent a solution. The illustrative example provided in Section 3.2 is used to clarify the description of these components.

Service Act Sequence: A finite set of service acts, ordered concerning their periods, is referred to as a *service act sequence* and is represented by ϕ. The i^{th} service act of the service act sequence ϕ is represented by $\sigma(\phi, i)$. In a service act sequence, there cannot exist two service acts that have the same period since it is not possible for a MSF to be located at different nodes at a period according to the problem definition.

A service act sequence for the optimal solution of the illustrative example provided in Section 3.2 is $\phi = \{(2,2),(2,4),(5,5)\}$, where the first service act of ϕ is $\sigma(\phi,1) = (2,2)$ and the length of ϕ is $|\phi| = 3$.

Time-Indexed Path: For a refugee group that enters path p at time t, the sequence of node-time pairs, which provides the location of the group at each period, in increasing order of their periods, is referred to as a *time-indexed path* and is denoted

Algorithm 1 ALNS Framework

Input: Initial Solution, s_0

$s \leftarrow s_0$
$s_{best} \leftarrow s_0$
$w_h \leftarrow 1/|R|,\ \forall h \in R$ // set initial weights of repair heuristics
$w_h \leftarrow 1/|D|,\ \forall h \in D$ // set initial weights of destroy heuristics
$iter \leftarrow 1$
while Stopping-criterion not met **do**
    if $iter\%\eta = 0$ **then** // weight adjustment once in η iterations
        $w_R \leftarrow Adjust(w_R)$
        $w_D \leftarrow Adjust(w_D)$
    end if
    if $iter\%\zeta = 0$ **then** // local search once in ζ iterations
        $s' \leftarrow LocalSearch(s)$
    end if
    $h_{repair} \leftarrow Select(w_R)$
    $h_{destroy} \leftarrow Select(w_D)$
    $s' \leftarrow h_{repair}(h_{destroy}(s))$
    if $f(s') \leq f(s_{best})$ **then**
        $s_{best} = s'$
    end if
    if $Accept(s', s)$ **then**
        $s \leftarrow s'$
    end if
    $iter \leftarrow iter + 1$
end while
$s_{best} \leftarrow LocalSearch(s_{best})$

Output: s_{best}

by $\pi(p,t)$. The i^{th} node-time pair of the time-indexed path π is denoted by $\sigma(\pi,i)$. The set of all time-indexed paths for an MM-FLP-MD instance is represented by Π.

In the illustrative example provided in Section 3.2, the time-indexed path of the refugee group entering path 2 at period 3 can be represented as $\pi(2,3)$=$\{(1,3),(2,4)$ $,(4,5),(5,6)\}$. The length of π is $|\pi| = 4$.

Subpath of a Time-Indexed Path: A subset of the node-time pairs of a time-indexed path π is called a subpath of π if the corresponding refugee group receives service at each node-time pair of the subpath i.e. each node-time pair corresponds to a service act, and the service requirement for the group is satisfied.

In the optimal solution of the illustrative example provided in Section 3.2, a subpath of the time-indexed path of the refugee group entering path 2 at period 3, i.e. $\pi(2,3)$, is $\{(2,4)\}$ as it covers the service requirement of the corresponding refugee group.

The following property provides the conditions for identifying a service act sequence as a subpath.

Property 4.2.1. ***Identification of a service act sequence as a subpath of time-indexed path π:*** *There are three conditions for a service act sequence ϕ to be qualified as a subpath of a time-indexed path π:*

1. *Each node-time pair corresponding to a service act in ϕ should be an element of π, i.e., $\sigma(\phi,i) \in \pi$*

2. $t(\sigma(\phi,i+1)) - t(\sigma(\phi,i)) \leq \tau$, $\forall i = 1,...,(|\phi|-1)$.

3. *The first and the last service acts of ϕ should be among the the first and the last τ node-time pairs of π, respectively, i.e. $\sigma(\phi,1) \in \{\sigma(\pi,1),...,\sigma(\pi,\tau)\}$ and $\sigma(\phi,|\phi|) \in \{\sigma(\pi,|\pi|-\tau+1),...,\sigma(\pi,|\pi|)\}$.*

In the illustrative example provided in Section 3.2 where $\tau = 3$, let us consider the time-indexed path $\pi = \{(1,1),(2,2),(4,3),(3,4),(5,5)\}$. In the optimal solution, for the service act sequence $\phi = \{(2,2),(5,5)\}$, condition (i) is satisfied since $\sigma(\phi,1) = (2,2), \sigma(\phi,2) = (5,5) \in \pi$, condition (ii) is satisfied since $t((5,5)) - t((2,2)) =$

$5-2=3 \leq 3$ and condition (iii) is satisfied since $(2,2) \in \{(1,1),(2,2),(4,3)\}$ and $(5,5) \in \{(4,3),(3,4),(5,5)\}$. Therefore, the service act sequence ϕ is a subpath of π.

Visit Schedule: For MSF $m \in M$, the sequence of node-time pairs that provides the facility location at each period, in increasing order of their periods, is referred to as a *visit schedule* and is denoted by v^m. If any node-time pair (n,t) of a visit schedule v^m exists in any time-indexed path π, i.e. the MSF m and the refugee group corresponding to π are at node n at period t, then it is assumed that the MSF m provides service to that refugee group, and (n,t) is indicated as a service act of v^m.

Note that when the facility does not move at the end of period t the location at consecutive periods t and $t+1$ will be the same. Additionally, if the first service act that a MSF provides is at period $t>1$, then it is assumed that the facility is located at the node of the first service act at each $1 \leq t' \leq t$.

Suppose that the i^{th} service act of the visit schedule v^m is represented by

$$\sigma(v^m,i) = (n(\sigma(v^m,i)), t(\sigma(v^m,i))).$$

Then, the node-time pairs up to (and including) the first service act of v^m can be represented by the sequence

$$(n(\sigma(v^m,1)),1), ..., (n(\sigma(v^m,1)), t(\sigma(v^m,1))-1), \sigma(v^m,1).$$

The node-time pairs between (and including) any two consecutive service acts, $\sigma(v^m,i)$ and $\sigma(v^m,i+1)$, of v^m can be represented by the sequence

$$\sigma(v^m,i), (n(\sigma(v^m,i)), t(\sigma(v^m,i))+1), ..., (n(\sigma(v^m,i)), t(\sigma(v^m,i+1))-1), \sigma(v^m,i+1).$$

The node-time pairs after (and including) the last service act, $\sigma(v^m,|v^m|)$, of v^m, where $|v^m|$ denotes the number of service acts of v^m, can be represented by the sequence

$$\sigma(v^m,|v^m|), (n(\sigma(v^m,|v^m|)), t(\sigma(v^m,|v^m|))+1), ..., (n(\sigma(v^m,|v^m|)), T)).$$

If a new service act $\sigma^{\text{new}} = (n^{\text{new}}, t^{\text{new}})$ is planned to be inserted in between any two consecutive service acts, $\sigma(v^m,i)$ and $\sigma(v^m,i+1)$, of v^m, then the above sequence should be updated as

$$\sigma(v^m, i), (n(\sigma(v^m, i)), t(\sigma(v^m, i)) + 1), ...,$$

$$(n(\sigma(v^m, i)), t^{\text{new}} - 1), \sigma^{\text{new}}, (n^{\text{new}}, t^{\text{new}} + 1), ..., (n^{\text{new}}, t(\sigma(v^m, i+1) - 1), \sigma(v^m, i+1).$$

If an existing service act, which is the i^{th} service act of the visit schedule v^m, is planned to be removed from v^m, then the sequence should be updated as

$$\sigma(v^m, i-1), (n(\sigma(v^m, i-1)), t(\sigma(v^m, i-1)) + 1), ...,$$

$$(n(\sigma(v^m, i-1)), t(\sigma(v^m, i+1)) - 1), \sigma(v^m, i+1).$$

As an example, in the optimal solution of the illustrative example provided in Section 3.2, the single facility's visit schedule is expressed as $v^1 = \{\mathbf{(2,1)}, (2,2), (2,3), (2,4), \mathbf{(5,5)}, (5,6)\}$, where the bold node-time pairs correspond to the service acts. If a new service act $(3,3)$ is planned to be inserted into v^1, then the visit schedule after insertion would be $\{\mathbf{(2,1)}, (2,2), \mathbf{(3,3)}, (3,4), \mathbf{(5,5)}, (5,6)\}$. If the existing service act $(5,5)$ is planned to be removed from v^1, then the visit schedule after removal would be $\{\mathbf{(2,1)}, (2,2), (2,3), (2,4), (2,5), (2,6)\}$.

Solution Representation: An MM-FLP-MD solution corresponds to the set of visit schedules for each MSF utilized, therefore, it can be represented as s=$\{v^1, v^2, ..., v^m\}$.

To provide a simple visualization of a solution, Table 4.1 gives the optimal solution of the illustrative example provided in Section 3.2. Visit schedule v^m of the utilized MSF m is provided for the whole planning horizon. For each service act σ of the visit schedule v^m, the table provides the location, i.e. $n(\sigma)$, of the MSF m at the corresponding period $t(\sigma)$. For the remaining node-time pairs, the corresponding cell of the table is empty.

For a solution of a problem instance, if there exists a time-indexed path that does not have a subpath in the solution, then the corresponding time-indexed path is called an *uncovered time-indexed path* and the solution is identified as *infeasible.* The solution is *feasible* if there exists at least one subpath for every time-indexed path, i.e. if there is no uncovered time-indexed path. Therefore, identifying subpaths of a time-indexed path plays a critical role in determining the feasibility of a solution

<table>
<tr><td rowspan="2">MSF</td><td colspan="6">Period</td><td>Total</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>Cost</td></tr>
<tr><td>1</td><td></td><td>2</td><td></td><td>2</td><td>5</td><td>5</td><td>80</td></tr>
<tr><td colspan="7">Objective Function Value</td><td>80</td></tr>
</table>

Table 4.1: The optimal solution of the illustrative example provided in Section 3.2.

and in converting an infeasible solution to a feasible solution. However, the number of subpaths of a time-indexed path π with length $|\pi|$ is $O(2^{|\pi|})$, therefore finding all subpaths of all time-indexed paths in Π may be time-consuming. The following property shows that some subpaths of a time-indexed path are *dominated* by others and so, can be neglected.

Property 4.2.2. *A subpath of a time-indexed path π represented by a service act sequence ϕ is dominated, if the service act sequence ϕ' that is obtained by removing the i^{th} service act $\sigma(\phi, i) \in \phi$ from ϕ for some $i \in \{2, ..., (|\phi| - 1)\}$, is still a subpath of π, i.e. $t(\sigma(\phi, i+1)) - t(\sigma(\phi, i-1)) \leq \tau$.*

Proof. From definition of dominating sets, we know that, $t(\sigma(\phi, i)) - t(\sigma(\phi, i-1)) \leq \tau$ and $t(\sigma(\phi, i+1)) - t(\sigma(\phi, i)) \leq \tau$, $\forall i \in 2, ..., (|\phi| - 1)$. Let $\bar{\phi}$ represent a service act sequence after we remove any $\sigma(\phi, i)$ from ϕ. In $\bar{\phi}$, $\sigma(\phi, i+1)$ and $\sigma(\phi, i-1)$ will be consecutive service acts and there are two possible cases: (i). $t(\sigma(\phi, i+1)) - t(\sigma(\phi, i-1)) > \tau$, where $\bar{\phi}$ does not satisfy Property 4.2.1 and therefore, it cannot be a subpath of π; (ii). $t(\sigma(\phi, i+1)) - t(\sigma(\phi, i-1)) \leq \tau$, where $\bar{\phi}$ satisfies Property 4.2.1 and therefore it is a subpath of π. Since ϕ contains $\bar{\phi}$ (another subpath), it is a dominated subpath by definition of dominating sets. □

Minimal (Non-dominated) Subpath of a Time-Indexed Path: The subpath, represented by the service act sequence ϕ, of a time-indexed path π is a minimal subpath, if removing any service act $\sigma(\phi, i)$ from ϕ, where $i \in \{2, ..., (|\phi| - 1)\}$, does not result in a subpath of π, i.e. $t(\sigma(\phi, i+1)) - t(\sigma(\phi, i-1)) > \tau$.

Let us consider the subpath represented by $\phi = \{(2, 2), (3, 4), (5, 5)\}$ of time-indexed path $\pi(3,1)$ that corresponds to the refugee group entering path 3 at period 1. If service act $(2,2)$ is removed, the resulting service act sequence would not be a subpath of π since Property 4.2.1 (iii) would be violated. If service act $(3,4)$ is removed, since $t(\sigma(\phi,1)) - t(\sigma(\phi,3)) = t((2,2)) - t((5,5)) = 3$ and that is equal to τ, the resulting subpath is another subpath of π. Therefore, ϕ is not a minimal subpath of π. Consider the resulting subpath $\phi' = \{(2,2),(5,5)\}$. Again, if service act $(2,2)$ is removed, the resulting service act sequence would not be a subpath of π since Property 4.2.1 (iii) would be violated. Likewise, if service act $(5,5)$ is removed, the resulting service act sequence would violate Property 4.2.1 (iii). Hence, ϕ' is a minimal subpath of π.

We propose an efficient constructive method to derive all minimal subpaths, referred to as Φ'_π, of a given time-indexed path π. The pseudocode of the constructive method is provided in Algorithm 2. The algorithm relies on Property 4.2.2. The algorithm starts with creating an empty service act sequence for each of the first τ node-time pairs of π and the corresponding node-time pair into the newly created service act sequence (lines 3-7). We note that the service act sequence is created only if service can be provided at the corresponding node (see line 4). Each of these new service act sequences is inserted into the set Φ_π, which was initially empty. In the upcoming steps (lines 8-23), the algorithm pops a service act sequence, say ϕ, from Φ_π and creates τ new service act sequences based on ϕ to be inserted into Φ_π as follows. Note the last service act in ϕ has a period of $t(\phi,|\phi|)$. According to Property 4.2.1, the period of the next service act to be inserted into the service act sequence should be within τ periods of the last service act of ϕ. Therefore, the algorithm creates τ new service act sequences from ϕ as follows. A new service act sequence $\bar{\phi}$ is obtained by appending a service act for one of the upcoming τ node-time pairs of π at the end of ϕ. We again note that $\bar{\phi}$ is created only if service can be provided at the corresponding node (see line 14). If $\bar{\phi}$ results in a dominated subpath of π (see Property 4.2.2), i.e. removing the second to last service act of $\bar{\phi}$ results in a subpath of π, then $\bar{\phi}$ is neglected. Otherwise, if the last service act of $\bar{\phi}$ corresponds to the last $(\tau-1)$ node-time pairs of π, i.e. the refugee group of π's service requirement can be covered through the service act sequence $\bar{\phi}$, then this service act sequence is a minimal subpath of π. In this case, the subpath is inserted into the set of subpaths of π, which is represented by Φ'_π. Otherwise, the service sequence $\bar{\phi}$ is placed into the set Φ_π. The algorithm continues until Φ_π is empty and ends with returning the minimal subpath set Φ'_π of π.

The following sections describe the components of the ALNS algorithm.

4.3 Destroy Heuristics

We, next, describe the three destroy heuristics of the proposed ALNS. Given the current solution s, each destroy heuristic obtains a destroyed solution by removing ϵ service acts from s, where ϵ is a random number having a discrete uniform distribution on the set $\{\underline{\epsilon}, \underline{\epsilon}+1, \ldots, \bar{\epsilon}\}$. In this set, $\underline{\epsilon}$ and $\bar{\epsilon}$ are calculated as rounding the numbers $b_1|s|$ and $b_2|s|$, respectively. We call b_1 and b_2 as destroy degree parameters.

4.3.1 Random Service Act Removal (RSAR)

The RSAR selects a random service act, provided by a MSF and removes it from the visit schedule of the corresponding MSF. This process is repeated until ϵ service acts are removed from the solution.

For the illustrative example provided in Section 3.2, suppose that given the current solution s presented in Table 4.2a, the randomly selected service acts to be removed are $(1,1)$, $(1,3)$, $(2,4)$, and $(5,6)$ (where $\epsilon = 4$). Table 4.2b represents the destroyed solution after RSAR is applied to s. We note that the destroyed solution is no longer feasible since there is no subpath for the time-indexed path $\pi(2,3) = \{(1,3),(2,4),(4,5),(5,6)\}$ that corresponds to the refugee group entering path 2 at period 3 in the destroyed solution and thus, the demand of $\pi(2,3)$ is not covered.

4.3.2 Random MSF Removal (RMFR)

The RMFR selects a random MSF and removes its visit schedule and the corresponding service acts from the solution. This process is repeated until at least ϵ service acts are removed from the solution.

Algorithm 2 Generate All Minimal Subpaths of a Time-Indexed Path $\pi(p,t) \in \Pi$

Input: a time-indexed path $\pi(p,t)$ that corresponds to the refugee group entering path p at period t

$\Phi_\pi \leftarrow \emptyset$ // Set of service act sequences
$\Phi'_\pi \leftarrow \emptyset$ // Set of subpaths
for $i = 1$ to τ **do**
  if $a_{n(\pi,i)} = 1$ **then**// If a MSF can provide service at the i^{th} node of π
    $\phi \leftarrow \emptyset$ // Create an empty service sequence ϕ
    $\phi \leftarrow \phi \cup \{\sigma(\pi,i)\}$ // Create a service act corresponding to the i^{th} node-time pair of π and insert into ϕ
    $\Phi_\pi \leftarrow \Phi_\pi \cup \{\phi\}$ // Insert π to the set of service act sequences
  end if
end for
while $\Phi_\pi \neq \emptyset$ **do**
  Pop the first service act sequence ϕ from Φ_π.
  for $i = 1$ to τ **do**
    if $a_{n(\pi,t'-t+i+1)} = 1$ **then**// If a MSF can serve at $(t'-t+i+1)^{\text{th}}$ node of π
      $\bar{\phi} \leftarrow \phi \cup \sigma(\pi, t'-t+i+1)$ // Create a new service act sequence by adding a new service act (the $(t'-t+i+1)^{\text{th}}$ node-time pair of π) at the end of ϕ
      if $t(\bar{\phi},|\bar{\phi}|) - t(\bar{\phi},|\bar{\phi}-2|) \leq \tau$ **then** // Removing the second to last $((|\bar{\phi}|-1)^{\text{th}})$ service act of $\bar{\phi}$ results in a subpath of π
        Neglect $\bar{\phi}$ since it is dominated
      else if $t(\bar{\phi},|\bar{\phi}|) + \tau - 1 \geq t(\pi,|\pi|)$ **then** // The last service act of $\bar{\phi}$ corresponds to the last $(\tau-1)$ node-time pairs of π, therefore ϕ is a subpath
        $\Phi'_\pi \leftarrow \Phi'_\pi \cup \{\bar{\phi}\}$ // Insert $\bar{\phi}$ into subpath set Φ'_π
      else
        $\Phi_\pi \leftarrow \Phi_\pi \cup \{\bar{\phi}\}$ // Insert $\bar{\phi}$ into service act sequence set Φ_π
      end if
    end if
  end for
end while

Output: Φ'_π: the set of minimal subpaths of time-indexed path π.

MSF	Period						Total Cost
	1	2	3	4	5	6	
1	1	2	3				70
2				5			50
3				3	5		70
4			1	2		5	90
Objective Function Value							280

(a) current solution, s

MSF	Period						Total Cost
	1	2	3	4	5	6	
1		2	3				60
2				5			50
3				3	5		70
Objective Function Value							180

(b) destroyed solution

Table 4.2: A sample application of RSAR

For the illustrative example provided in Section 3.2, suppose that Table 4.3a represents the current solution s and the randomly selected MSF is the third MSF with $\epsilon = 2$. Table 4.3b represents the destroyed solution after RMFR is applied. We note that the destroyed solution is no longer feasible since there is no subpath for the time-indexed path $\pi(2,3) = \{(1,3),(2,4),(4,5),(5,6)\}$.

MSF	Period						Cost
	1	2	3	4	5	6	
1		2	3				60
2				5		5	50
3			1	3	5		90
Objective Function							200

(a) current solution, s.

MSF	Period						Total Cost
	1	2	3	4	5	6	
1		2	3				60
2				5		5	50
Objective Function Value							110

(b) destroyed solution

Table 4.3: A sample application of RMFR.

4.3.3 Shaw Removal (SR)

According to the Shaw Removal heuristic, proposed by [Shaw, 1997] and [Shaw, 1998], it is more promising to remove the related components of the solution and a relatedness measure is used to determine the related components to be removed. In the SR of the proposed ALNS, service acts are removed according to a relatedness measure, R. The relatedness of two service acts σ_1 and σ_2, $R(\sigma_1, \sigma_2)$, is calculated through Equation 4.1 as a weighted sum of three criteria: (i). normalized distance between their nodes, (ii). normalized time difference between them, and (iii). the similarity of the refugee groups that they serve. $R(\sigma_1, \sigma_2)$ being close to 0 indicate that the service acts σ_1 and σ_2 are more related.

$$R(\sigma_1, \sigma_2) = w_1 \frac{c_{n(\sigma_1),n(\sigma_2)}}{\max_{(i,j)\in A}\{c_{ij}\}} + w_2 \frac{|t(\sigma_1) - t(\sigma_2)|}{T} + w_3 \Big(1 - \frac{2h(\sigma_1, \sigma_2)}{H(\sigma_1, \sigma_2)}\Big) \tag{4.1}$$

In this definition, $h(\sigma_1, \sigma_2)$ denotes the number of refugee groups that both σ_1 and σ_2 serve and $H(\sigma_1, \sigma_2)$ provides the summation of the number of refugee groups served by σ_1 or σ_2.

As an example, suppose that σ_1 and σ_2 are the two service acts that serve the set of refugee groups $\{1, 2, 3\}$ and $\{2, 3, 4\}$, respectively. Since refugee groups 2 and 3 are served by both σ_1 and σ_2, $h(\sigma_1, \sigma_2) = 2$ and since each of these service acts serve 3 refugee groups, $H(\sigma_1, \sigma_2) = 3+3 = 6$. Therefore, the third term of the relatedness measure can be calculated as $1 - \frac{2h(\sigma_1,\sigma_2)}{H(\sigma_1,\sigma_2)} = 1 - \frac{4}{6} = 0.33$. In this example

SR starts with removing a random service act σ from the solution and inserting σ into the removed service acts set D, which is initially empty. Then, the most related service act to a random service act selected from D is removed and inserted into the set D. This process is repeated until at least ϵ service acts are removed from the solution.

4.4 Repair Heuristics

Next, we describe the four repair heuristics of the proposed ALNS. Given the destroyed solution, each repair heuristic, first, identifies the time-indexed paths that are not covered, i.e. the ones that do not have a subpath. Then, for each uncovered time-indexed path, the missing service acts of a subpath of the corresponding time-indexed path are inserted into the solution. In this step, the repair order of the uncovered time-indexed paths is determined randomly in all repair heuristics, except the regret insertion. A missing service act is inserted into the visit schedule of a MSF that results in the smallest increase in the objective.

We also note that, before performing a repair heuristic, an empty vehicle is inserted into the solution to prevent MSF capacity infeasibility in the second step while inserting new service acts. At the end of the repair heuristic, the newly inserted vehicle is removed if it does not have a service act in the recovered solution.

4.4.1 Random Subpath Insertion (RSI)

For an uncovered time-indexed path π, the RSI selects a random minimal subpath of π, referred to as $\widetilde{\phi}_\pi$, among the set of minimal subpaths of π, Φ'_π. The service acts of $\widetilde{\phi}_\pi$ that do not exist in the solution are inserted into the visit schedules of the MSFs of the solution starting from the first service act of $\widetilde{\phi}_\pi$.

For the illustrative example provided in Section 3.2, suppose that Table 4.2b represents the destroyed solution to be repaired through the RSI. We note that the time-indexed path $\pi(2,3) = \{(1,3),\ (2,4),\ (4,5),\ (5,6)\}$ is not covered in the destroyed solution. Suppose that the minimal subpath $\{(1,3),(5,6)\}$ is the random subpath selected among the subpaths in Φ'_π. Both $(1,3)$ and $(5,6)$ are missing in the destroyed solution. It is impossible to add service act $(1,3)$ to the visit schedule of the first MSF since there is already a service act of this facility at period 3. Inserting the service act $(1,3)$ to the visit schedule of the second MSF and to the visit schedule of the third MSF increases the total cost by 40 and 20, respectively. Since forth, i.e. new, the MSF is not used in the solution, inserting a service act into its visit schedule incurs a fixed cost of 50. The service act $(1,3)$ is inserted into the visit schedule of the third MSF 3 since this inclusion increases the total cost least. Next, the service act $(5, 6)$ is inserted into the visit schedule of the second (or third) MSF without any increase in the objective function value, as the last visited node before period 6 is node 5. There is no more uncovered time-indexed path after this step. The recovered solution is presented in Table 4.4.

MSF	Period						Cost
	1	2	3	4	5	6	
1		2	3				60
2				5		5	50
3			1	3	5		90
Objective Function Value							200

Table 4.4: The solution after applying the RSI to the destroyed solution presented in Table 4.2b.

4.4.2 Least Cost Subpath Insertion (LCSI)

For an uncovered time-indexed path π, the LCSI selects the minimal subpath of π, referred to as $\widetilde{\phi}_\pi$, that has the least traveling cost in itself, among the set of minimal subpaths of π, Φ'_π. The service acts of $\widetilde{\phi}_\pi$ that do not exist in the solution are inserted into the visit schedules of the MSFs of the solution starting from the first service act of $\widetilde{\phi}_\pi$.

For the illustrative example provided in Section 3.2, suppose that Table 4.3b represents the destroyed solution to be repaired through the LCSI. We note that the time-indexed path $\pi(2,3) = \{(1,3),(2,4), (4,5),(5,6)\}$ is not covered in the destroyed solution. The set of minimal subpaths of $\pi(2,3)$ is $\Phi'_\pi = \big\{\{(1,3),(5,6)\}, \{(2,4)\}\big\}$. Considering all possible insertions to the visit schedules of all MSFs, the best cost increases of the subpaths in Φ'_π are 40, and 0, in the given element order of the set. Minimal subpath $\{(2,4)\}$ is selected among the minimal subpaths since it has a cost increase of 0. The resulting recovered solution is represented in Table

4.5.

MSF	Period						Total
	1	2	3	4	5	6	Cost
1		2	3	2			70
2				5		5	50
Objective Function Value							120

Table 4.5: The recovered solution after applying the LCSI to the destroyed solution presented in Table 4.3b.

4.4.3 Worst Insertion (WI)

For an uncovered time-indexed path π, the WI selects the worst subpath of π, referred to as $\widetilde{\phi}_{\pi}$. The worst, i.e. the most costly, subpath of π is itself. Therefore, service acts of π that do not exist in the solution are inserted into the visit schedules of the MSFs of the solution starting from the first service act of π.

For the illustrative example provided in Section 3.2, suppose that Table 4.3b represents the destroyed solution to be repaired through the WI. As the time-indexed path $\pi(2,3) = \{(1,3),(2,4),(4,5),(5,6)\}$ is not covered in the destroyed solution, the WI inserts all missing service acts of $\pi(2,3)$ into the solution. Note that $(4,5)$ is not a service act, since $a_4 = 0$. The resulting recovered solution is represented in Table 4.6.

4.4.4 n-Regret Insertion (RI-n)

Regret heuristics were introduced by [Potvin and Rousseau, 1993] for the vehicle routing problem with time. The aim of a regret heuristic is enhancing the solution quality by integrating future information with greedy searches [Ropke and Pisinger, 2006]. In the RI-n developed for the proposed ALNS, all the minimal subpaths of an uncovered time-indexed path π are ordered in the non-decreasing order of their change in the objective function value when their missing service acts are inserted

MSF	Period						Total
	1	2	3	4	5	6	Cost
1		2	3	2			70
2			1	5		5	90
Objective Function Value							160

Table 4.6: The recovered solution after applying the WI to the destroyed solution presented in Table 4.3b.

into the solution. The k^{th} subpath in this order is the k^{th} best subpath of π and is denoted by $\phi^{\pi,k}$. Let $\Delta z_{\phi^{\pi,k}}$ be the objective function value change when the missing service acts of $\phi^{\pi,k}$ of π are inserted into the solution.

Regret value represents the importance of the a subpath. The "regret" term represents the case where it is impossible to insert current best subpath in the future and to cover the time-indexed path again other, supposedly worse, subpaths should be used.

Let $\phi^{\pi,k}$ denote the k^{th} best subpath of the time-indexed path π. Here, the best subpath means the subpath of π that increases the objective function value the least. Thus, $\phi^{\pi,1}$ is the best subpath of π to add to the solution.

In the basic version where $n = 2$, referred to as RI-2, the uncovered time-index path, whose subpath to be inserted next, is selected where the cost difference between $\Delta z_{\phi^{\pi,2}}$ and $\Delta z_{\phi^{\pi,1}}$ is the largest.

Then, in each iteration, depending on the regret heuristic used, i.e. the value of n, the regret heuristic chooses the uncovered time-index path, π^*, whose subpath to be inserted next, according to Equation 4.2:

$$\pi^* = \underset{\pi\in\Pi}{\text{argmax}} \left\{ \sum_{k=2}^{\min\{n,|\Phi'_\pi|\}} \left(\Delta z_{\phi^{\pi,k}} - z_{\phi^{\pi,1}}\right) \right\} \tag{4.2}$$

The minimal subpath of π^* that results in the smallest increase in the objective function value, among the set of minimal subpaths of π^* is identified and its unvisited service acts are inserted into the solution.

This process continues until all uncovered time-indexed paths are covered.

4.5 Selection of Destroy and Repair Heuristics and Adaptive Weight Determination

A destroy/repair heuristic may perform well on an instance but may not do so in other instances. Using different destroy and repair heuristics in different iterations, we adjust our algorithm for problem instances.

At each iteration, the roulette wheel selection procedure that allocates each heuristic an angle proportional to its weight in the roulette is used to choose a destroy and a repair heuristic among the sets D and R, respectively.

Each destroy or repair heuristic $i \in D \cup R$ has a weight, ω_i, and initial weights of the heuristics in D and R are set to $1/D$ and $1/R$, respectively.

The weight adjustment of the destroy/repair heuristics is performed at the end of every η iterations based on their past performance as follows. Let β_i denote the score of heuristic $i \in D \cup R$ and suppose that starting from the first iteration, the iterations of the algorithm are grouped into disjoint iteration segments, each including η consecutive iterations.

Destroy and repair heuristic weights are adjusted automatically in the ALNS framework by their performances in the earlier iterations. Quality of a heuristic can be measured by its diversification and intensification success. Therefore, if a heuristic performs well in terms of intensification and diversification, it gets a higher weight. The weights are determined at the end of every η iterations. This iteration group is called a segment. The number of iterations during each segment is decided by parameter tuning.

At the end of each iteration, if the recovered solution, s', is accepted according to the acceptance criteria, the scores of the destroy and repair heuristics used in the corresponding iteration are increased by ϵ_1, if s' improves the incumbent solution, s_{best}; by ϵ_2, if s' improves only the current solution, s; or by ϵ_3, if s' improves neither s_{best} nor s but is still accepted.

We note that the first two cases reward intensification and the last one rewards diversification. If a resulting solution is accepted, it cannot be known whether this is due to the performance of destroy heuristic or the repair heuristic. Therefore, both destroy and repair heuristics are rewarded.

After each block of η iterations, the weight of each heuristic $i \in D \cup R$ is recalculated using Equation 4.3 a constant ρ, the scores of the heuristics, the number of usages and the weights in the previous segment. In the first segment, all weights are set to be equal. Then, for each heuristic i,

$$\omega_i = (1-\rho)\omega_i + \rho\frac{\beta_i}{\upsilon_i} \tag{4.3}$$

where υ_i represents the number of times heuristic i is used in the last η iterations and ρ is referred to as the *reaction factor* in the literature [Ropke and Pisinger, 2006] that allows to control how quickly the weight adjustment algorithm reacts to the changes in scores.

After all weights are updated, the weights of all destroy and repair heuristics are normalized in their own sets, i.e. in the sets D or R and all scores are set to zero to track the performance of the heuristics in the upcoming η iterations.

4.6 Initial Solution Construction - Constructive Heuristic

The ALNS algorithm starts with an initial solution s_0 as described in Algorithm 1. We employ the following simple constructive heuristic, referred to as CH, to obtain s_0.

We have tried several constructive heuristics. A natural one is to start the algorithm with the best solution obtained by a constructive algorithm. However, solutions that are not the best may also be good candidates for diversification. Therefore, a heuristic that sets the best balance between intensification and diversification is chosen as the initial solution heuristic as explained next.

Starting with an empty solution, at each iteration the CH randomly selects a time-indexed path $\pi \in \Pi$. If π is already covered, the algorithm continues with the next iteration. Otherwise, the minimal subpath of π, referred to as $\widetilde{\phi}_\pi$, that results in the smallest increase in the objective function value, among the set of minimal subpaths of π, Φ'_π is identified. The service acts of $\widetilde{\phi}_\pi$ that do not exist in the current partial solution are inserted into the visit schedules of the MSFs of the current partial solution starting from the first service act of $\widetilde{\phi}_\pi$. This process continues until all time-indexed paths in Π are covered.

As the last step of CH, the visit schedules of all MSFs used by the solution constructed are traversed so that if there exists a node-time pair (n, t) of any visit schedule v^m that also exists in any time-indexed path π, i.e. the MSF m and the refugee group corresponding to π are at node n at period t, then (n, t) is indicated as a service act of v^m.

4.7 Local Search (LS)

In our implementation, local search (LS) takes a solution and tries to improve it without leaving the feasible region. As defined in the Section 4.5, same segment definition with different number of iterations is defined in the Local Search, as well. LS is performed once in every ζ iterations. It is formed of two steps: service act removal and service act transfer/swap.

In the first step, i.e. service act removal, starting from the first service act of the first visit schedule, all service acts of the current solution s are traversed to check if removing the service act results in an infeasibility, i.e. an uncovered time-indexed path. If removing a service act σ from the visit schedule v^m does not result in an infeasibility, then σ is removed from the visit schedule v^m. Additionally, while a feasibility check is being performed the node-time pairs in which a refugee group and a MSF is met is placed into the solution as a service act.

In the second step, i.e. service act transfer/swap, starting from the first service act of the first visit schedule, all service acts of the current solution s are traversed to check if a better solution can be obtained through transferring the service act of a visit schedule to another visit schedule or through swapping the service act of a visit schedule with a service act of another visit schedule as follows. For a service act σ of a visit schedule v^m, all other visit schedules are evaluated for a potential transfer or swap. If there is no service act in period $t(\sigma_i)$ in a visit schedule $v^{m'}$, then the service act σ is transferred to, i.e. removed from v^m and inserted into, the visit schedule of $v^{m'}$. Otherwise, that is if there is a service act, say σ', in period $t(\sigma_i)$ of the visit schedule $v^{m'}$, then the service acts σ and σ' are removed from v^m and $v^{m'}$, respectively, and σ is inserted into $v^{m'}$ and σ' is inserted into v^m. Suppose that $\Delta z_{\sigma,m,m'}$ denote the change in the objective function value when the service act σ of the visit schedule m is transferred to or swapped with a service act of the visit schedule m'. If $\Delta z_{\sigma,m,m'}$ is negative for some m', i.e. the objective function value improves by the change, σ is transferred to or swapped with a service act of the visit schedule m^*, where $m^* = \arg\min_{m'} \Delta z_{\sigma,m,m'}$.

Consider the recovered solution provided in Table 4.5. In the service act removal step, the LS starts from the first service act of the first visit schedule, which is $(2, 2)$. The service act $(2, 2)$ is not removed, since the time-indexed path $\pi(2, 1)$ would not be covered if it is removed. However, it is feasible to remove the service act $(3, 3)$. The solution after this removal is presented in Table 4.7a. It is also feasible to remove the service act $(5, 4)$. However, it is not feasible to remove the service acts $(5, 5)$, and $(5, 6)$. The solution obtained after this step is presented in Table 4.7b.

MSF	Period						Total
	1	2	3	4	5	6	Cost
1		2		2			50
2				5	5	5	50
Objective Function Value							100

(a) after removing service act $(3, 3)$

MSF	Period						Total
	1	2	3	4	5	6	Cost
1		2		2			50
2					5	5	50
Objective Function Value							100

(b) after service act removal step

Table 4.7: A sample application of LS on the solution provided in Table 4.5.

Continuing with service act transfer/swap on the solution presented in Table 4.7b, the service act $(2, 2)$ can be transferred to the visit schedule of the second MSF since there is no service act in v^2 at that period, $(2, 2)$. Similarly, service act $(2, 4)$ can be transferred to v^2. After this transfer, the solution uses only one facility, and therefore, no more service act can be transferred or swapped. The resulting optimal solution is presented in Table 4.8.

MSF	Period						Total
	1	2	3	4	5	6	Cost
1		2		2	5	5	80
Objective Function Value							80

Table 4.8: The solution after string swap heuristic.

4.8 Acceptance and Stopping Criteria

The proposed ALNS employs the acceptance criteria of simulated annealing. Therefore, an improving solution s' obtained in an iteration is accepted with probability 1 and a non-improving solution s' is accepted with probability $e^{\frac{z(s)-z(s')}{temp_{iter}}}$, where $z(s)$ is the objective function value of solution s and $temp_{iter}$ is the temperature of the iteration. The initial temperature is set as $temp_0 = \frac{z(s_0)}{ln(2)}$, where s_0 is the initial solution. The initial temperature has been set to this value to increase the chance of selecting a non-improving movement at the beginning of the iterations and search larger solution space. In each iteration $iter$, temperature is updated as $temp_{iter} = \alpha\ temp_{iter-1}$, where α is the cooling rate.

The ALNS algorithm uses the maximum number of iterations, χ_1, or the maximum number of non-improving iterations χ_2 as the stopping criteria.

Chapter 5

CASE STUDY: 2018 HONDURAS MIGRATION CRISIS

We applied the proposed solution approach on the real life case of Honduras Migration Caravan Crisis in 2018. In October 2018, a refugee movement has started from Central America to USA. People from Honduras, Guatemala and Nicaragua gathered to leave their country because of various reasons such as poverty, crime rates in their country and political pressures. Their first goal was to seek a new life with better living conditions and better jobs. On October 12th, the group started their march with 160 people from San Pedro Sula, Honduras. With the help of social media and other communication channels, the march became popular and the number of supporters escalated rapidly. It is estimated that more than 7000 people attended the walk. By mid-November, more than 1500 refugees arrived to Tijuana, Mexico, a US-Mexico border city. Red Cross organizations and other humanitarian groups (such as American Friends Service Committee and Kino Border Initiative) served aid to the refugees on their path, mainly on a local basis. The New Sanctuary Coalition formed a new sanctuary caravan that helped people on their path. However, according to the interviews with the refugees, the aid service was not sufficient and some refugees had never been served [Jacob Rascon, 2018].

5.1 Data Generation

According to several news sources [Center for Immigration Studies, 2020], the refugee groups followed the *main* path shown in Figure 5.1. However, refugees were separated into smaller groups according to their travel modes, such as walking, public transportation, private cars, and sometimes hitchhiking. Some groups combined all these modes. For example, an initially walking group found free public transportation at some point, stopped walking for a certain amount of distance and returned to walking again at another point.

Network Nodes: Network nodes are determined using the path followed by walking refugees since walking is the slowest travel mode used by refugees. For both MSFs and refugees, city centers or urban areas are naturally safe places for camping. For refugees, it is easier to reach basic needs in cities. Moreover, the news sources report the number of the people and movement of the main group according to the cities. Therefore, the nodes of the network are primarily selected from cities on the main path.

Recall the assumption that a refugee group can traverse a link between the cities in 1 period. In the data generation process of this study, 1 period is determined as 1 day. To create the network of the main path, the news sources are examined daily and the locations of the refugee groups are noted. Reported cities are selected as the primary nodes according to the news, where MSFs can be placed on.

Secondary nodes, which are considered as camping points of refugee groups, are inserted between the primary nodes when there is more than 1 day between two reports. These insertions are made so that the arc between any two consecutive nodes can be traveled within one day by a walking refugee group. To be more consistent on the secondary node creating process, it is assumed that a person travels between 30-50 miles a day [Jacob Rascon, 2018]. However, it is not possible to place an MSF on secondary nodes.

As a result, we have $|V| = 30$ nodes in the main network $G = (V, A)$, 21 of which correspond to the city centers, where MSFs can locate, i.e. $|V_\nu| = 21$ and the remaining ones are the secondary nodes.

Network Links and Paths: Arcs of the network are determined so that they can be traversed by refugees using different travel modes in one period, which is a day. As a result, different travel modes lead to different paths on the network.

Due to security forces between countries, it is mostly harder to cross the borders. Since a large number of refugees do not have the right to pass the borders, they tend to use illegal ways, which slows them down. Hence, the arcs that cross the borders are considered to be shorter in terms of the distance for the paths of the refugees that do not have rights.

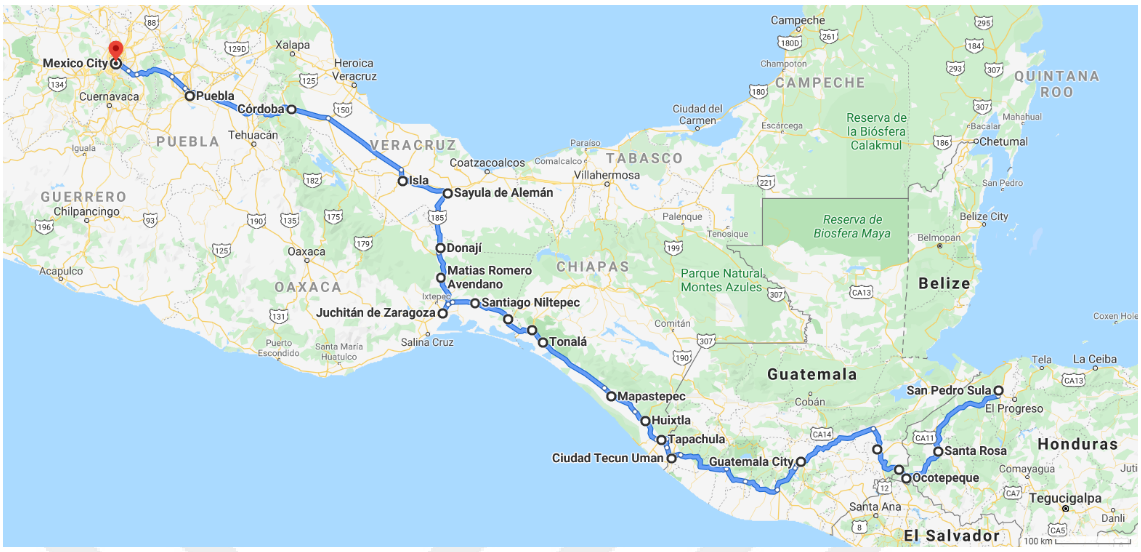


Figure 5.1: Map of Honduras Case

Government actions, tough conditions, and new opportunities convince some groups of refugees to end their travel before they reach the destination point on the main path. As a result, the ending nodes of the paths differ and some of the paths are shorter in terms of distance.

As a result, we have $P = 13$ refugee paths and $A_\rho = 42$ refugee arcs in the main network.

The MSF network is assumed to be complete, i.e. there is a bi-directional arc between each pair of nodes in V_ν. As V_ν=21, there are 420 arcs in A_ν. The cost of an arc $(i, j) \in A_\nu$, c_{ij}, is the shortest distance between the nodes i and j.

MSFs: The fixed cost of using a MSF, f_m, is determined as 600 based on a fixed cost analysis presented in Section 5.5. An upper bound on the number of MSFs is determined according to the number of refugee groups enter the system. First, number of MSF is set to the number of refugee groups. Then if a better solution is found with a less number of MSF, the number of MSF is decreased accordingly.

Planning Horizon: $|T|$ is equal to the period of the last refugee group that leaves the system. The service requirement level, τ is decided as 3 according to a service requirement level analysis provided in Section 5.4.

Test Instances: In addition to the main network, we generated two other networks based on the main network that correspond to the first 10 and 20 nodes of the main path to analyze different size instances. The large-size data set includes instances that use the main network. The medium and small-size data sets include instances that use the networks with 20 and 10 nodes, respectively. There are 25 instances in each data set and the instances of a data set differ in terms of the number of refugee groups or paths.

The name of each instance can be decomposed into three parts: the first part starts with the letter "P" and provides the number of different paths, the second part starts with the letter "E" and provides the number of refugee groups, and the third part starts with the letter 'N" and provides the number of nodes in the longest path of the instance. To differentiate the instances having the same number of paths, the same number of refugee groups and the same number of nodes in the longest path, instances are indexed. For example, the instance named P6E13N30_1, has 13 refugee groups that enter 6 different paths where the longest path has 30 nodes and this is the first instance among the instances with the same properties.

The refugee network of Honduras Case that corresponds to the instances P6E13-N30_1 in large-size data set is presented in Figure 5.2. Green nodes represent the camping nodes where the mobile facilities can be located. Red nodes are camp-only locations, i.e. secondary nodes. Nodes are labeled with node numbers. For the primary nodes, the corresponding city names are also provided. There are 6 paths in this instance and the nodes of each path are provided in Appendix Table A.2. The information of the refugee groups entering these paths are provided in Appendix Table A.1. The time-horizon of this instance can be calculated as 36, since the refugee group that enters the network on path 1 at period 6, leaves the system at period 36 and there is no other refugee group in the network after this period. Since there is no refugee group entering to the network after period 6, the periods after 6 are not presented in Appendix Table A.1.

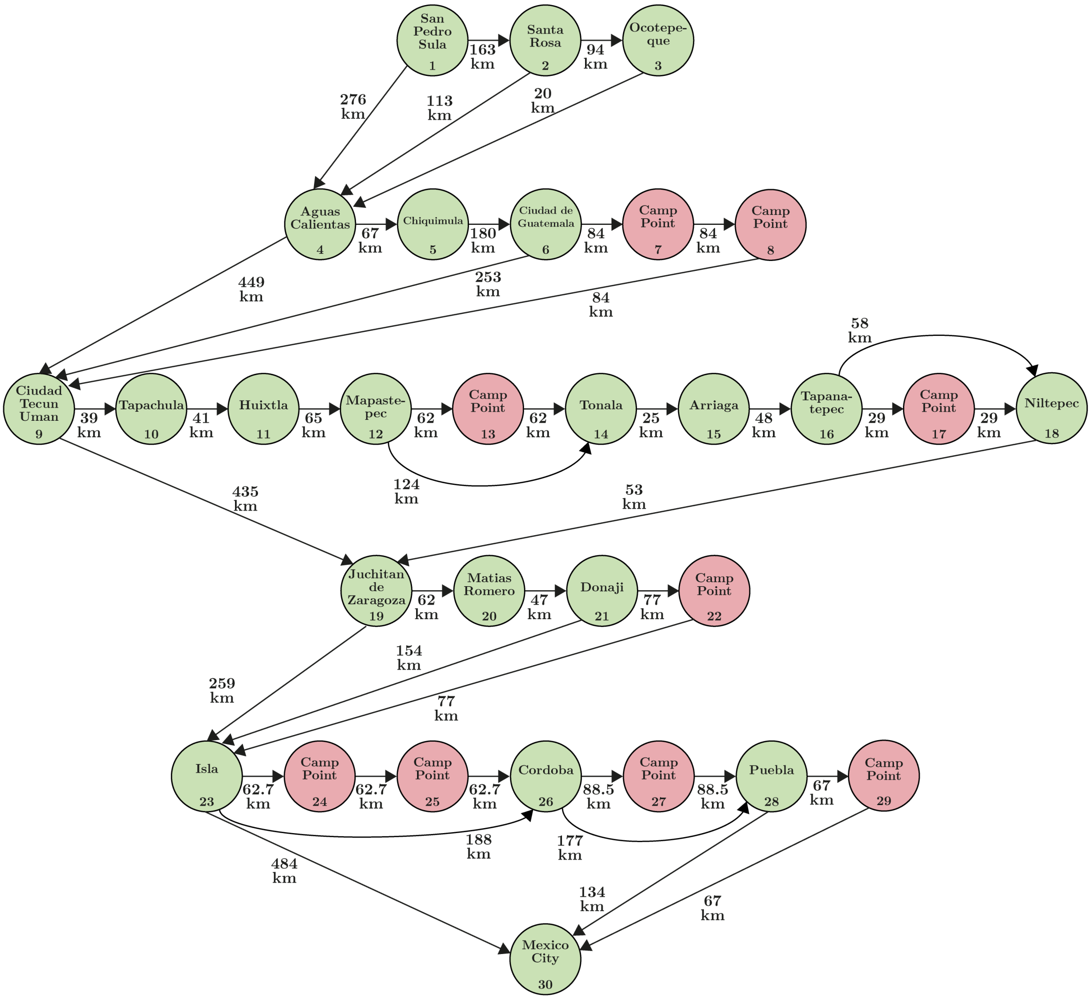


Figure 5.2: The Refugee Network of Honduras Case (Instance "P6E13N30")

5.2 *Computational Performance of ALNS*

In this section, we provide our computational experiments. ALNS performance is evaluated by comparison with the optimal solution of the MILP formulation in Section 3.4 for small, medium data sets, and with the best found total cost values for large-size data sets. Then, we study the performance of each algorithm component. Finally, we analyze our numerical results concerning service frequency and fixed costs.

We implemented the proposed ALNS in Java with Intellij IDEA 2019.3.3 programming platform and conducted all experiments on a workstation with 3.3 GHz

Intel Xeon PC with 16 processors and 32 GB of internal memory. We solved the mathematical model using CPLEX 12.9, with a run time limit of 4 h and an optimality gap limit of 0.01%.

5.2.1 Parameters and Parameter Tuning

The proposed approach uses 15 parameters: number of iterations (χ_1), number of non-improving iterations (χ_2), weight parameters in SR (w_1, w_2 and w_3), regret level of RI-n (n), cooling rate of acceptance criteria (α), the parameters that indicate the score accrual of the heuristics (ϵ_1, ϵ_2 and ϵ_3), reaction factor in heuristic weight adjustment (ρ), the periods in number of iterations of heuristic weight adjustment and local search (η and ζ, respectively), destroy degree parameters (b_1 and b_2).

Medium-sized instances were used as tuning instances. The parameter tuning process started with a trial-error step to find relatively acceptable parameters. Various values were tested for a specific parameter, while other parameters were kept fixed. The parameter is tuned with the value that gives the best average gap among the test instances. The process then continued with the next parameter without changing the tuned parameters. The process was performed for each parameter once.

The parameter vector (χ_1, χ_2 w_1, w_2, w_3, n, α, ϵ_1, ϵ_2, ϵ_3, ρ, η, ζ, b_1, b_2) is set as (25000, 3000, 1, 1, 1, 4, 0.999, 50, 20, 12, 0.1, 50, 25, 0.2, 0.5) after parameter tuning. However, ALNS framework is quite flexible in terms of parameter values, i.e., yield optimal solutions for most of cases. In small-size instances even in 1000 non-improving iterations, ALNS was able to find the optimal solution. Therefore, for small-size instances, χ_2 was set to 1000.

We conducted 8 ALNS replications running in parallel for each instance in our analysis. The best solution and the longest run time of these replications are reported.

5.2.2 Evaluation of the performance of the ALNS heuristic

To evaluate the performance of the proposed ALNS heuristic, we solve the mathematical model (MILP) provided in Section 3.4 and the ALNS approach on small and medium-size instances. Large-sized instances were computationally too expensive for obtaining optimal solution with CPLEX. In such cases, the execution time of CPLEX was restricted to 4 hours and the best feasible solution was reported with lower bound values.

For small-size instances, both MILP and ALNS find optimal solutions in less than 6 seconds. The objective function values of small-size instances are reported in Appendix B.1.

For some instances of medium-size data set, the MILP cannot find the optimal solution within 4 h run time limit. For these instances, the optimal objective function value is found through running the MILP warm starting from the best solution found by the ALNS without a run time limit. Table 5.1a reports the optimality gap of the best found solutions and the run times of the ALNS and MILP (in columns ALNS and MILP) under *Gap (%)* and *CPU Sec.* subtitles, respectively. The last column reports the number of replications in which the optimal solution of the corresponding instance found over 8 replications. The optimal objective function values of medium-size instances are reported in Appendix B.2.

According to the results in Table 5.1a, the MILP cannot find the optimal solution within the run time limit of 4 h for some of the medium-size instances with a maximum and an average gap of 4.8 % and 0.65 %, respectively, whereas the ALNS finds the optimal solution for medium-size instances in less than 7 min on average and less than 30 min at maximum.

Table 5.1b presents the results for the large-size instances and reports the run times of the ALNS and MILP (in columns ALNS and MILP) under *CPU Sec.* subtitles. A run-time limit of 4 h is used for both approaches. Warm starting the MILP with the ALNS solution does not result in an improved solution for large-size instances. Therefore, the percentage of improvement in the objective value provided by the ALNS compared to the MILP is provided for each instance in column *Diff*

Instance	**Gap (%)**		**CPU Sec.**		**# of**
	ALNS	**MILP**	**ALNS**	**MILP**	**Opt.**
P01E01N20_1	0.00	0.00	4	8	8
P01E02N20_1	0.00	0.00	10	5	8
P02E02N20_1	0.00	0.00	4	9	8
P02E04N20_1	0.00	0.00	16	21	8
P02E04N20_2	0.00	0.00	18	4222	8
P03E03N20_1	0.00	0.00	37	78	8
P03E04N20_1	0.00	0.00	155	3730	8
P03E04N20_2	0.00	0.00	40	254	8
P03E06N20_1	0.00	0.00	404	14402	6
P03E06N20_2	0.00	0.00	179	1678	8
P03E06N20_3	0.00	4.35	401	14418	3
P03E06N20_4	0.00	0.00	294	14412	3
P03E06N20_5	0.00	0.00	94	4370	8
P04E04N20_1	0.00	0.00	37	305	8
P04E08N20_1	0.00	0.00	209	1230	8
P04E08N20_2	0.00	0.00	431	11292	3
P04E08N20_3	0.00	4.80	350	14402	2
P04E08N20_4	0.00	1.49	453	14416	1
P04E08N20_5	0.00	0.03	213	14402	3
P04E10N20_1	0.00	0.34	980	14407	8
P04E10N20_2	0.00	0.00	857	13361	1
P06E10N20_1	0.00	1.28	758	14415	1
P06E10N20_2	0.00	2.50	647	14414	8
P06E10N20_3	0.00	0.76	810	14409	6
P06E13N20_1	0.00	0.78	1704	14403	8
Min.	0.00	0.00	4	5	1
Max.	0.00	4.80	1704	14418	8
Avg.	0.00	0.65	364	7963	5.96

(a) medium-size instances

Instance	**Diff**	**CPU Sec.**	
	(%)	**ALNS**	**MILP**
P01E01N30_1	0.00	7	183
P01E02N30_1	0.00	32	34
P02E02N30_1	0.00	15	5532
P02E04N30_1	0.00	119	10121
P02E04N30_2	0.15	198	14400
P03E03N30_1	5.80	535	14400
P03E04N30_1	7.31	2007	14400
P03E04N30_2	3.17	1618	14400
P03E06N30_1	22.89	5919	14400
P03E06N30_2	3.32	5196	14400
P03E06N30_3	19.24	5769	14400
P03E06N30_4	9.37	4573	14400
P03E06N30_5	6.61	708	14400
P04E04N30_1	0.00	906	14400
P04E08N30_1	6.20	7452	14400
P04E08N30_2	22.28	14400	14400
P04E08N30_3	26.41	11185	14400
P04E08N30_4	11.34	12372	14400
P04E08N30_5	22.05	10220	14400
P04E10N30_1	12.79	14400	14400
P04E10N30_2	21.40	14400	14400
P06E10N30_1	17.92	14400	14400
P06E10N30_2	26.77	14400	14400
P06E10N30_3	23.29	14400	14400
P06E13N30_1	10.27	14400	14400
Min.	0.00	7	34
Max.	26.77	14400	14400
Avg.	11.14	6785	12731

(b) large-size instances

Table 5.1: The ALNS and MILP Results for Medium and Large-Size Instances

(%).

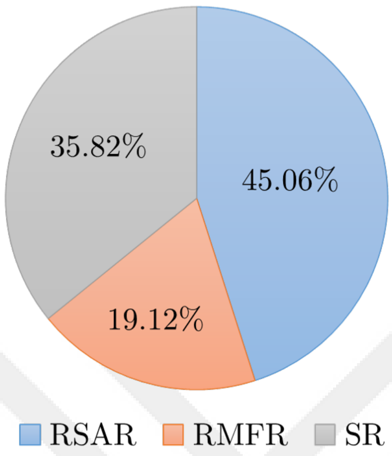
According to the results presented in Table 5.1b, for each instance, the best-found solution of the ALNS is at least as good as that of the MILP. found the same or better solutions in all instances. For 5 of the 25 large-size instances, both the MILP and ALNS find the optimal solution, however, the run time of the ALNS is considerably better than that of the MILP. For all large-size instances, the total cost of the best found ALNS solutions are on the average 11.14 % better than that of the MILP solutions, whereas the average run time of the ALNS is nearly half of that of the MILP. The best-found objective function values of large-size instances are reported in Appendix B.3.

5.3 Evaluation of the Performance of the ALNS Components

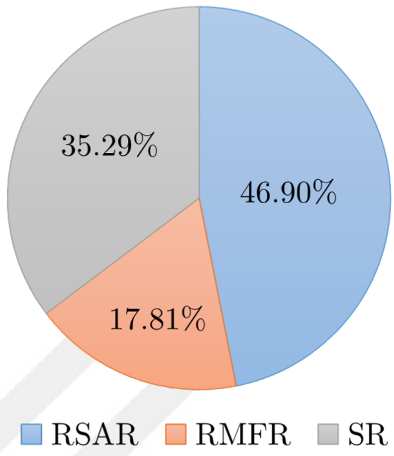
Next, we evaluate the performance of the ALNS components such as destroy and repair heuristics and local search, disjointly. The evaluations are based on large-size instances. As mentioned in Section 4.5, when an iteration of ALNS is accepted, scores of both destroy and repair heuristic are increased for that iteration. This is because it is not possible to find if the acceptance is caused by the performance of the destroy heuristic or repair heuristic. Therefore, we compare the heuristics within their own results and not with each other.

Figures 5.3a, 5.3b, 5.3c and 5.3d provide the percentage of iterations that each destroy heuristic used, led to an incumbent solution, led to an improving solution (compared to the current solution), and led to a non-improving but accepted solution, respectively, among all iterations for all instances. We note that for each instance, the statistics of the replication that lead to the best found solution is used to obtain the pie charts.

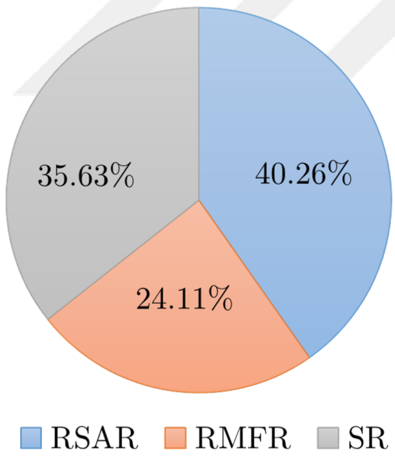
According to Figures 5.3a and 5.3b, the destroy heuristic that has the best performance in leading to an incumbent solution is the RSAR and therefore, it is the most frequently used destroy heuristic among all. The RMFR is the worst performance and the least frequently used destroy heuristic. According to Figure 5.3c, the RSAR, and SR have very close performance in leading to an improving solution.



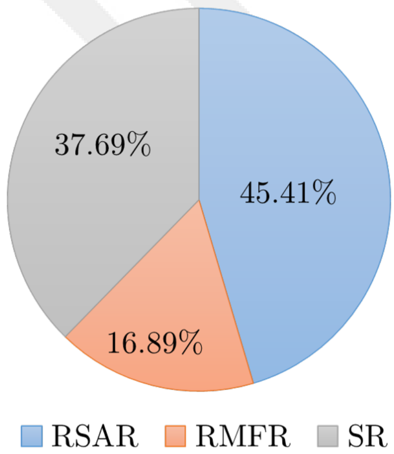
(a) used



(b) led to an incumbent solution



(c) led to an improving solution



(d) led to a non-improving but accepted solution

Figure 5.3: Percentage of iterations that each destroy heuristic used, led to an incumbent solution, led to an incumbent solution, and led to a non-improving but accepted solution

Figure 5.3d demonstrates that the acceptance probability of the solutions obtained through using the RMFR is considerably lower compared to the others. This shows that the RMFR tends to lead solutions that are worse in total cost. Thus, the score for this heuristic cannot be increased and making it harder to be selected in later iterations.

Figures 5.4a, 5.4b, 5.4c and 5.4d provide the percentage of iterations that each repair heuristic used, led to an incumbent solution, led to an improving solution (compared to the current solution), and led to a non-improving but accepted solution, respectively, among all iterations for all instances.

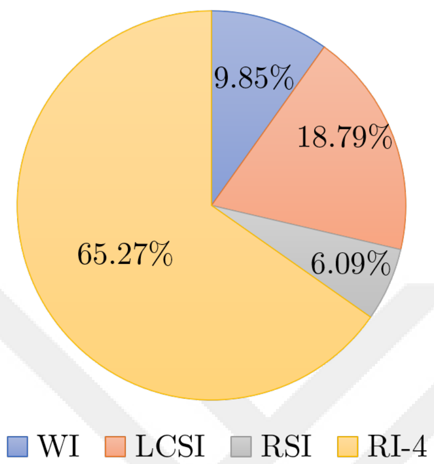
According to Figures 5.4a-5.4d, the RI-4 dominates the others in all cases. Although the RI-4 is used in $\tilde{\%}$ 62 percent of the iterations, the incumbent solutions found throughout all iterations are generated by the use of it in more than %90 percent of the cases. The RSI is the worst performing repair heuristic. Results are similar for Figures 5.4c and 5.4d, as well.

Local Search is another important component of the proposed ALNS algorithm. In 2 out of 25 large-size instances, the optimal solution was obtained directly by the CH. In 23 out of 25 large-size instances, on average, in %46 of the iterations in which an incumbent solution was obtained, Local Search was the reason for finding the new incumbent.

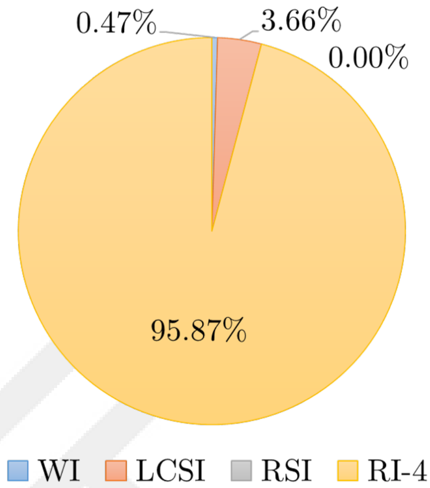
5.4 *Service Frequency Analysis*

According to the service requirement, every refugee group should be served at least once in every consecutive τ periods. Therefore, *tau* that determines the service frequency is an important parameter for the MM-FLP-MD problem. We, next, the change in the best found objective value for varying levels of service frequency. Figure 5.5 presents the average objective function value and the average decrease rate in the objective function value for varying levels of τ for the optimal solutions of the medium-size instances.

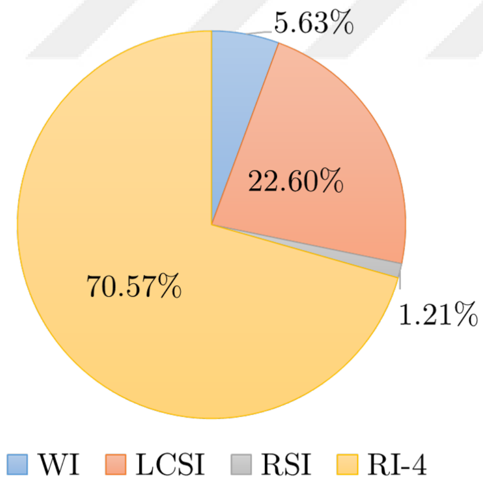
We note that for two MM-FLP-MD problem instances with all the same parameter values except τ, the instance with $\tau = x + 1$ is a relaxation of the instance with



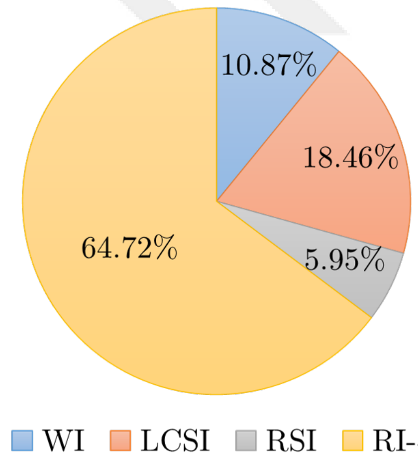
(a) used



(b) led to an incumbent solution



(c) led to an improving solution



(d) led to a non-improving but accepted solution

Figure 5.4: Percentage of iterations for each repair heuristic

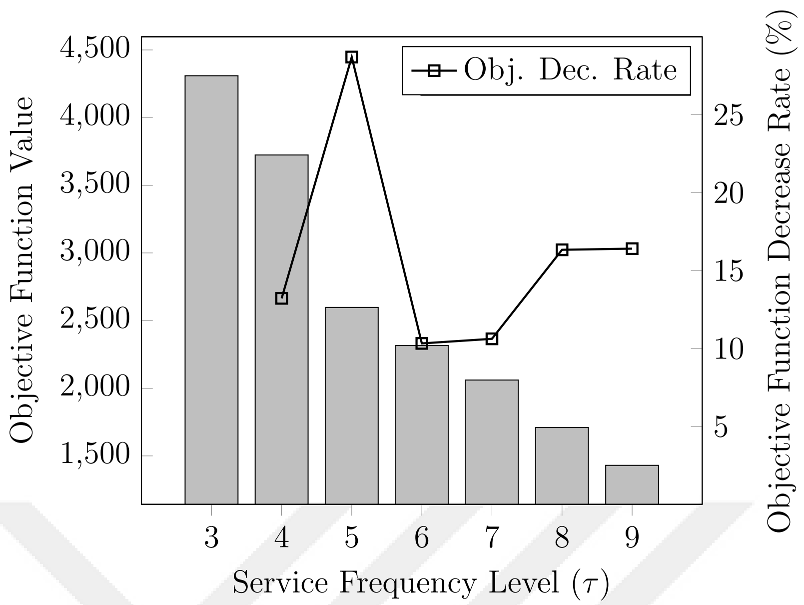


Figure 5.5: The change in the optimal objective value and objective function value decrease rate for varying levels of τ for medium-size instances

$\tau = x$ in terms of complexity since the required number of service acts and so the number of subpaths of time-indexed paths are smaller for the instance with $\tau = x+1$ compared to those for the instance with $\tau = x$. Therefore, as seen in Figure 5.5, the average optimal objective function value decreases as the service requirement is relaxed, i.e. as τ increases.

Between some levels, marginal decrease rates change dramatically, for example between $\tau = 4$ and $\tau = 5$. In instances of the problem, the increase in the level of τ extinguishes the requirement of covering paths which have length l_p less than τ. As a result, some of the entries do not need to be covered. Attempting to decrease the objective function value through increasing τ should be considered carefully since this change may trigger negative consequences in real life, such as not providing enough care for walking refugees.

Decision of service frequency level is critical and the results show that the solution methodology can be used effectively for such analysis. If the service frequency is too high, i.e. τ is too small, then the optimal objective value, i.e. optimal total cost, is too large. If it is too large, then, demand from the refugee groups may not be satisfied regularly enough and the control may be lost.

5.5 *Fixed Cost Analysis*

As provided in Equation 3.1, the objective function of a solution consists of two terms: the total fixed cost of mobile facilities used and the total transportation cost. The fixed cost of a mobile facility, f_m, corresponds to its operating costs that include costs such as vehicle renting cost, the salary of the operator, and vehicle maintenance cost. Next, we analyze the change in the optimal objective value and the optimal number of mobile facilities used for varying levels of the fixed cost of vehicles. Figure 5.6 presents the average percentage of the total fixed cost within the total cost and the average number of mobile facilities used for varying levels of τ for the optimal solutions of the medium-size instances.

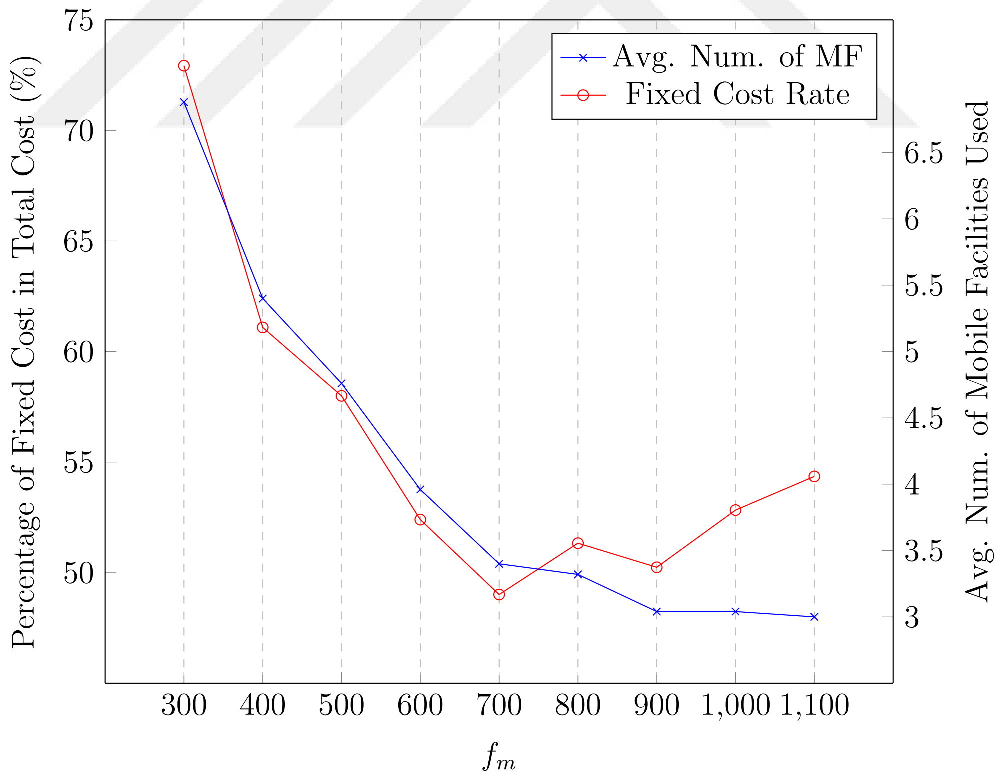


Figure 5.6: Fixed Cost Analysis for Medium Size Instances

According to Figure 5.6, the contribution of the total fixed cost to the total cost and the average number of mobile facilities used are the largest for the smallest value of f_m, and both decrease as f_m increases up to some point ($f_m = 800$). This is because when the facility fixed cost is low, it is better to locate each facility at a node and not to move them throughout the planning horizon which leads to zero transportation cost for the corresponding facilities. In addition, as expected, the average number of mobile facilities used decreases as the fixed cost mobile facilities increases.

After $f_m = 800$, although the average number of mobile facilities used remains the same as f_m increases, the contribution of the total fixed cost to the total cost increases. For $f_m >= 800$, it is impossible to use less number of mobile facilities before the problem becomes infeasible.

5.6 Insights on Largest Instance

In this section, the best found solution for the largest of the large-size instances, P6E13N30_1, is focused. As explained in Appendix A, 6 paths are in this instance including one path that contains all the nodes, one path that contains all the nodes except the camp points, one path that contains only metropolitan cities and three path that finish before reaching the last node. In this instance, 13 refugee groups enter the network on different periods. Since the last refugee group leaves the network on period 35, planning horizon consists of 35 periods. 9 nodes out of 30 are camping points, i.e. no mobile facility can be placed on these nodes. Recall that the fixed cost and service requirement level are determined as 600 and 3, respectively. Some representations of the best solution can be found in Appendix E.

Solution Properties: The best solution found by ALNS uses 3 mobile facilities. 51 different service acts are provided by these vehicles according to the solution and 15 different nodes are used as service points. The objective function is calculated as 4899. Since the fixed cost is equal to 1800, total traveling cost is equal to 3099 according to this solution. On nodes 3 (Ocotepeque) and 6 (Ciudad de Guatemala), the maximum number of groups that a service is provided is achieved with 12 different groups. On nodes 9 (Ciudad Tecun Uman), 12 (Mapastepec) and 26 (Cordoba), the maximum number of service acts performed is seen with 6 service acts.

Utilization Levels of Mobile Facilities: According to the solution, since there are 51 service acts performed in 35 periods, on average, 1.46 mobile facilities are used in every period. On 6 out of 35 periods, all 3 of the mobile facilities are used. The usage levels of vehicles are 54, 60 and 31 percent for each mobile facility in planning horizon, respectively. Thereof, it can be stated that, on average, the utilization level of a mobile facility is 0.49 per cent in this solution. Additionally, the travelling cost for each facility is 1006, 1173 and 920, respectively. The utilization levels of the mobile facilities seem to be unfair. However, when we compare the total travel cost, the total distances travelled by the mobile facilities constitute the fairness between the mobile facilities. The fairness can be constituted with mathematical model in which a fairness term can be added to objective by which difference between mobile facilities can be penalized.

Fixed Facility Strategy: Consider the strategy where we do not use any mobile facilities, but all the service acts must be performed with fixed facilities. When we force the algorithm to solve such kind of an instance, we get 10 fixed mobile facilities as service nodes. Thus, total cost would be 10*600=6000 which is %22.5 greater than current objective function. Furthermore, the average utilization would be equal to 15% which is 34% less than the current solution. As a result, using mobile facilities is better compared to using only fixed facilities in terms of both total cost and utilization.

Fixing Mostly Used Facility: In another strategy, we use the best found solution and fix an additional mobile facility to one of the nodes where the maximum number of service acts is performed. 3 nodes facilitate this feature: 9 (Ciudad Tecun Uman), 12 (Mapastepec) and 26 (Cordoba). When the solution is modified to fix a mobile facility to these nodes, the objective function appears as 5421, 5492 and 5496, respectively. Even in the best case, the objective function increases at least %10 units.

Fairness Between Refugee Groups: It is required in the problem definition for a refugee group to be provided a service at least once in every consecutive prefixed number (τ) of periods. According to this instance, a refugee group must be served at least $\left\lfloor \frac{l_p}{\tau} \right\rfloor$ times, since the number of periods that the group is in the network is equal to the length of their path. When we examine the service acts that is served to refugee groups, 11 out of 13 refugee groups are provided exactly this amount of service acts. As a result, only 15% of the refugee groups are provided more service than they need. We can see such kind of a provided excess service for the refugee group 10. The excess service for this groups is provided, since, when there is a service act is required to satisfy the service level requirement for another refugee group between periods 22 and 24, this group is at the same node coincidentally. Additionally, this provision of service does not increase the cost.

Start Point: Even though all 13 of the refugee groups start their travel from the first node (San Pedro Sula), there is no service provided on this node. Therefore, this solution organizes the operations the aid for the cities in between the start and destination points.

Chapter 6

CONCLUSIONS

In this study, an optimization problem arising in humanitarian operations is addressed. The problem involves the provision of relief aid in the form of services or essential needs to en route refugees as they march to their targeted destinations. Providing aid to mobile refugee groups is a crucial mission from a humanitarian perspective and serves to diminish security problems, as well. To be more effective, it is also critical to control and help the refugee groups regularly. Mobile service facilities are generally used as a practical solution to achieve this goal since they are more practical compared to fixed-location facilities. Nevertheless, selecting the locations of the facilities over time to provide periodic services as refugee groups advance towards their destinations on various paths is a complex optimization problem to solve. This thesis focuses on this problem.

One of the most important characteristics of the problem is the service level requirement. It guarantees that each refugee group is served at least once in each predetermined number of consecutive periods. We show that the problem is an $\mathcal{NP}$-hard problem. In the literature, there are some problem definitions close to MM-FLP-MD, however, up to our knowledge, this is the first problem to consider an aid setting with mobile facilities and moving demand under a predetermined service frequency level.

Two solution methods are developed: an exact MILP formulation and a problem-specific ALNS algorithm as a heuristic. The performance of these two methods are tested and compared computationally using a case study. Data for the case study is generated from the 2018 Honduras Migrant Caravan crisis mainly by extracting information from the news articles.

While the MILP formulation can solve the problem exactly in small instances, it is impractical in terms of run times when the size of the instances grows to represent real cases. The proposed ALNS algorithm overcomes this problem. For the tested medium and large-size instances, the run-time limit is determined as 4 hours, since the 4-hour limit is tolerable while operations need to proceed. The ALNS algorithm finds the optimal solution for all the generated instances of medium-size problems. In 20 out of 25 large-size instances, ALNS found better solutions compared to the MILP model. In the rest of the 5 instances, both methods found the same objective function values. Additionally, the objective function values of the best solutions found by ALNS is %11.14 better than the objective function values of the best solutions found by MILP.

The proposed ALNS algorithm consists of three destroy heuristics, and four repair heuristics. In each iteration, a destroy, and a repair heuristic is selected from the destroy and repair heuristic pools according to the probabilities determined by the success rates of the heuristics in the earlier iterations. The search is supported by a local search heuristic, as well.

The fixed cost level is an important feature of the problem. As the fixed cost increases, the number of mobile service facilities used in the solutions decreases. However, the fixed cost percentage of the total cost has a U-shaped graph. The service requirement level is another important parameter of the problem. An increase in the frequency of service, which determines the service level in some sense, decreases the total cost. It is important for decision-makers to determine the service requirement level considering humanitarian, security, and budget aspects.

The solution of the largest instance in the data set is analyzed in the last section. A comparison between strategies that allow only mobile facilities and that allow only fixed-location facilities is made and the mobile facility strategy is assessed to be 22.5% more efficient than the fixed-location strategy in terms of objective function. Furthermore, according to the solution, the fairness between the refugee groups is examined. The minimum number of service acts that a refugee group needs can be calculated according to the length of the refugee group's path and the service level requirement. In terms of fairness, 11 out of 13 groups are provided the minimum number of services they could be provided.

Future Works:

In the definition of MM-FLP-MD, the paths of the moving refugee groups and the refugee group entrances to a path in each period are assumed to be known. However, in real life, this may not be the case since the refugee groups are usually decide their paths on the way. Therefore, stochastic versions of MM-FLP-MD in which the paths and entrances to paths are assumed to be uncertain can be explored as an extension of the problem addressed in this thesis as future work. Moreover, a multi-scenario approach can be a natural further direction, where paths may change with respect to weather/political conditions.

In another research direction, mobile facilities may have an upper-bound on the number of people they can serve in a period, making the problem a capacitated one. In this case, the number of people in each refugee group would also be an important parameter of the problem, and there may be a need for more than one vehicle to serve a refugee group.

Finally, the case where more than one humanitarian organization provides aid and coordinates their efforts can be studied. The effectiveness of the operations may be increased with by centralized optimization. The multi-agent extension of this problem can be a vital study when the operations can be more effective with with the use of less resources.

BIBLIOGRAPHY

[Ahmadi-Javid et al., 2017] Ahmadi-Javid, A., Seyedi, P., and Syam, S. S. (2017). A survey of healthcare facility location. *Computers & Operations Research*, 79:223–263.

[Ahmadian et al., 2013] Ahmadian, S., Friggstad, Z., and Swamy, C. (2013). Local-search based approximation algorithms for mobile facility location problems. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1607–1621. SIAM.

[Alkan Group, 2020] Alkan Group (2020). Özel amaçlı mobil projeler. `https://www.alkangroup.com.tr/projelerimiz/ozel-amacli-mobil-projeler/`. (last accessed on May 24th, 2020).

[Anari et al., 2016] Anari, N., Fazli, M., Ghodsi, M., and Safari, M. (2016). Euclidean movement minimization. *Journal of Combinatorial Optimization*, 32(2):354–367.

[Arabani and Farahani, 2012] Arabani, A. B. and Farahani, R. Z. (2012). Facility location dynamics: An overview of classifications and applications. *Computers & Industrial Engineering*, 62(1):408–420.

[Ares et al., 2016] Ares, J. N., De Vries, H., and Huisman, D. (2016). A column generation approach for locating roadside clinics in africa based on effectiveness and equity. *European Journal of Operational Research*, 254(3):1002–1016.

[Armon et al., 2014] Armon, A., Gamzu, I., and Segev, D. (2014). Mobile facility location: combinatorial filtering via weighted occupancy. *Journal of Combinatorial Optimization*, 28(2):358–375.

[Balcik and Beamon, 2008] Balcik, B. and Beamon, B. M. (2008). Facility location in humanitarian relief. *International Journal of Logistics*, 11(2):101–121.

[Balinski, 1965] Balinski, M. L. (1965). Integer programming: methods, uses, computations. *Management science*, 12(3):253–313.

[Ballou, 1968] Ballou, R. H. (1968). Dynamic warehouse location analysis. *Journal of Marketing Research*, 5(3):271–276.

[Bastia and Skeldon, 2020] Bastia, T. and Skeldon, R. (2020). *Routledge Handbook of Migration and Development*. Routledge.

[Bayram, 2016] Bayram, V. (2016). Optimization models for large scale network evacuation planning and management: A literature review. *Surveys in Operations Research and Management Science*, 21(2):63–84.

[Berman, 1997] Berman, O. (1997). Deterministic flow-demand location problems. *Journal of the Operational Research Society*, 48(1):75–81.

[Berman et al., 1995] Berman, O., Bertsimas, D., and Larson, R. C. (1995). Locating discretionary service facilities, ii: maximizing market size, minimizing inconvenience. *Operations Research*, 43(4):623–632.

[Berman and Krass, 1998] Berman, O. and Krass, D. (1998). Flow intercepting spatial interaction model: a new approach to optimal location of competitive facilities. *Location science*, 6(1-4):41–65.

[Berman et al., 1992] Berman, O., Larson, R. C., and Fouska, N. (1992). Optimal location of discretionary service facilities. *Transportation Science*, 26(3):201–211.

[Boccia et al., 2009] Boccia, M., Sforza, A., and Sterle, C. (2009). Flow intercepting facility location: Problems, models and heuristics. *Journal of Mathematical Modelling and Algorithms*, 8(1):35–79.

[Boffey and Narula, 1998] Boffey, B. and Narula, S. C. (1998). Models for multi-path covering-routing problems. *Annals of Operations Research*, 82:331–342.

[Brešar et al., 2011] Brešar, B., Kardoš, F., Katrenič, J., and Semanišin, G. (2011). Minimum k-path vertex cover. *Discrete Applied Mathematics*, 159(12):1189–1195.

[Burgess, 2004] Burgess, A. (2004). Health challenges for refugees and immigrants. *Refugee Reports*, 25(2):1–3.

[Campbell, 1990] Campbell, J. F. (1990). Locating transportation terminals to serve an expanding demand. *Transportation Research Part B: Methodological*, 24(3):173–192.

[Center for Immigration Studies, 2020] Center for Immigration Studies (2020). Migrant caravan map. `https://cis.org/Migrant-Caravan-Map`. (Last Access: 2020-06-16).

[Comellas et al., 2015] Comellas, R. M., Makhashvili, N., Chikovani, I., Patel, V., McKee, M., Bisson, J., and Roberts, B. (2015). Patterns of somatic distress among conflict-affected persons in the republic of georgia. *Journal of Psychosomatic Research*, 78(5):466–471.

[Daskin et al., 1992] Daskin, M. S., Hopp, W. J., and Medina, B. (1992). Forecast horizons and dynamic facility location planning. *Annals of Operations Research*, 40(1):125–151.

[de Vries et al., 2020] de Vries, H., van de Klundert, J., and Wagelmans, A. P. (2020). The roadside healthcare facility location problem a managerial network design challenge. *Production and Operations Management*, 29(5):1165–1187.

[Demaine et al., 2009] Demaine, E. D., Hajiaghayi, M., Mahini, H., Sayedi-Roshkhar, A. S., Oveisgharan, S., and Zadimoghaddam, M. (2009). Minimizing movement. *ACM Transactions on Algorithms (TALG)*, 5(3):1–30.

[Durukan Sonmez, 2012] Durukan Sonmez, A. (2012). *Facility Location and Relocation Problem: Models and Decomposition Algorithms*. PhD thesis.

[Farahani et al., 2009a] Farahani, R. Z., Abedian, M., and Sharahi, S. (2009a). Dynamic facility location problem. In *Facility Location*, pages 347–372. Springer.

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

[Farahani et al., 2009b] Farahani, R. Z., Drezner, Z., and Asgari, N. (2009b). Single facility location and relocation problem with time dependent weights and discrete planning horizon. *Annals of Operations Research*, 167(1):353–368.

[Friggstad and Salavatipour, 2011] Friggstad, Z. and Salavatipour, M. R. (2011). Minimizing movement in mobile facility location problems. *ACM Transactions on Algorithms (TALG)*, 7(3):1–22.

[Gendreau et al., 2000] Gendreau, M., Laporte, G., and Parent, I. (2000). Heuristics for the location of inspection stations on a network. *Naval Research Logistics (NRL)*, 47(4):287–303.

[Gendreau et al., 1997] Gendreau, M., Laporte, G., and Semet, F. (1997). The covering tour problem. *Operations Research*, 45(4):568–576.

[Groen, 2016] Groen, D. (2016). Simulating refugee movements: Where would you go?

[Groen, 2018] Groen, D. (2018). Development of a multiscale simulation approach for forced migration. In *International Conference on Computational Science*, pages 869–875. Springer.

[Gulland, 2015] Gulland, A. (2015). The refugee crisis: what care is needed and how can doctors help? *Bmj*, 351:h4881.

[Güneş et al., 2019] Güneş, E. D., Melo, T., and Nickel, S. (2019). Location problems in healthcare. In *Location science*, pages 657–686. Springer.

[Gzara and Erkut, 2009] Gzara, F. and Erkut, E. (2009). A lagrangian relaxation approach to large-scale flow interception problems. *European Journal of Operational Research*, 198(2):405–411.

[Haghani and Oh, 1996] Haghani, A. and Oh, S.-C. (1996). Formulation and solution of a multi-commodity, multi-modal network flow model for disaster relief operations. *Transportation Research Part A: Policy and Practice*, 30(3):231–250.

[Halper et al., 2015] Halper, R., Raghavan, S., and Sahin, M. (2015). Local search heuristics for the mobile facility location problem. *Computers & Operations Research*, 62:210–223.

[Hamacher and Nickel, 1998] Hamacher, H. W. and Nickel, S. (1998). Classification of location models. *Location Science*, 6(1-4):229–242.

[Hodgson, 1990] Hodgson, M. J. (1990). A flow-capturing location-allocation model. *Geographical Analysis*, 22(3):270–279.

[Hodgson et al., 1996] Hodgson, M. J., Rosing, K. E., and Zhang, J. (1996). Locating vehicle inspection stations to protect a transportation network. *Geographical Analysis*, 28(4):299–314.

[IFRC, 2020] IFRC (2020). Migration and displacement. `https://media.ifrc.org/ifrc/what-we-do/migration-and-displacement/`. (last accessed on May 23th, 2020).

[Jacob Rascon, 2018] Jacob Rascon (2018). Migrant caravan: Kprc2's jacob rascon reports in mexico. `https://www.click2houston.com/news/2018/11/01/migrant-caravan-kprc2s-jacob-rascon-reports-in-mexico/`. (Last Access: 2020-07-10).

[Jena et al., 2016] Jena, S. D., Cordeau, J.-F., and Gendron, B. (2016). Solving a dynamic facility location problem with partial closing and reopening. *Computers & Operations Research*, 67:143–154.

[Jena et al., 2017] Jena, S. D., Cordeau, J.-F., and Gendron, B. (2017). Lagrangian heuristics for large-scale dynamic facility location with generalized modular capacities. *INFORMS Journal on Computing*, 29(3):388–404.

[Kim and Song, 2012] Kim, H. and Song, H. Y. (2012). Formulating human mobility model in a form of continuous time markov chain. *Procedia Computer Science*, 10:389–396.

[Konstantinou et al., 2020] Konstantinou, T., Kepaptsoglou, K., and Kopelias, P. (2020). Mobile vehicle emission inspection service optimization: a flow intercepting location model and application. *Transportation Planning and Technology*, 43(2):174–187.

[Kuby and Lim, 2005] Kuby, M. and Lim, S. (2005). The flow-refueling location problem for alternative-fuel vehicles. *Socio-Economic Planning Sciences*, 39(2):125–145.

[Laporte et al., 2019] Laporte, G., Nickel, S., and Saldanha-da Gama, F. (2019). Introduction to location science. In *Location science*, pages 1–21. Springer.

[Maxwell et al., 2010] Maxwell, M. S., Restrepo, M., Henderson, S. G., and Topaloglu, H. (2010). Approximate dynamic programming for ambulance redeployment. *INFORMS Journal on Computing*, 22(2):266–281.

[Melo et al., 2009] Melo, M., Nickel, S., and da Gama, F. S. (2009). Facility location and supply chain management – a review. *European Journal of Operational Research*, 196(2):401 – 412.

[Melo et al., 2006] Melo, M. T., Nickel, S., and Da Gama, F. S. (2006). Dynamic multi-commodity capacitated facility location: a mathematical modeling framework for strategic supply chain planning. *Computers & Operations Research*, 33(1):181–208.

[Min and Melachrinoudis, 1999] Min, H. and Melachrinoudis, E. (1999). The relocation of a hybrid manufacturing/distribution facility from supply chain perspectives: a case study. *Omega*, 27(1):75–85.

[MRS, 2019] MRS (2019). Our network. `https://www.mobilerefugeesupport.org/our-network`.

[Nagurney, 1989] Nagurney, A. (1989). Migration equilibrium and variational inequalities. *Economics Letters*, 31(1):109–112.

[Nagurney, 1990] Nagurney, A. (1990). A network model of migration equilibrium with movement costs. *Mathematical and Computer Modelling*, 13(5):79–88.

[Nickel and Saldanha-da Gama, 2019] Nickel, S. and Saldanha-da Gama, F. (2019). Multi-period facility location. In *Location science*, pages 303–326. Springer.

[Novotnỳ, 2010] Novotnỳ, M. (2010). Design and analysis of a generalized canvas protocol. In *IFIP International Workshop on Information Security Theory and Practices*, pages 106–121. Springer.

[OHCHR, 2016] OHCHR (2016). Situations of migrants in transit. `https://www.ohchr.org/Documents/Issues/Migration/StudyMigrants/OHCHR_2016_Report-migrants-transit_EN.pdf`. (last accessed on February 28th, 2019).

[Open Arms, 2020] Open Arms (2020). Who are we? `https://www.openarms.es/en/who-are-we`. (last accessed on May 24th, 2020).

[Pan and Nagurney, 1994] Pan, J. and Nagurney, A. (1994). Using markov chains to model human migration in a network equilibrium framework. *Mathematical and Computer Modelling*, 19(11):31–39.

[Pisinger and Ropke, 2019] Pisinger, D. and Ropke, S. (2019). Large neighborhood search. In *Handbook of Metaheuristics*, pages 99–127. Springer.

[Potvin and Rousseau, 1993] Potvin, J.-Y. and Rousseau, J.-M. (1993). A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3):331–340.

[Raghavan et al., 2019] Raghavan, S., Sahin, M., and Salman, F. S. (2019). The capacitated mobile facility location problem. *European Journal of Operational Research*, 277(2):507–520.

[Ropke and Pisinger, 2006] Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472.

[Rosenkrantz et al., 2000] Rosenkrantz, D. J., Tayi, G. K., and Ravi, S. (2000). Algorithms for path-based placement of inspection stations on networks. *INFORMS Journal on Computing*, 12(2):136–149.

[Šelmić et al., 2011] Šelmić, M., Bešinović, N., and Teodorović, D. (2011). Locating weigh-in-motion checkpoints in traffic networks using genetic algorithm. *E-Soc J*, 2(1):55–66.

[Šelmić et al., 2010] Šelmić, M., Teodorović, D., and Vukadinović, K. (2010). Locating inspection facilities in traffic networks: an artificial intelligence approach. *Transportation planning and technology*, 33(6):481–493.

[Sharara and Kanj, 2014] Sharara, S. L. and Kanj, S. S. (2014). War and infectious diseases: challenges of the syrian civil war. *PLoS pathogens*, 10(11).

[Shaw, 1997] Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK.*

[Shaw, 1998] Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *International conference on principles and practice of constraint programming*, pages 417–431. Springer.

[Shortall et al., 2017] Shortall, C. K., Glazik, R., Sornum, A., and Pritchard, C. (2017). On the ferries: the unmet health care needs of transiting refugees in greece. *International Health*, 9(5):272–280.

[Srivastava et al., 2016] Srivastava, S. K., Amula, A., and Ghagare, P. (2016). Service facility relocation decision: a case study. *Facilities*.

[Sweeney and Tatham, 1976] Sweeney, D. J. and Tatham, R. L. (1976). An improved long-run model for multiple warehouse location. *Management Science*, 22(7):748–758.

[Taymaz and Dellaert, 2013] Taymaz, S. and Dellaert, N. (2013). Network planning of walk-in clinics on roadsides in africa. Master's thesis, Middle East Technical University Turkey.

[Tian and Liu, 2014] Tian, G. and Liu, Y. (2014). Energy-efficient models of sustainable location for a vehicle inspection station with emission constraints. *IEEE Transactions on Automation Science and Engineering*, 12(1):238–243.

[Tian and Zhou, 2015] Tian, G. and Zhou, M. (2015). An expected value multi-objective optimization model to locate a vehicle inspection station. In *2015 International Conference on Advanced Mechatronic Systems (ICAMechS)*, pages 96–102. IEEE.

[Tian et al., 2016] Tian, G., Zhou, M., Li, P., Zhang, C., and Jia, H. (2016). Multi-objective optimization models for locating vehicle inspection stations subject to stochastic demand, varying velocity and regional constraints. *IEEE Transactions on Intelligent Transportation Systems*, 17(7):1978–1987.

[Türk Kızılayı, 2020] Türk Kızılayı (2020). Mobil mutfak bağışı. `https://www.kizilay.org.tr/Bagis/BagisYap/21/mobil-mutfak-bagisi`. (last accessed on May 24th, 2020).

[UN General Assembly, 1948] UN General Assembly (1948). Universal declaration of human rights. volume 217 (III) A, Paris.

[UN Network on Migration, 2020] UN Network on Migration (2020). About us. `https://migrationnetwork.un.org/about`. (last accessed on May 23th, 2020).

[UN Refugee Agency, 2020] UN Refugee Agency (2020). Refugee statistics. `https://www.unhcr.org/figures-at-a-glance.html`. (last accessed on July 4th, 2020).

[Wang et al., 2020] Wang, M., Li, H., Cui, J., Bhowmick, S. S., and Liu, P. (2020). Frost: Movement history–conscious facility relocation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(1):1–26.

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

[Wesolowsky, 1973] Wesolowsky, G. O. (1973). Dynamic facility location. *Management Science*, 19(11):1241–1248.

[WHO, 2019] WHO (2019). Migrants and refugees at higher risk of developing ill health than host populations, reveals first-ever who report on the health of displaced people in europe. `http://bit.ly/2RHpKFa`. (last accessed on February 17th, 2019).

[Worley et al., 2000] Worley, C. W., Worley, K. A., and Kumar, P. L. (2000). Infectious disease challenges in immigrants from tropical countries. *Pediatrics*, 106(1):e3–e3.

[Wu and Lin, 2003] Wu, T.-H. and Lin, J.-N. (2003). Solving the competitive discretionary service facility location problem. *European Journal of Operational Research*, 144(2):366–378.

[Yadav et al., 2016] Yadav, T., Sadhukhan, K., and Mallari, R. A. (2016). Approximation algorithm for n-distance minimal vertex cover problem. *arXiv preprint arXiv:1606.02889*.

Appendix A

THE PROPERTIES OF INSTANCE P6E13N30_1

In this chapter, the properties of largest instance is provided. In this instance, there are 13 refugee groups entering 6 different paths. The main path contains 30 nodes and 3 different refugee groups enter this path. The shortest path consist of 6 nodes and only 1 refugee group enters this path. The followers of third path do not stay in camp points and they reach their destination by travelling from cities to cities. 3 refugee groups enter this path. The last three paths are the paths that the refugee groups enter these paths finish their travel before reaching the destination point. There are 2 refugee groups entering to each of these paths. The entrance data and the node sequences of paths can be seen in Tables A.1 and A.2, respectively.

p	**Refugee groups entering path p on period t (d_{pt})** t					
	1	**2**	**3**	**4**	**5**	**6**
1	+				+	+
2					+	
3	+				+	+
4	+	+				
5	+	+				
6	+	+				

Table A.1: The refugee group data for the instance "P6E13N30"

p	l_p	k^{th} node of path p (n_{pk}) k																													
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
1	30	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
2	6	1	4	9	19	23	30																								
3	21	1	2	3	4	5	6	9	10	11	12	14	15	16	18	19	20	21	23	26	28	30									
4	10	1	2	3	4	5	6	7	8	9	10																				
5	11	1	2	3	4	5	6	7	8	9	10	11																			
6	12	1	2	3	4	5	6	7	8	9	10	11	12																		

Table A.2: The path data for the instance “P6E13N30”

Appendix B

RESULTS FOR INSTANCES

The first table of this appendix presents the optimal results, number of MSFs and time comparisons for small-size instances. First column, *Instance*, represents the name of the instance. Second column gives the objective function value of the instance and third column provides the number of MSF used in the optimal solution. The following two columns present the time comparison of small-size instances in terms of CPU seconds. The last column reports the number of replications that found the optimal solution of the corresponding instance.

The second and third tables provide the optimal results for medium-size instances and best found objective function values for large-size instances. Additionally, the number of MSFs used in these solutions are given, as well.

Note that, for all these instances, f and τ are set to 600 and 3, respectively.

Instance	Opt. Obj.	# of MSF	% Cost of Using MSF	% Cost of Traveling	CPU Sec.		# of Opt.
					ALNS	MILP	
P01E01N10_1	1100	1	%54.5	%45.5	1	1	8
P01E02N10_1	1100	1	%54.5	%45.5	1	0	8
P02E02N10_1	1120	1	%53.6	%46.4	1	0	8
P02E04N10_1	1356	1	%44.2	%55.8	1	0	8
P02E04N10_2	1669	2	%71.9	%28.1	1	1	8
P03E03N10_1	1120	1	%53.6	%46.4	1	0	8
P03E04N10_1	1689	2	%71.0	%29.0	1	1	8
P03E04N10_2	1120	1	%53.6	%46.4	1	0	8
P03E06N10_1	1720	2	%69.8	%30.2	1	1	4
P03E06N10_2	1356	1	%44.2	%55.8	1	0	8
P03E06N10_3	1800	3	%100.0	%0.0	2	2	8
P03E06N10_4	1800	3	%100.0	%0.0	1	2	8
P03E06N10_5	1689	2	%71.0	%29.0	2	1	8
P04E04N10_1	1356	1	%44.2	%55.8	1	0	8
P04E08N10_1	1356	1	%44.2	%55.8	2	1	8
P04E08N10_2	2020	2	%59.4	%40.6	2	3	3
P04E08N10_3	1989	2	%60.3	%39.7	3	3	8
P04E08N10_4	1990	2	%60.3	%39.7	3	3	5
P04E08N10_5	2020	2	%59.4	%40.6	2	4	4
P04E10N10_1	1990	2	%60.3	%39.7	4	2	7
P04E10N10_2	2060	2	%58.3	%41.7	5	5	7
P06E10N10_1	2082	3	%86.5	%13.5	5	6	8
P06E10N10_2	2063	2	%58.2	%41.8	4	5	5
P06E10N10_3	1840	3	%97.8	%2.2	3	3	8
P06E13N10_1	1933	3	%93.1	%6.9	5	4	2
	Min.	1	%44.2	%0.0	0.56	0.26	2
	Max.	3	%100.0	%55.8	4.89	6.31	8
	Avg.	1.84	%65.0	%35.0	2.04	2.03	6.92

Table B.1: Optimal Solution Values and Time Comparison for Small-Size Instances

Instance	**Opt. Obj.**	**# of MSF**	**% Cost of Using MSF**	**% Cost of Traveling**
P01E01N20_1	1488	1	%40.3	%59.7
P01E02N20_1	1492	1	%40.2	%59.8
P02E02N20_1	1508	1	%39.8	%60.2
P02E04N20_1	1839	2	%65.3	%34.7
P02E04N20_2	2594	2	%46.3	%53.7
P03E03N20_1	1716	1	%35.0	%65.0
P03E04N20_1	2669	2	%45.0	%55.0
P03E04N20_2	2043	2	%58.7	%41.3
P03E06N20_1	2910	2	%41.2	%58.8
P03E06N20_2	2372	2	%50.6	%49.4
P03E06N20_3	2940	3	%61.2	%38.8
P03E06N20_4	2837	3	%63.4	%36.6
P03E06N20_5	2614	2	%45.9	%54.1
P04E04N20_1	2043	2	%58.7	%41.3
P04E08N20_1	2372	2	%50.6	%49.4
P04E08N20_2	2784	3	%64.7	%35.3
P04E08N20_3	2940	3	%61.2	%38.8
P04E08N20_4	3093	3	%58.2	%41.8
P04E08N20_5	2940	3	%61.2	%38.8
P04E10N20_1	3264	3	%55.1	%44.9
P04E10N20_2	2960	3	%60.8	%39.2
P06E10N20_1	3215	4	%74.7	%25.3
P06E10N20_2	3244	3	%55.5	%44.5
P06E10N20_3	3311	4	%72.5	%27.5
P06E13N20_1	3217	3	%56.0	%44.0
	Min.	1	%35.0	%25.3
	Max.	4	%74.7	%65.0
	Avg.	2.4	%54.5	%45.5

Table B.2: Optimal Solution Values for Medium-Size Instances

Instance	Best Obj.	# of MSF	% Cost of Using MSF	% Cost of Traveling
P01E01N30_1	2129	1	%28.2	%71.8
P01E02N30_1	2133	1	%28.1	%71.9
P02E02N30_1	2552	2	%47.0	%53.0
P02E04N30_1	2883	3	%62.4	%37.6
P02E04N30_2	4098	3	%43.9	%56.1
P03E03N30_1	3003	2	%40.0	%60.0
P03E04N30_1	4374	3	%41.2	%58.8
P03E04N30_2	3965	3	%45.4	%54.6
P03E06N30_1	4613	3	%39.0	%61.0
P03E06N30_2	3667	3	%49.1	%50.9
P03E06N30_3	4642	4	%51.7	%48.3
P03E06N30_4	4439	4	%54.1	%45.9
P03E06N30_5	4123	3	%43.7	%56.3
P04E04N30_1	3540	2	%33.9	%66.1
P04E08N30_1	4157	3	%43.3	%56.7
P04E08N30_2	5137	5	%58.4	%41.6
P04E08N30_3	5200	6	%69.2	%30.8
P04E08N30_4	5496	6	%65.5	%34.5
P04E08N30_5	5150	6	%69.9	%30.1
P04E10N30_1	5563	7	%75.5	%24.5
P04E10N30_2	5375	7	%78.1	%21.9
P06E10N30_1	5351	6	%67.3	%32.7
P06E10N30_2	5565	5	%53.9	%46.1
P06E10N30_3	5686	8	%84.4	%15.6
P06E13N30_1	4899	3	%36.7	%63.3
	Min.	1	%28.1	%15.6
	Max.	8	%84.4	%71.9
	Avg.	3.96	%52.4	%47.6

Table B.3: Best Found Solution Values for Large-Size Instances

Appendix C

RESULTS FOR INSTANCES WITH DIFFERENT FIXED COST VALUES

In this section, objective function values, number of MSF used and CPU seconds elapsed for best found solutions of medium-size and large-size instances are provided when parameter f takes different values such as 300, 400, 500, 700, 800, 900, 1000, and 1100 in Tables C.1 and C.2. Since $f = 600$ is the base case and reported earlier, the values for that instance are not provided again. The parameter for service level requirement, τ, is determined as in the base case, 3.

Instance	$f = 300$			$f = 400$			$f = 500$			$f = 700$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N20_1	1188	1	15	1288	1	12	1388	1	14	1588	1	14
P01E02N20_1	1192	1	21	1292	1	19	1392	1	22	1592	1	21
P02E02N20_1	1208	1	14	1308	1	14	1408	1	16	1608	1	16
P02E04N20_1	1239	2	26	1439	2	26	1639	2	28	2039	2	30
P02E04N20_2	1731	4	58	2125	3	31	2394	2	32	2794	2	35
P03E03N20_1	1416	1	47	1516	1	44	1616	1	53	1816	1	52
P03E04N20_1	1786	4	144	2180	3	165	2469	2	176	2869	2	166
P03E04N20_2	1443	2	54	1643	2	53	1843	2	51	2243	2	47
P03E06N20_1	1800	6	281	2390	3	509	2690	3	552	3110	2	423
P03E06N20_2	1678	3	210	1972	2	238	2172	2	237	2572	2	196
P03E06N20_3	1800	6	411	2333	4	483	2640	3	421	3267	2	374
P03E06N20_4	1792	5	311	2237	3	330	2537	3	285	3133	2	290
P03E06N20_5	1731	4	115	2125	3	121	2414	2	108	2814	2	91
P04E04N20_1	1443	2	53	1643	2	57	1843	2	55	2243	2	53
P04E08N20_1	1678	3	313	1972	2	340	2172	2	249	2572	2	220
P04E08N20_2	1786	4	372	2184	3	460	2484	3	469	3084	3	434
P04E08N20_3	1800	6	396	2333	4	501	2640	3	460	3240	3	418
P04E08N20_4	1800	6	424	2335	5	557	2793	3	547	3396	3	424
P04E08N20_5	1800	6	338	2340	3	335	2640	3	302	3240	3	319
P04E10N20_1	1800	6	794	2335	5	1171	2835	5	1208	3564	3	1015
P04E10N20_2	1800	6	866	2335	5	1112	2660	3	1048	3260	3	775
P06E10N20_1	1800	6	779	2343	5	999	2815	4	843	3597	2	785
P06E10N20_2	1800	6	897	2335	5	821	2835	5	769	3544	3	721
P06E10N20_3	1800	6	646	2377	5	911	2877	5	1032	3621	3	872
P06E13N20_1	1800	6	1028	2343	5	1457	2843	5	1741	3517	3	1793

Table C.1: Best Found Solution Values for Medium-Size Instances when $f = 300, 400, 500, 700$ and $\tau = 3$

Instance	$f = 800$			$f = 900$			$f = 1000$			$f = 1100$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N20_1	1688	1	16	1788	1	20	1888	1	22	1988	1	22
P01E02N20_1	1692	1	22	1792	1	26	1892	1	27	1992	1	27
P02E02N20_1	1708	1	17	1808	1	20	1908	1	22	2008	1	25
P02E04N20_1	2239	2	34	2439	2	32	2639	2	37	2839	2	38
P02E04N20_2	2994	2	36	3194	2	34	3394	2	42	3594	2	42
P03E03N20_1	1916	1	49	2016	1	50	2116	1	55	2216	1	51
P03E04N20_1	3069	2	174	3269	2	194	3469	2	140	3669	2	172
P03E04N20_2	2443	2	50	2643	2	50	2843	2	48	3043	2	49
P03E06N20_1	3310	2	476	3510	2	457	3710	2	429	3910	2	525
P03E06N20_2	2772	2	181	2972	2	201	3172	2	189	3372	2	183
P03E06N20_3	3467	2	354	3667	2	382	3867	2	302	4067	2	360
P03E06N20_4	3333	2	324	3533	2	293	3733	2	324	3933	2	362
P03E06N20_5	3014	2	113	3214	2	122	3414	2	124	3614	2	102
P04E04N20_1	2443	2	51	2643	2	50	2843	2	52	3043	2	53
P04E08N20_1	2772	2	264	2972	2	200	3172	2	243	3372	2	224
P04E08N20_2	3333	2	488	3533	2	423	3733	2	517	3933	2	481
P04E08N20_3	3481	2	314	3681	2	308	3881	2	296	4081	2	377
P04E08N20_4	3634	2	395	3834	2	354	4034	2	380	4234	2	364
P04E08N20_5	3481	2	317	3681	2	322	3881	2	294	4081	2	309
P04E10N20_1	3864	3	959	4164	3	900	4464	3	1038	4764	3	1004
P04E10N20_2	3560	3	900	3860	3	859	4160	3	868	4460	3	914
P06E10N20_1	3797	2	891	3997	2	1078	4197	2	1034	4397	2	1113
P06E10N20_2	3844	3	623	4144	3	744	4444	3	614	4744	3	705
P06E10N20_3	3921	3	802	4221	3	881	4521	3	802	4821	3	1020
P06E13N20_1	3817	3	1743	4117	3	1759	4417	3	1732	4717	3	1804

Table C.2: Best Found Solution Values for Medium-Size Instances when $f = 800, 900, 1000, 1100$ and $\tau = 3$

Instance	$f = 300$			$f = 400$			$f = 500$			$f = 700$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N30_1	1829	1	6	1929	1	7	2029	1	11	2229	1	10
P01E02N30_1	1833	1	33	1933	1	33	2033	1	35	2233	1	36
P02E02N30_1	1952	2	17	2152	2	20	2352	2	20	2752	2	22
P02E04N30_1	1983	3	121	2283	3	129	2583	3	151	3115	2	90
P02E04N30_2	2808	7	385	3429	4	189	3798	3	216	4405	3	135
P03E03N30_1	2318	3	1243	2603	2	604	2803	2	604	3203	2	561
P03E04N30_1	2930	9	3327	3622	5	2104	4074	3	2034	4674	3	2414
P03E04N30_2	2506	5	1941	3006	5	2604	3569	4	2147	4265	3	1675
P03E06N30_1	2958	9	9178	3789	6	8851	4287	5	8060	4913	3	6381
P03E06N30_2	2565	5	6420	3052	4	5257	3367	3	5405	3967	3	4768
P03E06N30_3	2930	9	9185	3730	5	7308	4229	5	8929	5064	3	6480
P03E06N30_4	2914	8	5602	3576	5	3929	4039	4	3899	4835	3	3837
P03E06N30_5	2810	7	942	3434	4	771	3823	3	752	4423	3	882
P04E04N30_1	2506	5	2044	2960	3	1203	3260	3	1371	3740	2	932
P04E08N30_1	2795	8	12839	3479	4	9810	3857	3	9153	4457	3	7675
P04E08N30_2	2986	8	8574	3784	7	12446	4484	7	11788	5593	4	13347
P04E08N30_3	2992	9	10749	3830	7	14406	4530	7	14406	5728	5	14406
P04E08N30_4	2994	9	9298	3894	9	13203	4655	7	13945	6024	5	14408
P04E08N30_5	2994	9	7011	3830	7	7331	4530	7	6942	5689	4	11894
P04E10N30_1	2992	9	14410	3894	9	14407	4759	8	14409	6100	5	14410
P04E10N30_2	2992	9	14410	3832	8	14411	4642	8	14410	6067	6	14414
P06E10N30_1	2994	9	13991	3894	9	14409	4715	7	14410	5864	5	14409
P06E10N30_2	2992	9	14406	3892	9	14407	4749	8	14409	6065	5	14409
P06E10N30_3	3000	10	14207	3984	9	14409	4884	9	14408	6393	6	14408
P06E13N30_1	2932	9	14408	3753	7	14410	4423	6	14409	5202	3	14410

Table C.3: Best Found Solution Values for Large-Size Instances when $f = 300, 400, 500, 700$ and $\tau = 3$

Instance	$f = 800$			$f = 900$			$f = 1000$			$f = 1100$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N30_1	2329	1	11	2429	1	12	2529	1	13	2629	1	17
P01E02N30_1	2333	1	33	2433	1	34	2533	1	34	2633	1	36
P02E02N30_1	2952	2	23	3152	2	22	3352	2	23	3552	2	27
P02E04N30_1	3315	2	90	3515	2	88	3715	2	86	3915	2	89
P02E04N30_2	4701	3	149	4927	2	161	5127	2	207	5327	2	186
P03E03N30_1	3403	2	388	3603	2	454	3803	2	490	4003	2	383
P03E04N30_1	4974	3	2284	5274	3	2001	5574	3	2142	5874	3	2264
P03E04N30_2	4565	3	1702	4865	3	1477	5165	3	1612	5465	3	1246
P03E06N30_1	5213	3	6509	5513	3	5445	5813	3	5824	6113	3	5656
P03E06N30_2	4267	3	2955	4503	2	3473	4703	2	3321	4903	2	3104
P03E06N30_3	5339	3	5810	5639	3	4681	5932	3	6239	6232	3	6316
P03E06N30_4	5138	3	3764	5435	3	3499	5735	3	3765	6035	3	3268
P03E06N30_5	4723	3	809	5023	3	711	5323	3	543	5596	2	681
P04E04N30_1	3940	2	696	4140	2	773	4340	2	774	4540	2	711
P04E08N30_1	4757	3	7299	5057	3	7182	5357	3	7346	5657	3	7211
P04E08N30_2	5993	4	12870	6393	4	13606	6793	4	12490	7193	4	13749
P04E08N30_3	6174	4	14408	6574	4	11501	6974	4	9837	7374	4	11678
P04E08N30_4	6524	5	13840	6981	4	11721	7381	4	11169	7781	4	11220
P04E08N30_5	6089	4	9784	6489	4	8682	6889	4	8262	7289	4	8492
P04E10N30_1	6596	5	14412	7020	4	14413	7420	4	14413	7820	4	14414
P04E10N30_2	6659	6	14410	7217	5	14412	7713	5	14415	8287	5	14416
P06E10N30_1	6321	5	14412	6802	4	14412	7202	4	14413	7605	4	14414
P06E10N30_2	6565	5	14411	6989	4	14410	7389	4	14412	7789	4	14412
P06E10N30_3	6916	5	14410	7416	5	14412	7916	5	14413	8416	5	13531
P06E13N30_1	5499	3	14410	5803	3	14413	6100	3	14416	6399	3	14415

Table C.4: Best Found Solution Values for Large-Size Instances when $f = 800, 900, 1000, 1100$ and $\tau = 3$

Appendix D

RESULTS FOR INSTANCES WITH DIFFERENT SERVICE REQUIREMENT LEVELS

In this section, objective function values, number of MSF used and CPU seconds elapsed for best found solutions of medium-size and large-size instances are provided when parameter τ takes different values such as 4, 5, 6, 7, 8, and 9 Tables D.1 and D.2. Since $\tau = 3$ is the base case and reported earlier, the values for that instance are not provided again. The parameter for fixed cost, f, is determined as in the base case, 600.

Instance	$\tau = 4$			$\tau = 5$			$\tau = 6$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N20_1	1461	1	8	1356	1	5	1134	1	6
P01E02N20_1	1463	1	23	1357	1	15	1134	1	9
P02E02N20_1	1461	1	11	1356	1	6	1134	1	6
P02E04N20_1	1463	1	44	1357	1	16	1134	1	9
P02E04N20_2	2099	2	140	1487	1	46	1181	1	18
P03E03N20_1	1476	1	44	1356	1	16	1134	1	12
P03E04N20_1	2316	2	239	1651	1	80	1395	1	23
P03E04N20_2	1476	1	63	1382	1	32	1134	1	12
P03E06N20_1	2471	2	427	1889	3	76	1658	2	37
P03E06N20_2	1666	1	278	1361	1	63	1134	1	33
P03E06N20_3	2471	2	334	1889	3	88	1482	1	32
P03E06N20_4	2478	2	460	1685	1	117	1395	1	42
P03E06N20_5	2457	2	248	1667	1	143	1240	1	35
P04E04N20_1	1476	1	47	1382	1	29	1134	1	12
P04E08N20_1	1666	1	370	1408	1	133	1134	1	34
P04E08N20_2	2380	2	294	1767	2	192	1429	1	29
P04E08N20_3	2471	2	442	1928	3	112	1482	1	33
P04E08N20_4	2516	2	375	1928	3	132	1570	2	43
P04E08N20_5	2467	2	286	1977	3	139	1570	2	34
P04E10N20_1	2588	2	858	1977	3	328	1570	2	97
P04E10N20_2	2588	2	894	2000	3	405	1570	2	86
P06E10N20_1	2516	2	895	1977	3	287	1570	2	95
P06E10N20_2	2557	2	771	2000	3	333	1482	1	68
P06E10N20_3	2849	3	1041	2119	3	334	1676	2	80
P06E13N20_1	2650	2	1510	1954	3	461	1570	2	164

Table D.1: Best Found Solution Values for Medium-Size Instances and $f = 600$ when $\tau = 4, 5, 6$

Instance	$\tau = 7$			$\tau = 8$			$\tau = 9$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N20_1	1112	1	7	1112	1	8	745	1	9
P01E02N20_1	1112	1	11	1112	1	12	745	1	12
P02E02N20_1	1112	1	7	1112	1	9	745	1	10
P02E04N20_1	1112	1	12	1112	1	13	745	1	12
P02E04N20_2	1181	1	23	1112	1	21	863	1	16
P03E03N20_1	1112	1	13	1112	1	13	745	1	12
P03E04N20_1	1187	1	32	1112	1	21	902	1	18
P03E04N20_2	1112	1	13	1112	1	13	745	1	12
P03E06N20_1	1496	2	33	1200	2	26	1023	1	20
P03E06N20_2	1116	1	37	1112	1	30	863	1	21
P03E06N20_3	1407	1	45	1112	1	33	902	1	23
P03E06N20_4	1240	1	37	1112	1	27	932	1	23
P03E06N20_5	1240	1	58	1112	1	35	885	1	24
P04E04N20_1	1112	1	13	1112	1	13	745	1	12
P04E08N20_1	1116	1	38	1112	1	30	863	1	21
P04E08N20_2	1259	1	49	1112	1	27	902	1	22
P04E08N20_3	1407	1	45	1112	1	33	902	1	23
P04E08N20_4	1446	2	31	1159	1	28	924	1	26
P04E08N20_5	1446	2	56	1159	1	29	924	1	27
P04E10N20_1	1446	2	52	1159	1	40	902	1	38
P04E10N20_2	1446	2	62	1159	1	46	902	1	32
P06E10N20_1	1446	2	71	1159	1	55	924	1	34
P06E10N20_2	1415	1	74	1112	1	60	902	1	45
P06E10N20_3	1638	2	74	1200	2	43	984	1	31
P06E13N20_1	1446	2	83	1159	1	70	924	1	52

Table D.2: Best Found Solution Values for Medium-Size Instances when $f = 600$ and $\tau = 7, 8, 9$

Instance	$\tau = 4$			$\tau = 5$			$\tau = 6$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N30_1	2102	1	30	1861	1	39	1676	1	26
P01E02N30_1	2104	1	84	1861	1	115	1676	1	62
P02E02N30_1	2293	2	41	1886	1	65	1872	1	38
P02E04N30_1	2313	2	174	1886	1	298	1872	1	111
P02E04N30_2	3505	3	704	2035	1	328	1876	1	161
P03E03N30_1	2565	2	249	1889	1	182	1872	1	113
P03E04N30_1	3583	2	1570	2486	2	1354	2179	1	360
P03E04N30_2	3546	3	950	2489	2	634	2241	2	173
P03E06N30_1	3971	3	3780	2850	3	2067	2656	4	1018
P03E06N30_2	2710	2	2409	2016	1	889	1878	1	853
P03E06N30_3	3788	2	3719	2640	3	2178	2395	2	886
P03E06N30_4	3775	2	2719	2572	2	2101	2187	1	751
P03E06N30_5	3665	2	1512	2625	2	1921	1965	1	610
P04E04N30_1	3255	2	581	2488	2	629	2295	2	203
P04E08N30_1	3354	2	1647	2608	3	3324	2299	2	1218
P04E08N30_2	4543	5	9488	2925	3	3508	2633	2	1578
P04E08N30_3	4532	6	9744	3001	4	3321	2677	3	1669
P04E08N30_4	4578	6	8552	3128	5	3382	2791	4	1915
P04E08N30_5	4618	6	8461	3176	4	4692	2734	3	1988
P04E10N30_1	4816	6	14415	3059	4	7729	2711	4	3051
P04E10N30_2	4807	6	14418	3059	4	8651	2658	3	3069
P06E10N30_1	4664	6	14415	3177	5	8412	2770	4	2140
P06E10N30_2	4881	5	14418	3059	4	7611	2655	3	2417
P06E10N30_3	5028	7	14415	3302	4	8627	2914	4	2168
P06E13N30_1	4106	3	14415	2849	3	4835	2411	2	2927

Table D.3: Best Found Solution Values for Large-Size Instances and $f = 600$ when $\tau = 4, 5, 6$

Instance	$\tau = 7$			$\tau = 8$			$\tau = 9$		
	Obj.	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**	**Obj.**	**MSF**	**CPU**
P01E01N30_1	1676	1	29	1507	1	35	1258	1	29
P01E02N30_1	1676	1	63	1507	1	79	1258	1	48
P02E02N30_1	1676	1	29	1507	1	35	1258	1	31
P02E04N30_1	1676	1	65	1507	1	82	1258	1	57
P02E04N30_2	1688	1	268	1508	1	190	1258	1	87
P03E03N30_1	1676	1	65	1507	1	69	1258	1	53
P03E04N30_1	1801	1	159	1508	1	172	1259	1	128
P03E04N30_2	2194	3	171	1734	2	135	1463	2	79
P03E06N30_1	2223	3	359	1875	3	512	1529	1	123
P03E06N30_2	1678	1	396	1507	1	241	1258	1	163
P03E06N30_3	2062	1	429	1756	1	354	1259	1	142
P03E06N30_4	1801	1	388	1590	1	365	1259	1	147
P03E06N30_5	1713	1	301	1562	1	589	1258	1	152
P04E04N30_1	2194	3	142	1734	2	124	1463	2	70
P04E08N30_1	2194	3	540	1734	2	426	1463	2	216
P04E08N30_2	2323	2	1201	1784	2	476	1532	2	181
P04E08N30_3	2369	3	709	1784	2	536	1584	2	203
P04E08N30_4	2400	4	679	1834	2	459	1688	2	261
P04E08N30_5	2400	4	775	1881	2	974	1688	2	346
P04E10N30_1	2400	4	1215	1928	2	1133	1584	2	363
P04E10N30_2	2400	4	1393	1928	2	1451	1584	2	362
P06E10N30_1	2400	4	1191	1881	2	1384	1688	2	453
P06E10N30_2	2369	3	1303	1887	2	1392	1584	2	376
P06E10N30_3	2400	4	1120	2025	3	1584	1584	2	313
P06E13N30_1	2125	2	1195	1768	1	2183	1474	1	420

Table D.4: Best Found Solution Values for Large-Size Instances when $f = 600$ and $\tau = 7, 8, 9$

Appendix E

SOLUTION REPRESENTATION OF INSTANCE P6E13N30_1

In this chapter, the best found solution for instance P6E13N30_1 is explained. In Tables E.1 and E.2 represent where the MSFs perform their service acts, as explained in Chapter 1.

In Tables E.3 and E.4, the same solution is represented in the perspective of refugee groups where the numbers in the tables represent the node numbers on which the refugee groups present at the corresponding period.

According to this solution, 3 MSFs are used and the objective function value is 4899. The number of service acts performed in the process is 51.

MSF	Period																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1			3	3		4	3	3		6	6	12	12	12	12	12	12	18
2						6	6	10	9	9	9	9	9	9	15	15	15	
3								19			14			18		20		

Table E.1: The best found solution of instance P6E13N30_1 according to MSFs between periods 1 and 18.

MSF	Period																	Total
	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	Cost
1	18			18	18	20	20											1606
2	15	15	21	21	23				23	23		26	26					1773
3	26				26	26		26		28				28	28			1520

Table E.2: The best found solution of instance P6E13N30_1 according to MSFs between periods 19 and 35.

Refugee	Period																	
Group	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1	2	**3**	4	5	**6**	7	8	**9**	10	11	**12**	13	14	**15**	16	17	**18**
2					1	**4**	9	**19**	23	30								
3	1	2	**3**	4	5	**6**	9	**10**	11	12	**14**	15	16	**18**	19	**20**	21	23
4	1	2	**3**	4	5	**6**	7	8	**9**	10								
5	1	2	**3**	4	5	**6**	7	8	**9**	10	11							
6	1	2	**3**	4	5	**6**	7	8	**9**	10	11	**12**						
7					1	2	**3**	4	5	**6**	7	8	**9**	10	11	**12**	13	14
8						1	2	**3**	4	5	**6**	7	8	**9**	10	11	**12**	13
9					1	2	**3**	4	5	**6**	**9**	10	11	**12**	14	**15**	16	**18**
10						1	2	**3**	4	5	**6**	**9**	10	11	**12**	14	**15**	16
11		1	2	**3**	4	5	**6**	7	8	**9**	10							
12		1	2	**3**	4	5	**6**	7	8	**9**	10	11						
13		1	2	**3**	4	5	**6**	7	8	**9**	10	11	**12**					

Table E.3: The best found solution of instance P6E13N30_1 according to refugee groups between periods 1 and 18. The bold node numbers represent the the nodes where the group is served

Refugee	Period																
Group	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
1	19	20	**21**	22	**23**	24	25	**26**	27	**28**	29	30					
2																	
3	**26**	28	30														
4																	
5																	
6																	
7	**15**	16	17	**18**	19	**20**	21	22	**23**	24	25	**26**	27	**28**	29	30	
8	14	**15**	16	17	**18**	19	**20**	21	22	**23**	24	25	**26**	27	**28**	29	30
9	19	20	**21**	23	**26**	28	30										
10	**18**	19	20	**21**	**23**	**26**	28	30									
11																	
12																	
13																	

Table E.4: The best found solution of instance P6E13N30_1 according to refugee groups between periods 19 and 35.

The bold node numbers represent the the nodes where the group is served