KADİR HAS UNIVERSITY
SCHOOL OF GRADUATE STUDIES
PROGRAM OF INDUSTRIAL ENGINEERING

DEBRIS REMOVAL STRATEGIES TO IMPROVE ACCESSIBILITY ROAD NETWORKS

TUĞÇE ÖZKENAR

MASTER'S THESIS

İSTANBUL, AUGUST,
2019

DEBRIS REMOVAL STRATEGIES TO IMPROVE ACCESSIBILITY OF ROAD NETWORKS

TUĞÇE ÖZKENAR

MASTER'S THESIS

Submitted to the School of Graduate Studies of
Kadir Has University in partial fulfillment of the requirements for the degree of
Master of Science in Industiral Engineering

İSTANBUL, AUGUST, 2019

DECLARATION OF RESEARCH ETHICS /
METHODS OF DISSEMINATION

I, TUĞÇE ÖZKENAR, hereby declare that;

- this master's thesis is my own original work and that due references have been appropriately provided on all supporting literature and resources;
- this master's thesis contains no material that has been submitted or accepted for a degree or diploma in any other educational institution;
- I have followed *Kadir Has University Academic Ethics Principles prepared in accordance with The Council of Higher Education's Ethical Conduct Principles.*

In addition, I understand that any false claim in respect of this work will result in disciplinary action in accordance with University regulations.

Furthermore, both printed and electronic copies of my work will be kept in Kadir Has Information Center under the following condition as indicated below (SELECT ONLY ONE, DELETE THE OTHER TWO):

☐ The full content of my thesis will be accessible from everywhere by all means.
☐ The full content of my thesis will be accessible only within the campus of Kadir Has University.
☒ The full content of my thesis will not be accessible for one year. If no extension is required by the end of this period, the full content of my thesis will be automatically accessible from everywhere by all means.

TUĞÇE ÖZKENAR
09.AUGUST.2019

KADİR HAS UNIVERSITY

SCHOOL OF GRADUATE STUDIES

ACCEPTANCE AND APPROVAL

This work entitled DEBRIS REMOVAL STRATEGIES TO IMPROVE ACCESSIBILITY OF ROAD NETWORKS prepared by TUĞÇE ÖZKENAR has been judged to be successful at the defense exam on 09.AUGUST.2019 and accepted by our jury as master's thesis.

APPROVED BY:

Asst. Prof. Dr. Burak Çavdaroğlu (Advisor) .....
Kadir Has University

Assoc. Prof. Dr. Ahmet Yücekaya .....
Kadir Has University

Assoc. Prof. Dr. S. Çağlar Aksezer .....
Işık University

I certify that the above signatures belong to the faculty members named above.

.....

Prof. Dr. Sinem Açıkmeşe

Dean of School of Graduate Studies

DATE OF APPROVAL: 09.AUGUST.2019

TABLE OF CONTENTS

DEBRIS REMOVAL STRATEGIES TO IMPROVE ACCESSIBILITY OF ROAD NETWORKS

ABSTRACT

Disasters are non-routine events which have occurred in every age of history and have resulted in losses of several lives and economic losses. Debris is one of the most devastating consequence of the disasters. Debris occurs by the collapse of buildings and roads, threatens life, causes environmental problems and interrupts daily life. It is very crucial to remove debris effectively to open blocked roads, reach affected areas and help the people in need. Debris removal operations require a rational decision making process. This thesis focuses debris clearance on the response stage of the disaster management. It is assumed that debris will shut down the roads and prevent the travel of emergency aid teams from reaching people in need. The aim of the *dynamic spanning tee (DST)* problem introduced in the thesis is to determine the sequence of blocked roads to be cleaned in order to increase the accessibility of the affected areas. The optimization model developed is a mixed integer programming model which is based on an efficient LP formulation of the minimum spanning tree problem. We develop four heuristic methods for the DST problem. With different damage scenarios for a small hypothetical network, the performances of these heuristic methods are compared with the exact optimal solutions. It has been observed that two heuristic methods give the closest results to the optimal solutions. The exact and heuristic methods are also tested on two realistic data sets representing the road networks of Güven and Şirinevler neighborhoods in İstanbul. Due to the large number of variables in the model, commercial solver experiences memory issues in these network instances. It has been shown that the same two heuristic methods performing best in hypothetical instance also give the best results in these data sets.

Keywords: Disaster, Debris, Minimum spanning tree (MST).

YOL AĞLARININ ERİŞİLEBİLİRLİĞİNİ GELİŞTİRMEYE YÖNELİK ENKAZ KALDIRMA STRATEJİLERİ

ÖZET

Tarihin her döneminde afetler büyük ölçüde can ve mal kaybına yol açmışlardır. Binaların yıkılması ve yolların çökmesiyle oluşan enkaz, afetlerin en olumsuz sonuçlarından biridir. Enkaz insan hayatı için tehlike oluşturmakta, çeşitli çevre sorunlarına yol açmakta ve yolları kapayarak günlük hayatı kesintiye uğratmaktadır. Kapalı yolların açılması, afet bölgelerine ulaşılması ve insanların kurtarılması için enkazın kaldırılması rasyonel bir karar verme süreci gerektirir. Bu tez çalışması, bir afet sonrasındaki enkaz kaldırma safhasına odaklanmaktadır. Çalışmada afet sonrasında enkazın yolları kapatacağı ve yardım ekiplerinin afet bölgelerindeki insanlara ulaşımını engelleyeceği varsayılmıştır. İlk defa bu tezde tanımlanan *Dinamik Kapsayan Ağaç (DKA)* probleminin amacı, ulaşılamayan bölgelere giden kapalı yolların hangi sıra ile açılacağına karar verilmesi ve afet bölgesinin ulaşılabilirliğinin artırılmasıdır. Çalışmadaki optimizasyon modeli, minimum kapsayan ağaç probemi (MKA) için daha önce geliştirilmiş efektif bir doğrusal programlama modeline dayanan, karışık tam sayı programlama modelidir. DKA problemi için 4 adet sezgisel metot geliştirilmiştir. Önce küçük bir örnek yol ağı oluşturulmuş ve bu ağ için farklı enkaz senaryoları tanımlanmıştır. Daha sonra sezgisel metotların bu ağdaki performansları optimizasyon yazılımından elde edilen optimal sonuçlarla kıyaslanmıştır. Özellikle iki sezgisel metodun optimal sonuca en yakın sonuçları verdiği gözlemlenmiştir. En son tam ve sezgisel metotlar Güven ve Şirinevler mahallelerini baz alan iki adet gerçek veri ile test edilmiştir. Modelde çok fazla değişken olduğu için optimizasyon yazılımı bu verilere ait problemleri çözerken yetersiz bellek hatası vermiştir. Diğer taraftan, örnek yol ağında en iyi performans gösteren aynı iki sezgisel metot bu data kümeleri içinde en iyi sonuçları vermiştir.

Anahtar Sözcükler: Afet, Enkaz, Minimum kapsayan ağaç (MKA).

ACKNOWLEDGEMENTS

I would first like to thank my thesis advisor Asst. Prof. Burak Çavdaroğlu. His door was always open whenever I ran into a trouble or had a question about my thesis. He consistently allowed this study to be my own work, but steered me in the right direction whenever he thought I needed it.

I owe my deepest gratitude to Assoc. Prof. Ahmet Deniz Yücekaya for his advice and support in my research and education. Many thanks to other professors in my department, who patiently answered my questions and problems.

Finally, I must express my very profound gratitude to my family for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of research and this thesis. This accomplishment would not have been possible without them. Thank you.

Tuğçe Özkenar

To my family...

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

MST	Minimum Spanning Tree
GDP	Gross Domestic Product
I	Inaccessibility
CI	Cumulative Inaccessibility
DST	Dynamic Spanning Tree

1. INTRODUCTION

Disaster can be defined as catastrophic, non-routine events which lead to economic and social loses, disrupts or paralysis of daily life and human activities. It is a natural or a human related phenomenon which suddenly occurs and exceeds the coping skills of the affected population (AFAD Açıklamalı Afet Yönetimi Terimleri Sözlüğü, 2014). Disasters are one of the most destructive and extreme events in our world throughout human history. Earthquake, flood, landslide, snow slide, heavy rainfall, storm, typhoon and hurricane are among the most frequently encountered natural disasters while fire, nuclear, biological, chemical, mining, industrial, transportation accidents, epidemics, work accidents and wars are among the most frequently encountered man-made disasters in the world. They may result in loss of life and property and can affect the daily life negatively, leading to economic losses and preventing the development of a country.

1.1 Natural and Man Made Disasters

Natural disasters are natural phenomena and cannot be controlled by humans. They occur in a short period of time and usually cannot be prevented by people after it starts. Some of the most frequently occurring natural disasters can be summarized as follows. Earthquakes are among the most encountered natural disasters and they are likely to occur in the seismically active regions. This extreme non-routine events can be defined as the shaking of surface because of the ruptures in the earth's crust (AFAD, 2018). Flood is another disruptive natural event. It occurs when water rises in its current area or comes from another place and covers the dry surfaces. The main factors causing flood are the lack of vegetation or the weakness and the impermeability of the layers on the ground. Likewise heavy rainfalls (even if they

last for a short period of time) may result in floods. Landslide can be defined as the slope of rock, soil or parts of the terrain because of the effect of external factors such as earthquakes or gravity. Storms, typhoons and hurricanes are the winds that harm people and environment. The force they apply to the surface can cause a great destruction through floods and fire. They may prevent marine transportation and can even cause ships to sink or become aground. They can also blow the roofs, trees and humans apart. They occur as a result of high pressure difference between neighboring regions (AFAD Açıklamalı Afet Yönetimi Terimleri Sözlüğü, 2014). The sliding of large snow layers as a result of gravity is called as avalanche or snow slide. This event is usually seen in mountainous and slopping terrains which do not have vegetation layer (AFAD, 2016). In addition to aforementioned natural disasters, rockfall, volcanic eruption, debris/mud flow, drought, hail and hailstorm, tornado, lighting, blizzard, acid rains, fog and icing are also among the natural disasters.

Man-made disasters are the extreme events which occur as a result of human actions and those are not related with nature. Definitions of some of the most encountered man-made disasters are detailed as follows. Fire is a disaster caused by combustion reactions of heat and oxygen. They may also considered as natural disasters if there is no intervention of humans such as fires occurred as a result of volcanic eruptions, lightnings or hurricanes. However, most of the fires are human-induced. Nuclear, biological and chemical accidents can be defined as technological disasters which have increased rapidly as a result of developing technology. These accidents and their consequences can cause great destruction, such as life and property loss. Mining accidents may happen during or after exploration, extraction, processing and deposition of mines or because of the mining waste. Industrial accidents can be defined as an emission, fire or explosion that result from uncontrolled events during transportation, processing and storing of hazardous materials. They might create great danger for environment and human health depending on their severity, influence area and impact duration. Transportation accidents are the events which are encountered unexpectedly and cause life losses and injuries. Those accidents usually occur by careless driving but sometimes they occur by a technical problem.

Most of them happen in highways because of the higher utilization rate of highways than that of other roads. Air accidents happen more rarely compared to car accidents; however, the consequences of the air accidents are more profound. We can also mention about marine accidents that result in economical losses. On the other hand, in some marine accidents, chemical products and substances contaminate the environment, disturb the ecological balance and negatively effect the human life. Epidemic diseases may sometimes be considered as man-made disasters. If sewer systems and water networks are not properly maintained, waste water can leak into the main water system causing epidemic diseases to emerge. The decaying corpses that are not securely conserved create a convenient environment for reproduction and growth of pathogenic micro organisms. Rats and mosquitoes get infected by those micro organisms, spread the disease to larger populations, and turn it into an epidemic. In addition; terrorism, uncontrolled migration, wars, dam explosions, nuclear and work accidents are among the other types of man-made disasters (AFAD Açıklamalı Afet Yönetimi Terimleri Sözlüğü, 2014).

1.2 Disaster Management and Debris Removal

Natural and man-made disasters are always an important risk for the society and the environment. Therefore, both pre-disaster and post-disaster activities should be planned and improved systematically. According to this perspective, disaster prevention and mitigation should be applied in all phases of a disaster. It is not possible to completely prevent the negative effects of a disaster, yet those effects can be minimized by effective disaster management operations. Disaster management is the process of planning interventions and identification of risk reduction techniques to be able to respond to any disaster promptly and effectively in order to create safer and improved living space for the communities affected by the disaster (AFAD Açıklamalı Afet Yönetimi Terimleri Sözlüğü, 2014). Disaster management literature consists of pre-disaster preventive strategies and post-disaster damage reduction strategies. The pre-disaster phase of disaster management is divided into two stages as *mitigation* and *preparedness*, while the post-disaster phase consists of *recovery* and

response stages. These four stages constitute the *disaster management cycle* shown in Figure 1.1. *Preparedness* spans the precautions taken shortly before a disaster, which enable people to react the extreme event instantly and effectively. *Mitigation* involves long-term strategies that are carried out long before the next disaster in order to lessen its negative impacts. Improving the resilience of civil and social infrastructure systems, developing governmental policies for construction and land use are some examples of mitigation efforts. *Response* is comprised of the actions that are taken in the first 72 hours after disaster occurs. Some of those actions are search and rescue, supplying medical aid and creating emergency evacuation. Last but not least, *recovery* usually refers to all activities that aims to turn back to normal living conditions. Depending on the severity and extent of the disaster, recovery stage lasts for one to ten years or more (Carter, 2008).

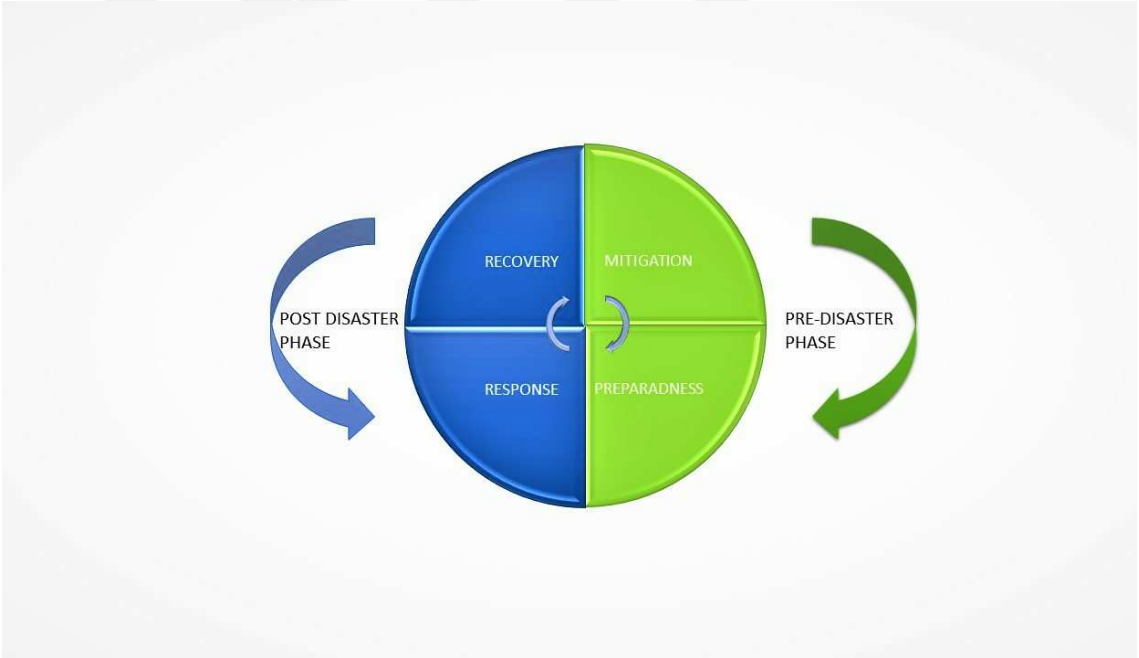


Figure 1.1 Disaster Management Cycle

Disasters can generate large amounts of debris, causing considerable challenges especially in the response and recovery stages of disaster management cycle. Debris is the accumulated waste resulting from a disaster and often includes building materials, sediments, vegetative debris, personal property and other materials. Removal of this debris can be time-consuming and costly (FEMA, 2007).

Disaster debris is regarded as one of the reasons for increasing rate of life loss in the aftermath of a disaster since it harms the accessibility of roadways for first responders. Large amounts of debris may come out due to the destruction of the structures, block roads and prevent the emergency aid crews from reaching people in need. Therefore, debris removal should begin as soon as it is safe for debris cleaning crews to be out. An initial activity in the response stage will likely be the clearing of roadways and ensuring that emergency vehicles of first responders can travel effectively. Disaster debris management, which is a crucial part of response stage of disaster management, requires an effective planning and decision making process. Debris management process divides into two steps. The aim of the first step is to remove debris from critical road networks to enable traffic flow and prevent life threatening situations. This step must be started right after the disaster and must be finished in 72 hours. The second step, on the other hand, starts after response stage. Its objective is to remove the rest of the debris during recovery stage (FEMA, 2007). Dozers (Figure 1.2) need to be used for the debris removal. These vehicles should always be ready to operate, be able to clean debris from blocked roads and travel through blocked roads in order to reach more critical blockages.

Figure 1.2 Dozer for Debris Removal

1.3 Problem Definition

One of the most efficient ways of limiting the negative impact of disasters on the first responders is the timely restoration of the road networks disrupted by the disaster debris. Officials responsible for the disaster debris management are often challenged with demanding choices in this restoration planning. The timely removal of the debris out of the roads is not only what is desirable, but is also required for the emergency vehicles of first responders to reach the people needing help on time. Since time is very critical at the response stage, it is essential to decide which locations require to be reached first. Schools, hospitals and potential shelter places are among the facilities that emergency response teams should access in the earliest time possible. To be able to perform this action, it might be necessary to travel through a road network that includes blocked roads due to disaster debris. In this case, blockage must be removed from those critical areas with debris clearance process. Especially *accessibility of a road network* is essential to be able to travel back and forth between these critical facilities and to reach people who need to be rescued. One popular way of measuring the accessibility level of a network is to find the cost (or total length) of the tree that is a subset of the original network and includes a path between any two nodes of the network. This particular tree is usually refereed as *spanning tree* in graph theory literature.

In graph theory, a tree is a subset of the arcs in a network in which any two nodes are connected by exactly one path. By definition, a tree cannot contain any cycle. Given a network $G = (N, A)$ where N and A are the sets of nodes and arcs respectively, a spanning tree T of network G is a tree which includes all of the nodes of G, with $|N-1|$ arcs. For instance, a directed network is given in Figure 1.3a and a spanning tree for this network is provided in Figure 1.3b. Note that a spanning tree can also be found for a directed network by omitting the directions of its arcs. *Minimum spanning tree* is a spanning tree whose sum of arc weights (usually expressed as the length of the arc) is as small as possible. For a given directed or undirected network, the problem of finding such trees are known as minimum spanning tree

(MST) problem (Taha, 2007).

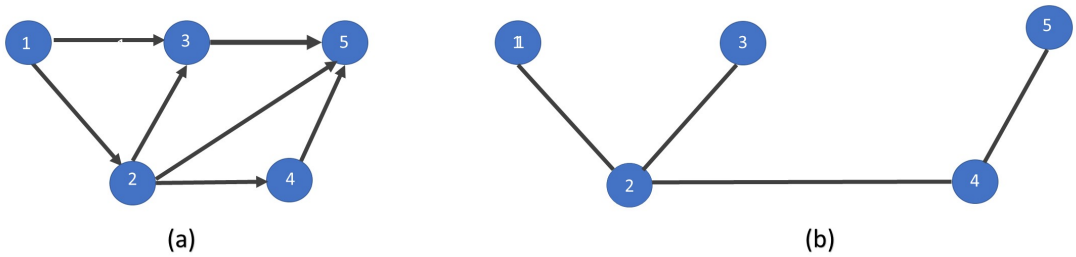


Figure 1.3 A Directed Network and Its Spanning Tree

After the disaster occurs, many roads (arcs) of the road network would be blocked by disaster debris. In return, this would cause emergency vehicles to traverse longer distances on the network while traveling back and forth among critical facilities and points where the help is needed (nodes). In extreme cases, the debris is so widespread that they may not even find a feasible path between some node pairs of the network. In other words, in some severe cases, identifying even a spanning tree may not be possible, which destroys the accessibility of the network altogether.

In this study, we aim to improve the accessibility of a road network which is quantified as the total length of the spanning tree defined on this network. Network modelling and optimization are frequently used in disaster debris management (as already discussed in the literature) in order to make an effective selection of roads for clearance and scheduling the debris removal operations particularly when the number of dozers are limited. Likewise, in the thesis, our objective is to decide in which order the roads blocked by disaster debris will be opened. The performance of this objective is evaluated by measuring the accessibility of the road network over time. We express the accessibility of the network at a given time as a ratio of the MST cost before disaster to the MST cost at that specific time. We consider a novel problem, *Dynamic Spanning Tree (DST)* problem, that can serve as a decision aid to formulate the debris removal efforts in road networks with blocked components. The problem tries to maximize the accessibility (or minimize the inaccessibility) over a planning time horizon. The problem is dynamic since the structure of the

network is subject to change over time after the restoration of a blocked arc in the network. To the best of our knowledge, this thesis is the first effort that addresses this problem.

1.4 Limitations of the Research

This thesis focuses on the debris removal on road networks taking place aftermath of a disaster. The aim of the thesis is to develop a debris removal plan that determines in which order the blocked roads will be opened. We have assumed that the size of the road network, which the debris removal plan is applied to, is such that a single dozer would be sufficient to clear the disaster debris in reasonable amount of time. If the planning needs to be done for larger-sized networks, our methodologies still apply after assigning each dozer to a certain region or neighborhood.

We presume that dozers are readily available immediately after disaster occurs. According to a debris removal strategy, we may prefer to skip some blockages and direct a dozer to a more critical debris clearance location. It is assumed that whenever required, the dozers are capable of passing through blocked roads in order to reach these critical blockages but this traveling time is not added to our mathematical model. Moreover, the distances traveled by the dozer from one debris removal location to another are not considered in this study, even though they may occasionally have a direct impact on the debris removal strategies.

Last but not least, we assumed the amount of time needed for each debris removal activity is discrete. This assumption is inevitable since the mathematical model which will be discussed in Chapter 3 has a "time" index representing each time period of the planning horizon. Here, each time period is defined as an equal spaced time interval and the debris removal on a road is assumed to take an integer number of time periods.

1.5 Outline of the Thesis

The reminder of the thesis is organized as follows. Chapter 2 provides details about the previous work so far. This chapter is divided into two sections. Section 2.1 summarizes previous work related to the socio-economic aspect of the disaster management, while Section 2.2 consists of previous study related to various methods, applications and algorithms designed for pre- and post-disaster phases of debris clearance strategies. Chapter 3 introduces a new problem and provides the optimization model generated for this problem. Chapter 4 proposes four heuristic methods and formally describes them using pseudo-codes. This chapter also demonstrates the implementation of these heuristics on an illustrative network and assess their performance by comparing them with the optimal results. Chapter 5 gives information about the data sets of two real road networks used in the thesis. It also provides the experimental results of the heuristic methods and compares their performance. Chapter 6 concludes the thesis with a summary and possible future research directions.

2. LITERATURE REVIEW

2.1 The Socio-Economic Aspect

Reinhardt et. al. (2011) underlines the importance of health related rehabilitation during disaster to reduce injuries, deaths and disabilities. Alexander (2012) focuses on disasters in terms of social vulnerability and indicates that vulnerability is a major factor of risk and the key aspect of disaster effects. The study submits a model which shows how the combination of culture, history and physical hazards affects the vulnerability to disasters. Usman et. al. (2013) underlines that disaster management is an important part of socio-economic well being and national security and it needs to be a sustainable process. Disaster management methods should be applied for two purposes: one of them is to enhance resiliency to natural disaster and the other is to prevent the increase of vulnerability to disasters. Felbermayr and Gröschl (2014) aim to show that natural disasters reduce GDP (gross domestic product) per capita and state that no previous study exists which shows the relationship between GDP and natural disasters. To support their claim, they build a comprehensive database for natural disasters. Büyükkaracığan (2016) explains that the massive growth of population, the demolition of nature, the unplanned urbanization and technological developments cause natural disasters which are increasing day after day. This work criticizes the methods applied for disaster management and defends that the crisis management in Turkey only focuses on disaster management period so that this leads to failures in economic, social and political crisis. With regard to this context, they investigate the crisis and disaster management legislation and studies in Turkey and make evaluations and recommendation for an efficient disaster management. Marin and Modica (2017) propose methods to generate a report based on the status of economic activities after natural disasters and with the aim of this

they use input-output models to give information regarding various socio-economic parameters such as population, employment rate, turnover and capital stock.

2.2 Mathematical Modelling and Optimization Aspect

Cho, Gordon and Richardson (2000) present a model for the impacts of earthquakes on industry and transportation, which also measures economic effects of the disaster. Feng and Wang (2003) aim to improve a scheduling model for highway emergency rehabilitation for the first 72 hours period for a post-earthquake phase when there are limitations on the time and resource. The objectives of the model are maximizing the performance of emergency rehabilitation, minimizing the risk for rescue workers and maximizing life savings. Murray, Matisziw and Grubesic (2007) propose an integer programming model which determines limitations of flow disruption after an interdiction in a telecommunication network. Matisziw and Murray (2009) formulate a model to determine node and arc blockages which are crucial and can prevent traffic flow. They conduct experiments in the Ohio interstate highway to assess the highway's vulnerability to disasters. Due to the fact that disruption of arcs and nodes in a network is unavoidable, it is important to restore damaged network in a most efficient way. With regard to this context, Matisziw, Murray and Grubesic (2010) propose an optimization model for disaster recovery stage of a network. Model can be used for two objectives such as cost minimization and system flow maximization. Günnec and Salman (2011) aim to evaluate the level of safety in İstanbul highway system and carry out a performance measurement after an earthquake when links are most probably damaged. Duque and Sörensen (2011) seek to repair a rural network after it has been disrupted by a disaster which happened naturally or by human intervention. The objective of the study is to maximize the accessibility of people in need, under a budget and time limitations. The accessibility of a town is demonstrated as the traveling time between the rural center and the nearest regional center. A weight is defined for the importance of each node and the aim is to minimize the weighted sum of the shortest paths. The study proposes a non-linear binary programming model which cannot be solved by a commercial solver and uses a

randomized adaptive search method and several meta-heuristics. Aksu and Ozdamar (2014) present a dynamic model both for response and recovery stages of disaster management. The purpose of the model is to maximize the cumulative network accessibility by restoring the roads as soon as possible under limitations; and in order to provide this, they use a dynamic path based mathematical model. Pramudita and Taniguchi (2014) study HTE debris collection and transportation routing problem after a disaster occured in Tokyo city. They use a capacitated arc routing method by adding a new constraint called access possibility by taking into account the limited transportation from one point to another because of the blocked roads. Özdamar, Aksu and Ergüneş (2014) focus on first 72 hours after a disaster. The purpose of their study is to maximize network accessibility. They use shortest path algorithm to measure the inaccessibility level of blocked road network after a disaster and to minimize completion time throughout debris clearance with limited number of equipments for a post-disaster road recovery problem. Study has two goals; first one is to maximize accessibility of the road network during operation application process and the second is to minimize completion time. Çelik, Ergun and Keskinocak (2015) develop a stochastic model for the debris clearance especially by taking various post-disaster cases into account. In the model, the limited information about the road debris rate is updated while blockage cleaning process continues. With the help of this model, they try to decide the order of the roads to be cleaned in each period in order to maximize sufficient relief demand. Onan, Ülengin and Sennaroğlu (2015) present a study based on disaster waste management. They present a framework to determine temporary waste storage locations and allocate waste source points to those storage points. Objective of the study is to minimize the cost and number of people who are under risk of exposure to hazardous waste. Study integrates disaster loss estimation methods with post disaster management. Proposed framework is applied to city of İstanbul according to predicted earthquake scenarios. Şahin, Kara and Karasan (2016) deal with debris removal in response stage after an earthquake. The problem is defined as removing debris from roads as quickly as possible in order to prevent a predicted disaster from affecting critical areas. The proposed method for the problem provides a critical path for an emergency response vehicle and a strategy

for the clearance of blocked arcs. An NP-hard mathematical model is used in this study. Objective of the model is to minimize total traveling time and debris removal time until all critical nodes are visited. Berktaş, Kara and Karasan (2016) develop a model which provides solution methodologies in the response stage by planning debris removal activities in specific blocked arcs so as to reach critical nodes such as hospitals and schools. In this context, this work presents two mathematical models with different objectives. First model aims to minimize the total time spent while reaching all critical nodes. The second model, on the other hand, aims to minimize the total weighted travel times where the weight of a critical node indicates its priority. Mixed integer programming is used for both models.

3. DYNAMIC SPANNING TREE PROBLEM

In this chapter, we discuss a mathematical model for the Dynamic Spanning Tree (DST) Problem. We assume that there is a damaged road network in which some arcs (roads) are blocked due to debris occurred after a disruptive disaster. In this problem, we are interested in determining (1) the set of blocked arcs that will be cleared (i.e. the network decisions) and (2) the order that the dozer will complete the debris clearance tasks (i.e. the scheduling decisions).

The proposed mathematical model will determine the roads to be cleared and the sequence of these clearances in order to minimize the inaccessibility level over time. Once a road is cleared from debris, it will become available for emergency vehicles to travel through. As mentioned earlier, in this study we measure the inaccessibility level at a time as a function of MST cost at that specific time. Therefore, as long as there is an opportunity to improve the cost of MST, the accessibility of the network can be improved.

Before proceeding to the mathematical model of DST problem, we first need to discuss how the classical MST problem can be mathematically formulated. Since the optimization model of classical MST needs to be embedded to the DST problem, its model is required to be computationally efficient.

3.1 Classical Minimum Spanning Tree Problem

For a given (directed or undirected) network, classical MST problem tries to find a spanning tree whose sum of arc costs is as small as possible. There may be more than one MST solutions with the same cost, if some of the arc costs are the

same. However, there will be a unique MST solution if each arc has a distinct cost. Unique solution is mainly the case in many realistic MST problems, such as the road networks where it is highly improbable to have any two paths with exactly the same length.

Prim's algorithm (Prim , 1957) and Kruskal's algorithm (Kruskal, 1956) are two famous methods that can solve classical MST problems. Both algorithms make the MST grow one arc (and node) at a time. Prim's algorithm starts with an arbitrary node and expand the tree T with the least-cost arc (x, y) such that x is in T and y is not yet in T. Kruskal's algorithm, on the other hand, makes the tree T grow by adding the least-cost arc to T at each step without forming a cycle. Even though both algorithms are greedy methods, they are capable of finding the optimal solution to the classical MST problem in polynomial time.

Besides these algorithms, it is also possible to solve the classical MST problem using linear programming (LP) models. We cover two popular LP formulations here. It should be noted that both formulations are defined for complete networks (i.e. networks in which every pair of nodes is connected with an arc). The first LP formulation of the MST problem for a given complete network $G = (N, A)$ is as follows.

$$min \sum_{i,j} c_{i,j} \cdot x_{i,j} \tag{3.1}$$

subject to:

$$\sum_{i,j} x_{i,j} = n - 1 \tag{3.2}$$

$$\sum_{(i,j)\in S} x_{i,j} = |S| - 1 \quad \forall S \subset N \tag{3.3}$$

$$0 \leq x_{i,j} \leq 1 \quad \forall (i,j) \in A \tag{3.4}$$

In this formulation, $x_{i,j}$ is the only variable which decides whether arc (i, j) is included in the selected spanning tree or not (1: included, 0: not included). $c_{i,j}$ denotes the cost for arc (i, j). Objective function is given by the expression in 3.1 and it minimizes the sum of arc costs selected for the spanning tree. Constraint 3.2 ensures that there are $n-1$ arcs in the spanning tree. Constraint 3.3 is the subtour elimination constraint and it guarantees that any subset of $|N|$ nodes must have at most $k-1$ arcs contained in that subset. Constraint 3.4 restricts the value of $x_{i,j}$ between 0 and 1. Even though, $x_{i,j}$ is defined as a binary variable, this LP model will find integral values for $x_{i,j}$ since the model is totally unimodular. Nevertheless, the formulation has at least $2^{|N|}$ constraints due to Constraint 3.3, which makes the size of the problem grow exponentially as the number of nodes in the network increases.

Martin (1991) presents another LP formulation for the classical MST problem with polynomial number of constraints. This LP formulation for the complete network $G = (N, A)$ is as follows.

$$min \sum_{i,j} c_{i,j} \cdot x_{i,j} \tag{3.5}$$

$$y_{i,j,k} + y_{j,i,k} = x_{i,j} \quad \forall (i,j) \in A, \quad \forall k \in N \tag{3.6}$$

$$\sum_{j \in N} y_{i,j,k} \leq 1 \quad \forall (i,j) \in A, \quad \forall k \in N, \quad i \neq k \tag{3.7}$$

$$y_{k,j,k} = 0 \quad \forall j, k \in N \tag{3.8}$$

$$0 \leq x_{i,j}, y_{i,j,k} \leq 1 \quad \forall (i,j) \in A \tag{3.9}$$

The variable $x_{i,j}$ and the parameter $c_{i,j}$ has the same exact definition as given in the first LP formulation. The variable $y_{i,j,k}$ takes the value of 1 if arc (i, j) is included

in the spanning tree and node k is on the side of node j; 0 otherwise. Equation 3.5 is the objective that minimizes the cost of the spanning tree. Constraint 3.6 ensures that there are $n-1$ arcs in any spanning tree. Constraint 3.7 denotes that if arc (i,j) is selected into the spanning tree (i.e. if $x_{i,j}=1$), then any node $k \in \{N\}$ must be either on the side of j or on the side of i. If arc (i,j) is not in the spanning tree (i.e. if $x_{i,j}=0$), then no node $k \in \{N\}$ can be on the side of j or i. Constraint 3.7 makes sure that if arc (i,j) is included in the spanning tree and a node $k \in \{N\}$ is on the side of node j (i.e. if $y_{i,j,k}=1$), node k cannot be on the side of any other node which is directly connected to node i in the spanning tree. Constraint 3.8 ensures that if arc (k,j) is included in the spanning tree, node k cannot be on the side of node j. Constraint 3.9 restricts both variables $x_{i,j}$ and $y_{i,j,k}$ between 0 and 1. Similar to the first formulation, Martin's formulation is totally unimodular. This means the LP formulation always finds integer optimal solutions. The advantage of this formulation over the first one is that it has polynomial number of constraints. In the worst case, any above expression (3.6-3.9) can generate at most $|N|^3$ constraints.

3.2 Mathematical Model of Dynamic Spanning Tree Problem

In this section, we propose a mixed integer programming (MIP) model for DST problem. This mathematical formulation basically involves an LP model of MST problem for each discrete time period. These LP models are formulated according to the network in each time period, structure of which is determined by the scheduling decisions. The definitions of sets, parameters and decision variables which are used in our dynamic spanning tree problem are as follows.

3.2.1 Sets

N = Set of all nodes$(1,...,n)$
A = Set of unblocked roads
A' = Set of blocked roads
T = Discrete time periods,$(1,...,\tau)$

3.2.2 Parameters

$\phi_{i,j}$: Debris removal time for arc $(i,j) \in \{A'\}$

$c_{i,j}$: travel time on arc $(i,j) \in \{AUA'\}$

$I(t)$: The inaccessibility level at time t

C^u : The cost of MST in the road network before disaster (unblocked)

$C(t)$: The cost of MST at time t

3.2.3 Decision Variables

$X^t_{i,j}$ = 1 if arc $(i,j) \in A \cup A'$ is included in the spanning tree at time period t; 0 otherwise.

$Y^t_{i,j,k}$ = 1 if arc $(i,j) \in A \cup A'$ is included in the spanning tree and node k is on the side of node j at time period t; 0 otherwise.

$\alpha^t_{i,j}$ = 1 if the restoration of arc $(i,j) \in A'$ is completed at the end of time period t; 0 otherwise.

$\beta^t_{i,j}$ = 1 if arc $(i,j) \in A'$ is already restored at time period t ; 0 otherwise.

We can categorize our decision variables into two groups. $X^t_{i,j}$ and $Y^t_{i,j,k}$ are "network" variables related to the mathematical model of the spanning tree problem, while $\alpha^t_{i,j}$ and $\beta^t_{i,j}$ are "scheduling" variables that decide in which order the disrupted arcs need to be restored. Note that all these variables have a time index "t" because of the dynamic nature of the problem.

3.2.4 Mixed Integer Programming Model for Dynamic Minimum Spanning Tree

Our objective function aims to minimize the inaccessibility measure through a certain planning horizon $T \in 0, ..., \tau$. The inaccessibility measure in time period t is given by Equation 3.10. Note that the inaccessibility level can be equal to 1 (i.e. 100%) if no spanning tree can be found on the damaged network.

$$I(t) = 1 - \frac{C^u}{C(t)} \tag{3.10}$$

As shown in Equation 3.11, the cumulative inaccessibility CI we aim to minimize can be found by summing the inaccessibility measure $I(t)$ over the planning horizon.

$$CI = min \sum_{i=1}^{\tau} (1 - \frac{C^u}{C(t)}) \tag{3.11}$$

Since C^u is a constant in Equation 3.11, the value of CI can only be minimized by minimizing $\sum_{i=1}^{\tau} C(t)$. Since $C(t) = \sum_{i,j}(C_{i,j} \times X_{i,j}^t)$ gives the cost of the minimum spanning tree in time period t, Equation 3.11 can be reduced to Equation 3.12, which is basically a linear function of variable $X_{i,j}^t$.

$$min \sum_{i=1}^{\tau} C(t) = \sum_{t=1}^{\tau} \sum_{i,j} c_{i,j} \cdot X_{i,j}^t \tag{3.12}$$

Our model contains three types of constraints: **i)** availability constraint (3.14) deciding whether an arc can be used for spanning tree, **ii)** constraints (3.15-3.18) related to the formulation of minimum spanning tree problem, and **iii)** constraints (3.19-3.23) to represent the scheduling decisions in the restoration process. Consequently, the entire mathematical formulation for the *Dynamic Minimum Spanning Tree* problem can be given as follows:

$$min \sum_{t=1}^{\tau} \sum_{i,j} c_{i,j} \cdot X_{i,j}^t \tag{3.13}$$

subject to:

$$x_{i,j}^t \leq \beta_{i,j}^t \quad \forall (i,j) \in A', \quad \forall t \in T \tag{3.14}$$

$$\sum_{i,j}(X_{i,j}^t) = n-1 \quad \forall t \in T \tag{3.15}$$

$$Y_{i,j,k}^t + Y_{j,i,k}^t = X_{i,j}^t \quad \forall (i,j) \in A \cup A', \quad \forall k \in N, \quad \forall t \in T \tag{3.16}$$

$$\sum_{j \in N} Y_{i,j,k}^t \leq 1 \quad \forall (i,j) \in A \cup A', \quad \forall k \in N, \quad \forall t \in T, \quad i \neq k \tag{3.17}$$

$$Y_{k,j,k}^t = 0 \quad \forall j,k \in N, \quad \forall t \in T \tag{3.18}$$

$$\sum_{(i,j) \in A'} \sum_{s=t}^{min(\tau, t+\phi_{i,j}-1)} \alpha_{i,j}^s \leq 1 \quad \forall t \in T \tag{3.19}$$

$$\beta_{i,j}^t - \beta_{i,j}^{t-1} = \alpha_{i,j}^{t-1} \quad \forall (i,j) \in A', \quad \forall t \in T \setminus \{1\} \tag{3.20}$$

$$\beta_{i,j}^t \leq \beta_{i,j}^{t+1} \quad \forall (i,j) \in A', \quad \forall t \in T \tag{3.21}$$

$$\sum_{t=1}^{\phi_{i,j}} \beta_{i,j}^t = 0 \quad \forall (i,j) \in A', \quad \forall t \in T \tag{3.22}$$

$$\sum_{t=1}^{\phi_{i,j}-1} \alpha_{i,j}^t = 0 \quad \forall (i,j) \in A', \quad \forall t \in T \tag{3.23}$$

$$0 \leq X_{i,j}^t, Y_{i,j,k}^t \leq 1 \quad \forall (i,j) \in A \cup A', \quad \forall t \in T \tag{3.24}$$

$$\alpha_{i,j,t}, \beta_{i,j,t} \in \{0,1\} \quad \forall (i,j) \in A', \quad \forall t \in T \tag{3.25}$$

Equation 3.11 denotes the cumulative inaccessibility level summed over the planning horizon T. Equation 3.13, on the other hand, gives the cumulative cost of MST problems summed over T. As discussed earlier, minimizing the objective function given by Equation 3.13 is equivalent to minimizing the cumulative inaccessibility. Constraint 3.14 ensures that a blocked arc can be included in a spanning tree only if it has been already restored at time t. Constraint 3.15 ensures that there are $n-1$ arcs in any spanning tree at any time point. Constraint 3.16 denotes that if arc

(i, j) is selected into the spanning tree at time t (i.e. $X_{i,j}^t = 1$), any node $k \in \{N\}$ must be either on the side of j (i.e. $Y_{i,j,k}^t = 1$) or on the side of i (i.e. $Y_{j,i,k}^t = 1$). If arc (i, j) is not in the spanning tree at time t (i.e. $X_{i,j}^t = 0$), then any node $k \in \{N\}$ cannot be on the side of j or i (i.e. $Y_{i,j,k}^t = Y_{j,i,k}^t = 0$). Constraint 3.17 makes sure that if arc (i, j) is included in the spanning tree and a node $k \in \{N\}$ is on the side of node j at time t (i.e. $Y_{i,j,k}^t = 1$), node k cannot be on the side of any other node which is directly connected to node i in the spanning tree. Constraint 3.18 ensures that at time period t if arc (k, j) is included in the spanning tree, node k cannot be on the side of node j. Constraints 3.15-3.18 are adopted from Martin's formulation (Martin, 1991) discussed in Section 3.1 . They ensure that the spanning tree is presented for a connected graph and provides continuity on the spanning tree network. Constraint 3.19 indicates that at most one blocked arc can be cleaned in time period t and if the restoration of an arc is completed at the end of time period s then we must work on it for $\phi_{i,j}$ successive periods from $s - \phi_{i,j} + 1$ through s. Constraint 3.20 expresses that a blocked arc becomes available at time period t only after it is completed in the previous period $t - 1$. Constraints 3.21 ensures that if an arc is available in time period t it is also available in subsequent periods (from period t to period τ). Constraints 3.22 is a logical constraint that makes sure of that an arc cannot become available in a time period earlier than its required processing time. Constraints 3.23 is another logical constraint dictating that at the earliest we can complete an arc at the end of a time period equal to its required processing time. Constraints 3.22-3.23 help to strengthen the linear programming relaxation of the problem. Constraint 3.24 and 3.25 gives the types of decision variables. Note that network variables ($X_{i,j}^t$ and $Y_{i,j,k}^t$) are not required to be binary due to the discussion we made in Section 3.1. Therefore, the formulation of the dynamic spanning tree problem is a mixed integer programming model. In the next section, we run this mathematical model for a hypothetical network.

3.3 An Illustrative Example

In this chapter we present the implementation of MIP model for DST on an illustrative example. To be able to perform this, we used a network consisting of 10 nodes and 23 arcs. The network is shown in Figure 3.1.

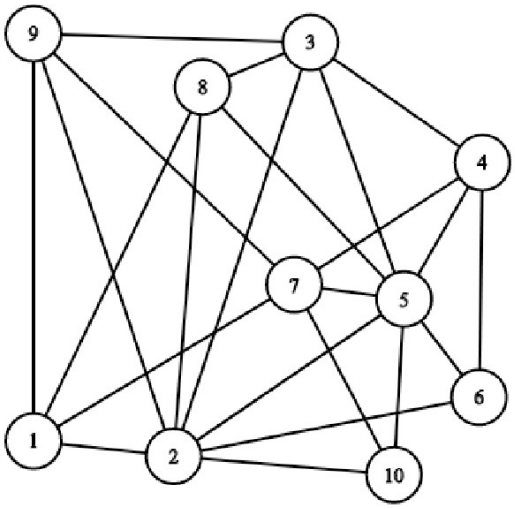


Figure 3.1 Small Network Graph

The distance and restoration time (if disrupted) of each arc are given in Table 3.1. The first number in each cell shows the distance, while the second number (in parenthesis) gives the restoration time.

Table 3.1 Distances and Restoration Times of Small Network

	1	2	3	4	5	6	7	8	9	10
1		14.74(6)					0.33(1)	7.34(3)	10.49(5)	
2			13.89(6)		9.95(4)	13.4(6)		16.51(7)	16.96(7)	5.76(3)
3				19.87(8)	13.39(6)			4.42(2)	10.73(5)	
4					18.87(8)	17.33(7)	3.03(2)			
5						0.89(1)	10.87(5)	7.02(3)		16.13(7)
6										
7									14.55(6)	10.29(5)
8										
9										
10										

Table 3.2 provides three possible damage scenarios, D1, D2 and D3, where 8, 11 and

16 arcs are disrupted respectively. Disruptions are labeled with "1" in the table.

Table 3.2 Damage Scenarios for Small Network

ARCS	1-2	1-7	1-8	1-9	2-3	2-5	2-6	2-8	2-9	2-10	3-4	3-5	3-8	3-9	4-5	4-6	4-7	5-6	5-7	5-8	5-10	7-9	7-10
D1			1	1				1	1				1	1		1		1					
D2	1	1		1		1				1		1	1	1		1		1			1		
D3	1	1	1		1		1	1		1	1		1	1		1		1	1	1		1	1

We solve MIP model of DST for each one of the damage scenario one by one. CPLEX optimization Studio 12.7 is used for optimally solving the mixed integer models. The script used in CPLEX Optimization Studio is given in Appendix A.1. These runs are executed on an Intel Core i7-2620M, 2.7 GHz computer with 8 GB of RAM. All experiments take less than 5 seconds. The results for D1, D2 and D3 are summarized in Table 3.3 (a), (b) and (c), respectively. Since τ is selected as $\tau = 20$ in the CPLEX model, the results till the end of 20th period are given in Table 3.3 as shown in the "TIME" column of the table. As it can be noted, in all three damage scenarios, best MST value (MST before the disaster) is attained way before reaching the end of the planing horizon.

To be more specific, the solutions found for D1, D2 and D3 can complete all necessary restorations and achieve the best possible MST (i.e. C^u) at the end of $t = 11$, $t = 16$ and $t = 13$ respectively. MST results can be seen on the "MST" column of the table. I denotes inaccessibility level at time t as we discussed before in Chapter 3. The aim of the model is to minimize the total inaccessibility measure of the spanning tree at time t in the resulting network. The "Restored Nodes" column denotes the blocked edges being restored at time t and therefore included into the spanning tree once restored. Since there is only one dozer, at most one blocked edge is cleaned in any time period t. If an edge is available for the first time in period t, it means that debris removal for the edge must have been completed at the end of the previous period $(t-1)$. Moreover, if an edge is available in time period t, it should also be available in time period $(t+1)$ and the following periods.

I measurement is calculated by subtracting the division of undamaged network MST to damaged network MST at time t from 1 (i.e. $I(t)$) and I can only take values between zero and one. So if the model could not find any spanning tree on the damaged network, this means MST cost is infinite and so that the value of I becomes 1. On the other hand, if the model finds a spanning tree on the damaged network whose cost is equal to the cost of minimum spanning tree on the undamaged network, then I measure becomes zero which is the best level that can be obtained. It can be seen on the table that numbers on both "MST" and "I" columns are decreasing as the model continues to restore nodes; and once the model reaches to the cost of undamaged MST (the best possible value), MST cost becomes constant in the following periods and I becomes zero. It can be seen that optimum result is obtained at the end of $t = 11, 16$ and 13 for the scenarios D1, D2 and D3 respectively.

Table 3.3 MIP Results of Damage Scenarios

	(a)			(b)			(c)		
TIME	MST (km)	I	Restored Nodes	MST (km)	I	Restored Nodes	MST (km)	I	Restored Nodes
T1	77.73	0.367	(5-6)	96.89	0.492	(5-6)	∞	1	(5-6)
T2	65.22	0.245	(3-8)	81.27	0.394	(1-7)	∞	1	(3-8)
T3	65.22	0.245	(3-8)	70.74	0.304	(3-8)	∞	1	(3-8)
T4	56.25	0.125	(1-8)	70.74	0.304	(3-8)	94.13	0.477	(1-7)
T5	56.25	0.125	(1-8)	61.28	0.197	(2-10)	75.60	0.349	(2-10)
T6	56.25	0.125	(1-8)	61.28	0.197	(2-10)	75.60	0.349	(2-10)
T7	53.30	0.076	(1-9)	61.28	0.197	(2-10)	75.60	0.349	(2-10)
T8	53.30	0.076	(1-9)	53.63	0.082	(1-9)	65.23	0.245	(1-8)
T9	53.30	0.076	(1-9)	53.63	0.082	(1-9)	65.23	0.245	(1-8)
T10	53.30	0.076	(1-9)	53.63	0.082	(1-9)	65.23	0.245	(1-8)
T11	53.30	0.076	(1-9)	53.63	0.082	(1-9)	55.61	0.115	(5-8)
T12	49.23	0		53.63	0.082	(1-9)	55.61	0.115	(5-8)
T13	49.23	0		49.57	0.007	(2-5)	55.61	0.115	(5-8)
T14	49.23	0		49.57	0.007	(2-5)	49.23	0	
T15	49.23	0		49.57	0.007	(2-5)	49.23	0	
T16	49.23	0		49.57	0.007	(2-5)	49.23	0	
T17	49.23	0		49.23	0		49.23	0	
T18	49.23	0		49.23	0		49.23	0	
T19	49.23	0		49.23	0		49.23	0	
T20	49.23	0		49.23	0		49.23	0	
Undamaged Network MST= 49.23m									
	CI=1.612			CI=2.521			CI=5.603		

CI values can be seen on the bottom of the table for each damage senario. CI is previously defined as cumulative inaccessibility level. This measurement can be found by summing the inaccessibility measure $I(t)$ over the planning horizon. Since CI measurement depends on I value, it can be seen that the model also aims to minimize cumulative CI value over the planning horizon. As our small example is performed over the same network but with different damage scenarios, it can be seen that CI value is directly proportional with the number of damaged arcs. Hence, the CI value of the scenario D3 (with 16 damaged arcs) is the highest, while CI value of scenario D1 (with 8 damaged arcs) is the smallest. The CI value of scenario D1 can also be derived from Figure 3.2, corresponding to the area under the graph.

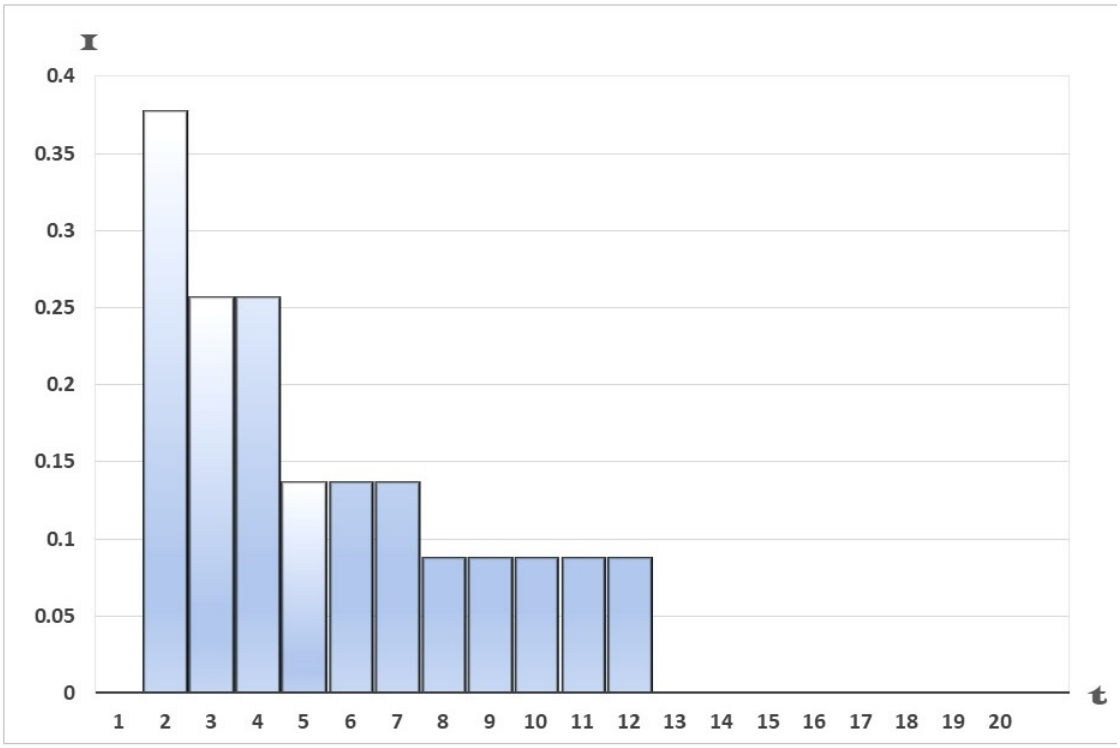


Figure 3.2 CI Graph for Small Network

4. HEURISTIC APPROACHES

In this chapter we propose heuristic methods. In real life, there will be situations where the optimization program cannot solve the dynamic spanning tree optimization model for real data due to complexity of the problem. Therefore we create heuristic methods for our mathematical problem proposed in Chapter 3. Our problem's complexity is directly proportional with the following parameters: the size of the network (number of nodes and arcs), the number of disrupted arcs and the number of debris removal vehicles. One of those complexities or combinations of them makes problem more difficult and time consuming or insoluble as in our case. Thus, heuristic methods need to be introduced to this problem as opposite to exact method (optimization model) to be able to obtain results. We developed four distinct heuristic approaches for blocked arc selection and generated codes in MATLAB 2017a. In order to create our heuristic methods, we used a well known algorithm, the Prim's algorithm.

First heuristic algorithm chooses the road which has the minimum debris removal time and then adds the selected road to minimum spanning tree. The second heuristic algorithm selects the blocked road which has maximum number of arc connections and puts this road into minimum spanning tree road network. If the heuristic cannot find a blocked road according to the previously stated rules, then algorithm selects the blocked road with the minimum distance and adds it into the network. The third heuristic algorithm calculates Prim's algorithm for both undamaged and damaged networks. Then it selects the blocked road which reduces the minimum spanning tree cost utmost and adds the selected road into network. The fourth heuristic method calculates Prim's algorithm both for before and after disaster networks, then checks all blocked roads individually by adding them one by one into

road network. Each of these steps generates a new road matrix. The heuristic calculates Prim's algorithm for each generated road matrix, then generates another matrix by subtracting current Prim's solution from previous solution and dividing the result to repairing time to corresponding damaged road. Finally, it adds the road which gives the maximum ratio and continues till all roads are opened.

In order to explain heuristic algorithms, we present the pseudo codes next (refer to Appendices B.1, B.2, B.3, B.4, B.5, B.6, B.7, B.8 and B.9 for the MATLAB codes used for heuristic methods).

4.1 Heuristic 1: Selecting the Damaged Road with Minimum Repairing time

To be able to decide the selection of roads, we used a well known method which is called Prim's algorithm. It is a frequently used algorithm to obtain minimum spanning tree. At first we take an undamaged network and form it as a symmetric adjacency matrix consisting from nodes and distances. This matrix is named as N. Then we calculate Prim's algorithm for this undamaged network, calling the solution as SOLglobal. Subsequently we assign some blocked roads to this network and generate a logical matrix that shows damaged roads of network as 1 and unblocked ones as 0, logical network matrix being called as betaD. We obtain the cost of damaged roads by element-wise multiplication of N and betaD, this output matrix being called as D. Afterwards, we obtain the cost matrix A for undamaged roads by subtracting D from N. We called solution of Prim's algorithm for network A as SOLBest. We start a while loop which will be working while counter is greater than zero and add a counter based on the number of damaged arcs. Hence, when there is no damaged arcs left, while loop ends. We start a for loop within while loop which will work as many times as the number of damaged arcs; and when there is no damaged arcs left, for loop ends.

For heuristic 1, we generate a repairing time matrix for which the time values are

assigned proportionally with road distances and as discreet values. To be able to decide the selection of roads we use repair times as a decision making procedure. After we define N, D and betaD matrices, we also define a repairing time matrix and called it T. We find the matrix of damaged road repairing times by multiplying T and betaD element wise and call the output matrix DART. Next, we subtract DART from T to find repairing times of undamaged arcs and name this matrix as ART. We start a while loop which will work as long as the counter is greater than zero and add a counter based on the number of damaged arcs. So when there is no damaged arcs left, the while loop ends. We start a for loop within the while loop which will work as many times as the number of damaged arcs and when there is no damaged arcs left, the for loop ends. The algorithm checks each damaged roads repairing time, adds the arc that has the minimum repairing time to ART matrix and finds the corresponding indices of the added arc in the undamaged A distance matrix. Then it adds this damaged arc to A, so that it becomes unblocked and we assign the cost of this arc as infinity in D and DART to prevent the reselection of the arc. Then the algorithm calculates Prim's. If there are damaged arcs which have equal repairing times with the minimum value, the algorithm compares the distances of the damaged arcs and adds the shortest one. If there are damaged arcs with same distance, the algorithm selects the arc which has the minimum indices and again the cost of the added arc is assigned as infinity in D and DART. Subsequently the algorithm again calculates Prim's. This loop starts from beginning till all damaged roads are opened. Finally, the algorithm gives a minimum spanning tree costs vector, the arc and nodes data of minimum spanning tree and an inaccessibility measure.

Algorithm 1 Algorithm 1 Pseudo-code for Heuristic 1

Initialize the algorithm as follows
Find size of network
Obtain cost matrix for damaged network
Obtain cost matrix for undamaged network
Obtain time matrix for damaged network
Obtain time matrix for undamaged network
Assign all zeros in damaged network matrix to infinity
Calculate number of damaged arcs
Set counter to number of damaged arcs
Find indices of mins of damaged time matrix
while $counter > 0$ **do**
    if *There is only one minimum repairing time in DART matrix* **then**
        Add damaged arc to undamaged time network which has the minimum time
    else
        Add damaged arc to undamaged network which has minimum distance
    end if
    Assign arc added from damaged network as infinity in damaged network
    Assign arc added from damaged time network as infinity in damaged time network
    Set corresponding indices of added arc to zero in betaD
    Find Prim's of undamaged distance network
    Show MST vector
    Calculate inaccessibility
end while

4.2 Heuristic 2: Selecting the Damaged Road with Maximum Weight

For heuristic 2, we add the weight of blocked arcs and generate a weight matrix by taking the sum of the connections of each node in each arc. To be able to decide the selection of roads, this time we use the weight dimension as a decision making procedure. After we define N, D and betaD matrices, we also define a weight matrix named as W. We find the matrix of damaged road weight by multiplying W matrix and betaD element-wise and call this matrix DARW. Later, we subtract DARW from W to find the weight of undamaged arcs and call this matrix ARW. We start a while loop which will work as the counter is greater than zero and add a counter based on the number of damaged arcs. When there are no damaged arcs left, the while loop ends. We start a for loop within while loop which will work as many times as the number of damaged arcs and when there are no damaged arcs left, the for loop ends. The algorithm checks each damaged road's weight and adds the arc that has the maximum weight into ARW matrix. Then it finds the corresponding indices of the added arc in undamaged A distance matrix and adds this damaged arc, so that it becomes unblocked. The cost of each unblocked arc is assigned as infinity in D and DARW to prevent reselection of arc. Then the algorithm calculates Prim's. If there are damaged arcs which have equal weights, the algorithm compares the distances of damaged arcs and adds the shortest one. If there are damaged arcs with the same distances, the algorithm selects the arc which has the minimum indices. This loop continues until all damaged roads are opened. Finally, the algorithm gives a minimum spanning tree costs vector, costs of the arcs and nodes, the data of minimum spanning tree and an inaccessibility measure.

Algorithm 2 Algorithm 2 Pseudo-code for Heuristic 2

Initialize the algorithm as follows
Find size of network
Obtain cost matrix for damaged network
Obtain cost matrix for undamaged network
Obtain weight matrix for damaged network
Obtain weight matrix for undamaged network
Assign all zeros in damaged network matrix to infinity
Calculate number of damaged arcs
Set counter to number of damaged arcs
Find indices of arcs maximum of damaged weight matrix
while $counter > 0$ **do**
    if *There is only one maximum weight of damaged weight matrix* **then**
        Add damaged arc to undamaged weight network which has the maximum weight
    else
        Add damaged arc to undamaged network which has minimum distance
    end if
    Assign arc added from damaged weight network as zero in damaged weight network
    Assign arc added from damaged network as infinity in damaged network
    Find Prim's of undamaged distance network
    Show MST vector
    Calculate inaccessibility
end while

4.3 Heuristic 3: Selecting the Damaged Road Minimizes Minimum Spanning Tree

The third heuristic method starts by defining N, D and betaD matrices. The algorithm checks each damaged arc, adds each damaged arc into undamaged network individually and calculates Prim's algorithm. It then compares Prim's solutions to choose the minimum one. If the minimum Prim's solution is less than SOLbest, it adds damaged road that gives this minimum solution once added. When selected road is added to the network, this road becomes unblocked and it is added to the undamaged road matrix. To prevent algorithm from choosing the added road again, we assign the cost of arc added from damaged network as infinity in damaged network. If the algorithm cannot find a Prim's solution after adding damaged arcs individually or cannot find a solution which is less than SOLBest or finds an equal solution, it compares the distances of damaged arcs and adds the shortest one. Afterwards, loop starts from the beginning till all damaged roads are opened. Finally, the algorithm gives a minimum spanning tree costs vector, the arc and nodes data of minimum spanning tree and the inaccessibility measure. Drawback of heuristic3 is that it does not take into account debris removal time of the arcs.

Algorithm 3 Algorithm 3 Pseudo-code for Heuristic 3

Initialize the algorithm as follows
Calculate SOLglobal for undamaged network
Find size of network
Obtain cost matrix for damaged network
Obtain cost matrix for undamaged network
Assign all zeros in damaged network matrix to infinity
Calculate SOLBest for damaged network
Generate an infinity matrix to assign solutions and name as SOL
Calculate number of damaged arcs
Set counter to number of damaged arcs
Assign SOLBest vector name as MST
Find indices of damaged arcs
while $counter > 0$ **do**
    for $i \in \{1...counter\}$ **do**
        Calculate Prim's by adding damaged arcs to undamaged network individually
    end for
    if *Prim's algorithm solution is less than SOLBest* **then**
        Assign new Prim's solution as SOLBest
        Add damaged arc to undamaged network that gives the minimum Prim's solution when added to network
    else
        Add damaged arc to undamaged network which has minimum distance
    end if
    Assign arc added from damaged network as infinity in damaged network
    Set corresponding indices of added arc to zero in betaD
    Show MST vector
    Calculate inaccessibility
end while

4.4 Heuristic 4: Selecting the Damaged Road with Maximum Clearance Ratio

The fourth heuristic method starts by calculating Prim's algorithm both for before and after disaster networks after defining N, D and betaD matrices. Then it starts to open blocked roads by checking all blocked roads individually by adding them one by one into road network and generating a new road matrix each time. Prim's algorithm is calculated for each generated road matrix, and another matrix is generated by subtracting current Prim's solution from the previous solution and dividing the result to repairing time for the corresponding damaged road. The road which gives the maximum ratio is added and the algorithm continues till all roads are opened. For heuristic 4, we use clearance ratio as the selection method for damaged roads. Clearance ratio means subtracting current Prim's solution from previous solution and dividing the result to repairing time of the corresponding damaged road. To generate the algorithm, we used heuristic 3 and added some more calculations. We define a matrix named as UI which consists of clearance ratio values. Subsequently, we define damaged networks UI value as UIbest. Since no roads are repaired at first, we assign UIbest as zero. We start a while loop which will work as the counter is greater than zero and add a counter based on the number of damaged arcs. When there is no damaged arcs left, the while loop ends. We started a for loop within the while loop which will work as many times as the number of damaged arcs and when there is no damaged arcs left, the for loop ends. Algorithm checks each damaged arc, adds each damaged arc into the undamaged network individually as explained above. It calculates Prim's algorithm and then finds UI for each blocked arc and later compares UI solutions to choose the maximum one and adds the corresponding arc into network. When the selected road is added to the network, this road becomes unblocked and is added to the undamaged road matrix. To prevent algorithm from choosing an added road again, we assign the cost of this arc as infinity and IU as zero in the damaged network. If the current UI result cannot beat the previous one, the algorithm compares the distances of damaged arcs and adds the shortest one. This loop continues until all damaged roads are opened. Finally, the algorithm

gives a minimum spanning tree costs vector, the arc and nodes data of minimum spanning tree and an inaccessibility measure.

Algorithm 4 Algorithm 4 Pseudo-code for Heuristic 4

Initialize the algorithm as follows
Apply steps of Heuristic 3 from 1 to 12
Assign UIglobal for undamaged network as infinity
Generate a zeros matrix to assign UI solutions and name as UI
Assign UIBest to zero
while $counter > 0$ **do**
    for $i \in \{1...counter\}$ **do**
        Calculate prims by adding damaged arcs to undamaged network individually
    end for
    if $SOLBest$ *inequal to* $infinity$ & UI *is equal or less than* $UIBest$ **then**
        Find UI by subtracting current SOL from previous and divide result to added arc's repairing time
        Find maximum indices of UI by checking SOL matrix
        Assign new prims solution as SOLBest
        Assign new UI solution as UIBest
        Add damaged arc to undamaged network that gives the maximum UI value when added to network
    else
        Assign UI as zero
        Add damaged arc to undamaged network which has minimum distance
    end if
    Assign arc added from damaged network as infinity in damaged network
    Assign added UI as zero in UI matrix
    Set corresponding indices of added arc to zero in betaD
    Show MST vector
    Calculate inaccessibility
end while

4.5 Implementation of the Heuristic Methods on the Illustrative Example

We use the heuristic methods on the illustrative example and on possible damage scenarios we introduced in Section 3.3. This example network consists of 10 nodes and 23 arcs as expressed in Table 3.1 and optimal MIP results obtained by CPLEX optimization Studio 12.7 are shown in Table 3.3. In order to perform heuristic methods, the numerical computing software MATLAB 2017a is used. These runs are executed on an Intel Core i5-82550U CPU, 1.8 GHz computer with a RAM of 8Gb. All experiments take less than 5 seconds. The CI results of each heuristic method for D1, D2 and D2 damage scenarios are given in Table 4.1 in details. As it is already stated in Section 3.3 the optimal CI results for MIP model are found as 1.612, 2.521 and 5.603 for three damage scenarios respectively, which are also seen on the last column of the Table 4.1. The first number in each cell shows the CI value , while the second number (in parenthesis) shows how much the heuristic results are far from the optimal CI values.

Table 4.1 Comparison of the Heuristic methods with MIP model

Damage Scenario	$HEU1$	$HEU2$	$HEU3$	$HEU4$	MIP
D1	1.612 (0.0%)	6.82(322.8%)	1.650 (2.3%)	1.612 (0.0%)	1,612
D2	2.832 (12.3%)	12.32 (388.7%)	2.521 (0.0%)	2.521 (0.0%)	2.521
D3	8.23 (46.9%)	33.6 (499.7%)	6.96 (24.2%)	6.12 (9.2%)	5.603

For D1 scenario, we can see that Heuristic 1 and 4 restore the same 4 arcs that the optimal solution has also restored, so they have the same CI result with the optimal solution. For damage scenario 2, the results of heuristic 3 and 4 are same with the optimal CI result. When we check the restored nodes, we find that both heuristic 3 and 4 and the optimal solution have restored the same arcs and with the same order. For damage scenario 3, we can see that heuristic 4 gives the best results and heuristic 3 gives the second best result. It can be seen from the table that heuristic 4 provides the minimum CI for all damage scenarios. Heuristic 3 reaches the optimal

CI result on D2 scenario while heuristic 1 reaches the optimal CI result on D1; so both of them can provide optimal result once. However, when we check their CI results for the three cases, we see that heuristic 4 results are more closer to optimal CI result. We can also see that heuristic 2 always gives the worst results in all cases.

The reason we test the heuristics on this illustrative example is because this is the only problem we can solve with optimality in CPLEX optimization Studio 12.7. While running the MIP model of DST for our neighborhoods, we ran into memory problems and failed to produce a solution. Therefore, unfortunately we are not able to report the results of the MIP model for those neighborhoods.

5. COMPUTATIONAL EXPERIMENTS

We will now discuss the input data used and explain experiments performed in detail and provide results of the experiments. We tested heuristic methods which are discussed in Chapter 4 for the dynamic spanning tree problem proposed in Chapter 3 with two different data sets representing road networks of neighborhoods in the city of İstanbul. We compared the heuristic methods based on inaccessibility measure given in objective function of mathematical method proposed in Chapter 3 to be able to rank heuristic applications and find the superior one. The manageable size of each neighborhoods enables us to solve our heuristic approaches in minutes. Therefore, we can conduct various analyses (results of analyses given in Section 5.2) for debris removal solutions of the dynamic spanning tree problem. Although we have only two road networks, each of the networks are highly complicated in size (number of nodes and arcs), the number of disrupted arcs and the number of debris removal vehicles. The size of the networks makes the original dynamic spanning tree problem unsolvable due to out of memory problem. Therefore, the experiments are carried out by concerning computational performance (computation time) instead of obtaining objective function result (heuristic application introduced in Chapter 4).

We assign 1 dozer to each neighborhood and assume that neighborhoods are big enough such that one dozer can make debris removal. We create the test scenarios based on blockages 10%, 25% and 50%. We assume that our road network will be an undirected graph after an earthquake occurs. This is because when debris shut down the roads, we will not consider the direction of the road to rescue people in need. Arc restoration times are taken directly proportional with arc distances. The longest removal time is assumed to be four days, time periods showed in mathematical model

and each time period is assumed as 12 hours. Arc weights are obtained by sum of number of each nodes connection with another node in an arc.

We obtain real neighborhood road maps from Google Maps. Afterwards we used ArcGIS 10.5 program. ArcGIS is an integrated geographic information system (GIS) tool. To be able to provide information, it uses geographic information and maps. We utilized this tool to obtain arcs, nodes, coordinates and distances data of the selected road networks. On ArcGIS each node on the map is defined one by one and arcs are constructed by following the map. We also used this tool to show minimum spanning tree road network on map after Prim's algorithm is applied to neighborhood networks (Appendix C.1).

5.1 Data Sets

We perform our tests for two road networks based on maps of two neighborhoods in Turkey for the response phase of a disruptive earthquake. An earthquake is the shaking of the earth's surface, caused by the fraction and displacement of earth's crust or the eruption of volcanoes (Earthquake, n. d.). It is a non-routine event which may cause life loss and paralyses social life and economy.

It has been known that Turkey has always been under various natural disaster risks due to its geological structure, tectonic formation and topographical and meteorological features. It is among the countries which frequently encounter natural disasters such as earthquake, flood, landslide, snow slide, heavy rainfall and storm and suffer significantly from them. Earthquakes are the most damaging disasters in terms of life and economy in Turkey. It is a fact that our country is located on the fault lines and the majority of the population lives under the risk of an earthquake. There are lots of negative effects of this non-routine natural event and debris is one of most crucial ones. Debris occurs by the collapse of buildings. Although the urbanization and construction of earthquake-resistant structures has become widespread in recent years, old buildings are still in majority and therefore this process will take time.

Hence, it is crucial to remove debris in an appropriate manner to be able to open blocked roads and reach affected places and prevent deaths of people.

City of İstanbul is a risky area in terms of earthquake. We perform our tests according to the response phase period after a probable İstanbul earthquake. In this context, two neighborhoods in İstanbul were chosen and it is assumed that debris will shut down the roads and prevent emergency aid teams to access humans who need to be rescued from disaster affected areas. These two road networks have different features in terms of magnitude, road intensity, population, number of buildings, road construction and population. The common feature of these road networks is the fact that both are among the high risk areas in terms of earthquake. In this context, blockage rate and availability of road networks has been randomly decided and several scenarios are constructed by considering which road blockages can occur because of the road and/or bridge collapse and build a debris barrier after a disruptive earthquake.

5.1.1 Güven Neighborhood

Güven neighborhood is located in Güngören district in İstanbul. According to the latest research results made for probable İstanbul earthquake, Güngören seems to be among the most risky areas in a post-earthquake scenario (Küçükali, 2018). Buildings of the neighborhood are under risk of collapsing after an earthquake. The reason that we selected this district to perform our test is that its earthquake risk, road and building density is high. Hence, after a disruptive earthquake, the debris amount per square will be too much. Tests are carried out for Güven neighborhood in Güngören district because it reflects properties of Güngören district in terms of road and building network construction. We used real data set consisting of the arcs and nodes in Güven neighborhood. The neighborhood consists from 80 nodes and 126 arcs. The ArcGIS map can be seen on Figure 5.1.

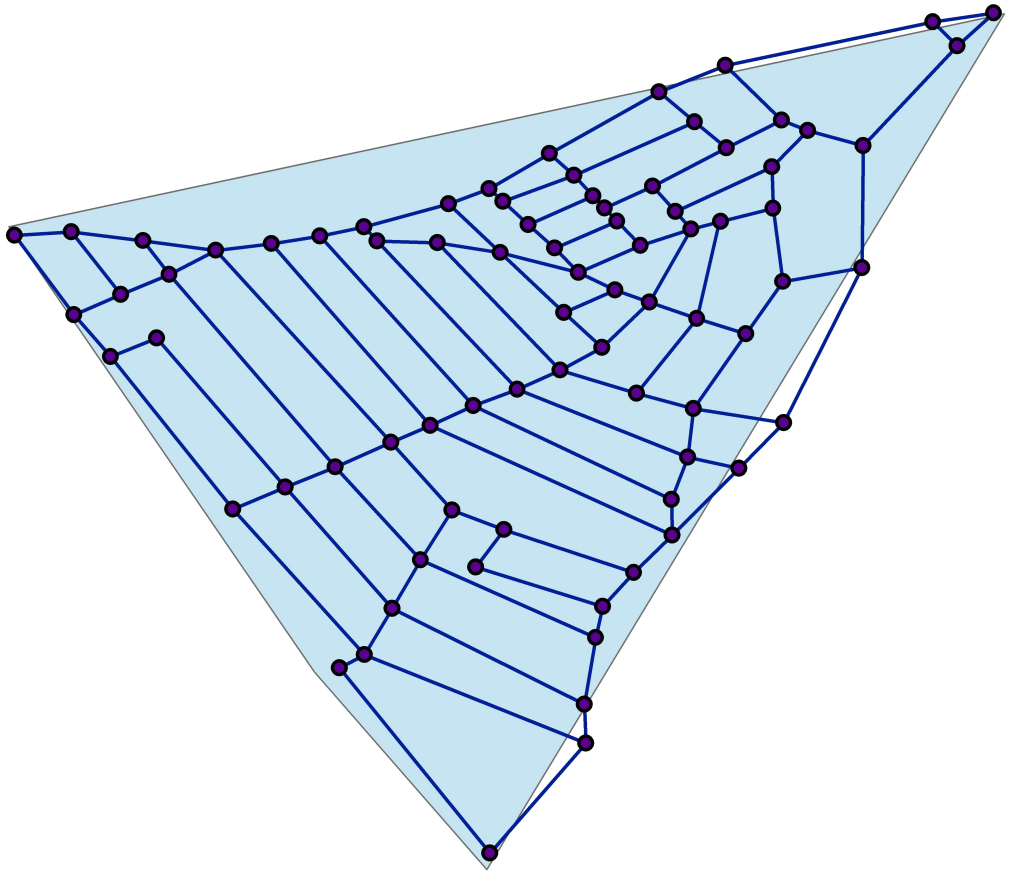


Figure 5.1 ArcGIS Map for Güven Neighborhood

5.1.2 Şirinevler Neighborhood

Şirinevler is a neighborhood of Bahçelievler which is a district of İstanbul. The reason that we chose Bahçelievler is that it is among the most risky and dangerous districts in İstanbul in terms of earthquake as a consequence of unplanned urbanization, high building density and road density (Küçükali, 2018). Bahçelievler has more than 20000 buildings and most of the buildings are old and roads are narrow. As a result of high building density there is a large population and for this reason if a disruptive earthquake occurs, most probably there will be a lot of people who needs to be rescued from debris in this district. To perform our tests, we chose Şirinevler neighborhood in Bahçelievler district because it has all the negative features of Bahçelievler in terms of high building and road density, large population,

narrow roads and unplanned urbanization, hence its riskiness level is high with regards to earthquake. The road network data is a real data set which represents road networks of Şirinevler neighborhood in Bahçelievler in İstanbul, Turkey. The neighborhood consists from 417 nodes and 632 arcs. The ArcGIS map can be seen on Figure 5.2.

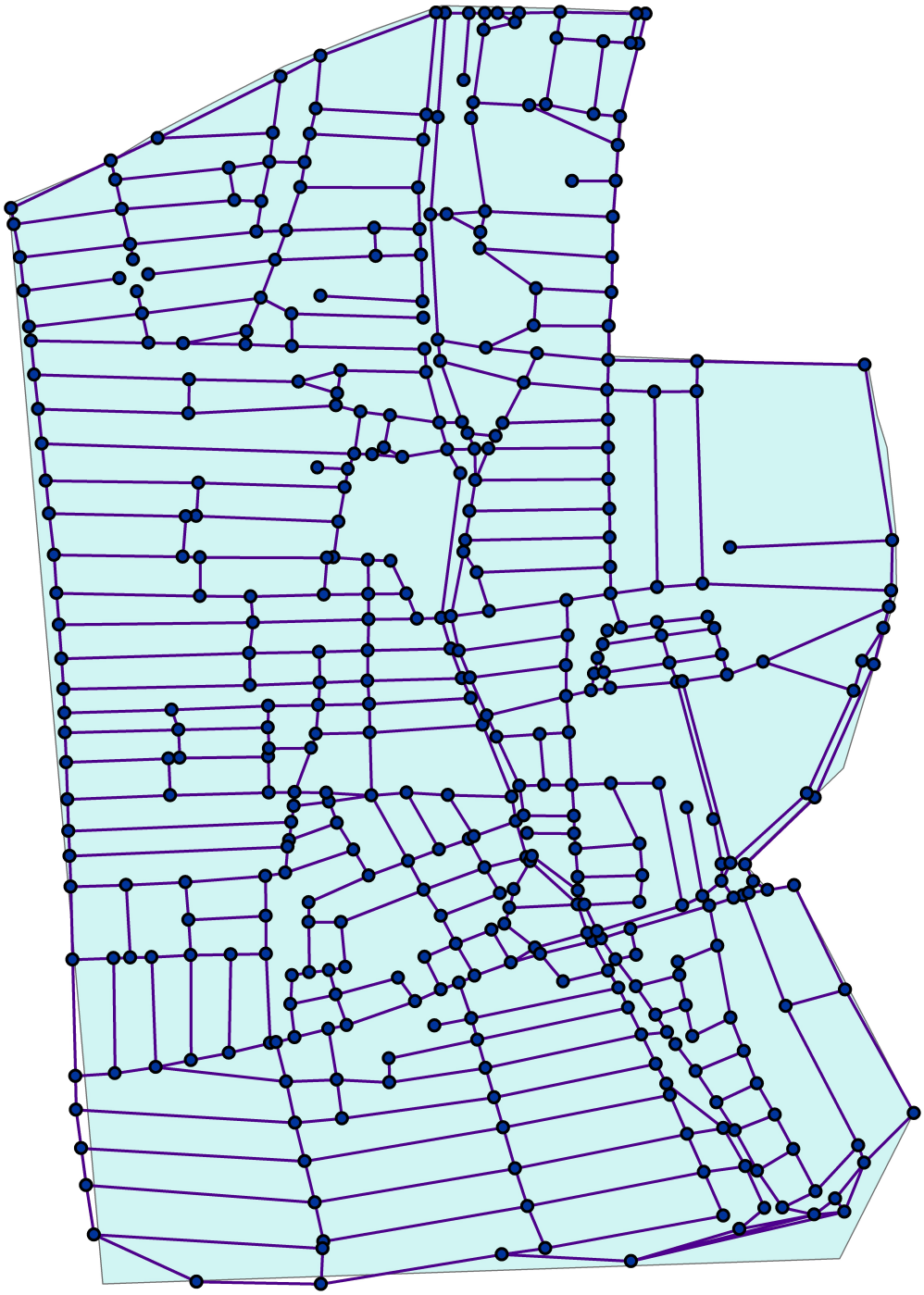


Figure 5.2 ArcGIS Map for Şirinevler Neighborhood

5.2 Results Of Experiments

While running the MIP model of DST for both Güven and Şirinevler neighborhoods, we ran into memory problems and failed to produce a solution. Therefore, unfortunately we are not able to report the results of the MIP model in this section. We present results of our experiments conducted with a data set of road networks in İstanbul. We used numerical computing software MATLAB 2017a. The MATLAB codes we have used in our study are shown in Appendix B. Our MATLAB code uses arcs, nodes, distance, damaged road, time and arc weight data to provide results. Our code uses this data in matrix format; therefore, it is necessary to convert the data into adjacency symmetrical matrix. Since we obtain data from ArcGIS program in list format, we fist prepare our data appropriate for MATLAB code so that it can detect data and provide solution. The code which is proposed in appendix B.1 is utilized to generate an adjacency matrix from list of nodes and arcs. After we receive road network data in adjacency matrix form, we use this data to solve Prim's algorithm. The MATLAB code for Prim's algorithm is shown in appendix B.2. This code is a network algorithm and gives the shortest road network by generating a connected graph. It ensures that each node is connected and prevents occurrence of a cycle. The codes given in appendix B.3 and appendix B.4 are used for locating minimum and maximum indices in adjacency matrix respectively. Appendix B.5 shows the code used for heuristic 1 given in Chapter 4. This code calculates Prim's algorithm both for before and after disaster networks, then starts to open blocked roads which have minimum repairing time, finds corresponding indices of time matrix within distance matrix and generates a new road matrix, calculates Prim's algorithm for each generated road matrix and this process goes on till all roads are opened. The second code that we introduced in appendix B.6 shows code for heuristic 2. This code calculates Prim's algorithm both for before and after disaster networks, then starts to open blocked roads which have maximum arc weight, finds corresponding indices of weight matrix within distance matrix and generates a new road matrix, calculates Prim's algorithm for each generated road matrix and this process goes on till all roads are opened. Arc weight means the sum of the con-

nections of the each nodes made in an arc. Appendix B.7 gives the code generated for heuristic 3 given in Chapter 4. This code calculates Prim's algorithm both for before and after disaster networks, then starts to open blocked roads by checking all blocked roads one by one by adding them into road network and generating a new road matrix, and calculates Prim's algorithm for each generated road matrix. It then adds the road which gives the minimum result and continues till all roads are opened. For heuristic 4 the code given in appendix B.8 and B.9 are utilized. The code calculates Prim's algorithm both for before and after disaster networks, then starts to open blocked roads by checking all blocked roads individually by adding them into road network and generating a new road matrix. It then calculates Prim's algorithm for each generated road matrix, generates another matrix by subtracting current Prim's solution from previous solution, divides the result to repairing time to corresponding damaged road and adds the road which gives the maximum ratio and continues till all roads are opened. Finally, each codes finds the same minimum spanning tree network by different road selection methods.

To be able to compare heuristic scenarios we carry out tests based on blockages of 10%, 25% and 50%. We then replicate five tests by creating randomly generated blockages with 10%, 25% and 50% blockage rates. We assign 1 dozer to each neighborhood and assume that neighborhoods are big enough such that one dozer can make debris removal. Then we calculate an inaccessibility ratio for all heuristic methods given in Chapter 4. Inaccessibility formulation given in objective function Chapter 3 is used to find which heuristic application is the best.

Among our experiments, we needed to run the MATLAB code for 8h, 3h and 30 mins respectively for the 50% ,25% and 10% damage scenarios we performed for both for heuristics 3 and 4 for Şirinevler. The rest of the heuristic solutions took under one minute. The results of four heuristic methods are summarized on Table 5.1. The first number in each cell shows the average CI value while the second number (in parenthesis) gives the average number of blocked arcs restored of five tests, and the last four columns of the table show the instances that each heuristic

reach the best result. Best result means the result with the minimum CI value.

For Güven neighborhood 64, 32 and 12 roads were blocked and for Şirinevler neighborhood 316, 158 and 64 roads were blocked for 10% ,25% and 50% blockages respectively.

Table 5.1 Heuristic Results

Damage Scenarios	CI Values				Number of Instances With Best CI			
Güven	$HEU1$	$HEU2$	$HEU3$	$HEU4$	$HEU1$	$HEU2$	$HEU3$	$HEU4$
10%	1.73(12)	1.976(12)	1.4(12)	1.2(12)	0	0	0	5
25%	19.9(26)	24.28(30)	9.82(20)	19.1(21)	0	1	2	2
50%	117.2(55)	164.6(63)	110.8(54)	117.2(55)	4	0	5	4
Şirinevler	$HEU1$	$HEU2$	$HEU3$	$HEU4$	$HEU1$	$HEU2$	$HEU3$	$HEU4$
10%	68.4(55)	81.8(63)	32.7(48)	68(52)	0	1	3	1
25%	238.01(146)	326.4(182)	213.8(132)	223.8(135)	0	0	4	1
50%	514.4(290)	577.2(311)	467(273)	512, 8(284)	0	0	5	0

For Güven neighborhood, we can see that heuristic 4 gives the minimum CI value for all damage scenarios and has 11 best instances for all damage scenarios. Heuristic 3 gives the second best result for 2 scenarios and has 7 instances. Then heuristic 1 gives the third best result with 4 instances totally but it provides the only best result in 50% blockage scenario. Finally, heuristic 2 has only one instance and gives the worst result among the four heuristics. On the other hand, when we check the number of damaged roads in each heuristic we see that heuristic 3 can reach the minimum spanning tree by restoring a minimum number of arcs and opens 86 roads totally, while heuristic 1, 2 and 4 open 93, 105 and 88 roads respectively.

Regarding Şirinevler neighborhood, we see that heuristic 3 has 12 best instances totally and provides the best result for all three cases. Heuristic 4 gives the second best result and has 2 instances and it gives the best results for all two scenarios. Heuristic 2 reaches best result only once within 10% blockage scenario while heuristic 1 never provides a best result. According to the restored road amount, we see the

same ranking as on Güven neighborhood and heuristic 1, 2, 3 and 4 opens 491, 556, 453, 471 roads respectively.

6. CONCLUDING REMARKS

6.1 Summary of Research

The losses caused by disasters have shown that the traditionally applied disaster management methods may sometimes be insufficient. It has been understood that disaster management should be managed systematically before and after the disaster. Therefore, risk management for disaster management has become crucial more than ever. In the event of a disruptive disaster, it is crucial to limit its forthcoming effects. This can be provided by taking proactive actions before the disaster and performing effective emergency response after disaster occurs. Disaster management systems are sometimes insufficient and in order to reduce the negative effects of disasters on humans it is necessary to contribute to them with effective and quick solutions.

Providing a model which effectively selects the blocked road for debris removal is an important step to contribute the disaster management process. In this thesis, we focus on the response stage of the post disaster period. Our motivation is to obtain an effective method for road selection to remove debris out of it, enable roads for transportation so that it will be possible to reach people in need. Our aim is to generate an efficient method to reduce life and property loss after a disruptive disaster.

First, we introduced a problem named as Dynamic Spanning Tree for the planning of debris removal. A model is then introduced for this problem that dynamically tries to obtain the minimum spanning tree on the damaged road network. The mathematical model of the Dynamic Spanning Tree problem ranks the blocked roads for

debris removal operations, and the objective function minimizes the total inaccessibility of the damaged road network over time and provides a connected road network whenever possible. Our optimization model is an mixed integer programming model which helps to assign dozers for debris removal and obtain traveling routes for emergency vehicles. It has a dynamic structure because the road network becomes more accessible as the roads are cleared from debris. We solved the optimization model on CPLEX Optimization Studio 12.7, but the solver cannot handle real life situations due to the out-of-memory problems. Hence, a small hypothetical network is generated to reach optimal results with the model.

Heuristic methods have been developed to achieve the closest result to optimal solution. The algorithm of each heuristic was coded with MATLAB. We used a real data set from two neighborhoods of İstanbul with different characteristics in terms of the population and the complexity of road network in each district. We carried out experiments to test our mathematical model and heuristic methods in these data sets. The road maps of both neighborhoods are taken from Google Maps. Afterwards, we used ArcGIS 10.5, which is a geographic information system tool utilized for various purposes, was used to obtain road data information such as nodes, arcs, distances and coordinates for showing results on maps.

Subsequently, we generated several damage scenarios and assigned various blockage rates to all road networks and we applied our heuristics in each scenario. Afterwards, we made comparisons of heuristic methods to be able to find the superior one. We compared them based on cumulative inaccessibility level. Finally, we presented the best heuristic method as a result of our analysis. It has been observed that Heuristic 3 and 4 give the closest results to the optimal solution for the hypothetical network and when we apply these heuristic methods to two real data cases, they also give the best results. We observe that heuristic 4 gives best results for the hypothetical network and Güven network while heuristic 3 gives the best results for Şirinevler network. When we check the construction of those three networks we see that both hypothetical and Güven networks have more spare construction of arcs and

nodes while Şirinevler has a more dense connections of arc and nodes. From this perspective, we conjecture that heuristic 3 gives best results for dense networks and heuristic 4 provide best results for spare networks.

The superiority of our method is that we obtain a shortest route on a connected graph rather than obtaining a disjoint road network. This is important because this allows the rescue team to transport one location to another through a connected road cluster. Also our heuristic methods are sufficient enough to achieve results which are close to optimal solution so that they can ensure the decision making to be rapid and reasonably good. Furthermore we provide an integrated decision making tool with ArcGIS and MATLAB as we took data from ArcGIS and than perform heuristic 3 so we can observe results visually. We present blocked map of 50% damage scenario for Sirivenler. ArcGIS results can be seen on Appendices C.1, C.2 and C.3 . Appendix C.1 shows the minimum spanning tree network for Şirinevler when it is undamaged so no roads are blocked. Appendix C.2 shows the map for 50% damaged Şirinevler and red lines indicate the blocked roads and the rest of the roads are not blocked. Appendix C.3 shows solution for 50% damaged Sirinevler for heuristic 3 and red lines indicate the blocked roads are opened to reach minimum spanning tree.

6.2 Future Research Opportunities

In this section, we discuss several important future research opportunities which can be possible extensions for Dynamic Spanning Tree problem.

In Chapter 3 in and Chapter 4, we solve the optimization problem and apply heuristic methods by assigning only one dozer to each neighborhood. We select neighborhoods with reasonable sizes so that one dozer would be enough to unblock the roads in the area. In future, we plan to extend our optimization and heuristic methods for cases with more than one dozer so that our model can be used for areas bigger than small neighborhoods.

We assume that dozers are readily available immediately after disaster and the distances traveled by the dozer from one debris removal location to another are not considered in this study, even though they may occasionally have a direct impact on the debris removal strategies. As a future research, travel times of dozers can also added to mathematical problem.

We also assume that dozers can pass through debris to reach the debris removal operations with higher priority. The traveling time of the dozer passing through debris (without clearing it) is also not included in our problem. Therefore, as another future research option, we also plan to add this time to the definition of the problem.

REFERENCES

1 AFAD (2014). *In: Açıklamalı Afet Yönetimi Terimleri Sözlüğü Ankara: T.C.* Başbakanlık Afet ve Acil Durum Yönetimi Başkanlığı, page 33.

2 Aksu, T. D. and Ozdamar, L. (2014). *A mathematical Model for Post-Disaster Road Restoration: Enabling Accessibility and Evacuation.* Transportation Research Part E: Logistics and Transportation Review, 61, pp. 56 – 67.

3 Alexander, D. (2012). *Models of Social Vulnerability to Disasters.* Open Edition Journals, 4, pp. 22-40.

4 Berktaş, N. , Kara, B. Y. and Karaşan, O. E. (2016). *Solution Methodologies for Debris Removal in Disaster Response.* EURO Journal on Computational Optimization, 4(3 – 4), pp. 403 – 445.

5 Büyükkaracığan, N. (2016). *Legislation Evaluation Of Crisis And Disaster Management In Local Governments In Turkey.* Selcuk University Journal of Social and Technical Researches, 12, pp. 195 - 219.

6 Carter, W. N. (2008). *Disaster Management: A Disaster Manager's Handbook.* Mandaluyong City: Asian Development Bank, pp. 51 - 56.

7 Cho, S. , Gordon, P. and Richardson, H. W. (2000). *Analyzing Transportation Reconstruction Network Strategies: A Full Cost Approach.* Review of Urban and Regional Development, 12(3), pp. 212 – 227.

8 AFAD (2018). *Deprem Nedir?.* Available at: https://www.afad.gov.tr/tr/4379/Deprem-Nedir [Accessed 4 May 2019].

9 Duque, P. M. and Sörensen, K. (2011). *A GRASP Metaheuristic to Improve Accessibility After a Disaster.* OR Spectrum, 33(3), pp. 525–542.

10 Çelik, M. , Ergun, Ö. and Keskinocak, P. (2015). *The Post-Disaster Debris Clearance Problem Under Incomplete Information.* Operations Research, 63(1), pp. 65 – 85.

11 *Earthquake, n. d.*, Available at: https://en.oxforddictionaries.com/definition/earthquake [Accessed 04 May 2019].

12 AFAD (2016). *Çığa Hazırlıklı Olmak İçin Neler Yapılmalı?.* Available at: https://www.afad.gov.tr/tr/4409/Ciga-Hazirlikli-Olmak-Icin-Neler-Yapilmali [Accessed 4 May 2019].

13 Federal Emergency Management Agency (FEMA), 2017. *Public Assistance Debris Management Guide FEMA - 325, Washington, D. C.* . United States Federal Emergency Management Agency, p. 22.

14 Felbermayr, G. and Gröschl, J. (2014). *Naturally Negative: The Growth Effects of Natural disasters*. Journal of Development Economics, 111, pp. 92 - 106.

15 Feng, C. M. and Wang, T. C. (2003). *Highway Emergency Rehabilitation Scheduling in Post-Earthquake 72 Hours*. Journal of the Eastern Asia Society for Transportation Studies, 5, pp. 3276 – 3285.

16 Günneç, D. and Salman, F. S. (2011). *Assessing the Reliability and the Expected Performance of a Network Under Disaster Risk*. OR Spectrum, 33(3), pp. 499 – 523.

17 Hillier, F. and Lieberman, G. (2007). *Introduction to Operations Research. 2nd ed.*. New York: Tata McGraw-Hill, pp. 407 - 408.

18 *Industrial excavator and bulldozer loading, n.d. photograph*, Available at: https://tr.depositphotos.com/56066749/stock-photo-industrial-excavator-and-bulldozer-loading.html [Accessed 29 Apr. 2019].

19 Kruskal, J. B. (1956). *On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem*. Proceedings of the American Mathematical Society, 7(1) , pp. 48-50.

20 Küçükali, U. F. (2018). *Evaluation of Districts in İstanbul at High-Risk in the Event of a Disaster for Planning Purposes*. Planlama, 28(2), pp. 171 – 178.

21 Marin, G. and Modica, M. (2017). Socio-economic exposure to natural disasters, Environmental Impact Assessment Review,64, pp. 57 - 66.

22 Martin, R. K. (1991). *Using Separation Algorithms to Generate Mixed Integer Model Reformulations*. Operations Research Letters, 10, pp. 119 - 128.

23 Matisziw, T. C. , Murray, A. T. and Grubesic, T. H. (2010). *Strategic Network Restoration*. Networks and Spatial Economics, 10(3), pp. 345 – 361.

24 Matisziw, T. C. and Murray, A. T. (2009). *Modeling S-T Path Availability to Support Disaster Vulnerability Assessment of Network Infrastructure*. Computers and Operations Research, 36(1), pp. 16 – 26.

25 Murray, A. T. , Matisziw, T. C. and Grubesic, T. H. (2007). *Critical network Infrastructure Analysis: Interdiction and System Flow.* Journal of Geographical Systems, 9(2), pp. 103 – 117.

26 Onan, K. , Ülengin, F. and Sennaroğlu, B. (2015). *An Evolutionary Multi-Objective Optimization Approach To Disaster Waste Management: A Case Study of İstanbul, Turkey.* Expert Systems with Applications ,42(22), pp. 8850 – 8857.

27 Özdamar, L. , Aksu, D. T. and Ergüneş, B. (2014). *Coordinating Debris Clean-up Operations in Post Disaster Road Networks.* Socio-Economic Planning Sciences, 48(4), pp. 249 – 262.

28 Pramudita, A. and Taniguchi, E. (2014). *Model of Debris Collection Operation After Disasters and Its Application in Urban Area.* International Journal of Urban Sciences, 18(2), pp. 218 – 243.

29 Reinhardt, J. D. , Li, J. , Gosney, J. , Rathore, F. A. , Haig, A. J. , Marx, M. , Delisa, J. A. (2011). *Disability and health-Related Rehabilitation in International Disaster Relief.* Glob Health Action, 4: 1.

30 Prim, R. C. (1957). *Shortest Connection Networks and Some Generalizations.* The Bell System Technical Journal, 36(6), pp. 1389 - 1401 .

31 Sahin, H. , Kara, B. Y. and Karasan, O. E. (2016). *Debris Removal During Disaster Response: A Case for Turkey.* Socio-Economic Planning Sciences, 53, pp. 49 – 59.

32 Taha, H. A. (2007). *Operations Research an Introduction 8nd. ed* . New Jersey: Pearson Pretince Hall, pp. 237 - 239.

33 Usman, R. A. , Olorunfemi, F. B. , Awotayo, G. P. , Tunde, A. M. and Usman, B. A. (2013). *Disaster Risk Management and Social Impact Assessment: Understanding Preparedness, Response and Recovery in Community Projects.* in Silvern, S. and Young S. (eds.), p. 259 - 274. Environmental Change and Sustainability, Available at: https://www.intechopen.com/books/environmental-change-and-sustainability/disaster-risk-management-and-social-impact-assessment-understanding-preparedness-response-and-recove [Accessed 8 April, 2019].

APPENDIX A: CPLEX CODE FOR DST PROBLEM

Figure A.1 CPLEX Code for Dynamic Spanning Tree Problem

DynamicMST.mod

```
/*********************************************
 * OPL 12.7.1.0 Model
 * Author: Tugce Ozkenar
 * Creation Date: Jul 28, 2019 at 3:59:22 PM
 *********************************************/

tuple doublex {
  int fromnode;
  int tonode; }

tuple node_tuple {
  int nodeid;
  float xcoord;
  float ycoord; }

tuple arc_tuple {
   int arcid;
   int fromnode;
   int tonode;
   float arclength;
   string status;
   int DRT; }

{node_tuple} node_info = ...;
{int} Nodes = {id | <id,x,y> in node_info};
int NbNodes = card(Nodes);

{arc_tuple} arc_info = ...;
{doublex} RealArcs = {<f,t>|<id,f,t,l,s,d> in arc_info};
float cost1[RealArcs] = [<f,t> : l | <n,f,t,l,s,d> in arc_info];
{doublex} Arcs = {<f,t>|f,t in Nodes : f<t};
{doublex} NoneArcs = Arcs diff RealArcs;
float cost[Arcs];

execute {
    for (i in NoneArcs) {
            cost[i] = 1000000;
    }
}

execute {
    for (i in RealArcs) {
            cost[i] = cost1[i];
    }
}

{doublex} DamagedArcs = {<f,t>|<id,f,t,l,s,d> in arc_info : s == "1"
int drt[Arcs] = [<f,t> : d | <n,f,t,l,s,d> in arc_info];

int NbTimes = 25;
range Times = 1..NbTimes;

dvar float x[Arcs][Times] in 0.. 1;
dvar float+  y[i in Nodes,j in Nodes ,k in Nodes, t in Times];
dvar boolean beta[DamagedArcs][Times];
dvar boolean alpha[DamagedArcs][Times];
```

Page 1

APPENDIX B: MATLAB CODES FOR HEURISTIC METHODS

Figure B.1 MATLAB Code for Converting Road Data to Adjacency Matrix

```
n = size(list,1);
from=list(:,1);
to=list(:,2);
distance=list(:,3);
damage = list(:,4);
time = list(:,5);
all = [from ; to];
uniq=unique(all);
len=numel(uniq);
 count_from = zeros(len,1);
for i = 1:len
count_from(i) = numel(find(from==i));
end
 count_to = zeros(len,1);
for i = 1:len
count_to(i) = numel(find(to==i));
end
N = zeros(len,len);
betaD = zeros(len,len);
T = zeros(len,len);
W = zeros(len,len);
sum_count = count_from + count_to;
for i = 1: n
    ii=find(uniq==from(i));%485 to 417
    jj=find(uniq==to(i));%485 to 417
    N(ii,jj)= distance(i);
    betaD(ii,jj)  = damage(i);
    T(ii,jj)  = time(i);
    W(ii,jj)  = sum_count(ii)+sum_count(jj);
end
 for i = 1: len
   for j = 1:len
      if N(i,j) > N(j,i)
         N(j,i) = N(i,j);
      end
   end
end
 for i = 1: len
   for j = 1:len
      if T(i,j) > T(j,i)
         T(j,i) = T(i,j);
      end
   end
end
 for i = 1: len
   for j = 1:len
      if betaD(i,j) > betaD(j,i)
       betaD(j,i) = betaD(i,j);
      end
   end
end
 for i = 1: len
  for j = 1:len
    if W(i,j) > W(j,i)
      W(j,i) = W(i,j);
  end
 end
end
```

Figure B.2 MATLAB Code for Prims Algorithm

```
function [cost,M] = primB(A)
 % User supplies adjacency matrix A.  This program uses Prim's algorithm
% to find a minimum spanning tree.  The edges of the minimum spanning
% tree are returned in array mst (of size n-1 by 2), and the total cost
% is returned in variable cost.  The program prints out intermediate
% results and pauses so that user can see what is happening.  To continue
% after a pause, hit any key.
 [n,n] = size(A)       ;                    % The matrix is n by n, where n = # nodes.
A(A==0)=inf;
%A, n, pause;
 %if norm(A-A','fro') ~= 0                % If adjacency matrix is not symmetric,
%    disp(' Error:  Adjacency matrix must be symmetric ') % print error message and
quit.
%    return,
%end
 % Start with node 1 and keep track of which nodes are in tree and which are not.
 intree = [1];  number_in_tree = 1;  number_of_edges = 0;
notintree = [2:n]';  number_notin_tree = n-1;
 in = intree(1:number_in_tree)  ;            % Print which nodes are in tree and which
out = notintree(1:number_notin_tree); % are not.
 % Iterate until all n nodes are in tree.
mst=zeros(n-1,2);
costs=zeros(n-1,1);
while number_in_tree < n
        %   Find the cheapest edge from a node that is in tree to one that is not.
        mincost = Inf;                   % You can actually enter infinity into Matlab.
    for i=1:number_in_tree
        for j=1:number_notin_tree
            ii = intree(i);  jj = notintree(j);
            if A(ii,jj) < mincost
                mincost = A(ii,jj);
                jsave = j; iisave = ii;
                jjsave = jj ; % Save coords of node.
            end
        end
    end
        %    Add this edge and associated node jjsave to tree.  Delete node jsave from
list
    %    of those not in tree.
        number_of_edges = number_of_edges + 1  ;   % Increment number of edges in tree.
    mst(number_of_edges,1) = iisave;             % Add this edge to tree.
    mst(number_of_edges,2) = jjsave;
    costs(number_of_edges,1) = mincost;
    number_in_tree = number_in_tree + 1;       % Increment number of nodes that tree
connects.
    intree = [intree; jjsave]           ;      % Add this node to tree.
    for j=jsave+1:number_notin_tree           % Delete this node from list of those not
in tree.
        notintree(j-1) = notintree(j);
    end
    number_notin_tree = number_notin_tree - 1 ; % Decrement number of nodes not in
tree.
        in = intree(1:number_in_tree)        ;     % Print which nodes are now in tree
and
    out = notintree(1:number_notin_tree) ;% which are not.
    end
%disp(' Edges in minimum spanning tree and their costs: ')
%[mst costs]         ;                       % Print out edges in minimum spanning
tree.
cost = sum(costs);
M=[mst,costs];
```

Figure B.3 MATLAB Code for Finding Minimum Element Indices in Matrix

```
function [ i j ] = minloc( A )

A(A == 0) = inf;
[x y] = min(A);
[a b]=min(x);
j=b;
i=y(b);


end
```

Figure B.4 MATLAB Code for Finding Maximum Element Indices in Matrix

```
function [ i j ] = maxloc( A )

A(A == inf) = 0;
[x y] = max(A);
[a b]=max(x);
j=b;
i=y(b);


end
```

Figure B.5 MATLAB Code for Heuristic 1

```
function [ lists,M,IA,CIA,CIAS,sec,free ] = heu2(T,N,betaD,uniq )
n = size(T,1);
DART = T .* betaD;
ART = T - DART;
DART(DART==0)=inf ;
D=N.*betaD;
A=N-D;
D(D==0)=inf;
num_D = sum(sum(betaD))
lists=zeros(num_D/2,8); % list
 [r,c]=find(A~=0);
free=throwmirror([r,c,uniq(r),uniq(c),N(A~=0)]);
counter = 1;
while counter <= num_D/2
            DARTms=find(DART==min(min(DART)));%mins of dart
            selected=0;
            ifcontrol=0;
            if(size(DARTms,1)==1)
                selected=DARTms(1);
                ifcontrol=1;
            else
                Dm=min(D(DARTms));%min value in D
                Dms=find(D==Dm,1);%all mins in D,,,,
                selected=Dms(1);
                ifcontrol=2;
            end
                i=mod(selected-1,n)+1;
                j=ceil(selected/n);
                A(i,j) = D(i,j);
                A(j,i) = D(j,i);
                D(i,j) = inf;
                D(j,i) = inf;
                ART(i,j) = DART(i,j);
                ART(j,i) = DART(j,i);
                DART(i,j) = inf;
                DART(j,i) = inf;
                lists(counter,1)=i;
                lists(counter,2)=j;
                lists(counter,3)=uniq(i);
                lists(counter,4)=uniq(j);
                lists(counter,5)=N(i,j);
                lists(counter,6)=primB(A);
                lists(counter,7)=T(selected);
                lists(counter,8)=ifcontrol;
                counter = counter + 1
           [num_D]
end
 [cost,M]=primB(A);
M=[M,T(M(:,1)+n.*(M(:,2)-1))];%list
TIMES = lists(:,7)';%list
IA=1-[cost./(lists(:,6)')] ;%IA
CIA=IA.*TIMES ; %IA
IAi=find(IA==0);
TIMESadd=0;
if(size(IAi,1)>0)
    TIMESadd=TIMES(IAi(1));
end
CIAS = sum(CIA)/(sum(TIMES(IA>0))+TIMESadd);
end
```

Figure B.6 MATLAB Code for Heuristic 2

```
function [ lists, M,IA,CIA,CIAS,sec,free] = heu4( W,T,N,betaD,uniq )
n = size(W,1);
DARW = W .* betaD;
ARW = W - DARW;
D=N.*betaD;
A=N-D;
D(D==0)=inf;
num_D = sum(sum(betaD));
lists=zeros(num_D/2,9); %list
counter = 1;
[r,c]=find(A~=0);
free=throwmirror([r,c,uniq(r),uniq(c),N(A~=0)]);
counter = 1;
while counter <= num_D/2
        DARWmx=find(DARW==max(max(DARW)));%max of darw
        selected=0;
        ifcontrol=0;
        if(size(DARWmx,1)==1)
            selected=DARWmx(1);
            ifcontrol=1;
        else
            Dx=min(D(DARWmx));%max value in D
            Dmx=find(D==Dx,1);%all max in D,,,,
            selected=Dmx(1);
            ifcontrol=2;
        end
        i=mod(selected-1,n)+1;
        j=ceil(selected/n);
        A(i,j) = D(i,j);
        A(j,i) = D(j,i);
        D(i,j) = inf;
        D(j,i) = inf;
        ARW(i,j) = DARW(i,j);
        ARW(j,i) = DARW(j,i);
        DARW(i,j) = 0;
        DARW(j,i) = 0;
        lists(counter,1)=i;
        lists(counter,2)=j;
        lists(counter,3)=uniq(i);
        lists(counter,4)=uniq(j);
        lists(counter,5)=N(i,j);
        lists(counter,6)=W(i,j);%list
        lists(counter,7)=primB(A);
        lists(counter,8)=T(selected);%list
        lists(counter,9)=ifcontrol;%list
       counter = counter + 1
       [num_D]
end
 lists=throwmirror(lists);
 TIMES=(lists(:,8)');
[cost,M]=primB(A);
M=[M,W(M(:,1)+n.*(M(:,2)-1))]; %list
IA=1-[cost./(lists(:,7)')] ; % IA
IA'
CIA=IA.*TIMES;   % IA
IAi=find(IA==0);
TIMESadd=0;
if(size(IAi,1)>0)
    TIMESadd=TIMES(IAi(1));
end
CIAS = sum(CIA)/(sum(TIMES(IA>0))+TIMESadd);
end
```

Figure B.7 MATLAB Code for Heuristic 3

```
function [ lists, M,IA,CIA,CIAS,sec,TIMESadd,free] = heu1( N ,T,betaD,uniq)
disp('MST value before the disaster');
SOLglobal = primB(N);
n = size(N,1);
D = N .* betaD;
A = N - D;
D(D==0)=inf;
disp('MST value just after the disaster');
SOLbest = primB(A)
SOL = Inf(n);
SUMT = zeros(n)  % ıa;
num_D = sum(sum(betaD));
counter = num_D
lists=zeros(num_D+1,8);  % list
lists(1,:)=[0,0,0,0,0 SOLbest,0,0];
[r,c]=find(A~=0);
free=throwmirror([r,c,uniq(r),uniq(c),N(A~=0)]);
bi=find(betaD==1);
while counter > 0
       for i=1:counter
         A(bi(i))= N(bi(i));
         SOL(bi(i)) = primB(A);
         A(bi(i))= 0;
    end
    [i, j] = minloc(SOL);
    ifcontrol=0;
    if (SOL(i, j) < SOLbest)
        SOLbest = SOL(i, j);
        ifcontrol=1;
    else
       [i, j] = minloc(D);
       ifcontrol=2;
    end
    A(i,j) = D(i,j);
    D(i,j) = inf;
    bi(bi==(j-1)*n+i)=[];
    row=num_D-counter+2;
    lists(row,1)=i;
    lists(row,2)=j;
    lists(row,3)=uniq(i);
    lists(row,4)=uniq(j);
    lists(row,5)=N(i,j);
    lists(row,6)=SOLbest;
    lists(row,7)=T(i, j);
    lists(row,8)=ifcontrol;
    counter = counter - 1;
    [counter]
end
lists=throwmirror(lists);
MST=lists(:,6)';
TIMES=lists(:,7)';
[cost,M]=primB(A);
IA=1-[SOLbest./MST] ;%ıa
CIA=IA.*TIMES ; %ıa
IAi=find(IA==0);
TIMESadd=0;
if(size(IAi,1)>0)
    TIMESadd=TIMES(IAi(1));
End
CIAS = sum(CIA)/(sum(TIMES(IA>0))+TIMESadd);%ıa
[lists];
end
```

Figure B.8 MATLAB Code for Heuristic 4-1

```
function [lists,IA,CIA,CIAS,sec,p,A,TIMESadd,free] = heu3( N ,T, betaD,uniq
tic
disp('MST value before the disaster');
SOLglobal = primB(N);
UIglobal=inf;
n = size(N,1);
D = N .* betaD;
A = N - D;
D(D==0)=inf;
disp('MST value just after the disaster');
SOLbest = primB(A);
UIbest=0;
SOL = Inf(n);
UI=zeros(n);
num_D = sum(sum(betaD));
counter = num_D;
lists=zeros(num_D/2+1,9); % list
lists(1,:)=[0,0,0,0,0,UIbest,SOLbest,0,0];
p=[];
bi=find(betaD==1);
[r,c]=find(A~=0);
free=throwmirror([r,c,uniq(r),uniq(c),N(A~=0)]);
i=0;
j=0;
 while counter>0
        for c=1:counter
        i=mod( bi(c)-1,n)+1;
        j=ceil( bi(c)/n);
        A(i,j)= N(i,j);
        A(j,i)= N(j,i);
        SOL(i,j) = primB(A);
        SOL(j,i) = primB(A);
        SOLold=0;
        if(SOLbest~=inf)
            SOLold=SOLbest;
        end
        UI(i,j)=(SOLold-SOL(i,j))/T(i,j);
        UI(j,i)=(SOLold-SOL(j,i))/T(j,i);
        A(i,j)= 0;
        A(j,i)= 0;
    end
    ifcontrol=0;
    i=0;
    j=0;
    selected=0;
        if(size(find(UI>0),1)>0)
        UI(UI==inf)=0;
        UImx=find(UI==max(max(UI)));
        SOLUImx=SOL(UImx);
        SOLUImxmn=min(SOLUImx);
            if(SOLUImxmn<=SOLbest)
            SOLUImxmni=find(SOLUImx==SOLUImxmn);
            SOLUImxmni=SOLUImxmni(1);
            selected=UImx(SOLUImxmni);
            ifcontrol=0;
            i=mod(selected-1,n)+1;
            j=ceil(selected/n);
```

Figure B.9 MATLAB Code for Heuristic 4-2

```
        else
            [ii, jj] = minloc(SOL);
            if(SOL(ii, jj) <= SOLbest)
                i=ii;
                j=jj;
                ifcontrol=1;
            else
            [ii, jj] = minloc(D);
            i=ii;
            j=jj;
            ifcontrol=2;
            end
        end
    else
        [ii, jj] = minloc(D);
            i=ii;
            j=jj;
            UI(i,j)=0;
            ifcontrol=3;
    end
    UIbest=UI(i,j);
    SOLbest=SOL(i,j);
    UI(i,j)=0;
    UI(j,i)=0;
    SOL(i,j)=inf;
    SOL(j,i)=inf;
    A(i,j) = D(i,j);
    A(j,i) = D(j,i);
    D(i,j) = inf;
    D(j,i) = inf;
    bi(bi==(j-1)*n+i)=[];
    bi(bi==(i-1)*n+j)=[];
    row=((num_D-counter)/2)+2;
    lists(row,1)=i;%list
    lists(row,2)=j;%list
    lists(row,3)=uniq(i);%list
    lists(row,4)=uniq(j);%list
    lists(row,5)=N(i,j);%list
    lists(row,6)=UIbest;%list
    lists(row,7)=SOLbest;%list
    lists(row,8)=T(i,j);%list
    lists(row,9)=ifcontrol;%list
    counter=counter-2
    [counter]
end
MST=lists(:,7)'
TIMES = lists(:,8)';%list
IA=1-[SOLbest./MST];%IA
CIA=IA.*TIMES; %IA
IAi=find(IA==0);
TIMESadd=0;
if(size(IAi,1)>0)
    TIMESadd=TIMES(IAi(1));
end
CIAS = sum(CIA)/(sum(TIMES(IA>0))+TIMESadd);
sec=toc;
end
```

APPENDIX C: ARCGIS SOLUTIONS

Figure C.1 MST Map for Şirinevler Neighborhood

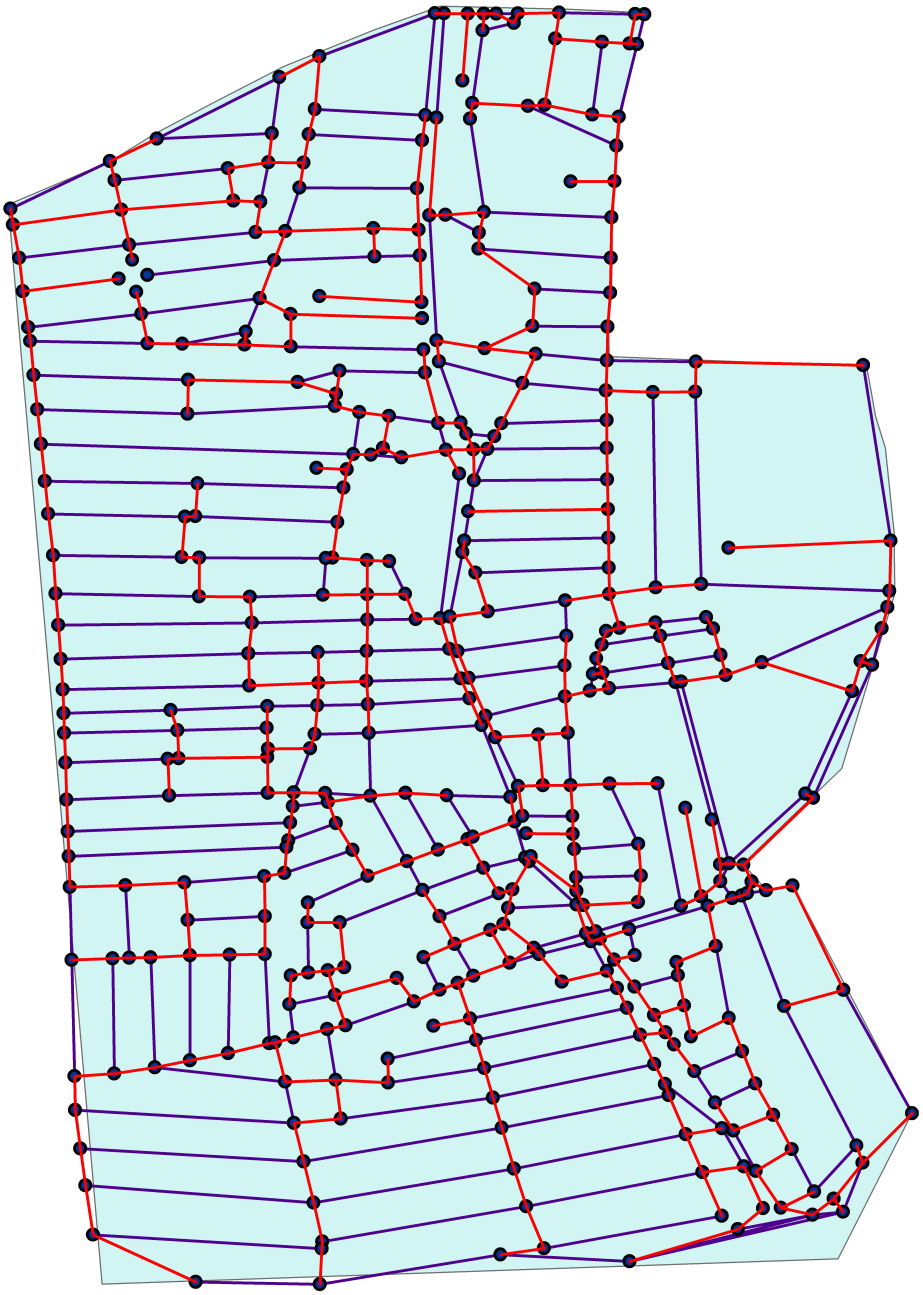


Figure C.2 50% Blocked Map for Şirinevler Neighborhood

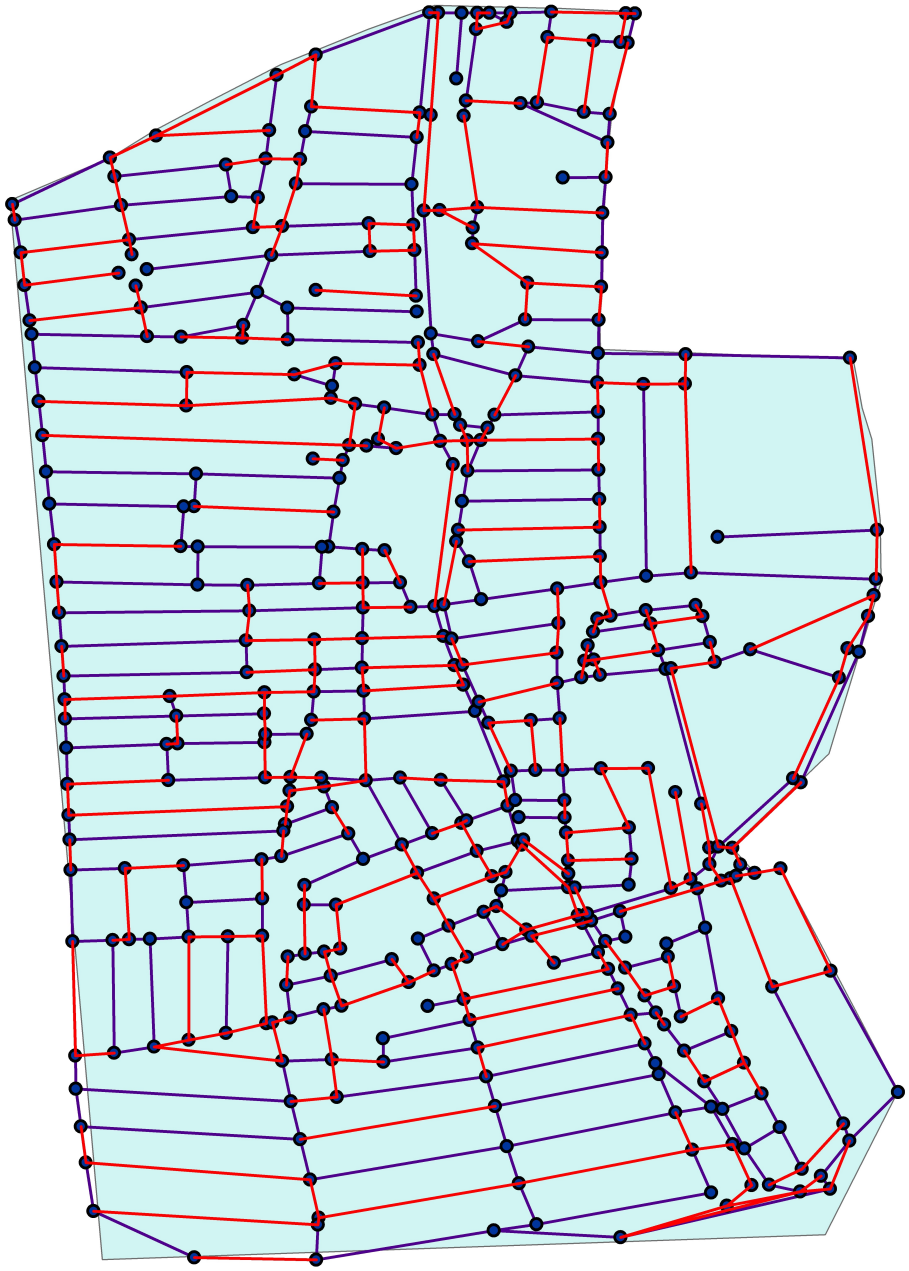
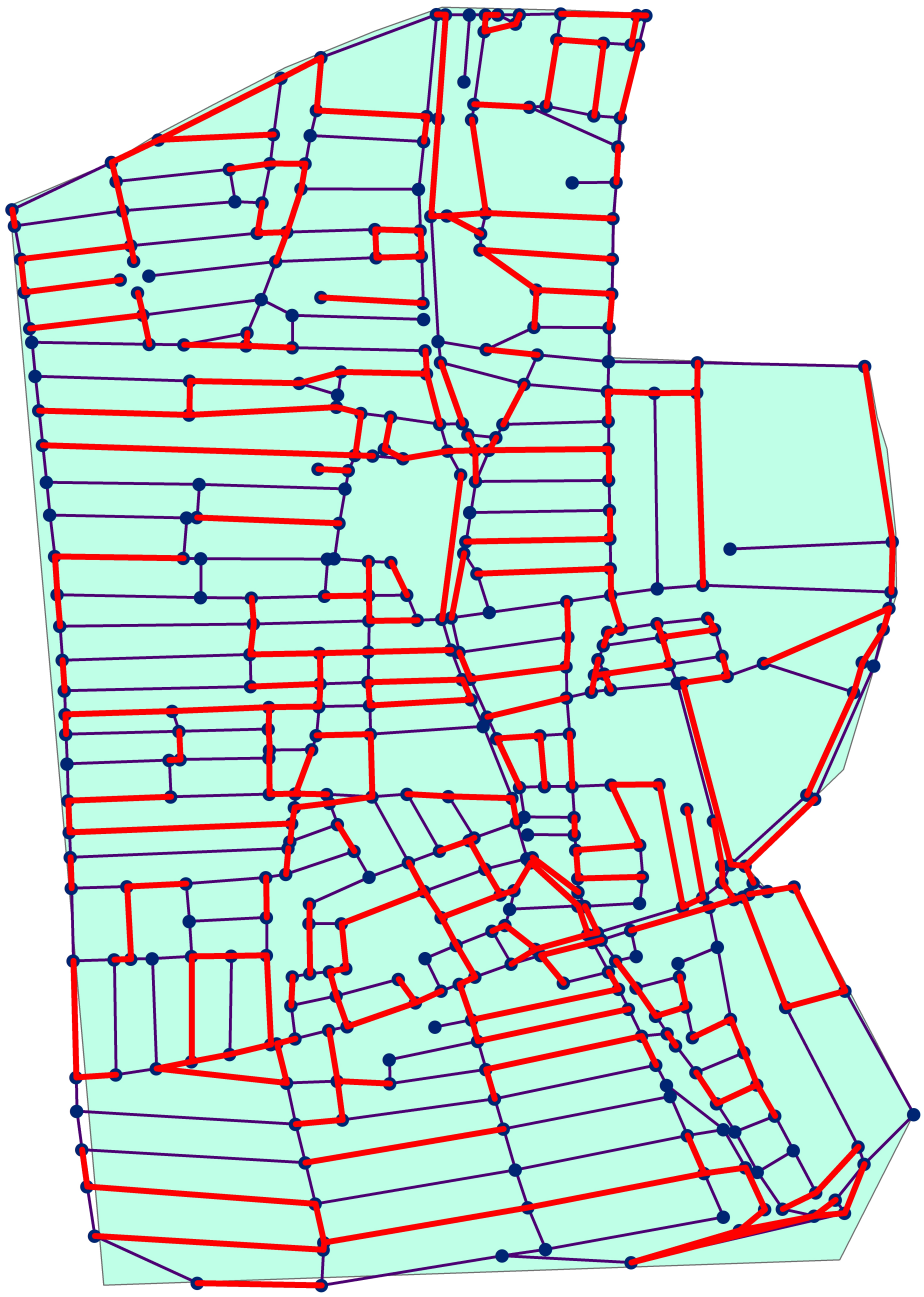


Figure C.3 Heuristic 3 Results for 50% Blocked Şirinevler Neighborhood



CURRICULUM VITAE

Personal Information

Name Surname : Tuğçe Özkenar
Place and Date of Birth : Istanbul , 05.09.1990

Education

Undergraduate Education : Istanbul Aydın University, Textile Engineering
Graduate Education : Kadir Has University , Industrial Engineering Master
Foreign Language Skills :English, Spanish

Work Experience

Name of Employer and Dates of Employment:
Cherryfield Trading Ltd. Sti. , Merchandiser-01.08.2017-
Eresya Giyim San.Ve Tic. Ltd. Sti., Merchandiser-19.02.2015-01.08.2017

Contact:

Telephone : +90(537 618 69 37)
E-mail Address : tugceozkenar@gmail.com