DEBRIS REMOVAL DURING DISASTER RESPONSE PHASE: A CASE FOR TURKEY

A THESIS
SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL
ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE OF
BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

by
Halenur Şahin
August 2013

I certify that I have read this thesis and that in my opinion it is full adequate, in scope and in quality, as a dissertation for the degree of Master of Science.

Assoc. Prof. Bahar Yetiş Kara (Advisor)

I certify that I have read this thesis and that in my opinion it is full adequate, in scope and in quality, as a dissertation for the degree of Master of Science.

Assoc. Prof. Oya Ekin Karaşan (Co-Advisor)

I certify that I have read this thesis and that in my opinion it is full adequate, in scope and in quality, as a dissertation for the degree of Master of Science.

Assoc. Prof. Osman Oğuz

I certify that I have read this thesis and that in my opinion it is full adequate, in scope and in quality, as a dissertation for the degree of Master of Science.

Asst. Prof. M.Alp Ertem

Approved for the Graduate School of Engineering and Science

Prof. Dr. Levent Onural
Director of the Graduate School of Engineering and Science

ABSTRACT

DEBRIS REMOVAL DURING DISASTER RESPONSE PHASE: A CASE FOR TURKEY

Halenur Şahin
M.S. in Industrial Engineering
Supervisor: Assoc.Prof. Bahar Y.Kara
Co-Supervisor: Assoc.Prof.Oya E.Karaşan
August 2013

In this study, a methodology to provide emergency relief supplies to the disaster affected regions is developed. As a result of destructive effects of disasters, debris, which is the ruin and wreckage of the structures, occurs. Proper removal of debris has significant importance since it blocks the roads and prohibits emergency aid teams to access the disaster affected regions. Wrong disaster management, lack of efficiency and delays in debris removal cause disruptions in providing sheltering, nutrition, healthcare and communication services to the disaster victims, and more importantly they result in loss of lives. Due to the importance of a systematic and efficient way of debris removal from the point of improving disaster victims' life quality and its contributions to transportation of emergency relief materials to the disaster affected regions, the focus of this study is providing emergency relief supplies to the disaster affected regions as soon as possible, by considering unblocking operations of roads through removing the accumulated debris.

To come up with a scientific solution methodology to the problem, mathematical models that select the paths in order to transport emergency aid materials in the presence of debris to the pre-determined disaster affected regions are developed. The performances of the models are tested on two distinct data sets from İstanbul. Since it is crucial to act quickly in an emergency case, a constructive and an improvement heuristic are also proposed.

Keywords*:* Disaster management, debris removal, emergency relief transportation.

ÖZET

AFET MÜDAHALE SAFHASINDA ACİL YARDIM MALZEMELERİNİN ULAŞIMI İÇİN ENKAZ KALDIRMA PROBLEMİ: TÜRKİYE UYGULAMASI

Halenur Şahin
Endüstri Mühendisliği Yüksek Lisans
Tez Yöneticisi: Doç. Dr. Bahar Yetiş Kara
Yardımcı Tez Yöneticisi: Doç.Dr.Oya Ekin Karaşan
Ağustos 2013

Bu çalışmada, afetten etkilenen bölgelere acil yardım malzemelerinin ulaşımını sağlayacak bir sistem geliştirilmiştir. Afetlerin yıkıcı etkileri enkaz oluşumuna sebep olmakta ve enkazın doğru biçim ve zamanda kaldırılmaması, afetten etkilenen bölgelere yardım ekibi ve ilk yardım malzemelerinin ulaştırılmasında aksaklıklara yol açmaktadır. Afet yönetiminde karşılaşılan eksikler ve enkazın kaldırılmasındaki aksamalar afetzedelere barınma, beslenme, sağlık ve iletişim hizmetlerinin ulaşmasını zorlaştırmakta ve en önemlisi, can kayıplarına sebebiyet vermektedir. Enkazın sistemli ve verimli bir biçimde kaldırılmasının yardım malzemelerinin afet bölgesine ulaştırılmasında ve afetzedelerin yaşam kalitesinde sağlayacağı olumlu etkiler göz önünde bulundurularak yapılan bu çalışmada, afet bölgelerine mümkün olan en kısa sürede ulaşılması ve bu doğrultuda, kapanan yollardaki enkazın geçişe imkân verecek şekilde kaldırılması öngörülmüştür. Problemin çözümü için geliştirilen matematiksel modeller, yardım malzemelerinin önceden belirlenmiş afet bölgelerine ulaştırılması sürecinde izleyeceği rotaları seçmekte ve bunu yaparken bölgedeki enkaz dolayısıyla kapanmış yolları göz önünde bulundurmaktadır. Modellerin performansları İstanbul iline ait iki farklı veri grubu kullanılarak test edilmiştir. Problem, yapısı itibariyle acil durumlarda çabuk karar vermeyi gerektirdiğinden büyük veri grupları için çok kısa sürelerde çözüm önerebilecek sezgisel yöntemler geliştirilmiştir.

Anahtar kelimeler: Afet yönetimi, enkaz kaldırma, acil yardım ulaştırma.

ACKNOWLEDGEMENT

I would like to express my gratitude to Assoc. Prof. Bahar Yetiş Kara and Assoc. Prof. Oya Ekin Karaşan for their guidance and support during my graduate study. Without their supervision, this thesis would not be possible.

I am grateful to Assoc. Prof. Osman Oğuz and Asst. Prof. Alp Ertem for accepting to read this thesis and for their valuable comments and helpful suggestions.

I am most thankful to my mom Dilek, dad Mahmut and brother Alperen for their unconditional love and support. Also, I would like to thank my uncle Alpaslan Fığlalı and aunt Nilgün Fığlalı for their impression on my choice of being an industrial engineer and an academician.

I would like to express my deepest gratitude to Pınar Okutgen for all valuable moments while sharing a home, a childhood, and a youth together. Also, I want to thank my dear sisters Ezgi İrem Bektaş, Emine Topaloğlu and Özge Özgenç for their love, encouragement and support in all phases of my life. Without the laughters we shared that brighten all the desperate times, nothing will be the same.

I am grateful to İrfan Mahmutoğulları for his contribution to every single line of this thesis, and I am thankful to him because he never gives up to being my pillar of strength.

I am thankful my precious friend Nur Timurlenk who always makes feel her support and sincerity. Also, I would like to thank to my dear friends Haşim Özlü, Başak Yazar, Meltem Peker, Okan Dükkancı, Bengisu Sert and Gizem Özbaygın who are more than officemates. We shared so many memorable moments together. For his definite support to this thesis and friendship I am also grateful to Fırat Kılcı, and for their kindness and moral support, I also would thank to Hatice Çalık, Feyza Güliz Şahinyazan and Görkem Özdemir.

I would like to acknowledge financial support of The Scientific and Technological Research Council of Turkey (TUBİTAK) for the Graduate Study Scholarship Program.

TABLE OF CONTENTS

Chapter 1: Introduction 1

Chapter 2: Problem Definition 8

Chapter 3: Related Literature 12

3.1: Arc Routing Problems 13

3.2: Node Routing Problems 34

3.3: Debris Removal Problem in the Response Phase and Its Relation to the Literature 37

Chapter 4: Model Development 40

4.1: MOD-1: Minimize Visiting Time 41

4.2: MOD-2: Minimize Distance Travelled 44

4.3: MOD-3: Minimize Total Effort 45

Chapter 5: Heuristic Solution Methodology 50

5.1: The Constructive Heuristic 51

5.2: The Improvement Heuristic with 2-opt 53

Chapter 6: Data and Computational Results 60

6.1: Data 60

6.2: Computational Analysis 64

Chapter 7: Conclusion & Future Research Directions 84

BIBLIOGRAPHY 88

APPENDIX 94

LIST OF FIGURES

Figure 1-1: Disaster timeline relating to debris operations..2
Figure 1-2: Seismicity map of Turkey..6
Figure 2-1: An example vehicle tour in the proposed system...10
Figure 3-1:Relation of the problems in the literature with proposed debris removal problem in the response phase..37
Figure 5-1: Flow chart of the constructive heuristic...53
Figure 5-2:Example of the 2-opt heuristic..54
Figure 5-3:Application of the improvement heuristic with 2-opt to a feasible instance of the problem 55
Figure 5-4: Output of the constructive heuristic 56
Figure 5-5: Output of the improvement heuristic with 2-opt...56
Figure 5-6: Flow chart of the improvement heuristic with 2-opt 58
Figure 5-7: Application procedure of the heuristics 59
Figure 6-1: The location of supplier and critical nodes in Kartal 62
Figure 6-2: The location of supplier and critical nodes in Bakırköy 62
Figure 6-3: Travel path of RESCUE for instances K-1…K-5 72
Figure 6-4: Travel path of RESCUE for instances K-16…K-20 73

LIST OF TABLES

Table 1-1: Debris quantities of previous disasters 3
Table 1-2: Top 10 most important earthquake disasters for the period 1900 to 2013 5
Table 3-1:Chinese postman problem (CPP) variations 16
Table 3-2: Summary of Chinese postman problem (CPP) variations 17
Table 3-3 Rural postman problem (RPP) variations 21
Table 3-4: Summary of rural postman problem (RPP) variations 22
Table 3-5: Capacitated arc routing problem (CARP) variations 29
Table 3-6: Summary of capacitated arc routing problem (CARP) variations 32
Table 6-1: Features of the data sets 61
Table 6-2:SOE and corresponding BAR values 63
Table 6-3:Kartal instances and the corresponding SOE, BAR and #of blocked arc settings 65
Table 6-4: Bakırköy instances and the corresponding SOE, BAR and #of blocked arc settings 65
Table 6-5: Model performances of Kartal instances with greater debris removal effort (Wij) 68
Table 6-6: Model performances of Kartal instances with smaller debris removal effort (Wij') 69
Table 6-7: Model performances of Bakırköy instances with greater debris removal effort (Wij) 75
Table 6-8: Model performances of Bakırköy instances with smaller debris removal effort (Wij') 76
Table 6-9: Heuristic performance summary of Kartal instances 77
Table 6-10: Heuristic performance summary of Kartal instances for each SOE class ... 79
Table 6-11: Heuristic performance summary of Bakırköy instances 81

Table 6-12: Heuristic performance summary of Bakırköy instances for each SOE class......82

Chapter 1

Introduction

Disaster is an event that causes physical damage, destruction, loss of lives or significant alteration in the natural environment. It may be the result of nature or human activities. Earthquake, flood, accidents, avalanche, landslide, fire and explosions are some of the disasters that may cause great losses. Type of the disaster, sphere of influence and severity of it are the factors that affect its impacts.

To minimize the negative effects of disasters and loss of lives, disaster management operations are of vital importance. Disaster management is comprised of the studies that both focus on preventive strategies for pre-disaster and damage reduction operations for post-disaster periods. The disaster management cycle consists of four phases: preparation, response, recovery and reconstruction.

The preparation phase covers the precautions that are taken beforehand in order to minimize negative outcomes of the disaster whereas the response phase starts immediately after the disaster. The response phase involves transporting all kinds of emergency services to the maximum possible number of disaster victims as soon as possible. During the recovery phase the main focus is to recover the disaster affected region in terms of communication, transportation and infrastructure; and finally, in the reconstruction phase, the main objective is to fully rehabilitate the disaster affected region and normalize disaster victims' daily lives.

Even though the severity of disaster and geographical or climatic specialities of the disaster affected region are the main factors that affect the number of disaster-victims; wrong disaster management, lack of efficiency and delays about debris removal also cause negative effects on people and more importantly they result in loss of lives.

As a result of destructive effects of disasters, debris, which is the ruin and wreckage of the structures, occurs. Proper removal of debris has significant importance since it blocks the roads and prohibits emergency aid teams to access the disaster affected region. Debris relevant operations in the disaster timeline are illustrated in the Figure 1-1.

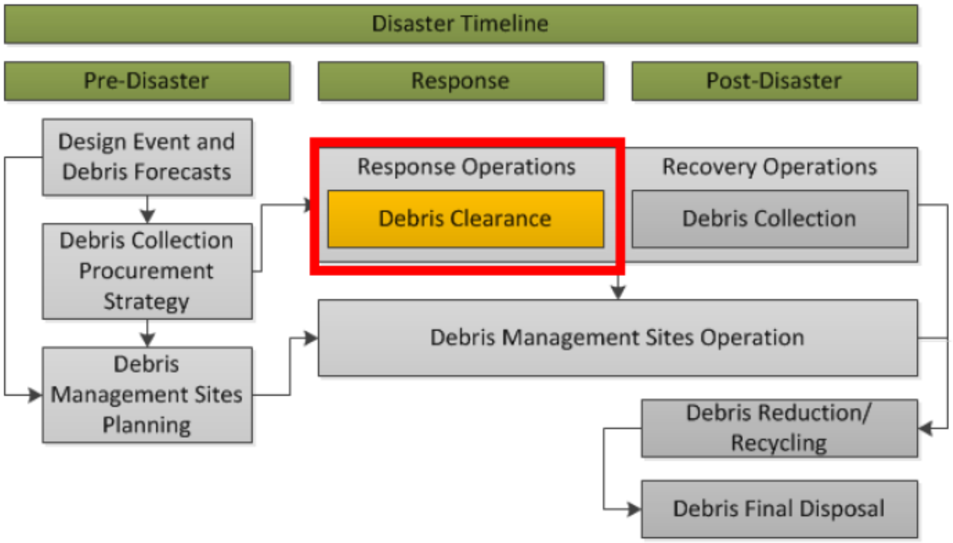


Figure 1-1: Disaster timeline relating to debris operations [1]

As it can be seen from the Figure 1-1, in the pre-disaster phase the main focus is on predicting the disaster characteristics and forecasting the resulting debris. According to these estimations, a proper way of debris collection strategy is determined, and the relevant debris management sites are planned. When it comes to disaster response phase, the pre-determined debris collection procurement strategy is applied to clear the debris. However, the complete removal of debris is postponed to the post-disaster phase where it is indicated as “debris collection” in the figure. Since both clearance and collection of debris require proper sites, debris management sites operations proceed along both the response and post-disaster phase. As depicted in the figure, debris reduction/recycling operations are done right after it is collected. Debris quantity depends on the nature and severity of the disaster. The following table shows worldwide disasters that resulted in intensive amount of debris over the last two decades.

Table 1-1: Debris quantities of previous disasters

Year	Event	Debris Amount
2005	Hurricane Katrina, USA	76 million m^3 [2]
2004	Tsunami, Indian Ocean	10 million m3 (Only Indonesia) [3]
2004	Hurricane Charley, USA	2 million m^3 [4]
1999	Marmara Earthquake, Turkey	13 million tons [5]
1995	Kobe Earthquake, Japan	15 million m^3 [6]
2011	Eastern Japan Great Earthquake	250 million tons [7]

As it can be seen from the table, disaster type and disaster region affect the amount of debris composed. Since the disaster debris has huge volume, it is important to apply debris reduction operations, such as grinding, in order to reduce the volume of debris into a manageable size. Also, not only the quantity, but also the type of debris may differ. Debris types can be classified as construction debris, hazardous domestic sewage, herbal waste and private property, where recycling of some components is possible and has many positive effects from the point of environment. When reducing and recycling operations of debris are over, the debris disposal is finalized. During the debris removal, by considering the damages that result from the characteristics of debris type, choosing the proper debris removal strategy has significant importance.

Among the disasters, in this study we focus on earthquake in consequence of its substantial financial and emotional damages. It is defined as the sudden shaking of Earth which results from the rapid vibrations that occur from the release of energy of the earth crust. It should be known that, this sudden natural event and its consequences show the incompetence of the human being against the nature.

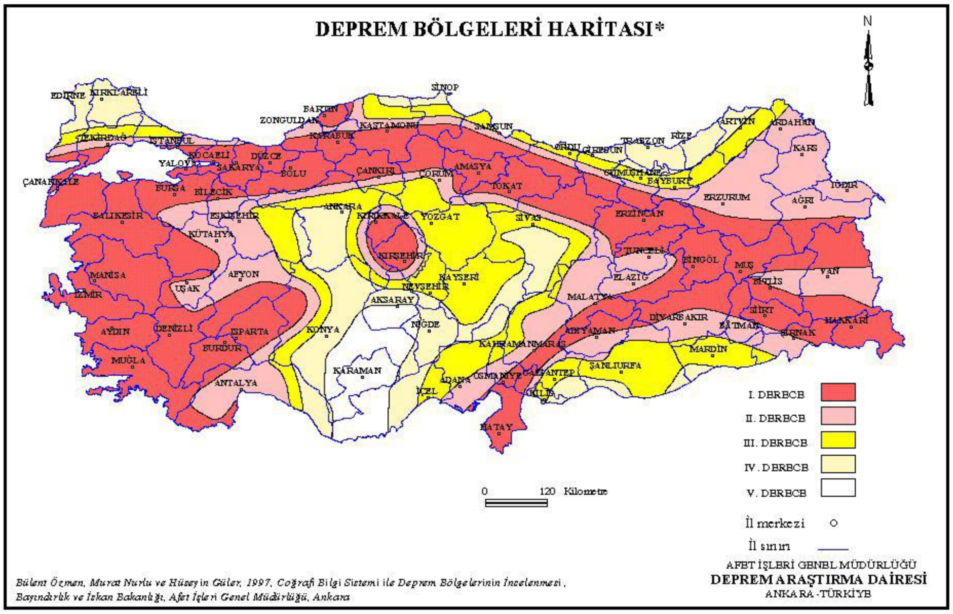
The following table shows the 10 most important earthquake disasters for the period of 1900 to 2013.

Table 1- 2: Top 10 most important Earthquake disasters for the period 1900 to 2013 [8].

Country	Date	#of Total Affected
China P Republic	12/05/2008	45,976,596
India	21/08/1988	20,003,766
India	26/01/2001	6,321,812
Pakistan	8/10/2005	5,128,309
China P Republic	3/02/1996	5,077,795
Guatemala	4/02/1976	4,993,000
Haiti	12/01/2010	3,700,000
Peru	31/05/1970	3,216,240
Indonesia	27/05/2006	3,177,923
China P Republic	1/11/1999	3,020,004
TOTAL		100,615,445

Table 1-2 shows that, more than a hundred million people were affected by earthquakes all over the world in the last decade. When Turkey is examined from the point of exposure from earthquakes, studies show that, statistically, a detrimental earthquake occurs every 8 months in Turkey [9].

According to the following seismicity map, 96% of the ground of Turkey is under different levels of earthquake risk and 98% of the population lives on these grounds. Also it is worth to note that, %66 of these regions have first and second level of earthquake risk [10].



Bülent Özmen, Murat Nurlu ve Hüseyin Güler, 1997, Coğrafi Bilgi Sistemi ile Deprem Bölgelerinin İncelenmesi, Bayındırlık ve İskan Bakanlığı, Afet İşleri Genel Müdürlüğü, Ankara

Figure 1- 2: Seismicity map of Turkey [11]

In this study we focus on the debris removal in the response phase of the earthquake. Complete debris removal may be postponed to the recovery phase whereas removing debris on the critical routes which are on the way of critical disaster affected regions have to be done in the response phase. Therefore, debris removal has significant importance to provide access to disaster victims.

Providing sheltering, nutrition, healthcare and communication services are of vital importance in the phase of response. In this study, we refer to all these services as emergency aid supplies and we intend to provide assistance to the critical disaster affected regions as soon as possible.

Turkey is exposed to many disasters over centuries, and unfortunately, not only the disasters but also the misapplications cause many losses of lives. In this context, the outstanding preventive strategy against this tragedy is to carry debris management studies with scientific methods by taking advantage of experiences.

In the next chapter, the importance of a systematic and efficient way of debris removal from the point of improving disaster victims' life quality and its contributions to transportation of emergency relief materials to the disaster affected regions is discussed. The chapter is concluded by defining the problem which is dealt with in this study.

In Chapter 3, the related literature to the problem is addressed. Namely, Arc Routing and Node Routing literature are reviewed and at the end of the chapter the defined problem is interrelated with the above mentioned literatures.

In Chapter 4, mixed integer linear programming mathematical models are proposed. Models select the paths in order to transport emergency aid materials to the pre-determined disaster affected regions by considering the roads on the network which are blocked by debris. The critical disaster affected regions, which get emergency aid materials, are determined by considering the existence of a school, hospital, shelter area etc. on a region. There exist three different models. The first two models have a periodic structure. However, since the results that are obtained from preliminary analysis are not satisfactory in terms of CPU consumption, a new model is developed.

In Chapter 5, a heuristic solution methodology to the problem is discussed. In this context, a constructive and an improvement heuristic are proposed.

In Chapter 6, the data sets which are used in the computational study are presented. There exist two different data sets with different sizes. Experimental results of the models and heuristics are presented and the performance of the models and heuristics are discussed for these data sets. Finally, in Chapter 7, the thesis is concluded and the future research directions are addressed.

Chapter 2

Problem Definition

Turkey is an earthquake-prone country, where there are many small, medium and large scale earthquakes in the history. Loss of lives, physiological problems, loss of property, damages in the buildings and roads are the main results of earthquakes.

The primary objective of the disaster management is to minimize the resulting negative effects and loss of lives. In this context, existence of a systematical debris removal in the response phase has vital importance.

Debris removal operations are under the responsibility of Republic of Turkey Prime Ministry Disaster and Emergency Management Presidency (T.C Afet ve Acil Durum Yönetim Başkanlığı (AFAD)), where Ministry of Environment and Urban Planning is the main solution partner.

As a result of the interviews with experts form Department of Recovery [12] and Department of Response [13], we learned that debris relevant studies are at the organizational level where operational services are conducted by Civil Defence Search and Rescue Unit Directory and Provincial AFAD Directories. Additionally, it is also learned that activity definitions are incomplete.

However, a systematic and efficient way of debris removal may positively affect the disaster victims’ life quality after an earthquake. Specifically, in the response phase, the goal is to transport emergency relief materials to the disaster affected regions as soon as possible.

In the phase of response, by considering the importance of rapidness and effectiveness, it is more efficient to determine critical districts where it is indispensable to access. Within this context, among all disaster affected districts, a subset of them is selected. Districts that contain schools, hospitals, potential shelter areas etc. are the ones, where it is critical to provide emergency aid as quickly as possible. In order to provide disaster aid to these critical districts, it is necessary to travel on a path which may include blocked roads as well. In such a case, it is required to unblock these roads by debris removing operations.

In this context, we define “*Debris Removal Problem in the Response Phase*” as, visiting pre-specified critical disaster affected districts as quickly as possible by traversing along a path which may include blocked arcs as well. In order to provide access, removing debris on such arcs is required. By means of this system, it is intended to utilize the use of resources, provide quick and effective access of emergency supplies. Achieving these operations in a timely manner helps to defuse the post disaster environment.

In accordance with this purpose, disaster affected region is assumed to be aggregated into districts. Then, the critical districts and the district which serves as a supplier to the critical ones are determined. Also, there exist some other districts which are neither critical nor supplier. The vehicle, where we call it as RESCUE (**Re**lief **S**upply **C**arrier **U**nder **E**mergency), that carries emergency aid materials, departs from supplier and travels to transport relief materials to the critical districts as soon as possible, by removing debris on the blocked arcs, if necessary. In other words, the proposed system decides the critical path, which is the travelling route of RESCUE that is used to transfer relief materials to the critical districts, and also the system decides the arcs which require debris removal in order to resolve blockage and provide access. It is worth to note that, all blocked arcs on the critical path have to be unblocked, and after an arc is opened it remains open forever. In common with traversal of a road, debris removal for the blocked roads also requires effort. This effort is defined in terms of time in our model. It is worth to note that, by means of problem characteristics, it is always possible to use an arc more than once in the critical path. Thus, model determines the critical path by taking advantage of the re-travel on an arc where the debris on it is removed earlier. By this means, once the debris removal effort is spent for this arc, it is never spent again. For each time, only the travel effort is spent.

An example of the vehicle tour can be seen in the Figure 2-1. The triangle represents the supply district. RESCUE departs from supplier and follows a path. Dashed lines symbolize the blocked roads on this path and white circles represent the intermediate nodes whereas the others are the critical districts, such as hospitals and schools.

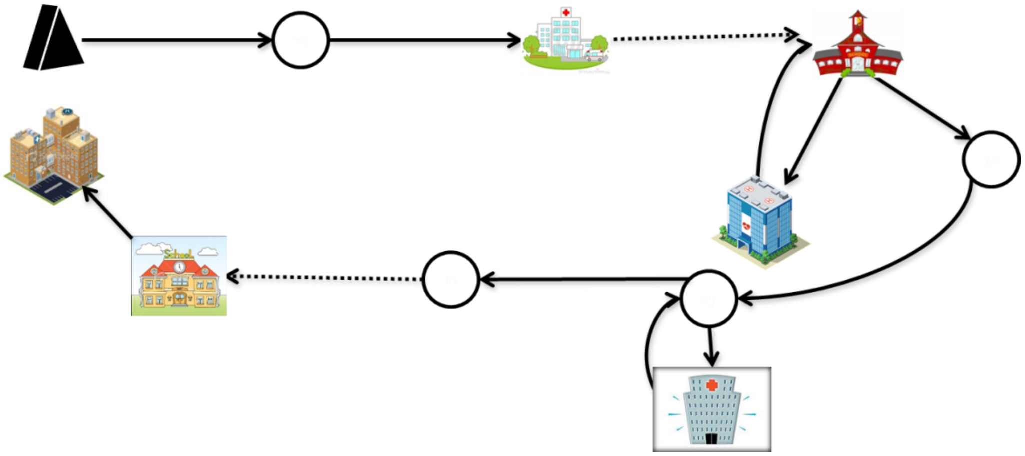


Figure 2-1: An example vehicle tour in the proposed system

Since there are required nodes that need to be visited, and since an arc routing aspect is present, our problem can be defined as a variant of general routing problem (GRP), which will be detailed in the following section.

However, different than GRP, our problem implies that the only reason to traverse an arc is to reach a required node.

To the best of the author’s knowledge, this variant of general routing problem has not been defined in the operations research literature.

Chapter 3

Related Literature

General routing problem (GRP) is a routing problem that aims to find a minimum cost vehicle route which starts and ends at the same node and visits the required nodes at least once by passing through the required edges at least one time. “Required nodes” is a subset of all nodes and “required edges” is a subset of all edges [14]. GRP includes both node and arc routing aspects, thus node routing and arc routing problems arise as special cases.

In order to make a comprehensive survey, we investigate both the arc and node routing literature. In this context, arc routing problems (ARP) and a node routing problem: vehicle routing problem (VRP), are investigated. Since one of the key aspects of our problem is debris removal on arcs in order to unblock them, arc routing literature is examined in a more detailed way.

3.1. Arc Routing Problems

The origin of the ARP is the famous Königsberg bridge problem which is solved by Euler. It aims to find a minimum cost route, which is a closed walk that traverses along each of the bridges in the city of Königsberg. In ARP the aim is to find a minimum cost vehicle tour that traverses through a specified arc subset, which begins and ends at the same node. Chinese postman problem (CPP), rural postman problem (RPP) and capacitated arc routing problem (CARP) are primary arc routing problems. The difference between the GRP and ARP is that, GRP also considers the node routing aspect by visiting some nodes of the graph. When the required nodes set is empty and the purpose is to visit all edges, the GRP reduces to CPP; where if there is a subset of edges that need to be visited with an empty required node set , then GRP reduces to the RPP [14],[15] .

3.1.1. Chinese Postman Problem

CPP is first defined by Kwan-Mei Ko in 1962[16] as to find a minimum cost tour that traverses all the arcs of the graph at least once. The problem is defined on a connected graph where the main elements of the graph are nodes, edges and the cost (or distance) matrix, which is defined for the edges [17]. Waste collection, street sweeping, and snow plowing operations are in the application area of CPP where it is required to pass through all arcs in the graph.

3.1.1.1. Undirected, directed and mixed Chinese Postman Problem

In the analysis, Eiselt et al. [18] summarizes many variations of Chinese postman problem. In their survey, they give details about proposed mathematical model for the undirected case of the CPP. Since the problem is polynomially solvable, a matching based integer linear programming (ILP) algorithm to solve the problem to optimality is investigated.

When the graph becomes directed, another polynomially solvable case arises. In their survey, the mathematical model and the suggested flow algorithm for directed CPP is analysed.

The mixed CPP where the graph both contains directed and undirected arcs is NP-Hard [19]. In this survey of Eiselt et al., mathematical models and some heuristics to solve mixed CPP are suggested, and branch and cut is proposed for the small size instances as an exact solution methodology.

3.1.1.2. Windy Postman Problem

The windy postman problem is another variation of CPP where the graph is undirected but the cost of traversing an edge is different for each travel direction. If the graph is Eulerian, then the problem is polynomially solvable [20], else it is NP-Hard [21], [22]. Some heuristic methodologies, a mathematical model and a cutting plane algorithm for this problem are investigated in the survey of Eiselt et al. [18].

3.1.1.3. Hierarchical Postman Problem

If a precedence relation is defined on the arcs of the graph and the service to these arcs is done according to this relation, the problem is referred as hierarchical postman problem, which is NP-Hard. However, if each subgraph is connected and order relation is complete, the problem can be solved in polynomial time [23]. A dynamic programming approach is developed to solve the problem for undirected case with small size instances. This type of problem shows itself in operations like snow plowing where streets have different priority levels [18].

3.1.1.4. Min- Max k-Chinese Postman Problem

Wøhlk [17] examines a different CPP variation , called min-max k-Chinese postman problem, which is defined on a connected and undirected graph. The aim of the problem is to find k tours, each starting and ending at the depot node, where every edge is covered by at least one of the tours, while keeping the length of the longest tour minimum. To serve the customers as early as possible, this type of objective is preferable.

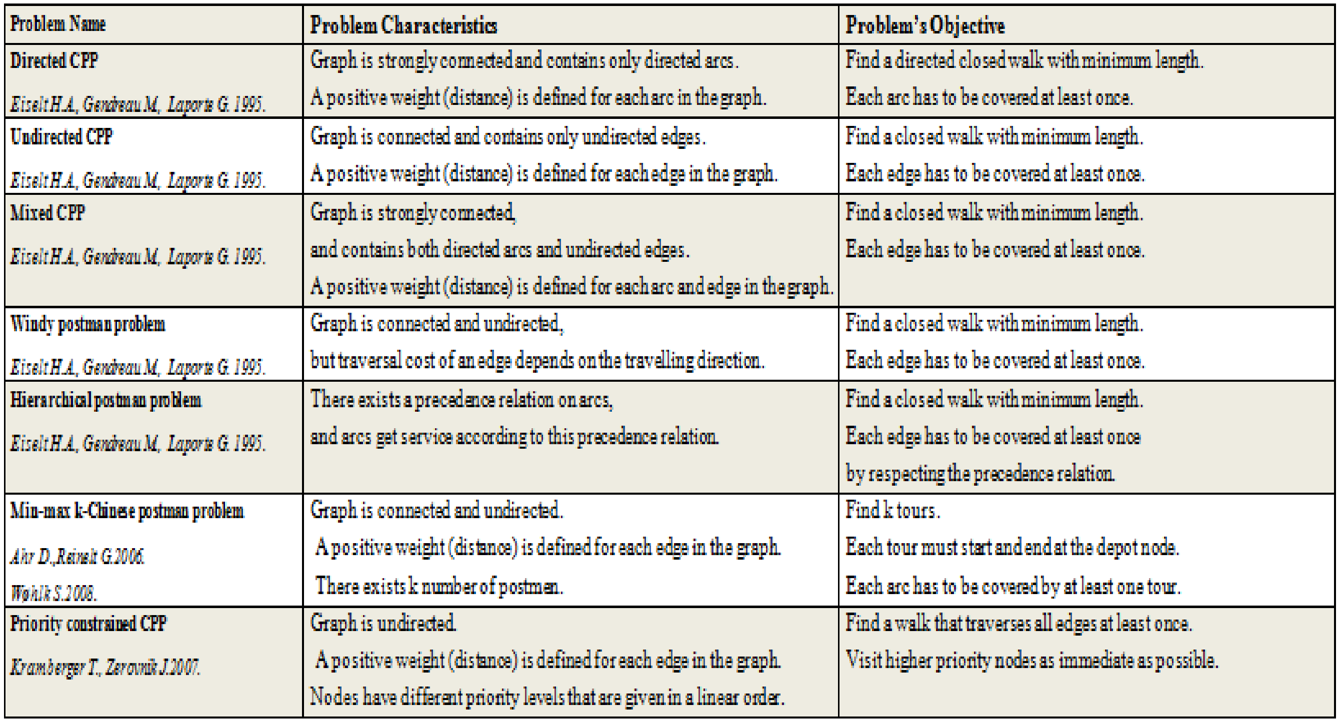
Ahr et al. [24] propose a tabu search heuristic which provides optimal or near optimal solutions in many cases for the min-max k-Chinese postman problem.

3.1.1.5. Priority Constrained Chinese Postman Problem

Kramberger et al. [25] analyse a different variation of CPP, where nodes have different priority levels given in a linear order. The problem is called as priority constrained Chinese postman problem and it aims to visit higher priority nodes as early as possible in such a way that all edges are traversed at least once. The optimal solution of the problem gives an Eulerian walk and they propose an algorithm that combines Fleury's algorithm [26] to construct an Eulerian walk and Dijkstra's algorithm to compute the shortest paths. In this paper, they focus on the salt gritting application of this problem.

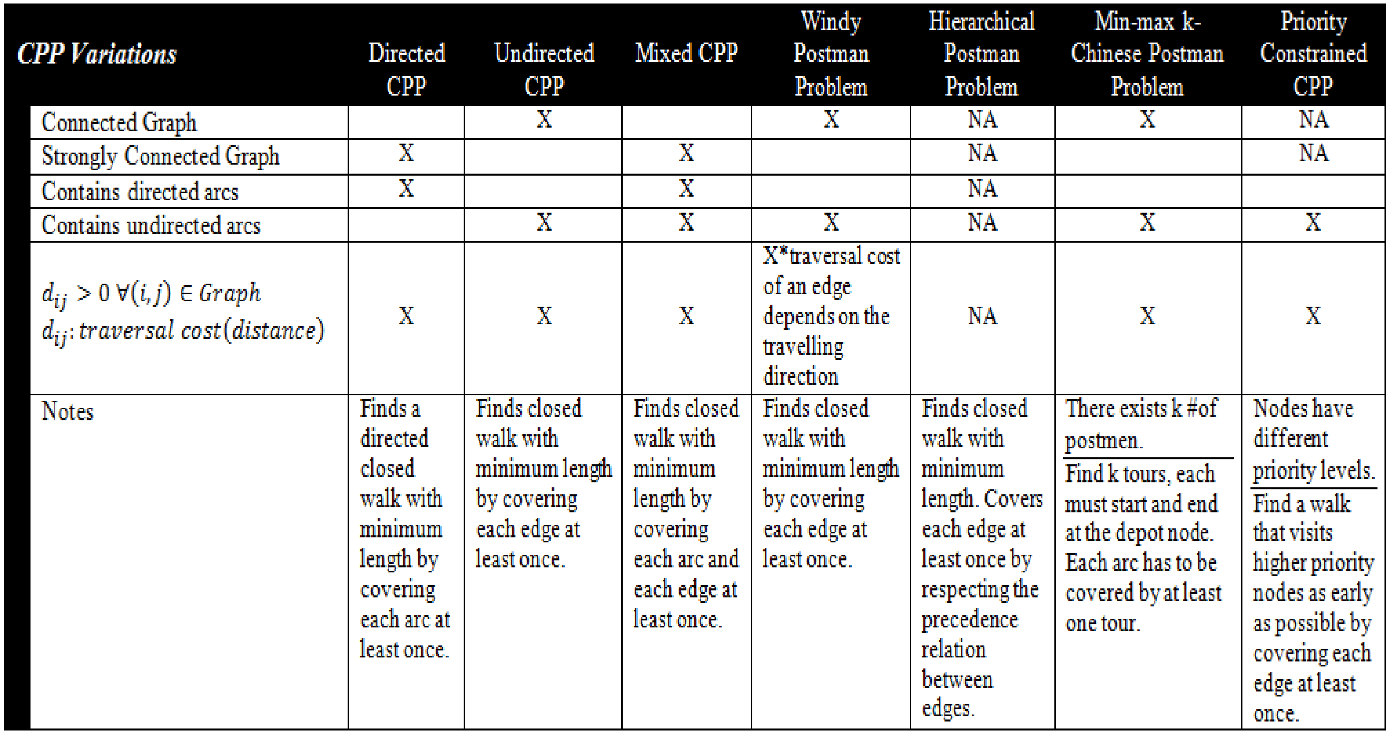
The following table summarizes the variations of CPP.

Table 3-1: Chinese postman problem (CPP) variations



Problem Name	Problem Characteristics	Problem's Objective
Directed CPP *Eiselt H.A, Gendreau M, Laporte G. 1995.*	Graph is strongly connected and contains only directed arcs. A positive weight (distance) is defined for each arc in the graph.	Find a directed closed walk with minimum length. Each arc has to be covered at least once.
Undirected CPP *Eiselt H.A, Gendreau M, Laporte G. 1995.*	Graph is connected and contains only undirected edges. A positive weight (distance) is defined for each edge in the graph.	Find a closed walk with minimum length. Each edge has to be covered at least once.
Mixed CPP *Eiselt H.A, Gendreau M, Laporte G. 1995.*	Graph is strongly connected, and contains both directed arcs and undirected edges. A positive weight (distance) is defined for each arc and edge in the graph.	Find a closed walk with minimum length. Each edge has to be covered at least once.
Windy postman problem *Eiselt H.A, Gendreau M, Laporte G. 1995.*	Graph is connected and undirected, but traversal cost of an edge depends on the travelling direction.	Find a closed walk with minimum length. Each edge has to be covered at least once.
Hierarchical postman problem *Eiselt H.A, Gendreau M, Laporte G. 1995.*	There exists a precedence relation on arcs, and arcs get service according to this precedence relation.	Find a closed walk with minimum length. Each edge has to be covered at least once by respecting the precedence relation.
Min-max k-Chinese postman problem *Ahr D.,Reinelt G.2006.* *Wøhlk S.2008.*	Graph is connected and undirected. A positive weight (distance) is defined for each edge in the graph. There exists k number of postmen.	Find k tours. Each tour must start and end at the depot node. Each arc has to be covered by at least one tour.
Priority constrained CPP *Kramberger T., Zerovnik J.2007.*	Graph is undirected. A positive weight (distance) is defined for each edge in the graph. Nodes have different priority levels that are given in a linear order.	Find a walk that traverses all edges at least once. Visit higher priority nodes as immediate as possible.

Table 3-2: Summary of Chinese postman problem (CPP) variations



CPP Variations	Directed CPP	Undirected CPP	Mixed CPP	Windy Postman Problem	Hierarchical Postman Problem	Min-max k-Chinese Postman Problem	Priority Constrained CPP
Connected Graph		X		X	NA	X	NA
Strongly Connected Graph	X		X		NA		NA
Contains directed arcs	X		X		NA		
Contains undirected arcs		X	X	X	NA	X	X
$d_{ij} > 0\ \forall(i,j) \in Graph$ $d_{ij}: traversal\ cost(distance)$	X	X	X	X*traversal cost of an edge depends on the travelling direction	NA	X	X
Notes	Finds a directed closed walk with minimum length by covering each arc at least once.	Finds closed walk with minimum length by covering each edge at least once.	Finds closed walk with minimum length by covering each arc and each edge at least once.	Finds closed walk with minimum length by covering each edge at least once.	Finds closed walk with minimum length. Covers each edge at least once by respecting the precedence relation between edges.	There exists k #of postmen. Find k tours, each must start and end at the depot node. Each arc has to be covered by at least one tour.	Nodes have different priority levels. Find a walk that visits higher priority nodes as early as possible by covering each edge at least once.

3.1.2. Rural Postman Problem

In 1974, Orloff [27] defined the RPP, where the objective is to find a minimum cost tour that traverses only a subset of arcs, which are called required arcs, at least once. RPP is also defined on a connected graph with nodes, edges and a cost matrix [17]. Lenstra and Rinnooy Kan [14] prove that both undirected and directed versions of RPP are NP-Hard. However if the required edges are all edges of the graph, then the problem becomes a CPP [18]. Many variations of RPP exist. Street sweeping, snow plowing, garbage collection, mail delivery, school bus routing and meter reading are the most common application areas of RPP.

3.1.2.1. Undirected, directed and mixed Rural Postman Problems

Eiselt et al. [28] come up with many variants of RPP. For the undirected and directed version of RPP which are both NP-Hard [14], mathematical models for each problem, branch and bound ILP based algorithms and some heuristics are presented in this survey. Another variant of RPP, named stacker crane problem, is defined on a mixed graph. It contains both directed arcs and undirected edges. The aim of the problem is to find a shortest circuit which traverses each directed arc of the graph at least once. The problem is NP-Hard [29] and there is no exact algorithm for the stacker crane problem; however some heuristic procedures are proposed.

3.1.2.2. Privatized Rural Postman Problem

In their study Araoz et al. [30] focus on the privatized rural postman problem where a profit function is defined for each edge that can be collected only the first time that the edge is traversed. The aim is to find the maximum profit-least cost tour, which starts and ends at the depot node. The solution of this problem is an Eulerian subgraph that starts and ends at the depot node, and isolated nodes. This indicates that, the solution must be connected to the depot node, however isolated nodes are also allowed. A branch and cut algorithm is presented as well as some heuristic methodologies to solve the problem.

It is indicated that privatized rural postman problem is the edge version of the TSP with profits. Additionally, it is informed that privatized Chinese postman problem is a variant of the privatized rural postman problem where it turns to the privatized rural postman problem when the graph is connected.

A solution algorithm for the prize- collecting rural postman problem which is the same as the previously defined privatized rural postman problem is presented by Araoz et al. [31]. A mathematical model is presented and a linear integer program is introduced. Also the proposed algorithm that gives very satisfactory results is explained. It has two phases where in the first phase upper bounds are obtained with an iterative LP-based cutting plane algorithm, and lower bounds are obtained with a heuristic. In the second phase of the algorithm, integer programming techniques are used with insertion of cutting planes. Collection of recycling bins by a private entity is one of the application areas of the prize-collecting rural postman problem.

3.1.2.3. Rural Postman Problem with Deadline Classes

Letchford and Eglese [32] come up with another variation of rural postman problem where the edges are classified according to their deadline classes. It is required that, edges must be served in their specified time limits. Additionally, interphase connectivity is an important constraint for this problem where the deadline classes can be considered as successive time periods, and the route of the postman should be connected between these phases. The edges that postman have to visit in each deadline class are the inputs of this problem. When the deadlines are removed, the problem turns into the standard rural postman problem. In this paper, a mathematical model and an algorithm based on the use of valid inequalities as cutting planes are introduced. Parcel delivery and salt gritting are some of the application areas of this problem.

3.1.2.4. Min-Max k-Vehicles Windy Rural Postman Problem

Another variation of rural postman problem is introduced by Benavent et al. [33], which is called min-max k-vehicles windy rural postman problem. There are k vehicles and the aim is to find k tours, one for each vehicle, where each tour starts and ends at the depot, and each required edge has to be serviced by exactly one of the vehicles, in a windy graph structure. The objective function of this problem considers minimizing the length of the longest tour which results to serve each customer as early as possible, as well as achieving a balanced tour schedule for the vehicles. A mathematical model, and a branch and cut method with separation procedures for the min-max k-vehicles windy rural postman problem are suggested in this paper.

The following table summarizes the variations of RPP.

Table 3-3: Rural postman problem (RP P) variations

Problem Name	Problem Characteristics	Problem's Objective
Directed RPP *Eiselt H.A., Gendreau M., Laporte G. 1995.*	Graph is strongly connected and contains only directed arcs. There is a subset of required arcs. A positive weight (distance) is defined for arcs.	Find a directed closed walk with minimum length. Each required arc has to be covered at least once.
Undirected RPP *Eiselt H.A., Gendreau M., Laporte G. 1995.*	Graph is connected and contains only undirected edges. There is a subset of required edges. A positive weight (distance) is defined for edges.	Find a closed walk with minimum length. Each required edge has to be covered at least once.
Stacker Crane problem *Eiselt H.A., Gendreau M., Laporte G. 1995.*	Graph is strongly connected and mixed. A positive weight (distance) is defined for each arc and edge in the graph.	Find a mixed closed walk with minimum length that traverses each directed arc of the graph at least once.
Privatized RPP *Aráoz J., Fernández E., and Zoltan C.2006.*	A nonnegative cost function and a nonnegative profit function is defined for each edge in the graph. Positive profit edges are considered to be required.	Find the maximum profit least cost tour which starts and ends at the depot node. Profit can be collected only the first time that the edge is traversed, while cost incurs at every traversal.
Prize-collecting RPP *Aráoz J., Fernández E., Meza O. 2009.*	A nonnegative cost function and a nonnegative profit function is defined for each edge in the graph. Positive profit edges are considered to be required.	Find the maximum profit least cost tour which starts and ends at the depot node. Profit can be collected only the first time that the edge is traversed, while cost incurs at every traversal.
RPP **with deadline classes** *Letchford A.N., Eglese R.W. 1998.*	Graph is connected. There is a subset of required edges and each belongs to a deadline class.	Find a minimum cost tour which starts and ends at the depot node. Edges must be served in their specified time limits.
min-max -k vehicles **windy RPP** *Benavent E., Corberán A.,* *Sanchis J.M., Plana I.2007.*	Graph is undirected but traversal cost of an edge depends on the travelling direction. There exists k number of vehicles.	Find k tours. Each tour must start and end at the depot node. Each required edge has to be covered by exactly one vehicle. The length of the longest tour should be minimized.

Table 3-4: Summary of rural postman problem (RPP) variations

RPP Variations	Directed RPP	Undirected RPP	Stacker Crane Problem	Privatized RPP	Prize-collecting RPP	RPP with deadline classes	Min-max k-vehicles windy RPP
Connected graph		X		NA		X	NA
Strongly connected graph	X		X	NA			NA
Contains directed arcs	X		X	NA		NA	
Contains undirected arcs		X	X	NA		NA	X
$d_{ij} > 0 \ \forall (i,j) \in Graph$ d_{ij}: $traversal\ cost(distance)$	X	X	X	* A nonnegative cost function and a nonnegative profit function is defined for each edge in the graph.	* A nonnegative cost function and a nonnegative profit function is defined for each edge in the graph.	NA	*Traversal cost of an edge depends on the travelling direction.
There is a subset of required arcs	X		X*only the directed arcs are considered to be required.				
There is a subset of required edges		X		X* Positive profit edges are considered to be required.	X* Positive profit edges are considered to be required.	X	X
Notes	Find a directed closed walk with minimum length by covering each required arc at least once.	Find a closed walk with minimum length by covering each required edge at least once.	Find a mixed closed walk with minimum length by covering each required arc at least once.	Find the maximum profit least cost tour, which starts and ends at the depot node. Profit can be collected only the first time that the edge is traversed, while cost incurs at every traversal.	Find the maximum profit least cost tour, which starts and ends at the depot node. Profit can be collected only the first time that the edge is traversed, while cost incurs at every traversal.	Find a closed walk with minimum length. Edges must be served in their specified time limits. Each edge belongs to a deadline class.	Find k tours and the length of the longest tour should be minimized. There exists k#of vehicles. Each required edge has to be covered by exactly one vehicle.

3.1.3. Capacitated Arc Routing Problem

When a capacity constraint of the vehicle is included, the problem is referred as the capacitated arc routing problem (CARP). CARP was first defined by Golden and Wong in 1981[34]. For CARP, besides the identical capacity restriction of the vehicles, graph is connected and main elements of the graph are nodes, edges, cost matrix and demand matrix. Just as the cost matrix, demand matrix is also defined for the edges. In CARP, the vehicle has finite capacity and needs to be refilled, emptied or recharged. This is achieved by returning to the depot or another specified station. There exists one vehicle and fleet of vehicles version of this problem.

We remark here that, CARP with strictly positive demands on the edges is called capacity constrained Chinese postman problem with m vehicles, which is defined by Christofides in 1973[35]. If the demands on the edges are defined as nonnegative, then the problem turns into capacity constrained rural postman problem with m vehicles, where the definition is done by Golden and Wong in 1981[34].

There are many variations of CARP, and their application areas are also various. Most of the CARPs are used in winter gritting, refuse collection, mail delivery, street sweeping operations and police patrols.

Eiselt et al. [28] analyse the capacitated arc routing problem (CARP) which is NP-Hard [34], [36], [37] where vehicles have finite capacity, and a nonnegative and non-identical demand or weight is defined for each arc in the graph. The aim of the problem is to find a minimum cost traversal of all arcs such that each arc is serviced by exactly one vehicle without exceeding the capacity of the vehicle. In their analysis, it is also emphasized that CARP is a generalization of the capacitated Chinese postman problem where each arc has positive demands [35]. Also, mathematical models, solution algorithms like branch and cut algorithm and some heuristic methodologies are explained for the CARP in this survey.

In their paper, Lancomme et al. [38] investigates the classical CARP, and a bi-objecitive genetic algorithm is suggested.

Capacitated arc routing problem has also many variations.

3.1.3.1. Capacitated Arc Routing Problem on Directed Graphs

In her work, Wøhlk [17] investigates many variants of CARP. It is said that, classical CARP is defined on an undirected graph. For the CARP on directed graphs, she investigates the suggested valid inequalities and separation algorithm for ILP formulation. For the CARP on mixed graphs, she examines the heuristics that are suggested for this problem, and also she refers to some solution algorithms like Memetic algorithm which is adapted to this problem. Additionally, a linear programming (LP) formulation that is used to obtain strong lower bounds in the cutting plane algorithm is examined.

3.1.3.2. Multi Depot Capacitated Arc Routing Problem

In classical CARP, there exists one depot node, and the tour starts and ends at this node. When there exist several depot nodes, where each tour must start and end at one of these depot nodes, the problem is called multi depot CARP (MD-CARP). The vehicle could end its tour in the depot where it starts the tour, or it is possible to return to another depot at the end of its tour. A unique solution strategy which is developed for MD-CARP is referred by Wøhlk [17].

Amberg et al. [39] also consider the same problem which is now called as multiple center capacitated arc routing problem. In this problem, the capacities that are taken into account are not only the vehicle capacities which are used to satisfy the demand but also the maximum allowed time duration. In this paper, a heuristic transformation of multiple center CARP into a multiple center capacitated minimum spanning tree problem is considered which takes into account the arc constraints.

3.1.3.3. Capacitated Arc Routing Problem with Intermediate Facilities and with Refill Points

Since the vehicles have finite capacity they need to be refilled, emptied or recharged. In classical CARP, this operation is done in the depot node, however another variation of CARP is defined where vehicles starts and ends their tour at the depot node, but the vehicles recharge their capacity in some nodes which are called intermediate facilities. This problem is called CARP with intermediate facilities (CARP-IF). Refuse collection is one of the application areas of CARP-IF where the dump sites can be considered as intermediate facilities. In her work two lower bounds and two heuristics that are developed for CARP-IF are examined by Wøhlk [17].

Amaya et al. [40] analyse the capacitated arc routing problem with refill point, where the vehicles can refill their capacity at any point in the graph, with the help of refill vehicles which are different from service vehicles. The aim of the problem is to find minimum cost routes for both refill and service vehicles. Since the refill points are also determined besides the routes of service vehicles, this problem can be considered as a location arc routing problem. In this paper, an ILP model is suggested to solve the problem, but since it has an exponential number of connectivity constraints, it solves the problem in reasonable time for small size instances only, by using branch and bound algorithm. Since in the optimal solution all connectivity constraints are not active, a cutting plane method can be used to solve the problem for small to medium size instances. When the problem size is larger, the method provides a lower bound. Road network maintenance to paint the road markings is one of the application areas of the CARP with refill points.

3.1.3.4. Capacitated Arc Routing Problem with Mobile Depots

When there exist two different types of service vehicles, where type1 vehicles unload onto the type2 vehicles, and type2 vehicles unload themselves at the depot node, the problem is called as CARP with mobile depots. In this problem, routing of both type of vehicles are considered as well as the time that two vehicles meet to perform the unload operation in some node. A variable neighbourhood descent algorithm is referred to solve the problem [17].

3.1.3.5. Periodic Capacitated Arc Routing Problem

Periodic CARP is another variation of CARP where a long time period is considered, and customers require service more than once. Mathematical formulation and heuristic methodologies that are developed for periodic CARP are discussed [17]. As an application area, refuse collection where a ménage requires service two or three times a week is considered.

3.1.3.6. Stochastic Capacitated Arc Routing Problem

When the demands on the edges are random variables, then the classical CARP turns into stochastic CARP. The Memetic Algorithm is developed for stochastic CARP is investigated in [17]. Refuse collection and snow removal are some of the application areas of stochastic CARP when the exact demand on the arcs is not known.

3.1.3.7. Capacitated Arc Routing Problem with Time Windows, and with Alternative Objective Functions

CARP with time windows is a variation of CARP where it is required to give service to the customers within a pre-determined time window. Two mathematical models and some heuristics are investigated in [17].

Wøhlk [17] analyses CARP with alternative objective functions. Minimizing the total number of vehicles used, equalizing the load of the tours, minimizing the length of the longest tour are investigated as different objective functions. Some heuristic methodologies are considered to solve these problems.

3.1.3.8. Undirected Capacitated Arc Routing Problem with Profits

Archetti et al. [41] define another CARP variation named the undirected capacitated arc routing problem with profits. A profit and a demand are defined for a subset of edges of the graph where a travel time is defined for all edges. There exists a fleet of capacitated vehicles, and the objective of the problem is to find a set of routes for the vehicles and collecting the maximum amount of profit while respecting the time limit of the routes and capacity of the vehicles. It should be noted that, at most one vehicle can collect the profit of an edge, and the vehicle which collects the profit has to satisfy the demand of this edge. This problem resembles the prize collecting rural postman problem. However, in prize collecting rural postman problem there is no associated demand for the edges, but only it is given that an edge requires service or not. Also the capacity constraints in terms of time and vehicle distinguish this problem from the prize collecting rural postman problem. In this paper, a variable neighbourhood search and two tabu search heuristics are presented to solve the problem, as well as a solution procedure based on column generation and a branch and prize algorithm is suggested.

3.1.3.9. Capacitated Arc Routing Problem with Vehicle/Site Dependencies

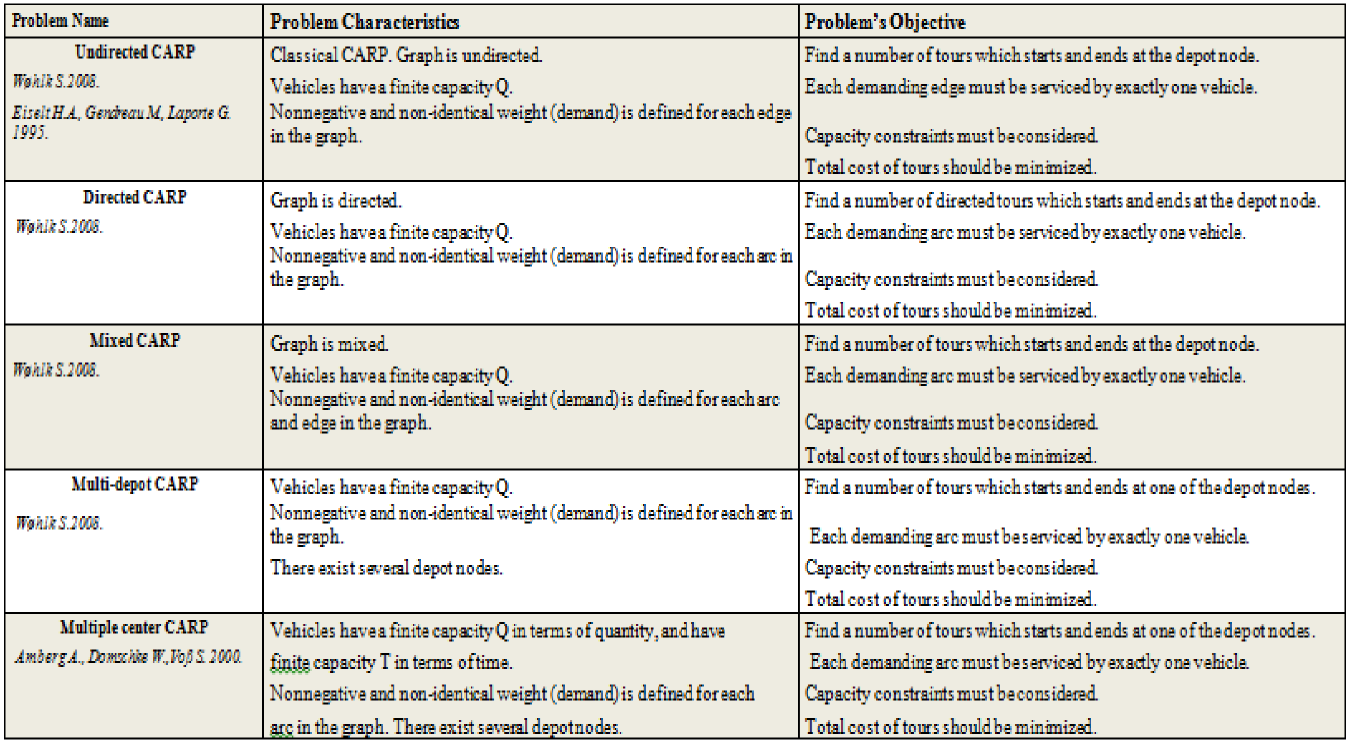
Sniezek and Bodin [42] study the CARP with vehicle/site dependencies. In this type of problem, there should be at least two different classes of vehicles where vehicle/site dependency on an arc describes that this arc cannot be traversed or serviced by a vehicle from some vehicle class if this arc is serviced or traversed by any vehicle from remaining vehicle classes. As a solution procedure; two mixed integer programs, the Initial Fleet Mix Generator, a mathematical programming procedure, and a measure of goodness function is proposed which comprises the Composite Approach. Additionally, as Wøhlk [17] analyses, a vehicle decomposition algorithm is presented to solve this problem.

3.1.3.10. Capacitated Arc Routing Problem with Deadheading Demands

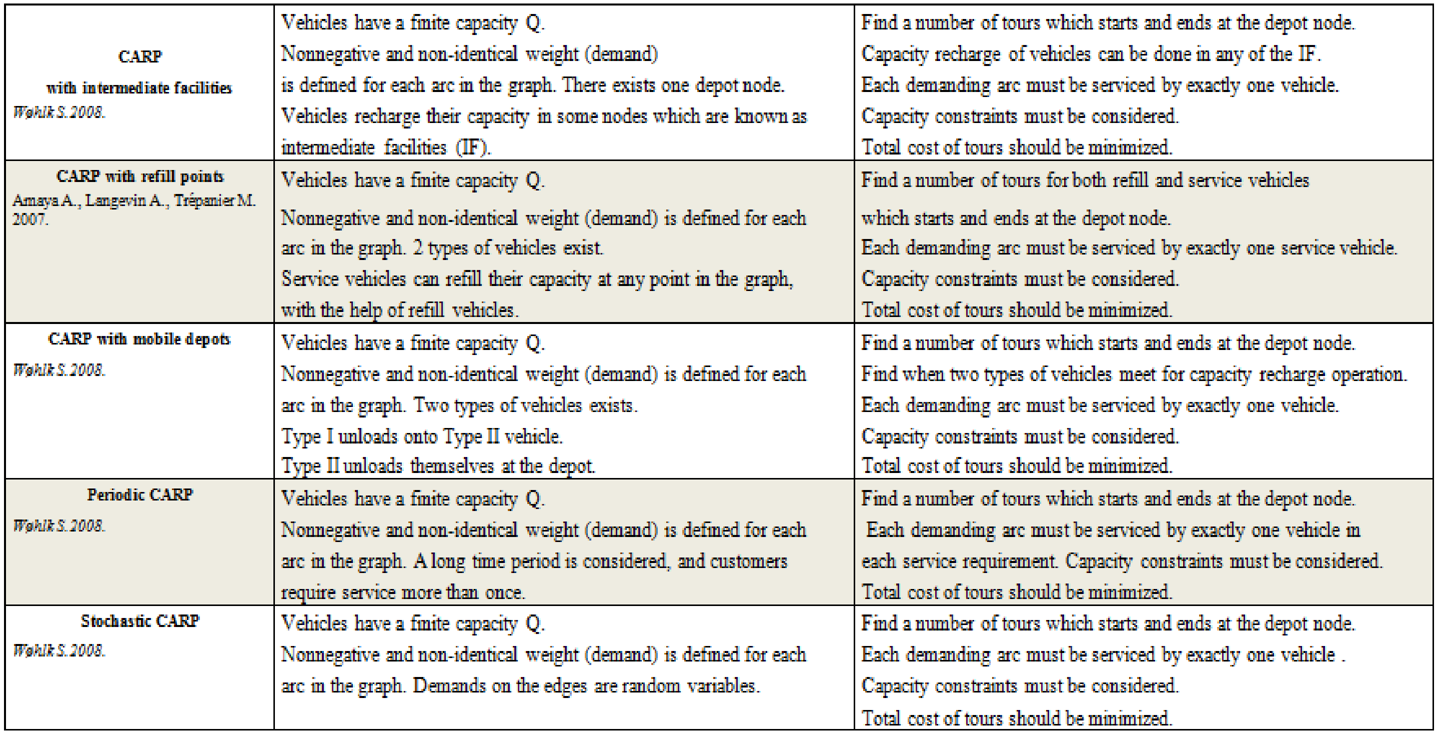
Another CARP variation where vehicle uses capacity not only while servicing but also during traversing the arcs is examined by Kirlik and Sipahioglu [43] which is named as CARP with deadheading demands(CARPDD). Deadheading refers the case when an edge is traversed without servicing, and if the capacity spent by deadheading is ignored, the problem turns into classical CARP. CARPDD is an NP-Hard problem, and an adaptation of Ulusoy heuristic [44] is used as a solution approach. Also a mathematical model is suggested by the authors. In this problem, the aim is to determine a path where every point in the given graph is covered at least once; vacuum cleaning and lawn mowers are typical application areas of this problem.

The following table summarizes the variations of CARP.

Table 3-5: Capacitated arc routing problem (CARP) variations



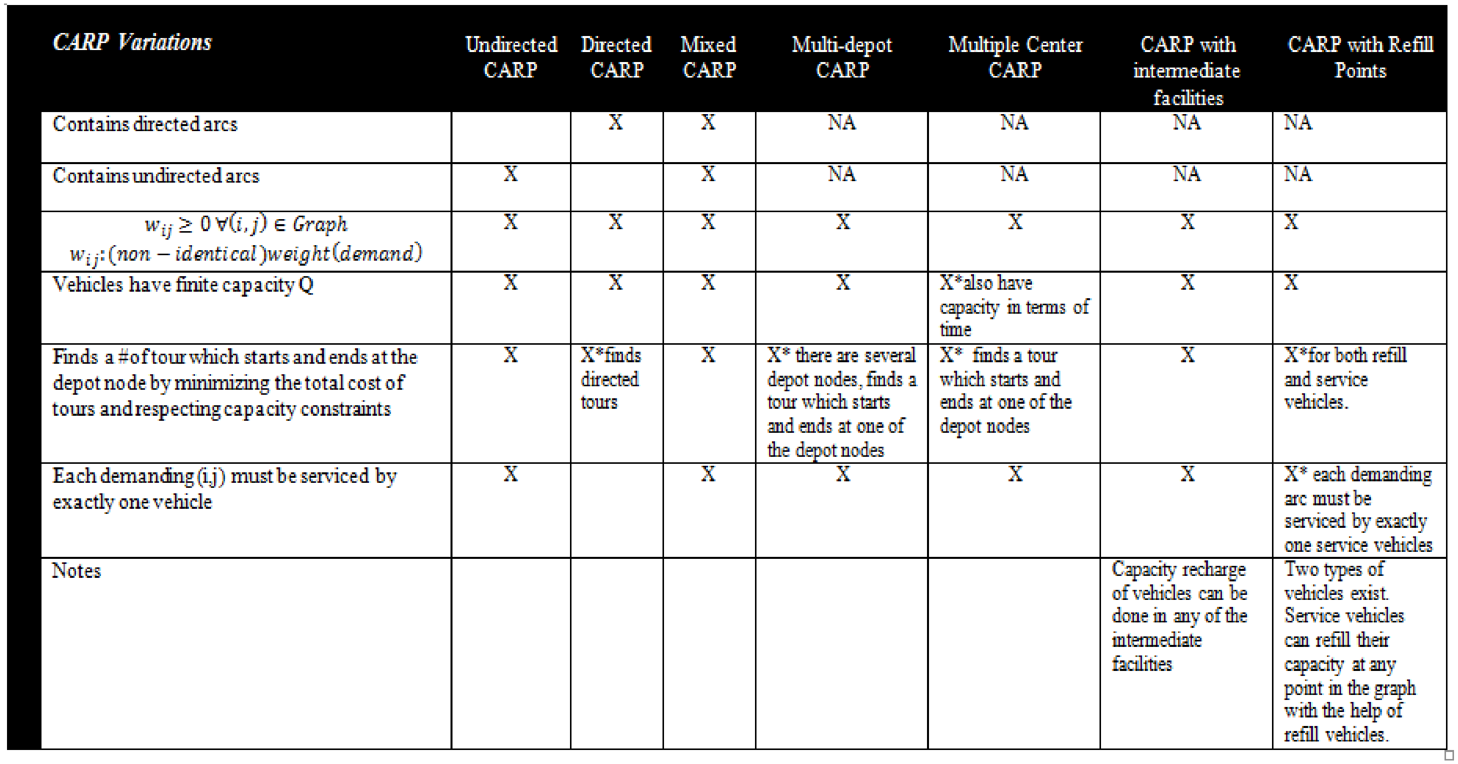
Problem Name	Problem Characteristics	Problem's Objective
Undirected CARP *Wøhlk S.2008.* *Eiselt H.A., Gendreau M, Laporte G. 1995.*	Classical CARP. Graph is undirected. Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each edge in the graph.	Find a number of tours which starts and ends at the depot node. Each demanding edge must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.
Directed CARP *Wøhlk S.2008.*	Graph is directed. Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph.	Find a number of directed tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.
Mixed CARP *Wøhlk S.2008.*	Graph is mixed. Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc and edge in the graph.	Find a number of tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.
Multi-depot CARP *Wøhlk S.2008.*	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. There exist several depot nodes.	Find a number of tours which starts and ends at one of the depot nodes. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.
Multiple center CARP *Amberg A., Domschke W., Voß S. 2000.*	Vehicles have a finite capacity Q in terms of quantity, and have finite capacity T in terms of time. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. There exist several depot nodes.	Find a number of tours which starts and ends at one of the depot nodes. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.



CARP **with intermediate facilities** *Wøhlk S. 2008.*	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. There exists one depot node. Vehicles recharge their capacity in some nodes which are known as intermediate facilities (IF).	Find a number of tours which starts and ends at the depot node. Capacity recharge of vehicles can be done in any of the IF. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.
CARP with refill points Amaya A., Langevin A., Trépanier M. 2007.	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. 2 types of vehicles exist. Service vehicles can refill their capacity at any point in the graph, with the help of refill vehicles.	Find a number of tours for both refill and service vehicles which starts and ends at the depot node. Each demanding arc must be serviced by exactly one service vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.
CARP with mobile depots *Wøhlk S. 2008.*	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. Two types of vehicles exists. Type I unloads onto Type II vehicle. Type II unloads themselves at the depot.	Find a number of tours which starts and ends at the depot node. Find when two types of vehicles meet for capacity recharge operation. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.
Periodic CARP *Wøhlk S. 2008.*	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. A long time period is considered, and customers require service more than once.	Find a number of tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle in each service requirement. Capacity constraints must be considered. Total cost of tours should be minimized.
Stochastic CARP *Wøhlk S. 2008.*	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. Demands on the edges are random variables.	Find a number of tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle . Capacity constraints must be considered. Total cost of tours should be minimized.

CARP with time windows Wøhlk S.2008	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. Pre-determined time windows are defined for demanding arcs.	Find a number of tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle within its pre-determined time window. Capacity constraint must be considered. Total cost of tours should be minimized.
Undirected CARP with profits Archetti C., Feillet D., Hertz A., Speranza M.G.2010.	Vehicles have a finite capacity Q. Routes of each vehicle have time duration limit T. Nonnegative and non-identical weight (demand) and profit is defined for a subset of edges of the graph. Travel time is defined for all edges.	Find a number of tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total amount of collected profit should be maximized.
CARP with vehicle/site dependencies Wøhlk S.2008. Sniezek J., Bodin L.2006.	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. There exists at least two different vehicle classes for vehicle/site dependency.	Find a number of tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints and vehicle/site dependencies must be considered. Total cost of tours should be minimized.
CARP with deadheading demands Kirlik G., Sipahioglu A.2012.	Vehicles have a finite capacity Q. Nonnegative and non-identical weight (demand) is defined for each arc in the graph. Deadheading also uses capacity of the vehicles.	Find a number of tours which starts and ends at the depot node. Each demanding arc must be serviced by exactly one vehicle. Capacity constraints must be considered. Total cost of tours should be minimized.

Table 3-6: Summary of capacitated arc routing problem (CARP) variations



CARP Variations	Undirected CARP	Directed CARP	Mixed CARP	Multi-depot CARP	Multiple Center CARP	CARP with intermediate facilities	CARP with Refill Points
Contains directed arcs		X	X	NA	NA	NA	NA
Contains undirected arcs	X		X	NA	NA	NA	NA
$w_{ij} \geq 0 \, \forall (i,j) \in Graph$ w_{ij}: $(non-identical)weight(demand)$	X	X	X	X	X	X	X
Vehicles have finite capacity Q	X	X	X	X	X*also have capacity in terms of time	X	X
Finds a #of tour which starts and ends at the depot node by minimizing the total cost of tours and respecting capacity constraints	X	X*finds directed tours	X	X* there are several depot nodes, finds a tour which starts and ends at one of the depot nodes	X* finds a tour which starts and ends at one of the depot nodes	X	X*for both refill and service vehicles.
Each demanding (i,j) must be serviced by exactly one vehicle	X		X	X	X	X	X* each demanding arc must be serviced by exactly one service vehicles
Notes						Capacity recharge of vehicles can be done in any of the intermediate facilities	Two types of vehicles exist. Service vehicles can refill their capacity at any point in the graph with the help of refill vehicles.

CARP Variations	CARP with mobile depots	Periodic CARP	Stochastic CARP	CARP with time windows	Undirected CARP with profits	CARP with vehicle/site dependencies	CARP with deadheading demands
Contains directed arcs	NA	NA	NA	NA	NA	NA	NA
Contains undirected arcs	NA	NA	NA	NA	NA	NA	NA
$w_{ij} \geq 0 \ \forall (i,j) \in Graph$ $w_{ij}: (non-identical) weight(demand)$	X	X	X*demand on the edges are random variables	X	w_{ij} is defined for a subset of edges, profit is defined for a subset of edges, travel time is defined for all edges.	X	X
Vehicles have finite capacity Q	X	X	X	X	X	X	X* deadheading also uses the capacity of the vehicles.
Finds a #of tour which starts and ends at the depot node by minimizing the total cost of tours and respecting capacity constraints	X	X	X	X	X	X*by considering the vehicle/site dependencies.	X
Each demanding (i,j) must be serviced by exactly one vehicle	X	X* in each service requirement	X	X* within its predetermined time window	X	X	X
Notes	Two types of vehicles exist. Type I unloads onto Type II vehicle. Type II unloads themselves at the depot. Finds when two types of vehicles meet for capacity recharge operation.	A long time period is considered, and customers require service more than once.		Predetermined time windows are defined for demanding arcs	Routes of each vehicle have time duration limit. Total amount of collected profit should be maximized.	There exist at least two different vehicle classes for vehicle/site dependency.	

3.2 Node Routing Problems

Node routing problems (NRP) are special cases of GRP and Vehicle routing problem (VRP) is one of the famous node routing problems. When there is a subset of nodes which require to be visited with an empty required edge set, the GRP reduces to VRP. Since the general VRP literature is too broad, we only focus on the VRP with blocked networks.

One of the problems from the shortest path classification is Canadian traveller problem (CTP). In their article, Xu et al. refer to CTP as an abstraction of the online shortest paths/routing problems [45].

3.2.1. Canadian Traveller Problem

It is first defined by Papadimitriou and Yannakakis, and proven that it is a #P-Hard problem [46]. The traveller knows the graph structure and edge costs but some edges may become blocked and traveller beholds this blockage only when he/she reaches the adjacent node of this blocked edge. The traveller does not know which edge will be blocked in advance, where this constitutes the online structure of the problem. Also it is assumed that, even if the blocked edges are removed, the subgraph is still connected [45]. It is defined for a single source and a single destination and the aim is to find the minimum cost route from source to destination. The classical version of the CTP is a stochastic problem and the blocked edges remain blocked forever [47], [48]. It is worth to note that, if all road blockages are known in advance, the optimal travel path can be obtained by applying shortest path algorithm from source to destination. However, as it is indicated before, the problem has online nature, where the future blockages are not known in advance. Therefore, the optimal travel strategy cannot be given by the shortest path [47].

In their study, Bar-Noy and Schieber introduce variations of CTP [47]. One of them is the recoverable-CTP where the blocked roads may become open again. There are both stochastic and deterministic versions of recoverable-CTP, where in the stochastic version, each edge has a blockage probability. In the deterministic version, there is a fixed bound on the total number of potential blockages. In the recoverable-CTP, there are recovery times of edges. It is assumed that the recovery times of blocked edges that are incident to the same node is the same. When all the recovery times are significantly large, recoverable-CTP becomes the classical CTP. k-CTP is another variant of CTP where k is a parameter that represents the maximum number of potential road blockages. When k equals the number of edges, k-CTP becomes classical CTP [47].

In CTP, the traveller selects a path and starts to travel without knowing the future blockages and when he/she encounters a blocked edge, it is required to determine whether to wait for reopening of the blocked edge, or look for another way. The main factor that is considered is the recovery time versus time to travel along another path. In this respect, if the problem structure becomes offline instead of online, Bar-Noy and Schieber state that, the optimal strategy is given by the shortest path from source to destination [47].

In the literature, there is no mathematical model developed for CTP. Instead, solution algorithms and heuristics are provided. Nikolova & Karger propose exact algorithms for special graphs, where it is not required to remember the edge costs. Some examples for these special cases are the directed acyclic graphs or the cases that edge costs that are adjacent to a node are resampled each time when this node is visited. For the first case dynamic programming approach and for the latter case a standard Markov decision process are appropriate to solve the problems in polynomial time [48]. They also point out that when traveller is free to return to edges whose values are fixed, this proposed exact algorithm does not work, and heuristics are provided [48].

Xu et al. offer a greedy strategy and a strategy that combines the greedy strategy and the reposition strategy where the reposition strategy implies the case that traveller turns back to the node were he/she starts when encountered with a blocked edge [45].

In their study, Bar-Noy and Schieber provide polynomial time travel strategy for the deterministic CTP for the cases where the gap between recovery times and travel times is not too big, and the maximum number of possible blockages known beforehand. Another polynomial time travel strategy for the stochastic version of CTP is also presented for the cases where recovery times are not very different than travel times [47].

Briefly, it can be said that, GRP is the most general version of routing problems which includes both arc and node routing aspects. The problem that only considers the arc routing aspects is ARP and it is a special case of GRP. CPP, RPP and CARP are the leading arc routing problems, where it is possible to reduce the CARP into CPP and RPP according to the definition of the demand function. NRP is the problem that considers only the node routing aspects, and CTP is a special case of NRP.

3.3. Debris Removal Problem in the Response Phase and Its Relation to the Literature

Figure 3-1 depicts the relations of the problems in the literature with proposed debris removal problem schematically:

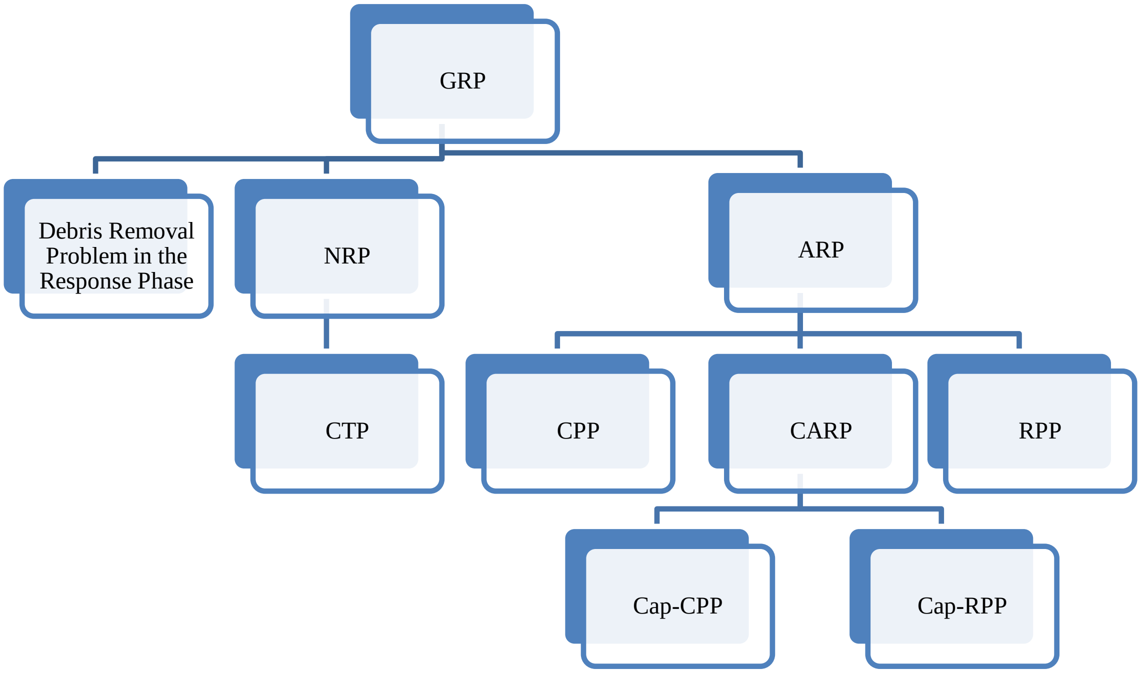


Figure 3-1: Relations of the problems in the literature with proposed debris removal problem in the response phase

Undirected GRP takes into account a subset of nodes and subset of edges that require service. The purpose is to give service to all required nodes and traversing all required edges. In Debris Removal Problem in the Response Phase, critical nodes, which demand disaster relief material, are the required nodes where their demand must be satisfied, however, the roads which are blocked by debris are the edges that may require service, but it is not an obligation to sweep all the blocked edges if the demand of the critical nodes can be satisfied without doing it. In addition to that, in Debris Removal Problem in the Response Phase, there is a vehicle which is both responsible for unblocking roads by sweeping the debris and delivering the disaster relief materials to the critical nodes.

In Debris Removal Problem in the Response Phase travel time is defined for all edges in the graph whereas an effort value, which is denominated in terms of time to be spent to unblock roads, is defined for only blocked edges. Both travel time and the service time occur on the edges which are blocked. The other edges that are already unblocked can only be used to traverse and each time they are traversed, the travel time is incurred. In this respect, the Debris Removal Problem in the Response Phase resembles an arc routing problem in some point of view but its definition does not exactly fit to any of them.

In the node routing literature, recoverable Canadian traveller problem is the most resembling problem to our problem, however, despite this fact there are significant differences. Besides the fact that there is no special effort to unblock the roads in CTP where traveller may wait a road to become open again without doing nothing or find another way; road unblocking operations have a significant importance in our problem. Moreover, their definition of recovery times is far from the definition of unblocking effort of edges in our problem. We take values according to the debris amount on it and it is independent from the node that they are adjacent. Additionally, in our problem, all the blocked edges are known in advance and there is no such case that an edge becomes blocked during the travel of the traveller. Also, once an arc is opened, it remains open, and after a blockage on an arc is resolved, it is possible to benefit from the advantage of re-using this arc. Moreover, in our problem there is one source and multi destination unlike the Canadian traveller problems' one destination structure. In this respect our problem is deterministic and has an offline structure unlike the online nature of the CTP.

Additionally, since it is only required to sweep the debris instead of totally removing it, there is no capacity restriction of RESCUE to unblocking the roads. From this point of view the Debris Removal Problem in the Response Phase is separated from those problems that have capacity restriction.

In conclusion, because debris removal problem has both arc routing and node routing aspects, it is a kind of general routing problem. The primary aim of the problem is to reach the critical nodes to support disaster relief materials. Unblocking the arcs is a necessity in order to achieve this primary objective but it is not an obligation to unblock all the blocked arcs.

Consequently, even though the debris removal problem matches up with previously defined problems in the literature to a large extent, it differs in certain points which are very important. So, a new problem is defined, which takes place in general routing literature with its both node and arc routing aspects. The application area of the problem is debris removal which is never studied in this literature.

Chapter 4

Model Development

Consider the disaster-affected region as an undirected graph. Districts compose the nodes and roads compose the edges. The districts which require assistance are the critical nodes and the district that involves a qualified supply unit is the supplier. It is targeted to provide assistance to the critical nodes as soon as possible by travelling along a path which may include blocked arcs to be unblocked, if necessary. To do so, a vehicle (RESCUE) departs from the supply node and accesses to the critical nodes by removing debris on its critical path.

Let $G= (N, E)$ be a complete network where N represents the nodes and E represents the edges. For each $\{i, j\} \in E$, (i, j) and $(j, i) \in A$ constitute the arc set of the model. It is worth to note that, even if the arcs are directed, the parameter settings of arcs (i, j) and (j, i) are symmetric. The node set contains supply node, critical nodes and intermediate nodes. Some arcs in the arc set are assumed to be blocked and the arcs that are blocked or not are represented by the parameter I_{kl}, which takes the value of 0 if the arc is blocked by debris, and takes value of 1, otherwise. t_{kl} is the required time to traverse through arc *(k,l)* and W_{kl} is the required effort in terms of time to remove debris on arc *(k,l)* if this arc is blocked. Since, the parameter settings for arcs (i,j) and (j,i) are symmetric, if the debris on the arc (i,j) is removed, than the arc (j,i) also becomes open. Let $DL \subseteq N$ be the set of critical nodes and $SL \in N$ be the chosen supply node.

At the first stage, we developed mathematical models which are based on a periodic structure. In these models, there are a limited number of capacitated time periods and all critical nodes have to be visited respecting these capacities. However, it is worth to note that, these models do not allow partial removal of debris in different time periods.

There are two different models which are based on this structure. These two models differentiate from each other by their objective functions. One of them intends to complete the visit of all critical nodes as early as possible, namely, within the least number of possible periods. The other model minimizes the total travelled distance of the vehicle.

4.1. MOD-1: Minimize Visiting Time

In addition to the above-stated parameters, number of time periods, |T|, and the total time capacity of each period, $Emax_t$ are defined for MOD-1.

The following are the variables of the model:

U_{it}= 1, if node $i \in DL$ is visited at time period $t \in T$ and 0, otherwise.

Y_{ij}^{t} =1, if the debris on arc (i,j) is removed at time period $t \in T$ and 0, otherwise.

Z_{ij}^{t} =1, if RESCUE uses arc (i,j) at time period $t \in T$ and 0, otherwise.

BN_{i}^{t} =1, if node $i \in N$ is the source node of RESCUE at time period $t \in T$ and 0, otherwise.

SN_i^t =1, if node $i \in N$ is the destination node of RESCUE at time period $t \in T$ and 0, otherwise.

v_i^t = (Miller-Tucker-Zemlin variable) the number of arcs visited before reaching node $i \in N$ at time period $t \in T$.

The mathematical model that minimizes the visiting time of all the critical nodes is as follows:

$$\textbf{minimize} \sum_{\boldsymbol{i \in DL}} \sum_{\boldsymbol{t \in T}} \boldsymbol{t * U_{it}}$$

subject to

$$\sum_{t \in T} U_{it} = 1 \qquad \forall i \in DL \qquad (4.1.1)$$

$$U_{jt} \leq \sum_{i \in N} Z_{ijt} \qquad \forall j \in DL, t \in T \qquad (4.1.2)$$

$$U_{jt} \geq \sum_{i \in N} Z_{ijt} \qquad \forall j \in DL, t \in T \qquad (4.1.3)$$

$$SN_{i0} = \quad 1 \qquad \forall\ i \in SL \qquad (4.1.4)$$

$$BN_{it} = SN_{it-1} \qquad \forall\ i \in N, t \in T \qquad (4.1.5)$$

$$\sum_{i \in N} BN_{it} \ = 1 \qquad \forall\ t \in T \qquad (4.1.6)$$

$$\sum_{i \in N} SN_{it} \ = 1 \qquad \forall\ t \in T \qquad (4.1.7)$$

$$\sum_{j \in N} Z_{ijt} \ - \sum_{j \in N} Z_{jit} \ = \ BN_{it} - SN_{it} \qquad \forall\ i \in N, t \in T \qquad (4.1.8)$$

$$Z_{jit} + Z_{ijt} \leq (I_{ij} + \sum_{p=1}^{t} Y_{ijp}) \qquad \forall\, i \in N, j \in N, t \in T: i < j \qquad (4.1.9)$$

$$Y_{ijt} + I_{ij} \leq 1 \qquad \forall i \in N, j \in N, t \in T: i < j \qquad (4.1.10)$$

$$\sum_{t \in T} Y_{ijt} \leq 1 \qquad \forall\, i \in N, j \in N : i < j \qquad (4.1.11)$$

$$Y_{ijt} \leq Z_{ijt} + Z_{jit} \qquad \forall\, i \in N, j \in N, t \in T: i < j \qquad (4.1.12)$$

$$\sum_{i \in N} \sum_{j \in N} t_{ij} * Z_{ijt} + \sum_{i \in N} \sum_{j \in N: i < j} W_{ij} * Y_{ijt} \leq Emax_t \qquad \forall\; t \in T \qquad (4.1.13)$$

$$v_{it} - v_{jt} + |DL| * Z_{ijt} \leq |DL| - 1 \qquad \forall i \in N, j \in N, t \in T \qquad (4.1.14)$$

$$U_{it} \in \{0,1\} \qquad \forall i \in N, t \in T \qquad (4.1.15)$$

$$Z_{ijt} \in \{0,1\} \qquad \forall i \in N, j \in N, t \in T \qquad (4.1.16)$$

$$SN_{it} \in \{0,1\} \qquad \forall i \in N, t \in T \qquad (4.1.17)$$

$$BN_{it} \in \{0,1\} \qquad \forall i \in N, t \in T \qquad (4.1.18)$$

$$Y_{ijt} \in \{0,1\} \qquad \forall i \in N, j \in N, t \in T \qquad (4.1.19)$$

Constraint (4.1.1) ensures that, a critical node is visited exactly once, and in constraints (4.1.2) and (4.1.3) it is guaranteed that, a critical node is considered to be visited just in the case a vehicle visits this node. Constraint (4.1.4) specifies the initial departure point of the RESCUE which is the pre-specified supplier node and in constraint (4.1.5) it is implied that, the last stop of the RESCUE for a period is the departure point of next period. In constraints (4.1.6) and (4.1.7), it is specified that there exists only one departure point and one terminal point for RESCUE in a period. Constraint (4.1.8) coordinates the departure and terminal points for the RESCUE according to the period base start and end points. Constraint (4.1.9) implies that, for a period, an arc is convenient to traverse if the arc is initially open or the blockage on the arc is eliminated by removing debris on the arc until that period. Constraint (4.1.10) implies that, in any period, debris removal operation for an arc is restricted if the arc is already open to traverse. Since the re-blockage of arcs is not the case for our problem, constraint (4.1.11) ensures that debris is removed at most once over an arc. In constraint (4.1.12), it is expressed that, it is unjustifiable to remove debris on an arc if the vehicle does not traverse along it. Constraint (4.1.13) guarantees that the total effort to traverse along arcs with the required effort to unblock the arcs in a period does not exceed the total capacity of the specified period. Constraint (4.1.14) removes illegal sub tours within each period. Finally, constraints (4.1.15)-(4.1.19) are the domain constraints.

4.2. MOD-2: Minimize Distance Travelled

Another variation of MOD-1 which intends to minimize the total distance travelled by the RESCUE is as follows:

$$\mathbf{minimize} \sum_{\boldsymbol{i} \in \boldsymbol{N}} \sum_{\boldsymbol{j} \in \boldsymbol{N}} \sum_{\boldsymbol{t} \in \boldsymbol{T}} \boldsymbol{t_{ij}} * \boldsymbol{Z_{ijt}}$$

subject to

$$\textstyle\sum_{i \in DL} \sum_{t \in T} \; t * U_{it} \quad \leq allowedTimeLimit \qquad (4.2.1)$$

(4.1.1 - 4.1.19)

All parameters and decision variables of MOD-2 are identical with MOD-1.

Different from MOD-1, the objective of this model is to transport emergency relief supplies to the critical nodes by having the minimum possible distance travelled by RESCUE.

All constraints are identical with MOD-1, except (4.2.1), which is the objective function of MOD-1, it comes along a constraint in MOD-2 which restricts the total visiting time of the critical nodes in a specified boundary, which is indicated as $allowedTimeLimit$, defined in terms of minutes.

Both models include $O(n^3)$ variables and $O(n^3)$ constraints where n= |N|.

However the periodic structure of these models does not allow partial debris removal in different time periods. Namely, it is not feasible to remove half of the debris in current period and continue for the remaining debris in the next period. Instead of that, models can unblock an arc if the remaining capacity of a period is sufficient. For that reason, since vehicle remains idle and waits for the next period for some cases, the efficient usage of resources is not the case, and additionally the models are not realistic for such a post disaster environment. Moreover, the preliminary analysis of the models shows that, these two models are cumbersome in terms of their CPU times since the periodic structure of the models brings on an additional index to some variables.

Therefore, we developed a new model which avoids the periodic structure, and gives better results in terms of resource efficiency.

4.3. MOD-3: Minimize Total Effort

Different than the previous models, MOD-3 avoids the periodic structure and briefly determines the visiting order of critical nodes and the travel path between two consecutive critical nodes, by considering the blocked roads as well. The objective of the model is to minimize the total effort that is spent for both travelling along paths and the debris removal effort for blocked arcs on these paths. The model separately considers the travel and debris removal efforts. The total effort spent is calculated by considering the travel effort until all critical nodes have been visited. Then, the required debris removal effort is added to the total effort spent. In order to come up with a mathematical model to the problem, we define the following decision variables:

TT : Total travel time until all critical nodes have been visited.

Y_{ij}= 1, if RESCUE visits the critical node $j \in DL$ right after the critical node $i \in DL$, and 0 otherwise.

X_{ijkl}= 1, if RESCUE uses arc (k, l) while traversing from the critical node $i \in DL$ to critical node$j \in DL$, and 0 otherwise.

C_{ij} is the cost(time) of traveling from critical node $i \in DL$ to critical node $j \in DL$, solely in terms of the traversal time. That is, the time effort to remove debris, if necessary, is not included in this value.

B_{kl}=1, if the debris on arc (k, l) is removed, and 0 otherwise.

Finally, P_i stands for the visiting time of critical node $i \in DL$ (again excluding the debris removal time).

The mathematical model that minimizes the total effort used until the visitation of all the critical nodes is completed is as follows:

$$\textbf{minimize } \boldsymbol{TT} + \sum_{\boldsymbol{k,l \in r : k<l}} \boldsymbol{B_{kl} * W_{kl}}$$

subject to

$$\textstyle\sum_{J \in DL \cup SL : i \neq j} Y_{ji} = 1 \qquad \forall i \in DL \tag{4.3.1}$$

$$\textstyle\sum_{J \in DL \cup SL : i \neq j} Y_{ij} = 1 \qquad \forall i \in DL \tag{4.3.2}$$

$$\textstyle\sum_{J \in DL} \quad Y_{ij} = 1 \qquad \forall i \in SL \tag{4.3.3}$$

$$\textstyle\sum_{l \in N} X_{ijkl} - \sum_{l \in N} X_{ijlk} = Y_{ij} \qquad \forall\, i,j \in DL \cup SL\,,\ k = i \tag{4.3.4}$$

$$\textstyle\sum_{l \in N} X_{ijkl} - \sum_{l \in N} X_{ijlk} = -Y_{ij} \qquad \forall\, i,j \in DL \cup SL\,,\ k = j \tag{4.3.5}$$

$$\textstyle\sum_{l \in N} X_{ijkl} - \sum_{l \in N} X_{ijlk} = 0 \qquad \forall\, i,j \in DL \cup SL\,, k \in N, k \neq i, k \neq j \tag{4.3.6}$$

$$P_i = 0 \qquad i \in SL \tag{4.3.7}$$

$$P_j \geq P_i + C_{ij} - M * \left(1 - Y_{ij}\right) \qquad \forall\, i \in DL \cup SL, j \in DL \tag{4.3.8}$$

$$TT \geq P_i \qquad \forall i \in DL \tag{4.3.9}$$

$$\textstyle\sum_{k,l \in N} X_{ijkl} \leq Y_{ij} * |N| * |N| \qquad \forall\, i,j \in DL \cup SL \tag{4.3.10}$$

$$C_{ij} = \textstyle\sum_{k,l \in N} X_{ijlk} * t_{kl} \qquad \forall\, i,j \in DL \cup SL \tag{4.3.11}$$

$$\textstyle\sum_{i,j \in DL \cup SL} X_{ijkl} + \sum_{i,j \in DL \cup SL} X_{ijlk} \leq (B_{kl} + I_{kl}) * |DL| * |DL| \qquad \forall k, l \in N : k < l \tag{4.3.12}$$

$$TT \geq 0 \tag{4.3.13}$$

$$P_i \geq 0 \qquad \forall\, i \in DL \cup SL \tag{4.3.14}$$

$$C_{ij} \geq 0 \qquad \forall\, i,j \in DL \cup SL \qquad (4.3.15)$$

$$X_{ijkl} \in \{0,1\} \qquad \forall\, i,j \in DL \cup SL\,, k \in N, l \in N \qquad (4.3.16)$$

$$B_{kl} \in \{0,1\} \qquad \forall\, k \in N, l \in N \qquad (4.3.17)$$

$$Y_{ij} \in \{0,1\} \qquad \forall\, i,j \in DL \cup SL \qquad (4.3.18)$$

The objective minimizes the total travelling time plus the total time spent for the debris removing operations until all critical nodes are visited.

Constraints (4.3.1),(4.3.2) and (4.3.3) are the assignment constraints where constraints (4.3.1) and (4.3.2) together form a visiting order to critical nodes which starts and ends at the supply node and visits each critical node one after another. It is worth to note that, even the constraints imply that the vehicle returns to the supply node, the objective function of the problem considers the path until all critical nodes have been visited. With constraint (4.3.3) it is ensured that RESCUE visits exactly one critical node right after it departs from supply node. Constraints (4.3.4), (4.3.5), and (4.3.6) establish a directed path between two consecutive critical nodes where the directed path is free to include intermediate non-critical nodes. Constraint (4.3.7) implies that in the first place RESCUE is positioned on the supply node. Constraint (4.3.8) assigns visiting time of critical nodes; without considering the time spent to remove debris on the blocked arcs, if any. Debris removal efforts are taken into account by the objective function. Additionally, constraint (4.3.8) eliminates sub tours between critical nodes and it is worth to note that sub tours are allowed between intermediate nodes appearing in different critical path segments. The objective function together with the constraint (4.3.9) minimize the most disadvantageous node’s visiting time. Constraint (4.3.10) guarantees that if there is no visit between a pair of critical nodes, there is no directed path between them. Constraint (4.3.11) correctly calculates the total time spent to travel. Constraint (4.3.12) guarantees that, it is possible to travel along an arc if it is already open or the debris on it is removed. Constraints (4.3.13) – (4.3.15) imply the non-negativity constraints, and constraints (4.3.16)-(4.3.18) are the domain constraints.

The proposed mathematical model has $O(n^4)$ variables and $O(n^3)$ constraints.

Chapter 5

Heuristic Solution Methodology

As the data enlarges and the number of critical nodes increases, it becomes harder to reach the optimal solution in reasonable amount of times. It takes hours to find the optimum for certain instances. However, by the nature of the problem it is necessary to make a decision immediately and it is impossible to wait for long durations. Therefore, we decided to develop heuristic methodologies within the scope of optimal vs. speed trade-off, which can find good solutions expeditiously without going too far away from the optimal solution.

For that purpose, we developed a fast constructive heuristic solution methodology that is based on Dijkstra’s algorithm. To have better optimality gaps, we also applied an improvement heuristic methodology which can be considered as a variation of 2-opt algorithm [49]. The improvement heuristic uses the output of the constructive heuristic as its input and provides better results.

5.1. The Constructive Heuristic

The heuristic starts from the source node and it applies Dijkstra's algorithm until all the critical nodes have been visited.

In this context, the algorithm first finds the shortest path tree which is rooted at the source node to other nodes until a critical node, say *j*, is reached. Then, having found the closest critical node *j*, travels along the path from source to node *j*. It unblocks the blocked arcs on this path, if any. Since after a blocked arc is opened it remains open, the debris removal cost for such an arc is not paid if it is used again. Then the algorithm considers the node *j* as the new source node and applies the same algorithm again and again until there is no critical node which is not visited.

The objective of the algorithm is to find a path for the vehicle that needs minimum effort until all critical nodes become visited. This effort involves both travel times and unblocking efforts of arcs.

In the following subsection the pseudo-code of the algorithm is presented.

5.1.1. Pseudo-code of the Constructive Heuristic

```
algorithm constructive
begin
    initialize
        $C_{uv} \coloneqq t_{uv} + W_{uv} * (1 - I_{uv})$ for all $u$ and $v$ in N
        Marked_Critical_Nodes :={source};
    while Marked_Critical_Nodes ≠ Critical_Nodes  do
          Apply Dijkstra's algorithm to find the shortest path tree rooted at "source"
          until a critical node is marked.
          Let node j be the closest node to source not in set Marked_Critical_Nodes
          source:= j;
          Marked_Critical_Nodes :=Marked_Critical_Nodes$\cup$ $\{j\}$;
          if a blocked arc (u,v) is traversed then
               $I_{uv} \coloneqq 1\ and\ I_{vu} \coloneqq 1$;
               update $C_{uv} \coloneqq t_{uv} + W_{uv} * (1 - I_{uv})$ for all $u$ and $v$ in N
end;
```

5.1.2. Flow Chart of the Constructive Heuristic

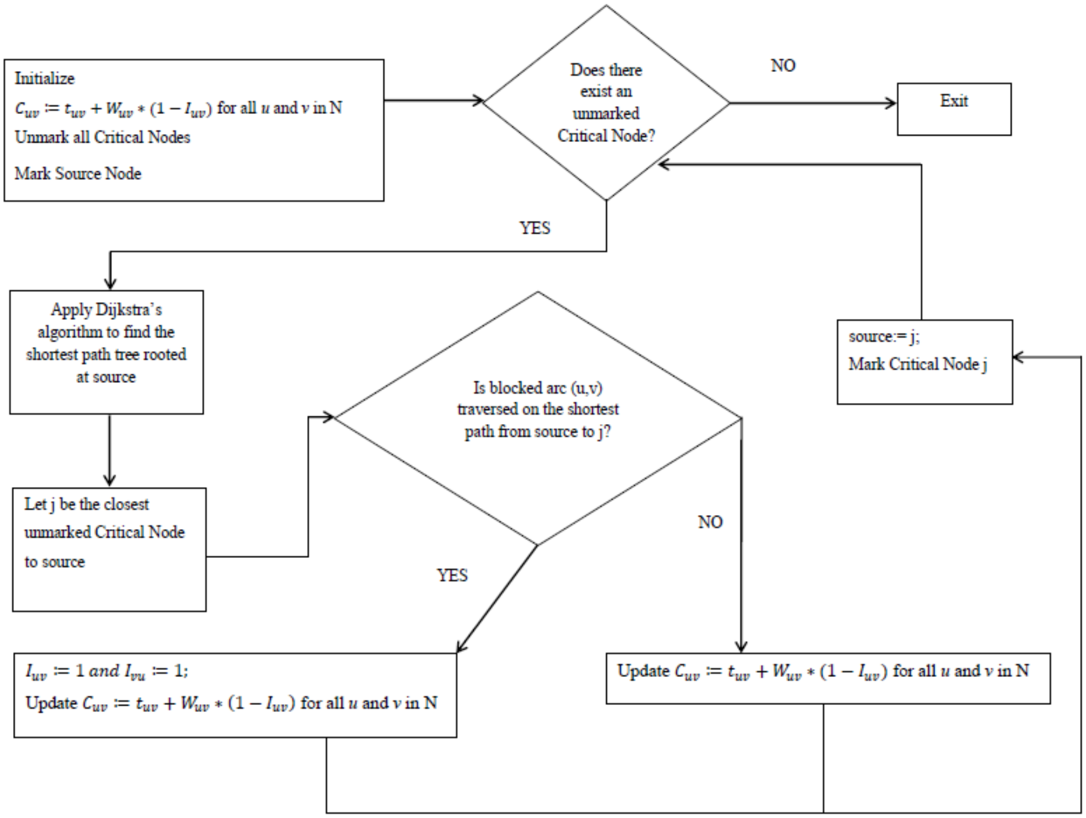


Figure 5-1: Flow chart of the constructive heuristic

5.2. The Improvement Heuristic with 2-opt

To improve the solution quality, by having the output of the constructive heuristic as input, we apply the 2-opt algorithm. The output of the constructive heuristic gives a path that visits all the critical nodes with some intermediate nodes, if necessary.

Let $start \rightarrow i_1 \rightarrow i_2 \ldots \rightarrow i_{h-1} \rightarrow i_h \rightarrow \ldots \rightarrow i_{e-1} \rightarrow i_e \rightarrow i_{e+1} \rightarrow \ldots \rightarrow i_k$ be the output path of the constructive heuristic. The 2-opt algorithm randomly selects two nodes, i_h and i_e, from this path. It preserves the same order for the nodes from start to i_{h-1} and from i_{e+1} to the end. It reverses the order for the nodes from i_h to i_e, namely, the resulting order is as follows:

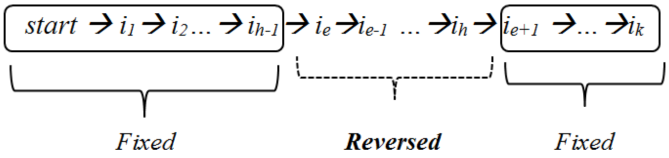


Figure 5-2: Example of the 2-opt Heuristic

By applying this procedure until no improvement is obtained, we get another path which contains the same nodes but in a different order with a better objective value.

To clarify the procedure, Figure 5-3 illustrates an example of 2-opt by using a feasible solution of the problem. Upper figure is the given route, where the node visited twice, node 45, is duplicated for clarification. Nodes 22 and 43 are chosen to apply 2-opt algorithm and in the subjacent figure, output of the 2-opt improvement heuristic is illustrated. The cross on the arc (16, 22) states the blockage on this arc. Certainly, to follow the path, the debris on it is removed and the blockage is resolved. Straight lines indicate the fixed arcs whereas dashed lines represent the arcs that the order of the nodes between them is changed. Namely, after RESCUE departs from node 16, it visits node 43 instead of 22, and follows the path 43-45-33-21-41-21-22, which is exactly the reverse order of the original path between nodes 22 and 43, that is shown in lower part of the Figure 5-3. Then right after RESCUE visits node 22, it visits node 45 and then follows the original path. To take into account problem characteristics, it is necessary to investigate the arcs that are used for each generated path. The total cost of the route is recalculated since the 2-opt algorithm may possibly replace a blocked arc with an unblocked arc or vice versa. If a blocked arc is included in the resulting route, the debris is removed to unblock this arc and corresponding debris removal cost is added to the objective. In the new instance, instead of arcs (16, 22) and (43, 45), the arcs (16, 43) and (22, 45) exist. As it can be seen from the figure, the arc (22, 45) is blocked, so it is necessary to remove debris on this arc for the new instance. It is worth to note that, the re-blockage of arcs is not possible. Therefore, after a blockage of an arc is resolved, it remains open forever. To represent this issue in the model, the *I*matrix is updated in the way that the 0 value of the arc, whose blockage is removed, is changed into1.

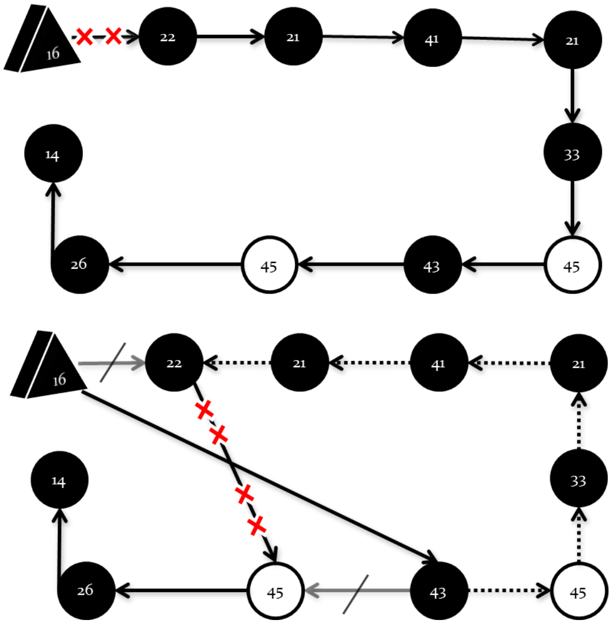


Figure 5-3: Application of the improvement heuristic with 2-opt to a feasible instance of the problem

Figures 5-4 and 5-5 show the former and resulting routes after 2-opt are applied, respectively, for the instance mentioned above.

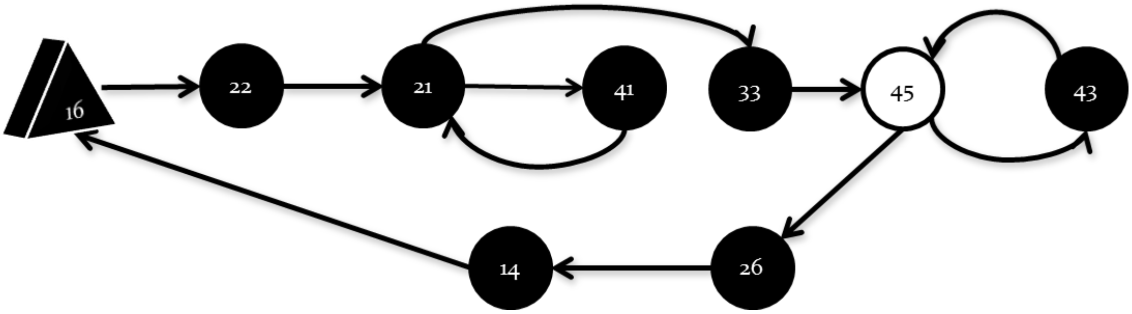


Figure 5-4: Output of the constructive heuristic

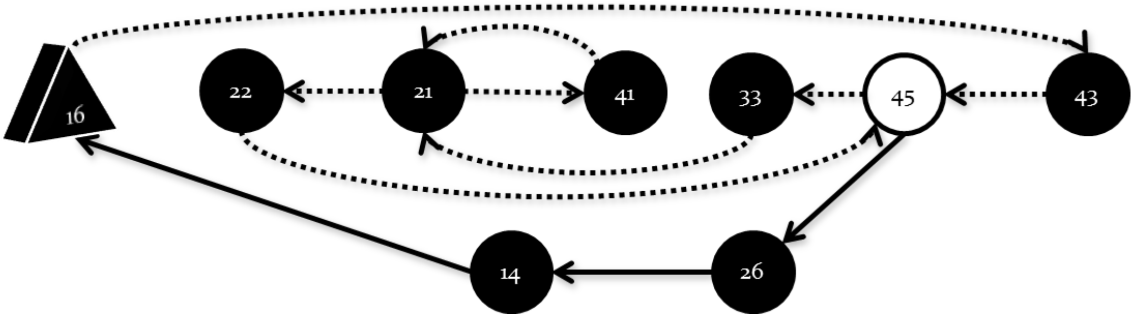


Figure 5-5: Output of the improvement heuristic with 2-opt

In the following subsection the pseudo-code of the algorithm is presented.

5.2.1. Pseudo-code of the Improvement Heuristic with 2-opt Algorithm

```
algorithm 2-opt
begin
    initialize
        $route$ := a feasible solution of the problem
        $bestDistance$ := objective value of route
    while improvement obtained in the objective do
        Apply 2-opt algorithm to $route$ to find $newRoute$
        Calculate newcost:= cost of newRoute
        if $newcost < bestDistance$ then
            $route$ = $newroute$;
            $bestDistance$= $newcost$;
end;
```

5.2.2. Flow Chart of the Improvement Heuristic with 2-opt

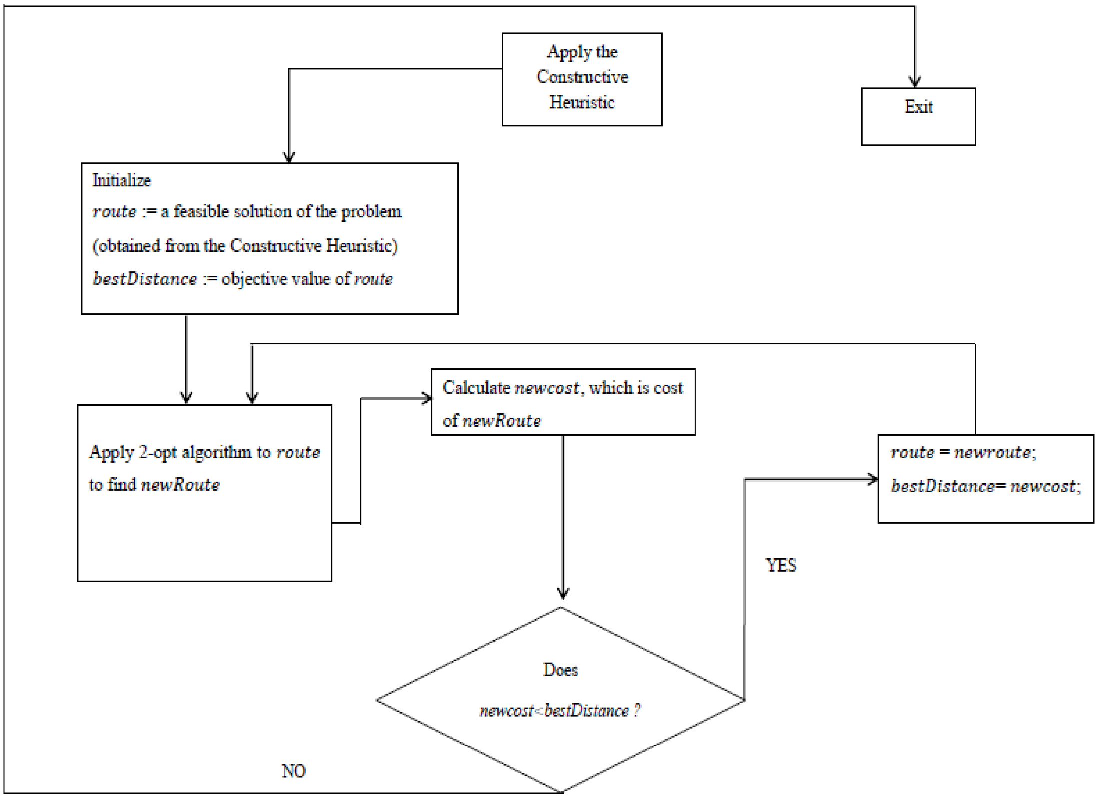


Figure 5-6: Flow chart of the improvement heuristic with 2-opt

To clarify the application procedure of these two heuristics, the following chart illustrates the process conceptually:

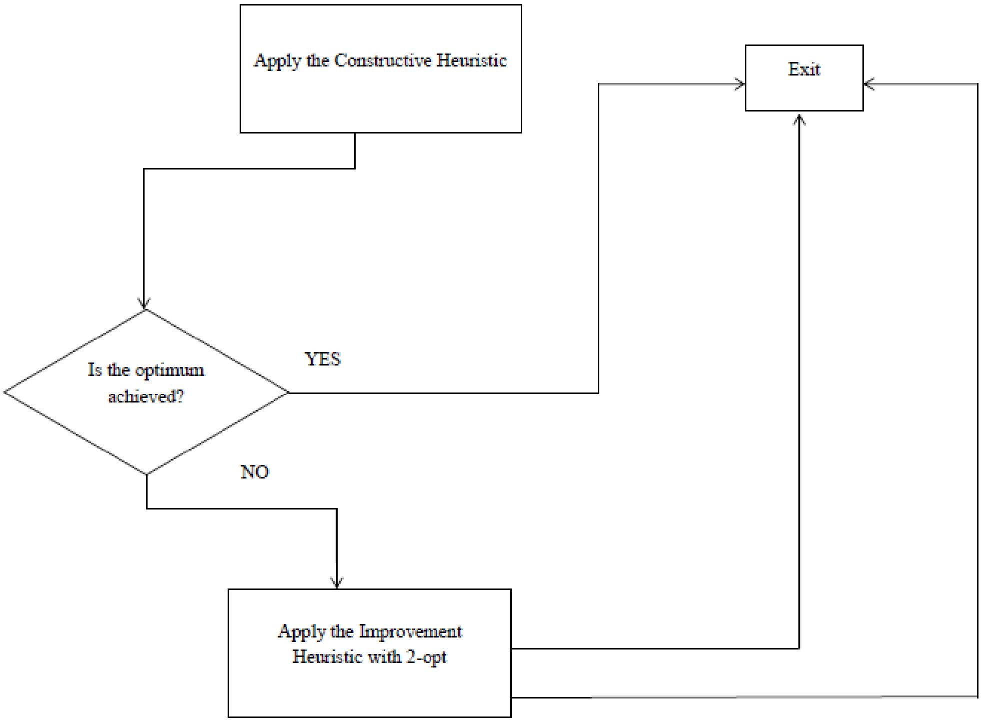


Figure 5-7: Application procedure of the heuristics

First, the constructive heuristic is applied and for the instances that the optimum is achieved, the process is over. However, if there exists such instances that the optimum is not reached yet, the improvement heuristic with 2-opt is applied. The improvement heuristic could find optimum for these instances, provides improvement in the objective values or it cannot achieve any improvement. In any case, the process terminates.

Chapter 6

Data and Computational Results

6.1. Data

To measure the effectiveness of the developed model, we test it by using two different data sets from Turkey which are based on Kartal and Barkırköy districts of İstanbul. Kartal is a relatively small data whereas Bakırköy is larger. For detailed information about these data sets, see Kılcı [50]. The neighbourhood units, namely the nodes, are investigated and the ones that contain school or hospital are selected as the critical ones. Thus, there exist 7 critical nodes in Kartal, and 15 critical nodes in Bakırköy. Additionally, Marmara Region Disaster Centre of the Turkish Red Crescent is located in Kartal, and a disaster coordination centre exists in Bakırköy which are extremely adequate to serve as the supplier of their respective districts.

The following table summarizes the features of these data sets. As it is stated in the Table 6-1, both Kartal and Bakırköy data is arranged in such a way that the distance matrices are symmetric, namely, $t_{ij} = t_{ji}$, and satisfy triangle inequality.

Table 6-1: Features of the data sets

	Kartal	**Bakırköy**
#of nodes	45	73
Symmetric distance matrix and triangle inequality requirement	Yes	Yes
Supply Node (**node number**, *name of place*)	***16****- Marmara Region Disaster Centre of the Turkish Red Crescent*	***7-*** *Disaster Coordination Centre*
Total #of critical nodes	7	15
(**node numbers**)	***14,21,22,26,33,41,43***	***16,17,18,19,20,21,22,55,515,47,65,36,34,67***
#of schools only	3	8
(node numbers)	***14,21,22***	***55, 5, 15, 47, 65, 36, 34, 67***
#of hospitals only	4	7
(node numbers)	***26,33,41,43***	***16,17,18,19,20,21,22***

The following maps show the locations of suppliers and critical nodes in Kartal and Bakırköy, where the red triangle represents the supplier, yellow circles illustrate the critical nodes with schools and green circles illustrate the critical nodes that include hospitals.

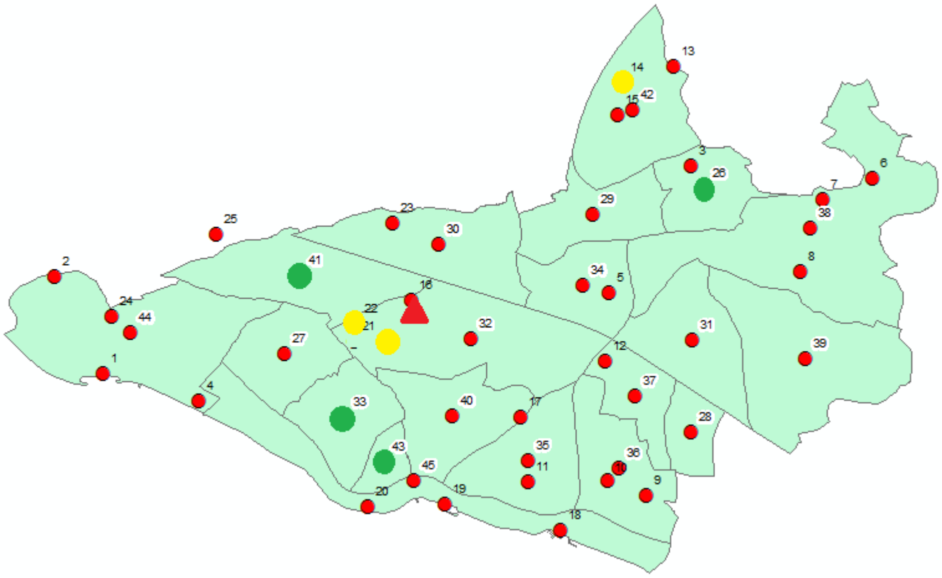


Figure 6-1: The location of supplier and critical nodes in Kartal

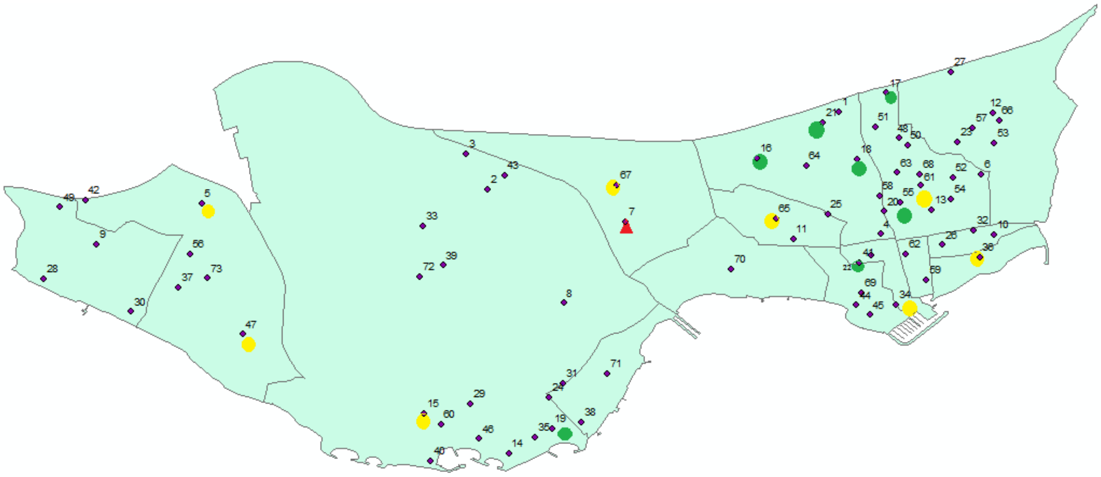


Figure 6-2: The location of supplier and critical nodes in Bakırköy

The node-to-node distance matrix of Kartal and Bakırköy data is utilized to calculate the travel time of a vehicle between nodes k and l, namely, t_{kl} matrix. It is assumed that the vehicle's speed is about 20km/hour which is equivalent to 334m/min and by dividing the distances to the vehicle's speed, t_{kl} matrix is formed. Then, we forced that t_{kl} matrix become a symmetric matrix and satisfies triangle inequality by Floyd-Warshall algorithm [51].

When it comes to constitute I_{kl} matrix, which indicates the blockages of the arcs and the W_{kl} matrix that stands for the required debris removal effort of arcs, the potential earthquakes are specified according to their severities. To this end, we constitute 4 groups of severities. We varied severity of earthquakes (SOE) from 1 to 4 where (4) is the most severe one. Following table illustrates the intervals of blockage ratios of arcs according to severity of earthquake. According to this classification and the corresponding BAR values stated in Table 6-2, zero values are randomly assigned to the arcs of I_{kl} matrix for those arcs which are blocked by debris. So we have higher number of blocked arcs for more severe earthquakes.

Table 6-2: SOE and corresponding BAR values

SOE	Blocked Arc Ratio(BAR)
SOE=1	0%-20%
SOE=2	20%-50%
SOE=3	50%-80%
SOE=4	80%-100%

Since the required effort to remove debris from the blocked arcs is directly related to the severity of earthquake, the required effort is calculated so as to have higher values for more severe earthquakes. To observe how the debris removal effort affects the computational results, we calculate W_{kl} in two different ways. Both of them are proportional to the severity of earthquake and the length of the relevant arc but one of them implies larger amount of debris removal effort.

The following equations show how W_{kl} is calculated:

$$W_{kl} = SOE * t_{kl} + U[0, \mathrm{argmax}_{i,j}\, t_{ij}]$$

$$W_{kl}{}' = SOE * t_{kl}$$

In this context, we create 20 different instances of Kartal and Bakırköy data, each, for our computational study. For each class of severity of earthquake there exist 5 different instances whose blocked arc ratios and accordingly the number of blocked arcs are equivalent but the arcs which are blocked are different. Additionally, to see the effect of debris removal effort to the results, these 20 instances are considered with two different debris removal effort (W_{kl}, W'_{kl}) values, which are calculated as mentioned above.

We repeat our experiments for three times, where critical nodes to visit are chosen as only hospitals, only schools and both hospitals and schools.

6.2. Computational Analysis

In this subsection, the computational results of the MOD-3 and the heuristics are discussed. The computational experiments of the MOD-3 are conducted with CPLEX 12.4.0.0 and a 4 x AMD Opteron Interlagos 16C 6282SE 2.6G 16M 6400MT computer, running under Linux operating system and the heuristic algorithms are coded in Java 1.6.0_23 on the computer.

The following table depict the SOE, BAR and number of blocked arcs setting of the instances, which are used in the computational experiments. Table 6-3 stands for Kartal instances and Bakırköy instances are depicted in a similar manner in Table 6- 4.

Table 6-3: Kartal instances and the corresponding SOE, BAR, and #of blocked arc settings

***Kartal* Instances**	**SOE**	**BAR**	**#of blocked arcs**
K-1...K-5 K-1'...K-5'	1	0.125	124
K-6...K-10 K-6'...K-10'	2	0.445	441
K-11...K-15 K-11'...K-15'	3	0.58	574
K-16...K-20 K-16'...K-20'	4	0.819	806

Table 6-4: Bakırköy instances and the corresponding SOE, BAR, and #of blocked arc settings

***Bakırköy* Instances**	**SOE**	**BAR**	**#of blocked arcs**
B-1...B-5 B-1'...B-5'	1	0.19	500
B-6...B-10 B-6'...B-10'	2	0.23	613
B-11...B-15 B-11'...B-15'	3	0.54	1423
B-16...B-20 B-16'...B-20'	4	0.82	2160

It is worth to note that, when the SEO =1, the number of blocked arcs for all computational experiments in Kartal data are 124, however, their locations differ, where each different configuration is a different instance. The same goes for other classes of SOE’s. To test the effects of this issue, we determined to have 5 instances for a class of SOE. Such as, when SOE=1, the corresponding instances are K-1, K-2… K-5.

In each row, the instances where the corresponding debris removal effort is greater (W_{kl}) is shown on the top, such as K-1…K-5; whereas the instances where the debris removal effort is smaller $(W_{kl}{}')$ stand right below them, (K-1'…K-5').

For both type of instances where debris removal effort is greater or smaller, same setting of SOE, BAR and number of blocked arcs are used.

The computational results of the experiments are summarized in the Tables 6-5, 6-6, 6-7 and 6-8. The total effort spent to visit all critical nodes in the optimum solution for the instances that optimum is achieved within the 4 hour time limit, or the gaps from optimal; CPU times and the arcs that the debris on them is removed in the resulting solution are shown. Since it is intended to observe the performance of the model for each class of SOE, a set of 5 instances, where the number of blocked arcs is the same but their locations are different are used in the computational experiments. The names of the instances are stated in the top of the tables and the corresponding settings, given in the Table 6-3 and 6-4, are used for each instance group. Additionally, to see the effect of the number of critical nodes, a column that indicates it, is stated. Moreover, as it is specified before, two different debris removal efforts settings, $(\boldsymbol{W_{ij}}, \boldsymbol{W'_{ij}})$, are used for the same instances. The results based on this difference are illustrated in different tables. Table 6-5 stands for the instances with $\boldsymbol{W_{ij}}$, and the results of the instances with $\boldsymbol{W_{ij}}'$ are summarized in Table 6-6. Finally, to analyze the performance of the model on different data sets, tables for the results from Kartal and Bakırköy are organized.

Table 6-5 summarizes the computational results for the Kartal instances where the debris removal effort is greater. Table is divided into 4 parts, where each part illustrates the results of the 5 set of instances belonging to a class of SOE, as reported in Table 6-3. In each part, for a class of SOE; the results for different amount of critical nodes are illustrated. Such as, for the instances K-1…K-5, earthquake severity is 1, namely, SOE=1, and experiments are conducted for 7, 4, and 3 critical nodes for this SOE class.

Additionally, since the instances K-1… K-5 differ in terms of the locations of the blocked arcs; each experiment is repeated 5 times, for a fixed critical node set. To clarify, when the number of critical nodes is 7, the corresponding 5 rows in the table coincide with the results of the 5 instances, K-1… K-5, respectively. Concisely, Table 6-5 and 6-6 illustrate the computational results of Kartal data; where one of the two debris removal effort settings is dealt with in each table, Tables 6-7 and 6-8 present results for Bakırköy data in a similar fashion.

Table 6-5: Model performances of Kartal instances with greater debris removal effort (Wij)

instances: K-1...K-5				instances: K-6...K-10			
#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs
7(all)	44	221.41	-	7(all)	48	223.26	-
	43	186.73	-		50	235.96	-
	44	190.82	-		51	270.41	-
	43	178.4	-		49	225.46	-
	43	183.2	-		48	261.39	-
4(hospitals)	35	8.81	-	4(hospitals)	38	15.39	-
	35	10.02	-		42	8.4	-
	36	6.27	-		40	9.09	-
	35	12.08	-		42	10.75	-
	35	11.69	-		39	7.04	-
3(schools)	30	1.4	-	3(schools)	29	2.52	-
	30	1.47	-		29	2.54	-
	30	1.61	-		30	2.28	-
	29	1.44	-		35	3.51	-
	29	1.44	-		30	3.04	-

instances: K-11...K-15				instances: K-16...K-20			
#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs
7(all)	53	315.37	-	7(all)	109	5167.18	(21,22)
	63	495.63	-		82	3319.8	-
	68	381.4	-		110	9136.87	(21,22)
	46	196.96	-		90	2915.86	(21,22)
	47	186.72	-		101	4541.84	(21,22),(22,41),(33,43)
4(hospitals)	35	4.87	-				
	53	9.51	-	4(hospitals)	84	22.32	-
	51	8.32	-		67	20.23	-
	38	8.69	-		70	38.91	(33,43)
	40	8.41	-		70	49.49	-
3(schools)	40	4.34	-		88	64.54	-
	35	2.98	-	3(schools)	57	6.62	(21,22)
	30	3.11	-		55	4.66	-
	35	2.85	-		68	7.08	(21,22)
	29	2.51	-		55	4.76	(21,22)
					45	3.62	(21,22)

Table 6-6: Model performances of Kartal instances with smaller debris removal effort (Wij')

instances: K-1'…K-5'				**instances: K-6'…K-10'**			
#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs
7(all)	44	219.89	-	7(all)	48	231.4	-
	43	213.25	-		49	210.34	(33,43)
	44	192.61	-		51	282.31	-
	43	156.37	-		49	259.62	-
	43	152.9	-		48	210.25	-
4(hospitals)	35	7.16	-	4(hospitals)	38	9.2	-
	35	10.12	-		41	12.63	(33,43)
	36	9.98	-		40	10.28	-
	35	7.02	-		42	13.18	-
	35	7.31	-		39	20.43	-
3(schools)	30	2.37	-	3(schools)	29	1.57	-
	30	1.77	-		29	2.75	-
	30	2.76	-		30	3.38	-
	29	1.86	-		32	3.8	(21,22)
	29	1.85	-		30	1.9	-
instances: K-11'…K-15'				**instances: K-16'…K-20'**			
#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs
7(all)	51	214.86	(21,22)	7(all)	97	4864.74	(3,26),(21,22),(43,45)
	63	316.91	-		78	2422.78	(22,41)
	67	336.34	(27,33)		95	3791.44	(14,15),(21,22),(33,43)
	46	198.85	-		81	2794.34	(21,22),(43,45)
	47	184.67	-		80	3258.78	(21,22),(22,41),(33,43)
4(hospitals)	35	7.84	-				
	53	8.37	-	4(hospitals)	80	54.89	(16,32),(43,45)
	50	16.5	(33,43)		67	17.18	(22,41)
	38	9.42	-		63	32.48	(33,43)
	40	13.71	-		70	26.91	-
3(schools)	38	3.42	(21,22)		73	14.96	(21,22),(22,41),(33,43)
	33	3.12	(21,22)	3(schools)	50	4.88	(21,22)
	30	2.93	-		51	3.48	(21,22)
	35	2.03	-		61	5.24	(21,22)
	29	2.65	-		48	4.73	(21,22)
					38	4.51	(21,22)

Since the number of blocked arcs increases as the SOE grows, it becomes compulsory to unblock arcs in order to visit critical nodes for some instances with greater SOE values. When the experiments are repeated for identical instances with different debris removal effort requirements, such as K-11 and K-11', it is observed that, it is not hesitated to travel on a path that includes blocked arcs on the drive when the required effort is less.

As it is indicated before, for a class of SOE, we repeat our experiment with 5 different instances, where their blocked arc ratios, and number of blocked arcs are same, but the arcs which are blocked are different. This issue on the CPU times has a marginal effect when the number of critical nodes remains constant. Namely, for the instances K-1, K-2, K-3, K-4 and K-5, when the number of critical nodes is 7, CPU times ranges from 183 seconds to 221 seconds; for 4 critical nodes, the range is 6 seconds to 12 seconds, and when it is required to visit only 3 critical nodes, all instances are solved to optimality within 1 second.

However, the number of critical nodes to visit affects CPU times dramatically. As the number of critical nodes increases, the CPU times exponentially increases. To exemplify, for instance K-1, the CPU time is 1.4 seconds for 3 critical nodes; 8.81 seconds for 4 critical nodes, and it becomes 221.41 seconds when the critical node number scales up to 7. Additionally, together with the number of critical nodes to visit, the severity of earthquake (SOE) significantly affects the CPU times. As it is expected, for more severe earthquakes, the solution times increases. To exemplify, when the SOE = 3 , the instance K-11’ is solved in 214.86 seconds for 7 critical nodes, whereas as the SOE increases to 4, the solution time of the instance K-16’ which intend to visit 7 critical nodes scales up to 4864.74 seconds. Both K-11’ and K-16’ intends to visit 7 critical nodes but due to the differences on the corresponding SOE’s, the numbers of blocked arcs differ for these instances. Also, when the CPU times for the instances where only the debris removal effort is changed is analysed, such as K-16 and K-16’, it is observed that, the CPU times for instances with smaller debris removal efforts are smaller. This issue comes into focus for the instances where the chance to encounter with a blocked arc increases, namely, as SOE and number of critical nodes increases. Additionally, when the objective values of the instances from Table 6-5 and Table 6-6 are compared, it is shown that, the objective values are the same for the instances from both tables, where there is no need to remove debris on an arc, however, the objective values of the instances with smaller debris removal effort is smaller in the case where the debris on an arc is removed. Such as, for the instances, K-1 and K-1’, the objective values are similar for all choice of number of critical nodes, because there is no debris removed arcs for these instances. However, for the instance K-7’, with 7 critical nodes, the objective is 49 and the arc (33, 43) is unblocked. When the same instance with greater debris removal effort, K-7, is investigated, it is shown that the objective is 50, and for no arc the debris removal operation is done.

When the resulting travel paths for the instances with the same number of blocked arcs with different locations are analysed, it is realized that, generally, the locations of critical nodes are more important than the locations of the blocked arcs for less severe earthquakes. For example, as it can be seen from the resulting travel paths from the solutions of instances K-1, K-2, K-3, K-4 and K-5 (Figure 6-3), even though the vehicle follows different travel paths for these instances due to the different locations of blocked arcs, the order of visit of the critical nodes remain the same. However, as the SOE increases, the number of blocked arcs also increases, and the order of visit of critical nodes differs a lot, as in the case with K-16, K-17, K-18, K-19 and K-20 depicted in Figure 6-4.

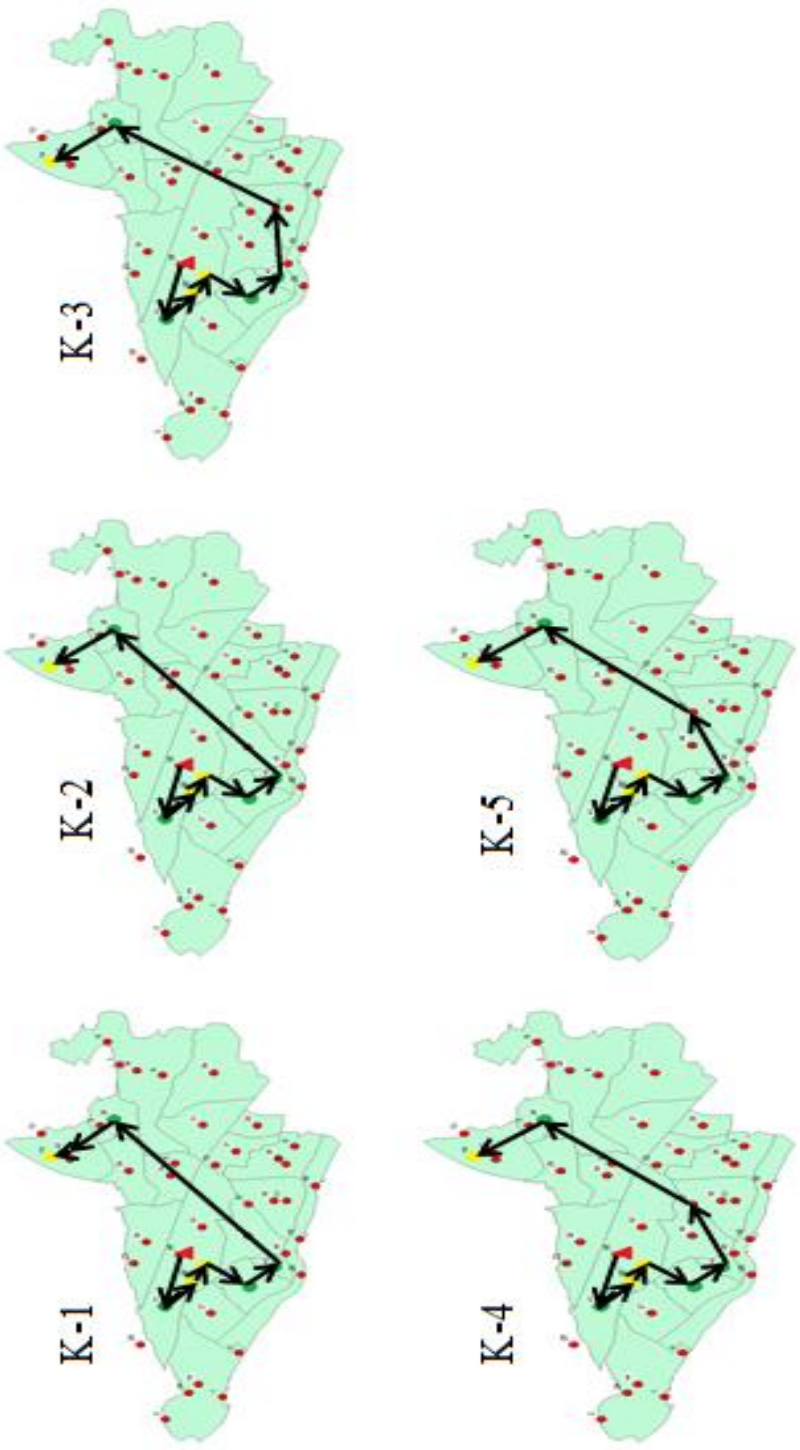


Figure 6-3: Travel path of RESCUE for instances K-1.. K-5

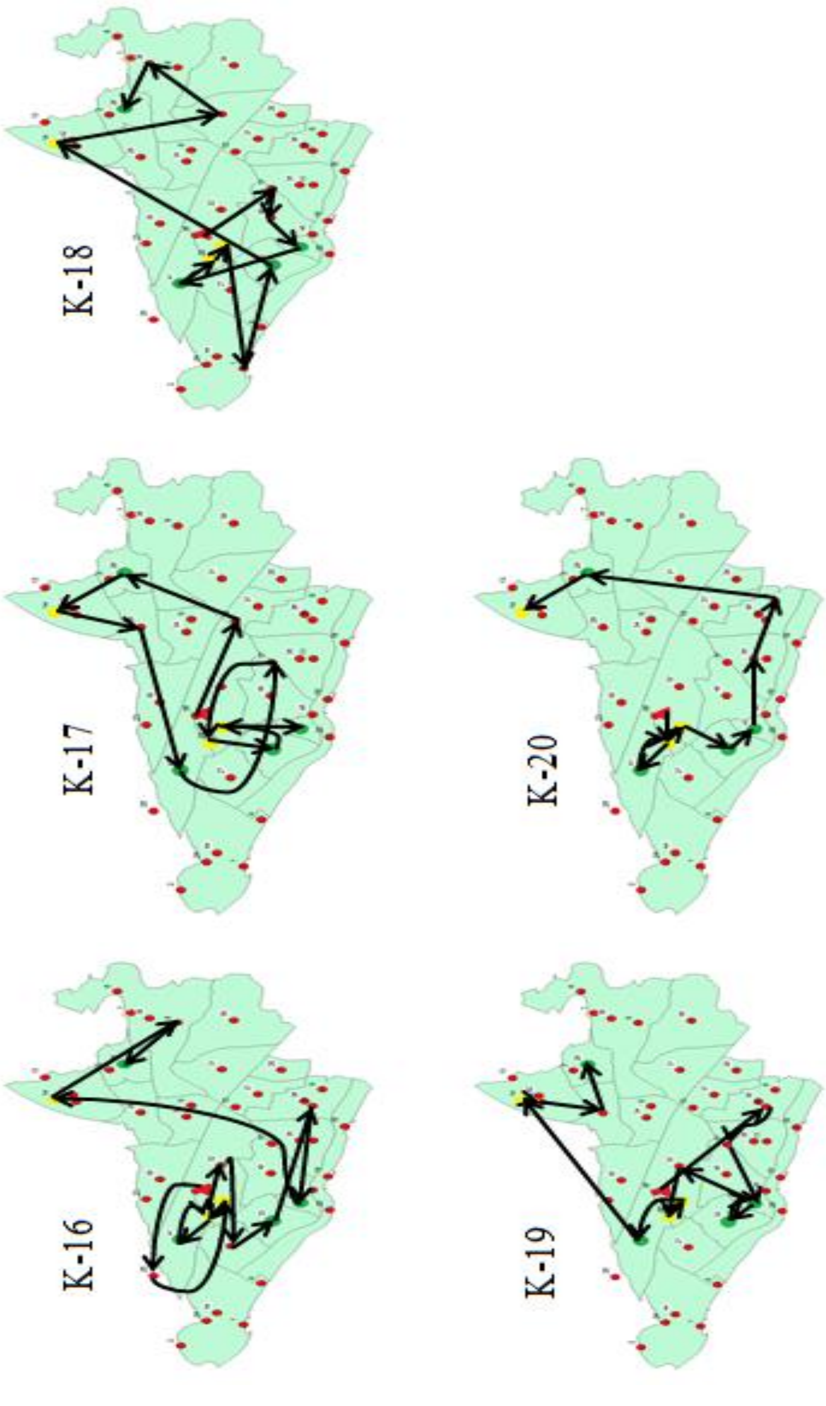


Figure 6-4: Travel path of RESCUE for instances K-16.. K-20

Similar arguments are valid for Bakırköy data as well. As the data set becomes larger, it becomes difficult to find optimum solutions for some instances. As it can be seen from the Tables 6-7 and 6-8, for the instances where it is required to visit both hospitals and schools, namely 15 nodes, the optimum cannot be reached, with 14400 second (4 hour) time bound. Additionally, as the SOE increases, it becomes harder to find the optimum for the instances where the number of critical nodes is 8 (only schools). Therefore, we developed heuristic methodologies. The following two tables summarize the results for Bakırköy instances with greater and smaller debris removal efforts, respectively.

Table 6-7: Model performances of Bakırköy instances with greater debris removal effort (Wij)

instances: B-1…B-5					**instances: B-6…B-10**				
#of critical nodes	Best Objective	cplex CPU(sec)	Gap (14400 sec)	Debris removed arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap% (14400 sec)	Debris removed arcs
15	73	14400	91.8%	-	15	81		92.6%	-
	85	14400	91.8%	-		88		90.9%	-
	75	14400	90.7%	-		77		93.5%	-
	75	14400	90.7%	-		86		93.0%	-
	80	14400	90.0%	-		76		93.4%	-
8(schools)	52	1981.32	-	-	8(schools)	52	1647.38	-	-
	61	3433.06	-	-		60	13108.26	-	-
	52	2586.69	-	-		52	2126.97	-	-
	54	2235.72	-	-		60	12207.84	-	-
	52	1753.38	-	-		52	12236.82	-	-
7(hospitals)	41	921.03	-	-	7(hospitals)	39	762.91	-	-
	38	761.61	-	-		39	686.03	-	-
	39	646.14	-	-		40	889.07	-	-
	40	821.21	-	-		42	647.82	-	-
	40	728.84	-	-		38	600.08	-	-
instances: B-11…B-15					**instances: B-16…B-20**				
#of critical nodes	Best Objective	cplex CPU(sec)	Gap%	Debris removed arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap% (14400 sec)	Debris removed arcs
15	93	14400	93.4%	-	15	182		100.0%	(19,38)
	106	14400	94.2%	-		170		100.0%	-
	107	14400	92.3%	-		169		99.1%	-
	99	14400	90.9%	-		144		100.0%	-
	104	14400	92.3%	-		145		99.5%	-
8(schools)	71	12711	-	-	8(schools)	96		71.0%	-
	80	14400	43.8%	-		78		31.7%	-
	74	13736.36	-	-		112		74.6%	-
	71	12507.97	-	-		84		45.2%	-
	77	14400	33.8%	-		87		63.2%	-
7(hospitals)	39	461.74	-	-	7(hospitals)	61	7279.04	-	-
	42	550.53	-	-		58	7653.28	-	-
	46	580.23	-	-		51	1162.77	-	-
	48	744.75	-	-		59	6006.05	-	-
	43	540.02	-	-		52	848.49	-	-

Table 6-8: Model performances of Bakırköy instances with smaller debris removal effort (Wij')

instances: B-1'…B-5'					**instances: B-6'…B-10'**				
#of critical nodes	Best Objective	cplex	Gap% (14400sec)	Debris removed arcs	#of critical nodes	Best Objective	cplex	Gap% (14400sec)	Debris removed arcs
		CPU(sec)					CPU(sec)		
15	74		91.9%		15	75		92.0%	
	84		92.9%	(7,67),(15,60)		80		90.0%	(7,67)
	78		92.3%			77		92.2%	
	80		92.5%			81		92.6%	(7,67)
	73		91.8%			76		92.1%	
8(schools)	52	1920.36			8(schools)	52	2785.56		
	58	2834.09		(7,67)		56		26.8%	(7,67)
	52	1849.6				52	2245.67		
	54	2194.44				56	12997.24		(7,67)
	52	1638.53				52	1971.06		
7(hospitals)	41	736.43		-	7(hospitals)	39	744.54		-
	38	706.57		-		39	708.93		-
	39	768.61		-		40	794.42		-
	40	649.6		-		42	729.86		-
	40	845.63		-		38	760.88		-
instances: B-11'…B-1+A625'					**instances: B-16'…B-20'**				
#of critical nodes	Best Objective	cplex	Gap% (14400sec)	Debris removed arcs	#of critical nodes	Best Objective	cplex	Gap% (14400sec)	Debris removed arcs
		CPU(sec)					CPU(sec)		
15	95		91.6%		15	135		92.6%	
	98		91.8%	(37,73)		131		100.0%	
	100		92.0%			165		95.2%	
	98		90.8%	(7,67)		134		99.3%	
	99		90.9%	(5,56)		328		99.7%	(11,25),(11,47),(13,26),(13,68),(21,64),(26,36)
8(schools)	67		34.0%	(7,67)	8(schools)	91		69.7%	(15,60)
	75		60.0%	(37,73),(55,58)		78		37.2%	
	73		57.5%	(7,67)		105		73.7%	(7,67),(15,60),(29,60)
	69		27.5%	(7,67)		79		44.1%	(7,67)
	75		50.7%	(7,67)		83		67.9%	(15,60),(37,56)
7(hospitals)	39	618.33			7(hospitals)	59	7433.39		(22,41)
	42	644.19				57	7577.35		(19,38)
	46	818.27				51	1352.84		
	48	1370.15				55	7261.47		(1,21)
	43	703.36				52	1191.86		

As it can be observed from the previous tables, generally, the mathematical model gives the optimum solutions after long durations, or it reports an optimality gap after 4 hours. The performances of the heuristic methodologies, which are developed to come up with good, feasible solutions fast for this problem, are summarized in Tables 6-9, 6-10, 6-11 and 6-12.

The solution times of the heuristics are less than seconds for each instance, therefore we do not report them. Initially, the constructive heuristic is implemented, then the 2-opt improvement heuristic is applied to the instances where the optimum is not reached by the constructive heuristic.

Table 6-9: Heuristic performance summary of Kartal instances

instances:		Constructive Heuristic			2-opt Improvement Heuristic			**Final**
K-1,…,K-20		Average	Maximum	Optimum	Average	Maximum	Optimum	**Optimum**
K-1',…,K-20'		Gap	Gap	Ratio	Gap	Gap	Ratio	**Ratio**
Kartal-with Wij	|DL|=7	2.80%	9.40%	50%	2.60%	7.50%	60%	**80%**
	|DL|=4	0.30%	2.90%	85%	2.20%	2.90%	0%	**85%**
	|DL|=3	1.20%	4.40%	70%	1%	4.40%	71.40%	**90%**
Kartal-with Wij'	|DL|=7	3.70%	18.80%	30%	3.10%	18.80%	50%	65%
	|DL|=4	1.60%	13.70%	75%	6.20%	13.70%	0%	**75%**
	|DL|=3	1.30%	5.30%	65%	1.10%	5.30%	71%	**90%**

Since, the mathematical model can find optimal solutions for all instances of Kartal, it is possible to compare the heuristic performances with the optimum solution of Kartal instances. Table 6-9 summarizes the heuristics performances for Kartal instances where the results are obtained from analysing all classes of SOE, namely, all instances from K-1 to K-20 and K-1’ to K-20’ , with different number of critical nodes. The first column under constructive and 2-opt improvement heuristic titles indicates the average gap of heuristic solution to the model’s optimum solution, second columns stands for the maximum of them, and the optimum ratio of the number of instances solved to optimality with the heuristic is illustrated in the third column. In the very last column of the table, the ratio of the instances that any heuristic finds the optimum solution is reported. Namely, the first row of the table indicates that the constructive heuristic finds optimum for 50% of the Kartal instances with Wij, where |DL|=7. Since there are 20 such instances, the constructive heuristic finds optimum for 10 of them. Then, for the remaining 10 instances, the 2-opt improvement heuristic is applied, and it gives optimum for 60% of them, which corresponds to 6 instances. In total, both heuristics finds optimum for the 16 instances, in other words, 80% of the total instances.

As it can be seen in the table, heuristic methodologies can find optimum solutions up to 90 % of Kartal instances. This indicates the success of the heuristics in terms of their solution quality, together with their ability to find such solutions in 1-2 seconds.

To observe the effect of SOE individually on the results, each quarter of the following table considers a set of 5 instances according to their belonging to a class of SOE. Namely, in the first quarter of the table, the instances K-1,…,K-5 and K-1',…,K-5' are discussed, with the same focus of the Table 6-9, where their corresponding SOE is 1. It is worth to note that, Table 6-9, gives the averages of all classes of SOE, and in Table 6-10, they are depicted separately. For each SOE, yellow rows report the heuristic performances for Kartal with greater $\left(W_{ij}\right)$ and smaller $\left(W_{ij}{}'\right)$ debris removal efforts individually, arranged over the number of critical nodes.

Table 6-10: Heuristic performance summary of Kartal instances for each SOE class

	instances:		Constructive Heuristic			2-opt Improvement Heuristic			**Final**
	K-1,…,K-5		Average	Maximum	Optimum	Average	Maximum	Optimum	**Optimum**
	K-1',…,K-5'		Gap	Gap	Ratio	Gap	Gap	Ratio	**Ratio**
SOE=1	Kartal- with Wij	|DL|=7	4.6%	4.7%	0%	0.9%	4.5%	80%	**80%**
		|DL|=4	0.0%	0.0%	100%	-	-	-	**100%**
		|DL|=3	1.4%	3.4%	60%	0.0%	0.0%	100%	**100%**
			2.0%	4.7%	53%	0.6%	4.5%	86%	**93%**
	Kartal- with Wij'	|DL|=7	4.6%	4.7%	0%	0.9%	4.5%	80%	**80%**
		|DL|=4	0.0%	0.0%	100%	-	-	-	**100%**
		|DL|=3	1.4%	3.4%	60%	0.0%	0.0%	100%	**100%**
			2.0%	4.7%	53%	0.6%	4.5%	86%	**93%**
	instances:		Constructive Heuristic			2-opt Improvement Heuristic			**Final**
	K-6,…,K-10		Average	Maximum	Optimum	Average	Maximum	Optimum	**Optimum**
	K-6',…,K-10'		Gap	Gap	Ratio	Gap	Gap	Ratio	**Ratio**
SOE=2	Kartal- with Wij	|DL|=7	0.8%	4.0%	80%	0.0%	0.0%	100%	**100%**
		|DL|=4	0.0%	0.0%	100%	-	-	-	**100%**
		|DL|=3	1.4%	3.4%	60%	0.0%	0.0%	100%	**100%**
			0.7%	4.0%	80%	0.0%	0.0%	100%	**100%**
	Kartal- with Wij'	|DL|=7	0.8%	4.1%	80%	0.0%	0.0%	100%	**100%**
		|DL|=4	0.0%	0.0%	100%	-	-	-	**100%**
		|DL|=3	1.4%	3.4%	60%	0.0%	0.0%	100%	**100%**
			0.7%	4.1%	80%	0.0%	0.0%	100%	**100%**
	instances:		Constructive Heuristic			2-opt Improvement Heuristic			**Final**
	K-11,…,K-15		Average	Maximum	Optimum	Average	Maximum	Optimum	**Optimum**
	K-11',…,K-15'		Gap	Gap	Ratio	Gap	Gap	Ratio	**Ratio**
SOE=3	Kartal- with Wij	|DL|=7	3.2%	9.4%	60%	3.8%	7.5%	50%	**80%**
		|DL|=4	0.0%	0.0%	100%	-	-	-	**100%**
		|DL|=3	1.2%	3.4%	60%	1.3%	2.5%	50%	**80%**
			1.5%	9.4%	73%	2.5%	6.4%	50%	**87%**
	Kartal- with Wij'	|DL|=7	3.9%	9.8%	20%	2.8%	9.80%	50%	**60%**
		|DL|=4	0.4%	2.0%	80%	2.0%	2.0%	0%	**80%**
		|DL|=3	1.2%	3.4%	60%	1.3%	2.60%	50%	**80%**
			1.8%	9.8%	53%	2.3%	9.80%	43%	**73%**
	instances:		Constructive Heuristic			2-opt Improvement Heuristic			**Final**
	K-16,…,K-20		Average	Maximum	Optimum	Average	Maximum	Optimum	**Optimum**
	K-16',…,K-20'		Gap	Gap	Ratio	Gap	Gap	Ratio	**Ratio**
SOE=4	Kartal- with Wij	|DL|=7	2.7%	7.9%	60%	6.7%	7.9%	0%	60%
		|DL|=4	1.3%	2.9%	40%	2.2%	2.9%	0%	40%
		|DL|=3	0.9%	4.4%	80%	4.4%	4.4%	0%	**80%**
			1.6%	7.9%	60%	4.1%	0.0%	0%	60%
	Kartal- with Wij'	|DL|=7	5.6%	18.8%	20%	7.0%	18.8%	0%	20%
		|DL|=4	5.8%	13.7%	20%	7.3%	13.7%.	0%	20%
		|DL|=3	1.1%	5.3%	80%	5.3%	5.3%	0%	**80%**
			4.2%	18.8%	40%	6.9%	18.8%	0%	40%

From the aspect of heuristic performances according to debris removal effort settings, heuristic results are almost the same for the instances in the first two quarters of the table, however, in the 3rd quarter, it can be seen that, heuristics give better results where the debris removal effort is greater, 87% of optimum, where it reduces to 73% for the instances with smaller debris removal efforts. Also, in the 4th quarter, for the greater debris removal efforts, heuristics find optimum at a rate of 60%, and it reduces to 40% for the cases with smaller debris removal effort. That is to say, as the SOE increases, the heuristics show better performances for the instances with W_{ij} compared with the instances with where the debris removal effort is smaller (W_{ij}').

When debris removal effort is smaller, the trade-off between unblocking a road and finding alternative paths is minor. However, with greater debris removal efforts on the arcs, and for more severe earthquakes, which implies more blocked arcs, the trade-off is obvious. Therefore, this may be the reason that lies behind the above fact which indicates that the heuristics give better results for the instances with greater debris removal efforts.

When the performances of heuristics are analysed from the point of number of critical nodes, Table 6-9, that summarizes results of all instances at a time, indicates that final optimum ratio is highest when the number of critical nodes is less, 3, and it reduces for higher number of nodes. However, contrary to this inference, the fact of having difficulty, in finding the optimum solutions when the number of critical nodes increases, is not valid when the instances are dealt separately, according to the classes of SOE, as in Table 6-10. Especially, the constructive heuristic gives better results for 4 critical nodes. Also, surprisingly, as it can be observed in the 4th quarter, it gives better ratios of optimum for 7 critical nodes than the cases with 4 critical nodes.

This may occur due to the differences of instances or since the number of critical nodes, 7, 4 and 3 are not very different from each other in a quantitative way, the locations of the critical nodes could be the main determinant factor of the heuristic performances.

Finally, from the point of how SOE effects the heuristic performances, the optimum finding rate is higher for less severe earthquakes and reduces when SOE=3 and SOE=4, as expected.

When it comes to deal with heuristics' performances of the Bakırköy instances, the following table summarizes the results obtained from analysing all instances from B-1 to B-20 and B-1' to B-20' with different number of critical nodes.

Table 6-11: Heuristic performance summary of Bakırköy instances

instances: B-1,...,B-20 B-1',...,B-20'		Constructive Heuristic		2-opt Improvement Heuristic		**Final Optimum Ratio**	**Final "Better" Ratio**
		Optimum Ratio	"Better" Ratio	Optimum Ratio	"Better" Ratio		
Bakırköy-with Wij	|DL|=15	?	60%	?	75%	?	75%
	|DL|=8	45.0%	-	36.4%	-	65.0%	-
	|DL|=7	15.0%	-	29.4%	-	40.0%	-
Bakırköy-with Wij'	|DL|=15	?	65.0%	?	80%	?	80%
	|DL|=8	45.0%	-	0.0%	-	45.0%	-
	|DL|=7	15.0%	-	29.4%	-	40.0%	-

Since the mathematical model is not able to find the optimal for some instances of Bakırköy, the heuristic performances could not be analysed based on optimum solutions for all instances. For that purpose, the performance summary table consists of columns where both optimum ratio and "better" ratio is reported. The "better" ratio indicates the proportion of the instances where the heuristics give better results than the model's best incumbent value. For this proportion of instances, the heuristics could find better feasible solutions in 1-2 seconds than the mathematical models feasible solution reported at the end of 4 hour. To exemplify, for the instances with 15 critical nodes, the mathematical model solves all instances with a certain gap from the real optimum. Therefore, it is not possible to know whether the heuristics find optimum even when they provide better results. That's why a question mark (?) is reported in the related cell of the table. But it is known that, the constructive heuristic gives better objectives than the model for 60% of the instances of Bakırköy with greater debris removal effort.

Additionally, it is observed that the final optimum ratio is higher when the number of critical nodes is 8, when compared with 7 critical nodes in the experimental setting. It is worth to note that, these two critical node sets are disjoint, and the observation expresses that the locations of critical nodes have higher impacts in the heuristic solution methodologies, than their quantity.

To observe the effect of SOE on the Bakırköy's results, as in the case of Kartal, each quarter of the following table considers a set of 5 instances according to their belonging to a class of SOE. Namely, in the first quarter of the table, the instances B-1,…,B-5 and B-1',…,B-5' are discussed where their corresponding SOE is 1; in the second quarter, the instances B-6,…,B-10 and B-6',…,B-10' are discussed which belong to the class of SOE=2, and so on.

Table 6-12: Heuristic performance summary of Bakırköy instances for each SOE class

	instances: B-1,…,B-5 B-1',…,B-5'		Constructive Heuristic		2-opt Improvement Heuristic		**Final Optimum Ratio**	**Final "Better" Ratio**
			Optimum Ratio	"Better" Ratio	Optimum Ratio	"Better" Ratio		
SOE=1	Bakırköy-with Wij	|DL|=15	?	40%	?	80%	?	80%
		|DL|=8	80.0%	-	100.0%	-	100%	-
		|DL|=7	0.0%	-	60.0%	-	60%	-
	Bakırköy-with Wij'	|DL|=15	?	60%	?	80%	?	80%
		|DL|=8	100.0%	-	0.0%	-	100%	-
		|DL|=7	0.0%	-	60.0%	-	60%	-
	instances: B-6,…,B-10 B-6',…,B-10'		Constructive Heuristic		2-opt Improvement Heuristic		**Final Optimum Ratio**	**Final "Better" Ratio**
			Optimum Ratio	"Better" Ratio	Optimum Ratio	"Better" Ratio		
SOE=2	Bakırköy-with Wij	|DL|=15	?	60%	0.0%	80%	?	80%
		|DL|=8	80.0%	-	0.0%	-	80%	-
		|DL|=7	20.0%	-	25.0%	-	40%	-
	Bakırköy-with Wij'	|DL|=15	?	60%	?	100%	?	100%
		|DL|=8	80.0%	-	0.0%	-	80%	-
		|DL|=7	20.0%	-	25.0%	-	40%	-
	instances: B-11,…,B-15 B-11',…,B-15'		Constructive Heuristic		2-opt Improvement Heuristic		**Final Optimum Ratio**	**Final "Better" Ratio**
			Optimum Ratio	"Better" Ratio	Optimum Ratio	"Better" Ratio		
SOE=3	Bakırköy-with Wij	|DL|=15	?	40%	?	40%	?	40%
		|DL|=8	20.0%	-	0.0%	-	20%	-
		|DL|=7	40.0%	-	0.0%	-	40%	-
	Bakırköy-with Wij'	|DL|=15	?	40%	?	40%	?	40%
		|DL|=8	0.0%	-	0.0%	-	0%	-
		|DL|=7	40.0%	-	0.00%	-	40%	-
	instances: B-16,…,B-20 B-16',…,B-20'		Constructive Heuristic		2-opt Improvement Heuristic		**Final Optimum Ratio**	**Final "Better" Ratio**
			Optimum Ratio	"Better" Ratio	Optimum Ratio	"Better" Ratio		
SOE=4	Bakırköy-with Wij	|DL|=15	?	100%	?	100%	?	100%
		|DL|=8	0.0%	-	60.0%	-	60%	-
		|DL|=7	0.0%	-	20.0%	-	20%	-
	Bakırköy-with Wij'	|DL|=15	?	100%	?	100%	?	100%
		|DL|=8	0.0%	-	0.0%	-	0%	-
		|DL|=7	0.0%	-	20.0%	-	20%	-

Once again, not surprisingly, it is observed that, as the SOE increases the heuristics' performances to find the optimum generally reduces for Bakırköy instances. Also, common with the Kartal heuristic performance results, the heuristics' conclusions are almost the same for greater and smaller debris removal efforts when the SOE=1 and SOE=2. But, as the severity increases, the heuristics performances are better for the instances with greater debris removal effort. As in the Kartal case, the reason behind this issue could be that the trade-off between unblocking a road and finding an alternative path is obvious when the debris removal effort is greater and the earthquake severity is higher, which implies more blocked arcs in network. Namely, both model and heuristic have tendency to use unblocked arcs rather than blocked arcs with relatively much higher costs.

Finally, as it is indicated before, the locations of critical nodes are important for the performances of heuristics, and they give better results when the number of critical nodes is 8. However, as the SOE increases, the number of blocked arcs in the network also increases; therefore, this issue reduces the effect of the locations of critical nodes for more severe earthquakes, and balance the difficulty to find optimum solutions. Namely, when SOE=4, constructive heuristic cannot find any optimum for both 7 and 8 critical nodes. The detailed results of heuristics performances can be seen in the Appendix.

As a conclusion, it is observed that, severity of earthquake (SOE) and the number of critical nodes are the main factors that affect the performances of both the mathematical model and the heuristics. In other respects, it is observed that the locations of the critical nodes affect the optimum travel path, and also, the heuristics performances are influenced from the locations of the critical nodes.

Chapter 7

Conclusion and Future Research Directions

Due to the importance of emergency aid transportation during the post-earthquake response phase, in this study, a solution methodology that provides emergency supplies to the pre-determined disaster affected regions, by considering the blockages on the transportation network, is developed. In the current system, there is no systematical way of aid transportation, responsibilities and authorities on this issue are not clear, and corresponding activity definitions are under development. The main contributions of the proposed methodology is increasing the quality of life of disaster victims, rescuing lives, and thus defusing the post-earthquake chaotic environment by providing disaster relief materials to the disaster affected regions as soon as possible. Since the problem characteristic both implies the node routing aspect, with the requirement of the vehicle to visit predetermined disaster affected regions; and an arc routing aspect, where it may be necessary to unblock some of the arcs on the travel path of the vehicle, the arc routing and node routing literatures, which are under the umbrella of general routing literature

(GRP) are investigated in Chapter 3. Then the problem is mathematically modeled as it is explained in Chapter 4. Since the first two models' periodic structure do not precisely fit the problem characteristics, yet another model which has $O(n^4)$ variables and $O(n^3)$ constraints, is developed which is free from the periodic nature. The objective of the model is to minimize the total effort spent until all the critical nodes have been visited, where "total effort" intends both travelling and debris removal efforts. The model assigns the visiting order of the critical nodes, and it decides the travelling path between them with the arcs where the blockage on them will be removed. Then, the heuristic methodologies are developed, due to the difficulties encountered when the dimension of the data increases. As it is stated in Chapter 5, a Dijkstra based constructive heuristic, and to improve the results, a 2-opt based improvement heuristic, which uses the results of the constructive heuristic as its inputs, are developed. The heuristics are extremely fast and arrive at the conclusion in seconds. Then the performances of the models and the heuristics are tested with two different data sets, Kartal and Bakırköy districts of İstanbul. As it is stated in Chapter 6, the nodes with hospitals and schools are selected as the critical ones. Also, since the debris amount is related to the severity of earthquake, and there are more blocked arcs in the case of more severe earthquakes, we defined 4 different classes of earthquake severities (SOE) and their corresponding blocked arc ratios (BAR). Travel times of arcs are adapted from the node-to-node distance matrices of the above mentioned data sets, and the required debris removal efforts of the arcs are calculated in such a way that, they are directly proportional to the length of the relevant arc and the earthquake severity (SOE). To see the effect of the amount of debris removal effort on the performances of the models and the heuristics, greater and smaller debris removal effort configurations are taken into consideration.

Initially, performance of MOD-3 is tested with Kartal instances with the critical node selection of only schools, only hospitals and both. The model solves entire Kartal instances to optimality and it is observed that, as SOE increases the CPU times also increase. In the same manner, increment on the number of critical nodes causes dramatic growth on the CPU times. Additionally, MOD-3 shows better performance for a specific instances' smaller debris removal effort configuration, when it is more likely to encounter with blocked arcs, namely in the cases where the SOE and the number of critical nodes are higher.

Thereafter MOD-3 is tested with the Bakırköy instances with the same critical node selection strategy, that is to say, only schools, only hospitals and both. However, even if model gives optimum results for some instances, as the SOE and number of critical nodes increase, it reports optimality gaps after 4 hours. Therefore, the above mentioned heuristics' performances are tested with the same instances of Bakırköy.

First, all Kartal instances without differentiating according to the classes of SOE, are analyzed, and it is shown that heuristic methodologies can find optimum solutions up to 90% of the Kartal instances, in seconds. When the instances with different SOE classes analyzed separately, it is observed that heuristic gives better optimal ratios for less severe earthquakes and the optimum ratio is also better for the instances with greater debris removal effort. Also, it is inferred that the locations of critical nodes are just as significant as the number of them, from the point of heuristics' performances. Then the heuristics are applied to the Bakırköy instances, when the instances are not differentiated according to the SOE classes, it is observed that, the heuristics can find "better" solutions than the model, up to 80% of the most difficult instances, namely the instances where the number of critical nodes is higher, and they give optimum solutions up to 65% of the moderate instances. When the performances of heuristic for different SOE classes are analyzed, it is shown that the performances are better for the instances where the severity of earthquake is lower. For such cases, they find "better" solutions than the model, up to 80% of the most difficult instances, and it is observed that, they can find optimum solutions 100% of the moderate cases. However, it is also observed that, even if the number of critical nodes and the severity of the earthquake are the significant factors that affect the performances, the locations of the critical nodes and the specific characteristics of the instances are the other factors on the performance results. In summary, the heuristics' performances are quite good in the sense of solution quality and speed.

Within the scope of this study, we proposed a methodology to provide emergency relief supplies to the disaster affected regions in the response phase of the earthquake. For that purpose, we developed mathematical models and heuristic methodologies which determine the route of the vehicle, named RESCUE, which both carries relief materials and removes the debris whenever it encounters with a blocked arc. The performances of the models and heuristics are tested for data sets which are designed to have distinctive characteristics to demonstrate the outcomes for the different experiment groups. Also, we introduced a new problem to the literature that includes both node and arc routing aspects.

In this study, all the critical nodes are assumed to have equal urgency to get relief materials, but as a future research direction, the critical nodes can be prioritized between each other. Also, the cases with multi suppliers and multi vehicles can be considered. Our solution methodology is easily adaptable to the case when there are multi suppliers and multi vehicles. Additionally, to improve the optimality gap of the model for greater data sets, valid inequalities can be derived.

BIBLIOGRAPHY

[1] Çelik, M., Ergun,Ö., & Keskinocak,P. (2013, June). Post Disaster Debris Clearance with Incomplete Information. In 2013 *International IIE Conference*. *İstanbul,Turkey.*

[2] Luther, L. (2006, June). Disaster debris removal after hurricane Katrina: status and associated issues. Congressional Research Service, Library of Congress.

[3] Bjerregaard, M. (2009). MSB/UNDP Debris management guidelines. *Disaster Waste Recovery*.

[4**]** MSW (2006). MSW Management September/October: 36.

[5] Baycan, F. (2004, April). Emergency Planning for Disaster Waste: A Proposal based on the experience of the Marmara Earthquake in Turkey. In *2004 International Conference and Student Competition on Post-disaster Reconstruction "Planning for Reconstruction". Coventry, UK*.

[6] Baycan, F., & Petersen, M. (2002, July). Disaster waste management-C&D waste. In *Annual conference of the international solid waste association* (pp. 8-12).

[7**]** Norio, O., Ye, T., Kajitani, Y., Shi, P., & Tatano, H. (2011). The 2011 eastern Japan great earthquake disaster: Overview and comments. *International Journal of Disaster Risk Science*, *2*(1), 34-42.

[8] Result Disaster Profiles |EM-DAT [Internet]. [updated 2013 July 06, cited 2013 July 06]. Available from: http://www.emdat.be/result-disaster-profiles?period=1900%242013&disgroup=group&dis_type='Earthquake+(seismic+activity)','Mass+movement+dry','Volcano'%24Geophysical&Submit=Display+Disaster+Profile%3E

[9] Türkiye Ulusal Deprem Araştırmaları Programı, September 2005. [cited 2013 July 06]. Available from: http://www.tubitak.gov.tr/tubitak_content_files/ARDEB/kamag/Turkiye_Ulusal_Deprem_Arastirmalari_Programi.pdf.

[10] Can, H., & Özmen, B. (2010). Türkiyenin Deprem Gerçeği Paneli. Ankara: Gazi Üniversitesi Gazi Basımevi.

[11] Özmen, B., Güler, H., & Nurlu, M. (1997). *Coğrafi Bilgi Sistemi ile deprem bölgelerinin incelenmesi*. Bayındırlık ve İskan Bakanlığı Afet İşleri Fenel Mürdürlüğü.

[12] Yıldız, B. K., T.C. Afet ve Acil Durum Yönetimi Başkanlığı İyileştirme Dairesi. Personal communication. February 2012, Ankara.

[13] Sisman, Ç., T.C. Afet ve Acil Durum Yönetimi Başkanlığı Müdahale Dairesi Hasar Tespit Çalışma Grubu. Personal communication. February 2012, Ankara.

[14] Lenstra, J. K., & Kan, A. H. G. (1976). On general routing problems. *Networks*,*6*(3), 273-280.

[15] Corberán, A., & Sanchis, J. M. (1994). A polyhedral approach to the rural postman problem. *European Journal of Operational Research*, *79*(1), 95-114.

[16] Kwan, M. K. (1962). Graphic programming using odd or even points. *Chinese Math*, *1*(273-277), 110.

[17] Wøhlk, S. (2008). A decade of capacitated arc routing. In *The vehicle routing problem: latest advances and new challenges* (pp. 29-48). Springer US.

[18] Eiselt, H. A., Gendreau, M., & Laporte, G. (1995). Arc routing problems, part I: The Chinese postman problem. *Operations Research*, *43*(2), 231-242.

[19] Papadimitriou, C. H. (1976). On the complexity of edge traversing. *Journal of the ACM (JACM)*, *23*(3), 544-554.

[20] Win, Z. (1989). On the windy postman problem on Eulerian graphs.*Mathematical Programming*, *44*(1-3), 97-112.

[21] Brucker, P. (1981). The Chinese postman problem for mixed graphs. In*Graphtheoretic Concepts in Computer Science* (pp. 354-366). Springer Berlin Heidelberg.

[22] Guan, M. (1984). On the windy postman problem. *Discrete Applied Mathematics*, *9*(1), 41-46.

[23] Dror, M., Stern, H., & Trudeau, P. (1987). Postman tour on a graph with precedence relation on arcs. *Networks*, *17*(3), 283-294.

[24] Ahr, D., & Reinelt, G. (2006). A tabu search algorithm for the min–max k-Chinese postman problem. *Computers & operations research*, *33*(12), 3403-3422.

[25] Kramberger, T., & Žerovnik, J. (2007). Priority constrained chinese postman problem. *Logistics & Sustainable Transport*, *1*(1).

[26] Fleury, M. (1883). Deux problemes de geometrie de situation. *Journal de mathematiques elementaires*, 257-261.

[27] Orloff, C. S. (1974). A fundamental problem in vehicle routing. *Networks*, *4*(1), 35-64.

[28] Eiselt, H. A., Gendreau, M., & Laporte, G. (1995). Arc routing problems, part II: The rural postman problem. *Operations Research*, *43*(3), 399-414.

[29] Frederickson G. N., Hecht M.S., & Kim C.E. (1978). Approximation algorithms for some routing problems. SIAM J. Comp. 7, 178-193.

[30] Aráoz, J., Fernández, E., & Zoltan, C. (2006). Privatized rural postman problems. *Computers & operations research*, *33*(12), 3432-3449.

[31] Aráoz, J., Fernández, E., & Meza, O. (2009). Solving the prize-collecting rural postman problem. *European Journal of Operational Research*, *196*(3), 886-896.

[32] Letchford, A. N., & Eglese, R. W. (1998). The rural postman problem with deadline classes. *European Journal of Operational Research*, *105*(3), 390-400.

[33] Benavent, E., Corberán, A., Plana, I., & Sanchis, J. M. (2009). Min-Max K-vehicles windy rural postman problem. *Networks*, *54*(4), 216-226.

[34] Golden, B. L., & Wong, R. T. (1981). Capacitated arc routing problems.*Networks*, *11*(3), 305-315.

[35] Christofides, N. (1973). The optimum traversal of a graph. *Omega*, *1*(6), 719-732.

[36] Busch, I. K. (1991). *Vehicle routing on acyclic networks* (Doctoral dissertation, Johns Hopkins University).

[37] Labbé, M., Laporte, G., & Mercure, H. (1991). Capacitated vehicle routing on trees. *Operations Research*, *39*(4), 616-622.

[38] Lacomme, P., Prins, C., & Sevaux, M. (2003, January). Multiobjective capacitated arc routing problem. In *Evolutionary multi-criterion optimization* (pp. 550-564). Springer Berlin Heidelberg.

[39] Amberg, A., Domschke, W., & Voß, S. (2000). Multiple center capacitated arc routing problems: A tabu search algorithm using capacitated trees. *European Journal of Operational Research*, *124*(2), 360-376.

[40] Amaya, A., Langevin, A., & Trépanier, M. (2007). The capacitated arc routing problem with refill points. *Operations Research Letters*, *35*(1), 45-53.

[41] Archetti, C., Feillet, D., Hertz, A., & Speranza, M. G. (2010). The undirected capacitated arc routing problem with profits. *Computers & Operations Research*, *37*(11), 1860-1869.

[42] Sniezek, J., & Bodin, L. (2006). Using mixed integer programming for solving the capacitated arc routing problem with vehicle/site dependencies with an application to the routing of residential sanitation collection vehicles. *Annals of Operations Research*, *144*(1), 33-58.

[43] Kirlik, G., & Sipahioglu, A. (2012). Capacitated arc routing problem with deadheading demands. *Computers & Operations Research*, *39*(10), 2380-2394.

[44] Ulusoy, G. (1985). The fleet size and mix problem for capacitated arc routing.*European Journal of Operational Research*, *22*(3), 329-337.

[45] Xu, Y., Hu, M., Su, B., Zhu, B., & Zhu, Z. (2009). The canadian traveller problem and its competitive analysis. *Journal of combinatorial optimization*,*18*(2), 195-205.

[46] Papadimitriou, C. H., & Yannakakis, M. (1991). Shortest paths without a map.*Theoretical Computer Science*, *84*(1), 127-150.

[47] Bar-Noy, A., & Schieber, B. (1991, March). The canadian traveller problem. In*Proceedings of the second annual ACM-SIAM symposium on Discrete algorithms* (pp. 261-270). Society for Industrial and Applied Mathematics.

[48] Nikolova, E., & Karger, D. R. (2008, July). Route Planning under Uncertainty: The Canadian Traveller Problem. In *AAAI* (pp. 969-974).

[49] Croes, G. A. (1958). A method for solving traveling-salesman problems.*Operations Research*, *6*(6), 791-812.

[50] Kılcı,F. (2012,June). *A decision support system for shelter site selection with gis integration: Case for Turkey* (M.S. Thesis, Bilkent University).

[51] Floyd, R. W. (1962). Algorithm 97: shortest path. *Communications of the ACM*,*5*(6), 345.

APPENDIX

Appendix 1: Detailed model performance of Kartal instances with greater debris removal effort (Wij)

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-1	1	0.125	124	7(all)	44	221.41	-	16-**41-22-21-33-43-26**-15-**14**
K-2					43	186.73	-	16-**41-22-21-33-43-26-14**
K-3					44	190.82	-	16-**41-22-21-33-43**-35-**26-14**
K-4					43	178.4	-	16-**41-22-21-33-43**-17-**26-14**
K-5					43	183.2	-	16-**41-22-21-33-43**-17-**26-14**
K-1				4(hospitals)	35	8.81	-	16-**41**-27-**33-43-26**
K-2					35	10.02	-	16-**41-33-43**-17-**26**
K-3					36	6.27	-	16-**41-33-43**-12-**26-41**
K-4					35	12.08	-	16-**41-33-43-26**
K-5					35	11.69	-	16-**41-33-43-26**
K-1				3(schools)	30	1.4	-	16-**22-21-14**
K-2					30	1.47	-	16-**22-21**-15-**14**
K-3					30	1.61	-	16-**22-21-14**
K-4					29	1.44	-	16-**21-22-14**
K-5					29	1.44	-	16-**21-22-14**

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-6	2	0.445	441	7(all)	48	223.26	-	16-**22-21-41-33**-45-**43-26-14**
K-7					50	235.96	-	16-**41-22-21-33-21-43-26**-3-**14**
K-8					51	270.41	-	16-**22-21-22-43-33-41-26-14**
K-9					49	225.46	-	16-**22-41-21-33**-40-**43-26-14**
K-10					48	261.39	-	16-**22-21-22-41-33**-45-**43-26-14**
K-6				4(hospitals)	38	15.39	-	16-**41-33**-45-**43-26**
K-7					42	8.4	-	16-**41-33**-21-**43-26**
K-8					40	9.09	-	16-**43-33-41-26**
K-9					42	10.75	-	16-22-**41**-21-**33**-40-**43-26**
K-10					39	7.04	-	16-22-**41-33**-45-**43-26**
K-6				3(schools)	29	2.52	-	16-**21-22-14**
K-7					29	2.54	-	16-**21-22-14**
K-8					30	2.28	-	16-**22-21-22-14**
K-9					35	3.51	-	16-**22**-27-**21-14**
K-10					30	3.04	-	16-**22-21**-15-**14**

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-11	3	0.58	574	7(all)	53	315.37	-	16-**41**-27-**22**-27-**21-33-43-26**-15-**14**
K-12					63	495.63	-	16-**21-43**-27-**33-22**-32-**41**-29-**26-14**
K-13					68	381.4	-	16-**22-21**-40-**43**-27-**41**-32-**33**-11-**26-14**
K-14					46	196.96	-	16-**41-21-22-33-43**-17-**26**-15-**14**
K-15					47	186.72	-	16-**41-22-21-33**-45-**43**-17-12-**26-14**
K-11				4(hospitals)	35	4.87	-	16-**41-33-43-26**
K-12					53	9.51	-	16-**43**-27-**33**-32-**41**-3-**26**
K-13					51	8.32	-	16-**41**-32-**33**-35-**43-26**
K-14					38	8.69	-	16-**41**-21-22-**33-43**-17-**26**
K-15					40	8.41	-	16-**41**-21-**33**-45-**26**-45-**43**
K-11				3(schools)	40	4.34	-	16-27-**21**-27-**22-14**
K-12					35	2.98	-	16-**21**-27-**22**-42-**14**
K-13					30	3.11	-	16-**22-21-22-14**
K-14					35	2.85	-	16-41-**21-22-14**
K-15					29	2.51	-	16-**21-22-14**

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-16	4	0.814	806	7(all)	109	5167.18	(21,22)	16-25-**21-22-41-22**-32-27-**33**-10-**43-14**-8-**26**
K-17					82	3319.8	-	16-12-**26-14**-29-**41**-17-**22-33-21-43**
K-18					110	9136.87	(21,22)	16-17-40-**43-41-22-21**-1-**33-14**-31-38-**26**
K-19					90	2915.86	(21,22)	16-10-17-**43-33-43**-32-**22-21-41-14**-15-29-**26**
K-20					101	4541.84	(21,22),(22,41),(33,43)	16-**22-41-22-21-33-43**-35-9-**26-14**
K-16				4(hospitals)	84	22.32	-	16-25-**43**-10-**33**-27-32-22-**41**-22-**26**
K-17					67	20.23	-	16-12-**26**-19-**43**-19-**33**-35-**41**
K-18					70	38.91	(33,43)	16-17-40-**43-33-43-41-26**
K-19					70	49.49	-	16-35-21-**41**-17-**43-33-43**-32-**26**
K-20					88	64.54	-	16-**33**-19-37-**26**-9-35-**43**-35-18-**41**
K-16				3(schools)	57	6.62	(21,22)	16-25-**21-22**-23-15-**14**
K-17					55	4.66	-	16-20-**22**-33-**21-14**
K-18					68	7.08	(21,22)	16-17-45-**21-22**-41-31-**14**
K-19					55	4.76	(21,22)	16-45-**22-21**-41-**14**
K-20					45	3.62	(21,22)	16-22-**21-22**-27-**14**

Appendix 2: Detailed model performance of Kartal instances with smaller debris removal effort (Wij’)

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-1'	1	0.125	124	7(all)	44	219.89	-	16-**41-22-21-33-43-26**-15-**14**
K-2'					43	213.25	-	16-**41-22-21-33-43-26-14**
K-3'					44	192.61	-	16-**41-22-21-33-43**-12-**26-14**
K-4'					43	156.37	-	16-**41-22-21-33-43-26-14**
K-5'					43	152.9	-	16-**41-22-21-33-43-26-14**
K-1'				4(hospitals)	35	7.16	-	16-**41**-27-**33-43-26**
K-2'					35	10.12	-	16-**41-33-43-26**
K-3'					36	9.98	-	16-**41-33-43**-35-**26**
K-4'					35	7.02	-	16-**41-33-43-26**
K-5'					35	7.31	-	16-**41-33-43-26**
K-1'				3(schools)	30	2.37	-	16-**22-21-22-14**
K-2'					30	1.77	-	16-**21-22**-23-15-**14**
K-3'					30	2.76	-	16-**22-21-14**
K-4'					29	1.86	-	16-**21-22-14**
K-5'					29	1.85	-	16-**21-22-14**

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-6'	2	0.445	441	7(all)	48	231.4	-	16-**41-21-22-21-33**-45-**43-26-14**
K-7'					49	210.34	(33,43)	16-**41-22-21-33-43-26**-3-**14**
K-8'					51	282.31	-	16-**22-21-22-33-43-41-26-14**
K-9'					49	259.62	-	16-**22-41-21-33**-40-**43-26-14**
K-10'					48	210.25	-	16-**22-21-22-41-33**-45-**43-26-14**
K-6'				4(hospitals)	38	9.2	-	16-**41-33**-45-**43-26**
K-7'					41	12.63	(33,43)	16-**41-33-43-26**
K-8'					40	10.28	-	16-**43-33-41-26**
K-9'					42	13.18	-	16-22-**41**-21-**33**-40-**43-26**
K-10'					39	20.43	-	16-22-**41-33**-45-**43-26**
K-6'				3(schools)	29	1.57	-	16-**21-22-14**
K-7'					29	2.75	-	16-**21-22-14**
K-8'					30	3.38	-	16-**22-21-22-14**
K-9'					32	3.8	(21,22)	16-**22-21-14**
K-10'					30	1.9	-	16-**22-21-22-14**

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-11'	3	0.58	574	7(all)	51	214.86	(21,22)	16-**41**-27-**22-21-33-43-26**-15-**14**
K-12'					63	316.91	-	16-**21-43**-27-**33-22**-32-**41**-29-**26-14**
K-13'					67	336.34	(27,33)	16-**22-21**-45-**43**-27-**33**-27-**41-26-14**
K-14'					46	198.85	-	16-**41-21-22**-23-**43**-17-**26**-15-**14**
K-15'					47	184.67	-	16-**41-22-21-33**-45-**43**-45-**26-14**
K-11'				4(hospitals)	35	7.84	-	16-**41-33-43-26**
K-12'					53	8.37	-	16-**43**-27-**33**-32-**41**-3-**26**
K-13'					50	16.5	(33,43)	16-**41**-27-43-**33-43-26**
K-14'					38	9.42	-	16-**41**-21-22-**33-43**-17-**26**
K-15'					40	13.71	-	16-**41**-21-**33**-45-**43**-45-**26**
K-11'				3(schools)	38	3.42	(21,22)	16-27-**21-22-14**
K-12'					33	3.12	(21,22)	16-**21-22**-15-**14**
K-13'					30	2.93	-	16-**22-21-22-14**
K-14'					35	2.03	-	16-41-**21-22-14**
K-15'					29	2.65	-	16-**21-22-14**

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Debris removed arcs	Travel Path
K-16'	4	0.814	806	7(all)	97	4864.74	(3,26),(21,22),(43,45)	16-25-**21-22-41**-45-**43**-45-19-20-**33-26**-3-**14**
K-17'					78	2422.78	(22,41)	16-12-**26-14**-29-**41-22-33-21-43**
K-18'					95	3791.44	(14,15),(21,22),(33,43)	16-17-40-**43-33-43-41-22-21-22-41-26**-13-15-**14**
K-19'					81	2794.34	(21,22),(43,45)	16-45-**43-33-43**-45-**22-21-41-14**-42-29-**26**
K-20'					80	3258.78	(21,22),(22,41),(33,43)	16-**22-41-22-21-33-43**-35-9-**26-14**
K-16'				4(hospitals)	80	54.89	(16,32),(43,45)	16-32-22-6-**41**-45-**43**-45-19-20-**33-26**
K-17'					67	17.18	(22,41)	16-12-**26**-19-**43**-19-**33**-22-**41**
K-18'					63	32.48	(33,43)	16-17-40-**43-33-43-41-26**
K-19'					70	26.91	-	16-35-21-**41**-17-**43-33-43**-32-**26**
K-20'					73	14.96	(21,22),(22,41),(33,43)	16-22-**41**-22-21-**33-43**-35-9-**26**
K-16'				3(schools)	50	4.88	(21,22)	16-25-**21-22**-23-**14**
K-17'					51	3.48	(21,22)	16-20-**22-21-14**
K-18'					61	5.24	(21,22)	16-17-45-**21-22**-41-31-**14**
K-19'					48	4.73	(21,22)	16-45-**22-21**-41-**14**
K-20'					38	4.51	(21,22)	****16-**22-21-14**-16
						****(since the travel path too long, only the visiting order of critical nodes is reported)		

Appendix 3: Detailed model performance of Bakırköy instances with greater debris removal effort (Wij)

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap (14400 sec)	Debris removed arcs
B-1	1	0.19	500	15	73	14400	91.8%	-
B-2					85	14400	91.8%	-
B-3					75	14400	90.7%	-
B-4					75	14400	90.7%	-
B-5					80	14400	90.0%	-
B-1				8(schools)	52	1981.32	-	-
B-2					61	3433.06	-	-
B-3					52	2586.69	-	-
B-4					54	2235.72	-	-
B-5					52	1753.38	-	-
B-1				7(hospitals)	41	921.03	-	-
B-2					38	761.61	-	-
B-3					39	646.14	-	-
B-4					40	821.21	-	-
B-5					40	728.84	-	-

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap%	Debris removed arcs
B-6	2	0.23	613	15	81	14400	92.6%	-
B-7					88	14400	90.9%	-
B-8					77	14400	93.5%	-
B-9					86	14400	93.0%	-
B-10					76	14400	93.4%	-
B-6				8(schools)	52	1647.38	-	-
B-7					60	13108.26	-	-
B-8					52	2126.97	-	-
B-9					60	12207.84	-	-
B-10					52	12236.82	-	-
B-6				7(hospitals)	39	762.91	-	-
B-7					39	686.03	-	-
B-8					40	889.07	-	-
B-9					42	647.82	-	-
B-10					38	600.08	-	-

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap%	Debris removed arcs
B-11	3	0.54	1423	15	93	14400	93.4%	-
B-12					106	14400	94.2%	-
B-13					107	14400	92.3%	-
B-14					99	14400	90.9%	-
B-15					104	14400	92.3%	-
B-11				8(schools)	71	12711	-	-
B-12					80	14400	43.8%	-
B-13					74	13736.36	-	-
B-14					71	12507.97	-	-
B-15					77	14400	33.8%	-
B-11				7(hospitals)	39	461.74	-	-
B-12					42	550.53	-	-
B-13					46	580.23	-	-
B-14					48	744.75	-	-
B-15					43	540.02	-	-

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap%	Debris removed arcs
B-16	4	0.82	2160	15	182	14400	100.0%	(19,38)
B-17					170	14400	100.0%	-
B-18					169	14400	99.1%	-
B-19					144	14400	100.0%	-
B-20					145	14400	99.5%	-
B-16				8(schools)	96	14400	71.0%	-
B-17					78	14400	31.7%	-
B-18					112	14400	74.6%	-
B-19					84	14400	45.2%	-
B-20					87	14400	63.2%	-
B-16				7(hospitals)	61	7279.04	-	-
B-17					58	7653.28	-	-
B-18					51	1162.77	-	-
B-19					59	6006.05	-	-
B-20					52	848.49	-	-

Appendix 4: Detailed model performance of Kartal instances with smaller debris removal effort (Wij’)

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex	Gap% (14400sec)	Debris removed arcs
						CPU(sec)		
B-1'	1	0.19	500	15	74		91.9%	
B-2'					84		92.9%	(7,67),(15,60)
B-3'					78		92.3%	
B-4'					80		92.5%	
B-5'					73		91.8%	
B-1'				8(schools)	52	1920.36		
B-2'					58	2834.09		(7,67)
B-3'					52	1849.6		
B-4'					54	2194.44		
B-5'					52	1638.53		
B-1'				7(hospitals)	41	736.43		-
B-2'					38	706.57		-
B-3'					39	768.61		-
B-4'					40	649.6		-
B-5'					40	845.63		-

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex	Gap% (14400sec)	Debris removed arcs
						CPU(sec)		
B-6'	2	0.23	613	15	75		92.0%	
B-7'					80		90.0%	(7,67)
B-8'					77		92.2%	
B-9'					81		92.6%	(7,67)
B-10'					76		92.1%	
B-6'				8(schools)	52	2785.56		
B-7'					56		26.8%	(7,67)
B-8'					52	2245.67		
B-9'					56	12997.24		(7,67)
B-10'					52	1971.06		
B-6'				7(hospitals)	39	744.54		-
B-7'					39	708.93		-
B-8'					40	794.42		-
B-9'					42	729.86		-
B-10'					38	760.88		-

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap% (14400sec)	Debris removed arcs
B-11'	3	0.54	1423	15	95		91.6%	
B-12'					98		91.8%	(37,73)
B-13'					100		92.0%	
B-14'					98		90.8%	(7,67)
B-15'					99		90.9%	(5,56)
B-11'				8(schools)	67		34.0%	(7,67)
B-12'					75		60.0%	(37,73),(55,58)
B-13'					73		57.5%	(7,67)
B-14'					69		27.5%	(7,67)
B-15'					75		50.7%	(7,67)
B-11'				7(hospitals)	39	618.33		
B-12'					42	644.19		
B-13'					46	818.27		
B-14'					48	1370.15		
B-15'					43	703.36		

Minimize Total Effort	Severity of earhquake (4)>(3)>(2)>(1)	Blocked Arc Ratio	#of blocked arcs	#of critical nodes	Best Objective	cplex CPU(sec)	Gap% (14400sec)	Debris removed arcs
B-16'	4	0.82	2160	15	135		92.6%	
B-17'					131		100.0%	
B-18'					165		95.2%	
B-19'					134		99.3%	
B-20'					328		99.7%	(11,25),(11,47),(13,26), (13,68),(21,64),(26,36)
B-16'				8(schools)	91		69.7%	(15,60)
B-17'					78		37.2%	
B-18'					105		73.7%	(7,67),(15,60),(29,60)
B-19'					79		44.1%	(7,67)
B-20'					83		67.9%	(15,60),(37,56)
B-16'				7(hospitals)	59	7433.39		(22,41)
B-17'					57	7577.35		(19,38)
B-18'					51	1352.84		
B-19'					55	7261.47		(1,21)
B-20'					52	1191.86		

Appendix 5: Detailed heuristic performance of Kartal instances with greater debris removal effort (Wij)

MODEL/HEURISTICS COMPARISON			**Model**		**Constructive Heuristic**				**2-opt Improvement Heuristic**				
Instance #	Instance Features	#of critical nodes	Best Objective	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Improvement	Optimal?
K-1	SOE=1; BAR=0.125; #of Blocked Arcs=124	7	44	-	46	4.5%	-	**NO**	46	4.5%	-	0.0%	**NO**
K-2			43	-	45	4.7%	-	**NO**	43	0.0%	-	4.7%	YES
K-3			44	-	46	4.5%	-	**NO**	44	0.0%	-	4.5%	YES
K-4			43	-	45	4.7%	-	**NO**	43	0.0%	-	4.7%	YES
K-5			43	-	45	4.7%	-	**NO**	43	0.0%	-	4.7%	YES
K-6	SOE=2; BAR=0.445; #of Blocked Arcs=441		48	-	48	0.0%	-	YES					
K-7			50	-	52	4.0%	-	**NO**	50	0.0%	-	4.0%	YES
K-8			51	-	51	0.0%	-	YES					
K-9			49	-	49	0.0%	-	YES					
K-10			48	-	48	0.0%	-	YES					
K-11	SOE=3; BAR=0.580; #of Blocked Arcs=574		53	-	58	9.4%	-	**NO**	57	7.5%	-	1.9%	**NO**
K-12			63	-	63	0.0%	-	YES					
K-13			68	-	68	0.0%	-	YES					
K-14			46	-	46	0.0%	-	YES					
K-15			47	-	50	6.4%	-	**NO**	47	0.0%	-	6.4%	YES
K-16	SOE=4; BAR=0.814; #of Blocked Arcs=806		109	(21,22)	109	0.0%	(21,22)	YES					
K-17			82	-	82	0.0%	-	YES					
K-18			110	(21,22)	110	0.0%	(21,22)	YES					
K-19			90	(21,22)	95	5.6%	-	**NO**	95	5.6%	-	0.0%	**NO**
K-20			101	(21,22),(22,41),(33,43)	109	7.9%	(22,21),(33,43)	**NO**	109	7.9%	(22,21),(33,43)	0.0%	**NO**

MODEL/HEURISTICS COMPARISON			**Model**		**Constructive Heuristic**				**Improvement Heuristic**				
Instance #	Instance Features	#of critical nodes	Best Objective	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Improvement	Optimal?
K-1	SOE=1; BAR=0.125; #of Blocked Arcs=124	4	35	-	35	0.0%	-	YES					
K-2			35	-	35	0.0%	-	YES					
K-3			36	-	36	0.0%	-	YES					
K-4			35	-	35	0.0%	-	YES					
K-5			35	-	35	0.0%	-	YES					
K-6	SOE=2; BAR=0.445; #of Blocked Arcs=441		38	-	38	0.0%	-	YES					
K-7			42	-	42	0.0%	-	YES					
K-8			40	-	40	0.0%	-	YES					
K-9			42	-	42	0.0%	-	YES					
K-10			39	-	39	0.0%	-	YES					
K-11	SOE=3; BAR=0.580; #of Blocked Arcs=574		35	-	35	0.0%	-	YES					
K-12			53	-	53	0.0%	-	YES					
K-13			51	-	51	0.0%	-	YES					
K-14			38	-	38	0.0%	-	YES					
K-15			40	-	40	0.0%	-	YES					
K-16	SOE=4; BAR=0.814; #of Blocked Arcs=806		84	-	84	0.0%	-	YES					
K-17			67	-	67	0.0%	-	YES					
K-18			70	(33,43)	71	1.4%	-	**NO**	71	1.4%	-	0.0%	**NO**
K-19			70	-	72	2.9%	-	**NO**	72	2.9%	-	0.0%	**NO**
K-20			88	-	90	2.3%	(33,43)	**NO**	90	2.3%	(33,43)	0.0%	**NO**

MODEL/HEURISTICS COMPARISON			Model		Constructive Heuristic				Improvement Heuristic				
Instance #	Instance Features	#of critical nodes	Best Objective	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Improvement	Optimal?
K-1	SOE=1; BAR=0.125; #of Blocked Arcs=124	3	30	-	30	0.0%	-	YES					
K-2			30	-	30	0.0%	-	YES					
K-3			30	-	30	0.0%	-	YES					
K-4			29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-5			29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-6	SOE=2; BAR=0.445; #of Blocked Arcs=441		29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-7			29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-8			30	-	30	0.0%	-	YES					
K-9			35	-	35	0.0%	-	YES					
K-10			30	-	30	0.0%	-	YES					
K-11	SOE=3; BAR=0.580; #of Blocked Arcs=574		40	-	41	2.5%	-	**NO**	41	2.5%	-	0.0%	**NO**
K-12			35	-	35	0.0%	-	YES					
K-13			30	-	30	0.0%	-	YES					
K-14			35	-	35	0.0%	-	YES					
K-15			29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-16	SOE=4; BAR=0.814; #of Blocked Arcs=806		57	(21,22)	57	0.0%	(21,22)	YES					
K-17			55	-	55	0.0%	-	YES					
K-18			68	(21,22)	68	0.0%	(21,22)	YES					
K-19			55	(21,22)	55	0.0%	(22,21)	YES					
K-20			45	(21,22)	47	4.4%	(22,21)	**NO**	47	4.4%	(22,21)	0.0%	**NO**

Appendix 6: Detailed heuristic performance of Kartal instances with smaller debris removal effort (Wij')

MODEL/HEURISTICS COMPARISON			**Model**		**Constructive Heuristic**				**2-opt Improvement Heuristic**				
Instance #	Instance Features	#of critical nodes	Best Objective	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Improvement	Optimal?
K-1'	SOE=1; BAR=0.125; #of Blocked Arcs=124	7	44		46	4.5%	-	**NO**	46	4.5%	-	0.0%	**NO**
K-2'			43		45	4.7%	-	**NO**	43	0.0%	-	4.7%	YES
K-3'			44		46	4.5%	-	**NO**	44	0.0%	-	4.5%	YES
K-4'			43		45	4.7%	-	**NO**	43	0.0%	-	4.7%	YES
K-5'			43		45	4.7%	-	**NO**	43	0.0%	-	4.7%	YES
K-6'	SOE=2; BAR=0.445; #of Blocked Arcs=441		48	-	48	0.0%	-	YES					
K-7'			49	(33,43)	51	4.1%	(33,43)	**NO**	49	0.0%	(33,43)	4.1%	YES
K-8'			51	-	51	0.0%	-	YES					
K-9'			49	-	49	0.0%	-	YES					
K-10'			48	-	48	0.0%	-	YES					
K-11'	SOE=3; BAR=0.580; #of Blocked Arcs=574		51	(21,22)	56	9.8%	(21,22)	**NO**	56	9.8%	(21,22)	0.0%	**NO**
K-12'			63	-	64	1.6%	(21,22)	**NO**	63	0.0%	-	1.6%	YES
K-13'			67	(27,33)	68	1.5%	-	**NO**	68	1.5%	-	0.0%	**NO**
K-14'			46	-	46	0.0%	-	YES					
K-15'			47	-	50	6.4%	-	**NO**	47	0.0%	-	6.4%	YES
K-16'	SOE=4; BAR=0.814; #of Blocked Arcs=806		97	(3,26),(21,22),(43,45)	98	1.0%	(21,22),(33,43),(3,26)	**NO**	98	1.0%	(21,22),(33,43),(3,26)	0.0%	**NO**
K-17'			78	(22,41)	78	0.0%	(41,22)	YES					
K-18'			95	(14,15),(21,22),(33,43)	97	2.1%	(22,21),(21,33),(14,15)	**NO**	97	2.1%	(22,21),(21,33),(14,15)	0.0%	**NO**
K-19'			81	(21,22),(43,45)	86	6.2%	(22,21)	**NO**	86	6.2%	(22,21)	0.0%	**NO**
K-20'			80	(21,22),(22,41),(33,43)	95	18.8%	(22,21),(33,43)	**NO**	95	18.8%	(22,21),(33,43)	0.0%	**NO**

MODEL/HEURISTICS COMPARISON			**Model**		**Constructive Heuristic**				**Improvement Heuristic**				
Instance #	Instance Features	#of critical nodes	Best Objective	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Improvement	Optimal?
K-1'	SOE=1; BAR=0.125; #of Blocked Arcs=124	4	35		35	0.0%	-	YES					
K-2'			35		35	0.0%	-	YES					
K-3'			36		36	0.0%	-	YES					
K-4'			35		35	0.0%	-	YES					
K-5'			35		35	0.0%	-	YES					
K-6'	SOE=2; BAR=0.445; #of Blocked Arcs=441		38	-	38	0.0%	-	YES					
K-7'			41	(33,43)	41	0.0%	(33,43)	YES					
K-8'			40	-	40	0.0%	-	YES					
K-9'			42	-	42	0.0%	-	YES					
K-10'			39	-	39	0.0%	-	YES					
K-11'	SOE=3; BAR=0.580; #of Blocked Arcs=574		35	-	35	0.0%	-	YES					
K-12'			53	-	53	0.0%	-	YES					
K-13'			50	(33,43)	51	2.0%	(43,33)	**NO**	51	2.0%	(43,33)	0.0%	**NO**
K-14'			38	-	38	0.0%	-	YES					
K-15'			40	-	40	0.0%	-	YES					
K-16'	SOE=4; BAR=0.814; #of Blocked Arcs=806		80	(16,32),(43,45)	81	1.3%	(43,33)	**NO**	81	1.3%	(43,33)	0.0%	**NO**
K-17'			67	(22,41)	67	0.0%	(22,41)	YES					
K-18'			63	(33,43)	71	12.7%	-	**NO**	71	12.7%	-	0.0%	**NO**
K-19'			70	-	71	1.4%	(45,43)	**NO**	71	1.4%	(45,43)	0.0%	**NO**
K-20'			73	(21,22),(22,41),(33,43)	83	13.7%	(33,43)	**NO**	83	13.7%	(33,43)	0.0%	**NO**

MODEL/HEURISTICS COMPARISON			**Model**		**Constructive Heuristic**				**Improvement Heuristic**				
Instance #	Instance Features	#of critical nodes	Best Objective	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Improvement	Optimal?
K-1'	SOE=1; BAR=0.125; #of Blocked Arcs=124	3	30		30	0.0%	-	YES					
K-2'			30		30	0.0%	-	YES					
K-3'			30		30	0.0%	-	YES					
K-4'			29		30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-5'			29		30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-6'	SOE=2; BAR=0.445; #of Blocked Arcs=441		29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-7'			29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-8'			30	-	30	0.0%	-	YES					
K-9'			32	(21,22)	32	0.0%	(22,21)	YES					
K-10'			30	-	30	0.0%	-	YES					
K-11'	SOE=3; BAR=0.580; #of Blocked Arcs=574		38	(21,22)	39	2.6%	(22,21)	**NO**	39	2.6%	(22,21)	0.0%	**NO**
K-12'			33	(21,22)	33	0.0%	(21,22)	YES					
K-13'			30	-	30	0.0%	-	YES					
K-14'			35	-	35	0.0%	-	YES					
K-15'			29	-	30	3.4%	-	**NO**	29	0.0%	-	3.4%	YES
K-16'	SOE=4; BAR=0.814; #of Blocked Arcs=806		50	(21,22)	50	0.0%	(21,22)	YES					
K-17'			51	(21,22)	51	0.0%	(22,21)	YES					
K-18'			61	(21,22)	61	0.0%	(21,22)	YES					
K-19'			48	(21,22)	48	0.0%	(22,21)	YES					
K-20'			38	(21,22)	40	5.3%	(22,21)	**NO**	40	5.3%	(22,21)	0.0%	**NO**

Appendix 7: Detailed heuristic performance of Bakırköy instances with greater debris removal effort (Wij)

MODEL/HEURISTICS COMPARISON			**Model**			**Constructive Heuristic**				**2-opt Improvement Heuristic**			
Instance #	Instance Features	#of critical nodes	Best Objective	Gap (14400sec)	Debris removed arcs	Best Objective	Gap to model's best solution	Debris removed arcs	Optimal?	Best Objective	Gap to model's best solution	Debris removed arcs	Optimal?
B-1	SOE=1; BAR=0.19; #of Blocked Arcs=500	15	73	91.78%	-	76	4%	-	NO	73	0%	-	NO
B-2			85	91.76%	-	94	11%	-	NO	78	-8%	(7,67)	?
B-3			75	90.67%	-	74	-1%	-	?	74	-1%	-	?
B-4			75	90.67%	-	76	1%	-	NO	73	-3%	-	?
B-5			80	90.00%	-	77	-4%	-	?	73	-9%	-	?
B-6	SOE=2; BAR=0.23; #of Blocked Arcs=613		81	92.59%	-	74	-9%	-	?	72	-11%	-	?
B-7			88	90.91%	-	86	-2%	-	?	85	-3%	-	?
B-8			77	93.51%	-	78	1%	-	NO	76	-1%	-	?
B-9			86	93.02%	-	91	6%	-	NO	88	2%	-	NO
B-10			76	93.42%	-	75	-1%	-	?	70	-8%	-	?
B-11	SOE=3; BAR=0.54; #of Blocked Arcs=1423		93	93.41%	-	97	4%	-	NO	95	2%	-	NO
B-12			106	94.23%	-	112	6%	-	NO	106	0%	-	NO
B-13			107	92.25%	-	96	-10%	-	?	96	-10%	-	?
B-14			99	90.91%	-	104	5%	-	NO	100	1%	-	NO
B-15			104	92.31%	-	100	-4%	-	?	99	-5%	-	?
B-16	SOE=4; BAR=0.82; #of Blocked Arcs=2160		182	100.00%	(19,38)	127	-30%	-	?	127	-30%	-	?
B-17			170	100.00%	-	126	-26%	-	?	126	-26%	-	?
B-18			169	99.10%	-	167	-1%	-	?	167	-1%	-	?
B-19			144	100.00%	-	122	-15%	-	?	122	-15%	-	?
B-20			145	99.45%	-	128	-12%	-	?	127	-12%	-	?

MODEL/HEURISTICS COMPARISON			**Model**			**Constructive Heuristic**				**Improvement Heuristic**			
Instance #	Instance Features	#of critical nodes	Best Objective	Gap	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap from optimal	Debris removed arcs	Optimal?
B-1	SOE=1; BAR=0.19; #of Blocked Arcs=500	8	52	-	-	52	0%	-	**YES**				
B-2			61	-	-	64	5%	-	NO	61	0%	(7,67)	**YES**
B-3			52	-	-	52	0%	-	**YES**				
B-4			54	-	-	54	0%	-	**YES**				
B-5			52	-	-	52	0%	-	**YES**				
B-6	SOE=2; BAR=0.23; #of Blocked Arcs=613		52	-	-	52	0%	-	**YES**				
B-7			60	-	-	66	10%	-	NO	66	10%	-	NO
B-8			52	-	-	52	0%	-	**YES**				
B-9			60	-	-	60	0%	-	**YES**				
B-10			52	-	-	52	0%	-	YES				
B-11	SOE=3; BAR=0.54; #of Blocked Arcs=1423		71	-	-	72	1%	(55,58)	NO	72	1%	(55,58)	NO
B-12			80	43.77%	-	81	1%	-	NO	81	1%	-	NO
B-13			74	-	-	74	0%	-	**YES**				
B-14			71	-	-	72	1%	-	NO	72	1%	-	NO
B-15			77	33.77%	-	78	1%	-	NO	78	1%	-	NO
B-16	SOE=4; BAR=0.82; #of Blocked Arcs=2160		96	71.01%	-	96	0%	(15,60)	NO	96	0%	(15,60)	**YES**
B-17			78	31.71%	-	78	0%	-	NO	78	0%	-	**YES**
B-18			112	74.64%	-	118	5%	-	NO	112	0%	-	**YES**
B-19			84	45.24%	-	100	19%	-	NO	100	19%	-	NO
B-20			87	63.22%	-	91	5%	-	NO	91	5%	-	NO

MODEL/HEURISTICS COMPARISON			**Model**			**Constructive Heuristic**				**Improvement Heuristic**			
Instance #	Instance Features	#of critical nodes	Best Objective	Gap	Debris removed arcs	Best Objective	Gap	Debris removed arcs	Optimal?	Best Objective	Gap from optimal	Debris removed arcs	Optimal?
B-1	SOE=1; BAR=0.19; #of Blocked Arcs=500	7	41	-	-	44	7%	-	NO	41	0%	-	**YES**
B-2			38	-	-	42	11%	-	NO	38	0%	-	**YES**
B-3			39	-	-	42	8%	-	NO	39	0%	-	**YES**
B-4			40	-	-	42	5%	-	NO	41	3%	-	NO
B-5			40	-	-	44	10%	-	NO	42	5%	-	NO
B-6	SOE=2; BAR=0.23; #of Blocked Arcs=613		39	-	-	42	8%	-	NO	39	0%	-	**YES**
B-7			39	-	-	44	13%	-	NO	40	3%	-	NO
B-8			40	-	-	45	13%	-	NO	42	5%	-	NO
B-9			42	-	-	42	0%	-	**YES**				
B-10			38	-	-	42	11%	-	NO	40	5%	-	NO
B-11	SOE=3; BAR=0.54; #of Blocked Arcs=1423		39	-	-	42	8%	-	NO	41	5%	-	NO
B-12			42	-	-	42	0%	-	**YES**				
B-13			46	-	-	46	0%	-	**YES**				
B-14			48	-	-	52	8%	-	NO	52	8%	-	NO
B-15			43	-	-	50	16%	-	NO	50	16%	-	NO
B-16	SOE=4; BAR=0.82; #of Blocked Arcs=2160		61	-	-	62	2%	-	NO	62	2%	-	NO
B-17			58	-	-	64	10%	-	NO	64	10%	-	NO
B-18			51	-	-	52	2%	-	NO	51	0%	-	**YES**
B-19			59	-	-	62	5%	-	NO	62	5%	-	NO
B-20			52	-	-	58	12%	-	NO	56	8%	-	NO

Appendix 8: Detailed heuristic performance of Kartal instances with smaller debris removal effort (Wij')

MODEL/HEURISTICS COMPARISON			**Model**			**Constructive Heuristic**				**2-opt Improvement Heuristic**			
Instance #	Instance Features	#of critical nodes	Best Objective	Gap (14400sec)	Debris removed arcs	Best Objective	Gap to model's best solution	Debris removed arcs	Optimal?	Best Objective	Gap to model's best solution	Debris removed arcs	Optimal?
B-1'	SOE=1; BAR=0.19; #of Blocked Arcs=500	15	74	91.9%	-	76	2.7%	-	NO	73	-1%	-	?
B-2'			84	92.9%	(7,67),(15,60)	79	-6.0%	(7,67),(15,60)	?	74	-12%	(7,67),(15,60)	?
B-3'			78	92.3%	-	74	-5.1%	-	?	74	-5%	-	?
B-4'			80	92.5%	-	76	-5.0%	-	?	73	-9%	-	?
B-5'			73	91.8%	-	77	5.5%	-	NO	73	0%	-	NO
B-6'	SOE=2; BAR=0.23; #of Blocked Arcs=613		75	92.0%	-	74	-1.3%	-	?	72	-4%	-	?
B-7'			80	90.0%	(7,67)	82	2.5%	(7,67),(55,20)	NO	77	-4%	(7,67),(20,55)	?
B-8'			77	92.2%	-	78	1.3%	(55,20)	NO	76	-1%	(20,55)	?
B-9'			81	92.6%	(7,67)	78	-3.7%	(7,67)	?	76	-6%	(7,67)	?
B-10'			76	92.1%	-	75	-1.3%	-	?	70	-8%	-	?
B-11'	SOE=3; BAR=0.54; #of Blocked Arcs=1423		95	91.6%	-	97	2.1%	-	NO	95	0%	-	NO
B-12'			98	91.8%	(37,73)	105	7.1%	(55,20),(37,73)	NO	99	1%	(55,20),(37,73)	NO
B-13'			100	92.0%	-	97	-3.0%	(7,67)	?	97	-3%	(7,67)	?
B-14'			98	90.8%	(7,67)	96	-2.0%	(7,67)	?	93	-5%	(7,67)	?
B-15'			99	90.9%	(5,56)	105	6.1%	(7,67),(56,5)	NO	101	2%	(7,67),(56,5)	NO
B-16'	SOE=4; BAR=0.82; #of Blocked Arcs=2160		135	92.6%	-	125	-7.4%	(7,67),(15,60),(68,61)	?	125	-7%	(7,67),(15,60),(68,61)	?
B-17'			131	100.0%	-	122	-6.9%	(38,19)	?	122	-7%	(38,19)	?
B-18'			165	95.2%	-	152	-7.9%	(44,34),(19,35),(15,29),(56,5),(47,73)	?	152	-8%	(44,34),(19,35),(15,29),(47,73),(56,5)	?
B-19'			134	99.3%	-	118	-11.9%	(68,61),(1,21),(38,19)	?	118	-12%	(68,61),(1,21),(38,19)	?
B-20'			328	99.7%	(11,25),(11,47),(13,26),(13,68),(21,64),(26,36)	125	-61.9%	(56,5)	?	118	-64%	(56,5)	?

MODEL/HEURISTICS COMPARISON			**Model**			**Constructive Heuristic**				**2-opt Improvement Heuristic**			
Instance #	Instance Features	#of critical nodes	Best Objective	Gap from optimal	Debris removed arcs	Best Objective	Gap%	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Optimal?
B-1'	SOE=1; BAR=0.19; #of Blocked Arcs=500	8	52		-	52	0.0%	-	**YES**				
B-2'			58		(7,67)	58	0.0%	(7,67),(15,60)	**YES**				
B-3'			52		-	52	0.0%	-	**YES**				
B-4'			54		-	54	0.0%	-	**YES**				
B-5'			52		-	52	0.0%	-	**YES**				
B-6'	SOE=2; BAR=0.23; #of Blocked Arcs=613		52		-	52	0.0%	-	**YES**				
B-7'			56	26.8%	(7,67)	56	0.0%	(7,67)	NO	56	0%	(7,67)	NO
B-8'			52		-	52	0.0%	-	**YES**				
B-9'			56		(7,67)	56	0.0%	(7,67)	**YES**				
B-10'			52		-	52	0.0%	-	**YES**				
B-11'	SOE=3; BAR=0.54; #of Blocked Arcs=1423		67	34.0%	(7,67)	72	7.5%	-	NO	72	7%	-	NO
B-12'			75	60.0%	(37,73),(55,58)	76	1.3%	(37,73)	NO	76	1%	(37,73)	NO
B-13'			73	57.5%	(7,67)	73	0.0%	(7,67)	NO	73	0%	(7,67)	NO
B-14'			69	27.5%	(7,67)	69	0.0%	(7,67)	NO	69	0%	(7,67)	NO
B-15'			75	50.7%	(7,67)	75	0.0%	(7,67),(56,5)	NO	75	0%	(7,67),(56,5)	NO
B-16'	SOE=4; BAR=0.82; #of Blocked Arcs=2160		91	69.7%	(15,60)	91	0.0%	(7,67),(32,36),(15,60)	NO	91	0%	(7,67),(32,36),(15,60)	NO
B-17'			78	37.2%	-	78	0.0%	-	NO	78	0%	-	NO
B-18'			105	73.7%	(7,67),(15,60),(29,60)	108	2.9%	(7,67),(34,44),(15,29),(56,5),(47,73)	NO	108	3%	(7,67),(34,44),(15,29),(47,73),(56,5)	NO
B-19'			79	44.1%	(7,67)	98	24.1%	(61,68)	NO	98	24%	(61,68)	NO
B-20'			83	67.9%	(15,60),(37,56)	85	2.4%	(60,15),(5,56)	NO	85	2%	(60,15),(5,56)	NO

MODEL/HEURISTICS COMPARISON			**Model**			**Constructive Heuristic**				**2-opt Improvement Heuristic**			
Instance #	Instance Features	#of critical nodes	Best Objective	Gap from optimal	Debris removed arcs	Best Objective	Gap%	Debris removed arcs	Optimal?	Best Objective	Gap	Debris removed arcs	Optimal?
B-1'	SOE=1; BAR=0.19; #of Blocked Arcs=500	7	41	-	-	44	7.3%	-	NO	41	0%	-	**YES**
B-2'			38	-	-	42	10.5%	-	NO	38	0%	-	**YES**
B-3'			39	-	-	42	7.7%	-	NO	39	0%	-	**YES**
B-4'			40	-	-	42	5.0%	-	NO	41	3%	-	NO
B-5'			40	-	-	44	10.0%	-	NO	42	5%	-	NO
B-6'	SOE=2; BAR=0.23; #of Blocked Arcs=613		39	-	-	42	7.7%	-	NO	39	0%	-	**YES**
B-7'			39	-	-	44	12.8%	-	NO	40	3%	-	NO
B-8'			40	-	-	45	12.5%	-	NO	42	5%	-	NO
B-9'			42	-	-	42	0.0%	-	**YES**				
B-10'			38	-	-	42	10.5%	-	NO	40	5%	-	NO
B-11'	SOE=3; BAR=0.54; #of Blocked Arcs=1423		39	-	-	42	7.7%	-	NO	41	5%	-	NO
B-12'			42	-	-	42	0.0%	-	**YES**				
B-13'			46	-	-	46	0.0%	-	**YES**				
B-14'			48	-	-	52	8.3%	-	NO	52	8%	-	NO
B-15'			43	-	-	50	16.3%	-	NO	50	16%	-	NO
B-16'	SOE=4; BAR=0.82; #of Blocked Arcs=2160		59	-	(22,41)	61	3.4%	(22,41)	NO	61	3%	(22,41)	NO
B-17'			57	-	(19,38)	64	12.3%	(38,19)	NO	64	12%	(38,19)	NO
B-18'			51	-	-	52	2.0%	-	NO	51	0%	-	**YES**
B-19'			55	-	(1,21)	58	5.5%	(21,1),(4,20),(38,19)	NO	58	5%	(21,1),(4,20),(38,19)	NO
B-20'			52	-	-	58	11.5%	(61,68)	NO	56	8%	(68,61)	NO