STOCHASTIC MODELING OF POST-EARTHQUAKE ROAD CLEARANCE FOR MAXIMIZING ACCESSIBILITY AND METAHEURISTIC SOLUTION APPROACHES

by

Elifcan Yaşa

Submitted to Graduate School of Natural and Applied Sciences

in Partial Fulfillment of the Requirements

for the Degree of Doctor of Philosophy in

Systems Engineering

Yeditepe University

2022

STOCHASTIC MODELING OF POST-EARTHQUAKE ROAD CLEARANCE FOR MAXIMIZING ACCESSIBILITY AND METAHEURISTIC SOLUTION APPROACHES

APPROVED BY:

Prof. Dr. Dilek Tüzün Aksu ……
(Thesis Supervisor)
(Yeditepe University)

Prof. Dr. Linet Özdamar
(Thesis Co-Supervisor)
(Yeditepe University)

Assist. Prof. Dr. Onur Demir ……
(Yeditepe University)

Assist. Prof. Dr. Semih Yalçındağ ……
(Yeditepe University)

Prof. Dr. Temel Öncan ……
(Galatasaray University)

Assist. Prof. Dr. Seda Baş Güre ……
(Işık University)

DATE OF APPROVAL: ..../..../2022

I hereby declare that this thesis is my own work and that all information in this thesis has been obtained and presented in accordance with academic rules and ethical conduct. I have fully cited and referenced all material and results as required by these rules and conduct, and this thesis study does not contain any plagiarism. If any material used in the thesis requires copyright, the necessary permissions have been obtained. No material from this thesis has been used for the award of another degree.

I accept all kinds of legal liability that may arise in case contrary to these situations.

Elifcan Yaşa

Signature …..……

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to my supervisors, Prof. Dr. Dilek Tüzün Aksu and Prof. Dr. Linet Özdamar for their commitment, all of their time, useful suggestions, and feedback during this research. I also want to give my thanks to the other members of my dissertation committee, Assist. Prof. Dr. Onur Demir and Prof. Dr. Temel Özcan. Special thanks to Assist. Prof. Dr. Seda Baş Güre and Biket Ergüneş who provided the motivation, encouragement, guidance, and advice throughout the Ph.D. journey.

And last but not least, I would like to thank my beloved family, for their tireless support and endless love throughout the duration of my graduate studies.

ABSTRACT

STOCHASTIC MODELING OF POST-EARTHQUAKE ROAD CLEARANCE FOR MAXIMIZING ACCESSIBILITY AND METAHEURISTIC SOLUTION APPROACHES

Disasters have had a great impact on human life throughout the 21st century in terms of loss of life, property, and economic growth. Post-disaster debris clearance is of utmost importance in disaster response and recovery. The goal in planning debris clearance operations in emergency response is to maximize road network accessibility and enable casualty transport to medical facilities, primary relief distribution to survivors, and evacuation of survivors from the affected region. An optimal debris clearance plan leads to fast access to warehouses, temporary survivor camps and medic tents/hospitals. Debris clearance is also important in the affected region's post-disaster recovery because it impacts the speed of business re-openings.

Here, we develop a novel scenario-based mathematical model to represent the road network debris clearance scheduling problem with multiple cleaning crews. The goal is to maximize network accessibility throughout the clearance process of a given road network with blockages. The model's output schedule is consistent in the sense that it takes all clearing time scenarios into consideration. We ensure the reliability of any given solution by determining the required number of different clearance time estimate scenarios using the Sample Average Approximation Method (SAA). The proposed mathematical model can create schedules for small-scale networks due to the curse of dimensionality. Therefore, to deal with larger-scale networks, the designed metaheuristics include combinations or stand-alone versions of Biased Random Sampling (BRS), Tabu Search (TS), Simulated Annealing (SA), and Variable Neighborhood Search algorithms (VNS). To enable the usage of the model in practice, we also propose a rolling horizon approach to revise the initial schedule based on updated clearance time estimates received from the field.

ÖZET

ERİŞİLEBİLİRLİĞİ EN ÜST DÜZEYE ÇIKARMAK İÇİN DEPREM SONRASI YOL AÇIKLIĞININ STOKASTİK MODELLEMESİ VE META-SEZGİSEL ÇÖZÜM YAKLAŞIMLARI

21. yüzyıl boyunca felaketler can, mal kaybı ve ekonomik büyüme açısından insan hayatını büyük ölçüde etkiledi. Afet sonrası enkaz temizliği afete müdahale ve iyileştirmede büyük önem taşımaktadır. Acil durum müdahalesinde enkaz temizleme operasyonlarının planlanmasındaki amaç, yol ağı erişilebilirliğini en üst düzeye çıkarmak ve kazazedelerin tıbbi tesislere taşınmasını, hayatta kalanlara birincil yardım dağıtımını ve etkilenen bölgeden kurtulanların tahliyesini sağlamaktır. Eniyilenmiş bir enkaz temizleme planı depolara, hayatta kalanların geçici kamplarına ve ilaç çadırlarına / hastanelere hızlı erişim sağlar. Bu nedenle, enkaz temizleme operasyonları tüm alanı dikkate alarak eşit erişim sağlamalıdır. Enkaz temizliği, etkilenen bölgenin felaket sonrası kurtarılmasında da önemlidir çünkü bu, işlerin yeniden açılmasının hızını etkiler.

Bu tezde, birden fazla yol açma ekipli karayolu ağı enkaz temizleme çizelgeleme problemi için senaryoya dayalı yeni bir matematik model geliştirilmiştir. Amaç, deprem dolayısıyla tıkanan belirli bir yol ağının temizleme süreci boyunca ağ erişilebilirliğini en üst düzeye çıkarmaktır. Modelin sağladığı çizelgeleme, tüm temizleme zamanı senaryolarını dikkate alması açısından tutarlıdır. Elde edilen çözümün tutarlılığını sağlamak için gerekli olan temizleme süresi tahmin senaryosu sayısını belirlemek için Örnek Ortalama Kestirimi yöntemi kullanılmıştır. Boyutsallık lanetinden dolayı önerilen matematiksel model ancak küçük ölçekli ağlar için çizelge oluşturmasında kullanılabilir. Bu nedenle, daha büyük ölçekli ağlarla başa çıkmak için geliştirilen metasezgisel yöntemler; Yanlı Rastgele Örnekleme, Tabu Arama, Tavlama Benzetimi ve Değişken Komşuluk Arama algoritmalarının kombinasyonlarını veya bağımsız sürümlerini içerir. Modelin pratikte kullanılmasını sağlamak için, sahadan alınan güncellenmiş temizleme süresi tahminlerine dayalı olarak mevcut çizelgenin güncellenmesi için bir kayan ufuk yaklaşımı da önerilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

A	The link(arc) set
BA	Disrupted links set
b	A disrupted link
b^r	The bi-directional partner link of a disrupted link
C	Crew set
D	A bi-directional network
d_a	Travel distance of link a
E_n	Set of links entering node n
F	Function
F_h	The incumbent solution's fitness value
$g_N^m(\hat{x})$	Resulting permutation inserted objective function value
$\bar{g}_N^M(\hat{x})$	Average resulting permutation inserted objective function value
k_b	Integer variable indicating the unique position of precedence of a disrupted link b
L_n	The set of links leaving node n
L	Tabu list length
m_b^N	Independent scenarios
$\tilde{m}_b^\omega$	Clearance duration of a closed link b
M	Total number of replications
N	Total number of scenarios
NP_b	Link b 's biased probability
N_m	Neighborhood operator
N	The node set
$order_{bb'}$	Binary variable indicating if disrupted link $b \epsilon BA$ is cleared before the link $b' \epsilon BA$ by any crew
obj_b	The sum of the lengths of all shortest paths between all node pairs in the network when only link b is blocked and all remaining links are open
P_b	Link b 's priority
p_ω	Probability of occurrence of a scenario ω

$\mathrm{Rank_b}$	The rank broken links b in descending order of their $\mathrm{obj_b}$
S	The set of source nodes
SV	Set of (s, v) source-sink pairs
T	The length of the planning horizon
$\mathrm{t_i}$	The temperature value in iteration i
V	The set of sink nodes
$\widehat{\mathrm{V}}_{\mathrm{N}}$	Sample average function
$\overline{\mathrm{V}}_{\mathrm{N}}^{\mathrm{M}}$	Overall average of sample average function
$\mathrm{x_{sv}^{at\omega}}$	Positive variable indicating if arc a is on the shortest path between source s and sink v in period t in scenario ω
$\mathrm{y_b^{t\omega}}$	Binary variable indicating whether disrupted link $b \epsilon BA$ is traversable in period t in scenario ω
$\mathrm{z_{cb}^{t\omega}}$	Binary variable indicating if disrupted link $b \epsilon BA$ has been cleared by crew c in period t, in scenario ω
σ_{gap}^2	Variance of gap
$\tau_{\mathrm{b}}^{\omega}$	Clearance start time of a disrupted link $b \epsilon BA$ in scenario ω
β	A parameter in the range [0,1]
ω	A scenario
Ω	Set of repair time scenarios
Δ	The difference between the fitness values of the two neighboring solutions
$\%\ gap_{NM}$	Percentage optimality gap
BIG	A very large positive number
BRS	Biased Random Sampling
C	Caddebostan network
CPLEX	An optimization software package
CPU	Central processing unit
DCP	Debris Clearance Problem
gap_{NM}	Optimality gap

GIS	Geographic Information System
GRASP	Greedy Randomized Adaptive Search Procedure
HTSSA	Hybrid Tabu Search and Simulated Annealing
ILS	Iterated Local Search
K	The list of neighborhoods
L	Length of tabu list
MIP	Mixed integer programming
NI	Non-improving
P	Pendik network
PC	Personal Computer
RAM	Random-access Memory
S	Start-over
SA	Simulated Annealing
SAA	Sample Average Approximation
TS	Tabu Search
TSP	Traveling Salesman Problem
VNS	Variable Neighborhood Search

1. INTRODUCTION

The disaster management cycle consists of preparation, response, recovery, and reconstruction phases [1]. Here, we consider the road network debris clearance activities conducted in both post-disaster response and recovery phases. Post-disaster debris clearance involves pushing roadside blockages to the curbs and eventually transporting them to dump sites. This activity is essential in the early response phase where casualties need to be transported to medical facilities, and relief has to be distributed to survivors. Additionally, open roads are required for residents who wish to relocate from the affected region. In the recovery phase, debris clearance enables businesses to re-open so that life may return to normal. Çelik [2] emphasizes the post-earthquake situation in Haiti where survivors could not receive critical relief materials over an extended period of time due to blocked roads.

In this study, we deal with the post-disaster debris clearance scheduling problem with multiple crews with the goal of maximizing road network accessibility in each period of the planning time horizon. In a post-disaster situation, the road network's connectivity is observed via helicopter surveys and/or satellite images (as discussed in the GIS-based debris assessment study by Karimzadeh et al. [3]. At this point, we assume that the specific parts of the road network that are blocked are known with certainty. However, clearance durations for each blocked road segment are uncertain and estimated. Note that when satellite images of the road network are acquired after a disaster strikes, information regarding the blocked road segments and the estimates for the clearance durations of each blocked segment become available, but the actual debris clearance durations cannot be known with certainty until after the clearance job is completed. Due to this uncertainty of clearance durations, prioritization of clearance work needs to be made based on estimated clearance durations. In this article, we offer a stochastic solution approach that aims to create a clearance sequence for blocked road segments that maximizes the expected accessibility of the road network based on stochastic clearance durations. To achieve this, we propose a two-stage stochastic mathematical model. In the first stage of this model, clearance jobs of blocked segments are sequenced such that the expected network accessibility over multiple scenarios of clearance durations is maximized. In the second stage, assignment of clearance crews to blocked segments is performed individually under each scenario based on the clearance durations of that scenario and the clearance sequence obtained in the first stage. Here, we define the

network's accessibility measure as the total length of all shortest paths between all pairs of junctions (nodes) in the network. As illustrated by [4], this measure maximizes network accessibility dynamically in each period of the cleaning process. The model output provides an optimal sequence of clearance jobs that minimizes the expected value of the total length of shortest paths between all node pairs over the planning horizon based on a range of clearance duration estimates for blocked links across all scenarios. It is important to note that the model recommends a single sequence of clearance jobs as opposed to a different sequence for each scenario. Since scenarios only represent possible outcomes, the objective of the model is not to create a solution for each scenario, but to use these scenarios to provide a reliable plan (in the form of a sequence of clearance jobs) that can be used to prioritize the post-disaster clearance jobs. Note that the actual clearance duration of each segment is unknown until the clearance of that segment is completed. Therefore, immediately after a disaster, prioritization of clearance work has to be performed using duration estimates obtained from satellite imagery and/or helicopter surveys. However, as new information is communicated by clearance crews in the field, the model should be run again during the course of the recovery process to revise the clearance sequence based on the latest duration estimates. Figure 1.1 displays the rolling horizon approach that we propose to address the online nature of the debris clearance problem (DCP). Immediately following a disaster, DCP is initially solved using preliminary estimates from helicopter surveys and/or satellite images. This solution is then used to assign the available crew(s) to their first clearance jobs. As the jobs are in progress, clearance duration estimates are updated based on the information from the crew working in the field. Upon completion of the first clearance job, DCP is solved again using the latest duration estimates obtained so far. (To be more precise, the DCP run should be initiated some time before the job is completed to ensure that the updated sequence is available at the time of job completion.) The freed-up crew is then assigned to the first job on the revised sequence from the new DCP solution. This process is repeated until all blocked links have been cleared.

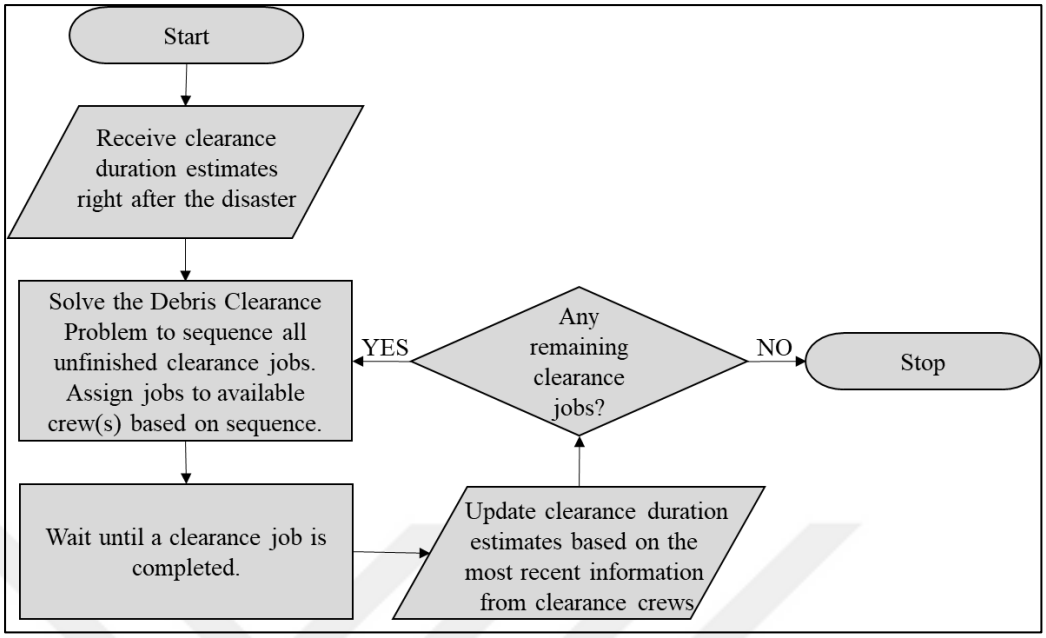

Figure 1.1 Rolling horizon approach proposed to solve the online debris clearance problem

The stochastic programming model for DCP assumes that a blocked link (also referred to as a disrupted or broken link) cannot be utilized on any path unless it is cleared. Both the status of the blocked links and the assignment of tasks to crews lead to binary variables, resulting in a Mixed Integer Program (MIP) that is intractable for larger-scale networks. Moreover, the online nature of the DCP requires it to be solved many times during the course of the recovery efforts as depicted in Figure 1.1. Therefore, we need an alternative approach that can produce solutions quickly so that the crew assignments can be communicated in a timely manner. For this purpose, we design metaheuristics involving Simulated Annealing (SA), Tabu Search (TS), Variable Neighborhood Search (VNS), and Biased Random Sampling (BRS) methods by combining them into new algorithms and embedding some novel neighborhood definitions in them. We test the performance of these algorithms on a couple of real road networks within different counties in Istanbul. After selecting the best performing algorithm among them, we employ the Sample Average Approximation (SAA) method (described in Kleywegt et al. [5]), to identify the optimal number of clearance duration scenarios for each broken link so that the problem is represented properly. Finally, we validate the reliability of the achieved solution according to some measures used in SAA.

The main contributions of this work can be summarized as follows:

- We introduce a new problem to post-disaster debris clearance literature for the coordination of debris clearance activity such that accessibility of the entire road network over the planning horizon is maximized.
- We propose a novel two-stage stochastic model for the DCP that provides a clearance schedule based on a range of disaster scenarios.
- We develop an efficient and effective solution methodology to solve the DCP. We first employ a BRS approach to create high-quality initial solutions and design efficient metaheuristics that provide high-quality solutions for large realistic instances of this problem. We also utilize an SAA technique to ensure that the number of scenarios used in the stochastic model is sufficient to provide a good representation of the true problem.
- We propose a rolling horizon approach for using the DCP in a practical setting, in which the clearance sequence is revised as clearance duration updates are communicated from the field in an online fashion.
- Overall, the proposed approach offers a straightforward methodology that can be implemented easily at the time of the actual disaster to prioritize debris clearance work and schedule work dynamically to clearance crews.

2. LITERATURE REVIEW

Debris clearance is the activity executed right after an extreme event, pushing roadside blockages to the curbs so that relief, evacuation, and other vital logistics operations can be carried out. In the roadside debris clearance problem, there are many different ways to represent the network and model the scheduling operations along with other response phase operations, such as relief distribution or evacuation of the survivors. As Çelik [2] points out, the debris clearance literature is quite new and limited. In the following three subsections we present a review of the related literature. The literature review section comprises of three subsections. The first one reviews articles published on the debris clearance problem, whereas the second subsection summarizes stochastic approaches to humanitarian logistics problem in general. Finally, third one discusses the literature on metaheuristic applications to stochastic problems.

2.1. DEBRIS CLEARANCE PROBLEM

We group the literature on the topic according to the structure of the objective functions described in the models. Basically, three categories of objectives exist: minimizing makespan (completing all tasks as soon as possible), maximizing satisfied demand over time, and maximizing network accessibility throughout the process. Below we review the literature on these three categories with particular emphasis on network accessibility. Additionally, under the category of hybrid objectives we present studies with combined objectives for the debris clearance problem.

Network accessibility models based on all shortest paths: In the model introduced by Özdamar et al. [6] , the objective is to minimize the pairwise distance between each pair of nodes in the network during a given planning horizon. The authors also consider minimizing the maximum completion time of all clearance work (that is, the makespan). The model is represented by a recursive mixed-integer program. A constructive heuristic that uses different job sequencing rules is introduced to solve the problem. The authors consider both deterministic and probabilistic debris clearance durations to analyze the effect of a priori knowledge on debris clearance durations. Averbakh and Pereira [7] present a flow-based network repair problem. The models minimize both makespan of the repair process and the

average completion times of each repair. The first objective model uses a single commodity network flow formulation of the minimum spanning tree problem with node recovery constraints. A Branch and Bound algorithm is used where the solution space is restricted by lower and upper bounds obtained by using simple heuristics.

Partial network accessibility models based on shortest paths between demand-supply nodes: Sakuraba et al. [8] consider the problem of minimizing the (weighted) shortest paths between the current locations of all work crews to all gathering points (temporary camps) in a region. This model targets opening the paths to temporary camps rather than considering all road junctions in the network. The model is deterministic, and the authors propose two constructive heuristics tested on small instances. Maya Duque et al. [9] limit the travel distance between demand and depot nodes by an upper bound based on pre-disaster shortest paths. The model defines a multi-period network repair crew scheduling and crew routing problem. They propose a Greedy Randomized Adaptive Search Procedure (GRASP) to deal with larger-scale networks. In an earlier study, Duque and Sörensen [10] minimize the weighted sum of shortest paths from all villages to the closest regional center under budget constraints. GRASP is combined with a variable neighborhood search (VNS) algorithm to solve the problem.

Demand satisfaction objectives: Yan and Shih [11] propose a time-space network flow formulation for the multi-crew road repair and relief distribution problem and solve it using a sequential heuristic approach that decomposes the two decisions. Later, the authors develop an ant-colony based hybrid algorithm [12]. Çelik et al. [13] propose a dynamic stochastic debris clearance problem with the goal of maximizing demand satisfaction. The debris amount on the roads is not known and the knowledge about the situation is updated in each period. A Markov decision process is used, and a heuristic approach is proposed. Moreno et al. [14] propose a mixed-integer linear programming model for crew scheduling and routing problem using a partial accessibility-related objective that minimizes the time intervals during which demand nodes stay inaccessible from the depot. A Branch-and-Benders-Cut algorithm is used where the master problem addresses crew scheduling decisions, and the sub-problem addresses the crew routing decisions. Ulusan and Ergun [15] aim to satisfy demand with an exponential earliness reward where a new centrality measure based on the shortest paths between supply-demand node pairs is defined to prioritize demand urgency. A constructive heuristic that prioritizes blocked links according to several

centrality measures is used to solve the problem. Morshedlou et al. [16] propose a model that maximizes the ratio of pre-disaster to post-disaster flows at demand nodes. The authors use a relaxed formulation by ignoring the timing of restoration crews to obtain a lower bound for the objective and employ some pre-processing and feasibility algorithms to solve timing conflicts of common works between groups.

Makespan related objectives: In Furuta et al. [17], the uncertainty related to delay in road restoration works due to aftershocks is considered. Repair time uncertainty is represented by the Gaussian distribution. The authors propose a genetic algorithm tailored to this problem. Tang et al. [18] present a road repair problem that is not related to post-earthquake conditions but has similarities to our problem. The study considers road repair crew routing in a time-space network formulation with stochastic vehicle traversal times and repair times. They present a robust optimization model that minimizes penalties for deviating from expected job completion times. Tuzun Aksu and Ozdamar [4] propose a path-based debris clearance scheduling model that allocates a limited number of clearance crews to clear several main paths with the goal of minimizing the total weighted clearance completion times of all paths. After solving the debris clearance model, the authors allocate clearance equipment to crews. An equity-based objective that minimizes the maximum difference between path completion times is also proposed. Sahin et al. [19] develop a variant of the generic vehicle routing problem for road repair to reach all critical nodes in minimal duration. Kasaei and Salman [20], Akbari and Salman [21], and Akbari and Salman [22] propose multi-vehicle arc routing models for the debris clearance scheduling problem. Kasaei and Salman [20] propose two arc routing models for clearing the route from a source node to a sink node. The first model's goal is to minimize traversal and unblocking time over the route (makespan). The second model considers two criteria: maximizing total collected prize and minimizing the walk time. Akbari and Salman [22] study multi-vehicle prize collecting connectivity arc routing problem to minimize the makespan of tours that start and end at depots. Tour lengths include traversing edges, debris clearance times, and waiting times. The model consists of constraints involving flow balance, traversing, and unblocking edges, cut-sets between connected and unconnected sets, time limit restrictions, earliest traversable time for an edge, traversal and arrival time association, and arrival time calculation. The authors use a relaxed version of their exact formulation by ignoring timing conditions and then apply local search procedure to improve the solution. In a similar study, Akbari and Salman [21] consider the

same model found in [22] to maximize the total benefit gained with connecting components within a time limit. In this work, they use Lagrangian relaxation to solve the relaxed formulation in [22] updating the Lagrangian multipliers in each step with the subgradient optimization technique. Akbari and Salman [23] aim to minimize the makespan of repair activities and use local search to solve the problem while relaxing crew timing conditions. Berktaş et al. [24] propose a Traveling Salesman Problem (TSP) model with two objectives, the first objective is makespan minimization and the second objective minimizes the weighted sum of critical node access times. The authors use a heuristic approach that routes the crews over the shortest paths between critical nodes. Ajam et al. [23] solve an arc routing problem adapted from the multi-vehicle TSP. A lower bound is derived for the makespan objective and metaheuristics are proposed to solve large-scale problems. The first method is a matheuristic that uses a TSP formulation position in a permutation of visit for each node without any blocked edges. The solution is a visiting order with minimum latency with shortest path times on the edges with objective correction. The second approach is a metaheuristic where solutions are improved with an iterative method. The authors suggest a local search-based metaheuristic with a local optimum escape procedure. Another study that minimizes makespan is found in Caunhye et al. [25] propose a set of deterministic and two stage robust optimization models solved with and without a priori knowledge of uncertainties with the goal of minimizing the makespan. The study uses a robust optimization approach to handle uncertainty in debris clearance times. The first stage selects edges “proactively” before disaster and the schedule is created after the disaster strikes.

*Hybrid objectives:*A multi-objective network optimization model for relief distribution and road repair is proposed by Ransikarbum and Mason [26] under budget and capacity constraints. The objectives considered are equity-based demand satisfaction, minimizing the sum of unmet demand, and minimization of repair and distribution costs. All objectives are assigned weights and summed up in a single function. Li and Teo [27] propose a bi-level model that considers both shortest path-based accessibility and demand satisfaction. In the first level, the model maximizes weighted cumulative accessibility of the network using a modified version of the accessibility measure proposed in Özdamar et al. [6]. The authors add expected traffic flow weights to link traversal times while calculating pre- and post-disaster shortest paths among all pairs of nodes. In the second level, total relief delivery time is minimized, thereby targeting demand satisfaction. The lower-level model performs relief

logistics as a network flow problem. The authors employ a parallel genetic algorithm as a solution method. Aslan and Çelik [28] compose a two-stage stochastic model that considers relief inventory prepositioning network in the first stage, and, road repair in the second stage. In the first stage, warehouse location and inventory prepositioning decisions are made where uncertainties pertain to relief demand and road damage. In the second stage, relief transport-related road repair decisions are established. The objective in the first stage consists of the fixed warehouse building and inventory costs whereas relief distribution, road repair, and outsourcing costs are considered in the second stage under budget constraints. The study focuses on predetermined paths reopening between potential warehouses and distribution centers where direct path availability, travel times, and demand are stochastic parameters. Similar to Aslan and Çelik [28], Sanci and Daskin [29] consider a two-stage stochastic integrated facility location and network repair model under demand, damage and repair time uncertainty with the goal of minimizing facility, relief distribution, and unmet demand costs. In the first stage, the number and the location of restoration equipment and emergency response facilities are decided to minimize facility costs and in the second stage, restoration equipment is assigned to damaged links with the goal of minimizing transport costs and costs of unmet demand. SAA method is used to solve a large stochastic model. Note that the two-stage models in [28] and [29] have more tactical decisions (such as inventory positioning and facility location) in the first stage and they address the operational decisions regarding road repair in the second stage. These studies assume that the uncertainty regarding repair durations is revealed completely at the time repair decisions are made. In our study, we define a more realistic problem setting in which actual repair times are not deterministically known until job completion. Shin et al. [30] propose an integrated repair crew scheduling and relief distribution model that minimizing last relief service time. The proposed solution method is the Ant Colony Algorithm. Rezaei et al. [31] compose a multi-stage stochastic mixed-integer model for relief distribution and debris clearance. The model minimizes a weighted unmet demand function that is defined between distribution centers and demand points while clearing predetermined links. Non-anticipative events are modeled through scenario trees. Ransikarbum and Mason [26] address emergency repair of supply-demand accessibility and relief distribution simultaneously under two setup perspectives as pooled-budget/separate budgets for restoration and relief and partial/full restoration. The restoration activities are organized so that the supply of relief is distributed equitably as well as other

objectives such as minimizing unsatisfied demand and minimizing cost and provide an efficient frontier.

In Table 2.1, we present a summary of the papers published in DCP literature that bear the most similarity with this dissertation in terms of the problem setting and the solution approach. In line with the classification used above, the first four columns identify the objective function used in the study. In the first column titled "Accessibility", PA refers to studies that focus on partial accessibility between certain demand and supply nodes. The last column indicates whether the study considers stochasticity in its input data.

Table 2.1 Contribution Summary

Citation	Accessibility	Makespan	Demand Satisfaction	Hybrid	Stochastic/Deterministic
Özdamar et al. [6]	✓				Deterministic
Averbakh and Pereira [7]	✓				Deterministic
Sakuraba et al. [8]	PA				Deterministic
Maya Duque et al. [9]	PA				Deterministic
Duque and Sörensen [10]	PA				Deterministic
Çelik et al. [13]			✓		Stochastic
Morshedlou et al. [16]			✓		Deterministic
Furuta et al. [17]		✓			Stochastic
Caunhye et al. [25]		✓			Deterministic/Stochastic
Li and Teo [27]				✓	Deterministic
Aslan and Çelik [28]				✓	Stochastic
Sanci and Daskin [29]				✓	Stochastic
Rezaei et al. [31]				✓	Stochastic
This Dissertation	✓				Stochastic

Accessibility of the entire network is important in a post-disaster situation because of the transportation requirements of the survivors, relief effort, and demand arising from other

situations such as fire, etc. The works in the literature that consider same network accessibility objective as ours are Özdamar et al. [6] and Averbakh and Pereira [7]. However, these studies both consider deterministic repair times. Our contributions to Özdamar et al. [6] are the stochastic setting of the problem and the metaheuristics that are proposed for the solution of the resulting stochastic problem. The other study that considers connectivity between all pairs of nodes is Averbakh and Pereira [7]. They present a flow-based network construction problem where they use a single commodity network flow formulation of the minimum spanning tree problem with node recovery constraints. However, in this dissertation we propose a new debris clearance mathematical model that comprises incremental repair activities represented as binary variables with a stochastic viewpoint and solve the stochastic model accordingly. Most of the studies in debris clearance literature given in Table 2.1 deal with reopening of a predetermined set of paths and focus on the connectivity between certain points in the network regarding the accessibility in the network. Sakuraba et al. [8], Maya Duque et al. [9], Duque and Sörensen [10] consider partial network accessibility objectives and deterministic clearance times in repair activities. Çelik et al. [13], Morshedlou et al. [16], Rezaei et al. [31] are the studies that focus on accessing a predetermined set of points. Çelik et al. [13] propose a dynamic stochastic debris clearance problem to maximize demand satisfaction. Although stochastic approach employed in this study resembles the one used in our dissertation, their study differs from ours in that it uses a demand satisfaction-based model that focuses on the connectivity between demand and supply points. Caunhye et al. [25] and Furuta et al. [17] also use a different objective function that minimizes the time it takes to complete all clearance activities. Under the hybrid objective category, the work of Li and Teo [27] has the same perspective of network accessibility as ours, however this study considers deterministic repair times. Finally, the two-stage stochastic models in Sanci and Daskin [29] and Aslan and Çelik [28] consider more tactical decisions (such as inventory positioning and facility location) in the first stage and they address the operational decisions regarding road repair in the second stage. These studies assume that the uncertainty regarding repair durations is revealed completely at the time repair decisions are made. The two-stage model we propose is a more realistic problem setting in which actual repair times are not deterministically known until job completion.

2.2. STOCHASTIC HUMANITARIAN LOGISTICS STUDIES

In this section of the literature review, stochastic models in humanitarian logistics literature will be discussed. Although stochastic debris clearance literature is limited, in humanitarian logistics literature, studies considering the inherent uncertainty have extensive applications. Stochastic programming, two-stage stochastic programming, robust optimization approaches are seen most often in humanitarian logistics literature. Not quite often as others, simulation, probability and statistics models and decision theory can be encountered as well, see [32].

Liberatore et al. [33] make a classification of stochastic components in studies for humanitarian logistics in their review. According to Liberatore et al. [33], five major parameters can be seen as stochastic components of studies such as demand, demand location, affected areas, supply, and the transportation network. Also, in this review author states that the most common stochastic component is demand in humanitarian logistics. Additionally, Balcik et al. [34] state a remark of high frequency in demand discretization in uncertain humanitarian supply chain studies.

For preparedness stage facility location and resource allocation Jia et al. [35], Balcik and Beamon [36], Rawls and Turnquist [37, 38], Salmeron and Apte [39] considered demand uncertainty or demand and supply location uncertainty. Falasca and Zobel [40] studied a two-stage procurement model for humanitarian relief supply chains with demand uncertainty. In Döyen et al. [41] the study composes a two-echelon stochastic facility location for humanitarian relief logistics and aside from demand and the condition of facilities, costs are uncertain. Rawls and Turnquist [38] used a scenario approach and considered the possibility of infrastructural damage in emergency response in pre-positioning planning. Wang et al. [42] proposed a two-stage stochastic model considering demand uncertainty after a disaster for emergency resource storage. Peeta et al. [43] consider the case where link failure is random on a network to manage links to be strengthened before a disaster occurrence for connectivity.

For the response stage Tzeng et al. [44] composed a fuzzy multi-objective linear programming model for relief delivery systems with an uncertainty analysis of achievement for three of each objective. Yan and Shih [45] considered error tolerance in their roadway

repair and relief distribution model to cope with the uncertainty in available information. Stepanov and Smith [46] consider uncertainty in demand for the stochastic emergency evacuation problem. Zhang et al. [47] took a secondary disaster occurrence into account in the multiple-disaster multiple-response emergency allocation and disaster relief model. Barbarosoglu and Arda [48] composed a two-stage stochastic programming model for transportation planning where road capacities, supply availability, and demand is uncertain. Mete and Zabinsky [49] proposed a two-stage stochastic program where demand for medical supplies is uncertain for the storage and distribution problem of medical supplies. In similar Bozorghi-Amiri et al. [50, 51] designed a relief supply chain where demand and the costs are uncertain. Rennemo et al. [52] used uncertainty in demand in a three-stage stochastic facility routing for relief transportation as well as road capacity uncertainty. Similarly, Noyan et al. [53] designed a two-stage stochastic programming supply chain network with recourse considering uncertainties in demand and road capacity. Further Najafi et al. [54] composed a multi-objective robust optimization model for logistics planning with uncertainty in the number of casualties meaning in demand. Since the study represented here focuses on debris operations after an earthquake, stochastic disaster operations management literature is not discussed in great detail. For further information see reviews on disaster logistics models that involve uncertainty Özdamar [55], Liberatore et al. [33], Grass and Fischer [56], Hoyos et al. [32].

2.3. METAHEURISTICS APPLIED TO STOCHASTIC OPTIMIZATION PROBLEMS

The following part of the literature review is dedicated to metaheuristics applied to stochastic optimization problems. For stochastic programming problems most used trajectory-based metaheuristics are Tabu Search (TS), Iterated Local Search (ILS), Simulated Annealing (SA), Variable Neighborhood Search (VNS), and the approaches are not limited to these. For further information, we refer the reader to review papers on metaheuristic applications to stochastic problems [57–59]. We focus our literature review on TS, SA, and VNS applications when uncertainty is present since these are the three metaheuristics that we use in our research.

TS is first introduced by Glover [60, 61] and is applied to several types of problems. In stochastic optimization, although there are many different solving approaches L-shaped method, Lagrangian relaxation, etc. [62, 63], TS has particular attention in vehicle routing literature when demand and/or customers are uncertain. Gendreau et al. [64] used TS for vehicle routing problem with stochastic demands and customers. Since the exact evaluation of a move is too time-consuming, it is proposed to use a proxy for the objective function. Used proxy is not the exact calculation of the objective function but rather a strongly related representative for the move evaluation. It was found important to compute the true value of the objective function for each of the five best neighbors determined by the proxy when selecting a move. For the neighborhood, they used p nearest neighbor insertion for randomly selected customers. Tabu length size is decided randomly in a predetermined interval. For chance constraint programming Roberto Aringhieri [65] used TS as short-term memory. The simulation was used to find approximate probabilities that each constraint was violated. Two different move evaluations are considered, allowing infeasible solutions with a penalty given to the objective function and avoiding infeasible solutions. For evaluation of moves Aringhieri [65] used average demand to evaluate neighborhood and test probabilistic feasibility. For the move acceptation, two versions were tested, one that satisfies chance constraints and another that allows infeasibility but penalizes in the objective function. Bianchi et al. [66] studied two different approximations of the objective function to see the eligibility of proxy functions. The first approximation is based on vehicle routing tour building whereas the second one uses tour length calculations as it were a TSP. The study is comprised of ILS, TS, SA, Ant Colony Optimization, Evolutionary Algorithms as search mechanisms. As aspiration criteria, tabu moves are evaluated and selected with a probability of 0.3. Haugland et al. [67] conducted TS to solve a two-stage stochastic program with recourse in designing delivery districts for the vehicle routing problem with stochastic demands. Composing delivery district decisions are the first stage decisions and vehicle routing decisions for each district are the second stage decisions. Since the calculating expectation of the objective function requires routing for each district, they used an approximation of objective function value with an upper bound calculation. Moving a vertex from a district to another with border and connectivity conditions satisfied is the neighborhood generation mechanism. District-vertex assignments are tabu and the high frequency of swapping two vertex is penalized in the objective function respective term. Sungur et al. [68] consider a variant of the vehicle routing problem with time windows as a

courier delivery problem. Uncertainty in the problem is the uncertain service times and probabilistic customers. They developed a heuristic approach based on TS and the insertion heuristic. A master plan is formed according to the high frequency of customers and worst-case service times. Additionally, the study solves one routing problem for each scenario accompanied by the exercise of skipping customers if not appear in the respective scenario and inserting new ones. Building an initial solution is comprised of these mentioned steps together with prioritization of unscheduled customers through the frequency of the customer is seen in scenarios and update the master plan. Two different neighborhood generation is used, and tabu list length is random while no improvement is the termination criteria. Shukla et al. [69] used algorithm portfolios for the vehicle routing problem with stochastic demands. The difference from the classical vehicle routing is that the vehicles are allowed the deviate from the given route. The study is comprised of four metaheuristic algorithms and the same algorithms with three different neighborhood generation mechanisms such as: exchanging, insertion, and reversing. Tabu starts with a feasible solution and explores the neighborhood, next re-exploration takes place. As the last step, the algorithm explores infeasible solutions. Teng et al. [70] approach the vehicle routing problem with stochastic demand with a comparative study regarding TS and SA. The neighborhood of a solution consists of only a node position exchanged with only nearest neighbors and the neighborhood size is restricted. Tabu list size is set to 5 for the smaller size of nodes and $\sqrt{n}$ for larger networks. The study keeps track of visited nodes and penalizes by frequency. Furthermore, for the tabu algorithm, the authors keep track of no-improvement and use a diminishing threshold value for the improvement percentage between visited solutions.

TS is used for other problems in stochastic programming as well. Costa and Silver [71] solved a stochastic problem in the cause and effect analysis framework. An upper bound is found to reduce the search area with implicit enumeration and used descriptive sampling as well. Similarly, to reduce the neighborhood size, the study considers only a predetermined number of random neighbors. Tabu list size is determined according to multiple descent iterations. The main contribution of this paper is move evaluation. While comparing the current solution and best neighbor a small number of scenarios are generated and the expectation of objective functions are calculated accordingly. With small sample results hypothesis tests such as Wilcoxon or Median tests, are conducted to seek evidence that the rest of the sample should be included in the move evaluation. To diversify solutions close

swaps in the permutations are forbidden. In a disaster-related study, Salman and Yücel [72] considered where to locate emergency facilities regarding random network damage after a disaster. The study takes correlated link failures into consideration and supply and demand point connection through enduring paths after a disaster. As for the tabu mechanism, the study starts with a random solution, uses pairwise exchange, takes the best neighbor, and accepts ten percent of the non-improving solutions. Şeker and Noyan [73] used TS to solve airport gate assignment problems where arrival and departure times are uncertain. For the initial solution, the study takes a relaxed solution that satisfies a part of the constraints taken from an exact solver. The moves are generated according to three schemes: swap, mutation, and insert and used interchangeably considering the current neighborhood, allowing infeasibilities with a penalty in the objective function, and tabu list length changes according to problem size. Termination of the metaheuristic depends on one of the three conditions: the maximum number of iterations, the maximum number of non-improving iterations, or max computation time. Liu et al. [74] start with a random initial solution to the facility layout problem with uncertain logistic flows. The study employs different goals for the fitness calculations, adds new introduced products central point, and checks feasibility, if the place is full then re-layouts. As tabu, the study uses the areas assigned on the contrary to products, considering the feasibility of the particular place. Marchand et al. [75] used TS in finding operating rules for hydropower planning where inflow is stochastic. TS is executed for n iterations where n is a random variable between a maximum and a minimum number interval given by the user. Used first improvement when consecutive two plants swap for neighborhood generation that is randomly sorted. For non-improving neighbors, the metaheuristic carries on with the minimum objective value. Schepler et al. [76] used TS in a two-stage stochastic programming environment where discrete berth allocation is considered with the uncertain arrival time of vessels. In the first stage, vessels are assigned to berths, and sequences of vessels are defined whereas in the second stage actual schedule is revealed. Neighborhood is defined as moving a vessel to a different berth and add this assignment to tabu list. Non-improving moves are penalized according to frequency of assignment to that place. The authors composed an iterated version of TS. Chiyoshi and Morabito [77] solves the problem of extended maximal availability location where the objective is to locate servers to maximize expected coverage of demand. The algorithm starts with a random solution and generates neighborhood with node insertion, the node to be inserted is tested against all other nodes that are not in the solution. Effect of the node insertion is calculated

with non-linear model for that particular configuration of servers. The leaving node is made tabu for a number of iterations and for non-improving neighborhoods a node is probabilistically selected. To intensify the search in more promising areas tabu is run for 15 times and the best solutions found are taken and start over the search with those solutions. Another phase of the algorithm is to find least frequently used nodes in the solution pool and run the algorithm. Search is limited by a predetermined number of iterations and for the tabu list length number of servers and half of number of servers and two random numbers that are between former upper bounds, and one are used.

Simulated Annealing is another metaheuristic that mimics the metal annealing process and was first introduced by Kirkpatrick et al. [78] to the literature. Walter J. Gutjahr [79] compiles two different metaheuristics for multi-objective stochastic combinatorial optimization problems and applies the methodology to bi-objective stochastic traveling salesperson problem with time windows problem where service times inherits stochasticity. The expectation of the objective function is calculated via sampling and taking the average of the scenario's objective values to obtain the overall value. For the diversification well-known 2-opt algorithm is used. Mohammadi and Forghani [80] applied SA to a stochastic berth allocation problem where arrivals and handling times are stochastic parameters. Designated encoding is pairing the vessels with the berths, and the neighborhood generation is accomplished by single or double swap or shifting. The algorithm starts with a 70 percent acceptance probability for non-improving moves. Tang et al. [81] considered a bike repositioning problem in a bike-sharing system with uncertain demand. The study modeled the system as a two-stage programming model where the first stage is routing, and the second stage is the loading and unloading decisions of repaired or damaged bikes to be collected. The initial solution is constructed with a greedy approach with the nearest consideration and adjusts loading and unloading decisions to make a feasible solution. Local search is performed on routing decisions while generating second stage decisions accordingly. Neighborhood structure consists of the swap, relocate, and 2-opt mechanisms with an equal probability of being applied. SA terminates when the temperature drops to a predetermined threshold value. Teng et al. [70] compared TS and SA algorithms for vehicle routing problem with stochastic demands. Initial solution is constructed randomly by a nearest neighborhood heuristic. In SA used a node exchange to generate a neighborhood solution only with the nearest nodes. Termination is made according to thermal equilibrium that is the last

improvement is smaller than a threshold value. Fattahi and Govindan [82] studied forward/reverse logistics network design as a two-stage stochastic model under return and forward demand uncertainty. The study considers a relaxation of a binary variable and finds an upper bound through which SA is used to search a subset of first-stage decision variables. The fitness is approximated with simulation to derive the upper bound. The local search to the overall problem is carried through a potential location matrix. Condradie et al. [83] modeled a coal handling facility schedule via SA. Coal blends are sent to consuming facilities in which demand is stochastic. The scheduling problem is handled as a two-stage problem where the first stage decisions are extraction, reclaiming, and stacking of coal and in the second stage additional load-in of coal from dead or strategic stockpiles. A matrix that contains stacking and reclaiming coals in tons is implemented for encoding and for the neighborhood generation a step size is considered. The study sets the initial temperature to 500 and reduces it slowly. Ritt and Costa [84] solved assembly line worker assignment and balanced the line as a two-stage stochastic programming model. In the first stage task assignments to workstations are carried out and in the second stage worker assignments to workstations are decided. To generate an initial solution the study simplifies the problem assigns minimum task time to stations and satisfies precedence constraints. Local search is carried out as shifting a task from its current station to another and swapping two tasks assigned to two different stations and best improving neighbor is selected. SA starts from a very low temperature and double the value until the acceptance probability exceeds a desired value where a geometric cooling schedule is implemented. Termination criterion are either a last number of n iterations that did not improve the solution or the acceptance probability dropped below to a threshold value.

Variable Neighborhood Search (VNS) is an algorithm, first introduced by Hansen and Mladenovic [85], that systematically alters neighborhood operators throughout the search. In debris clearance literature VNS is used in the following studies. Kasaei and Salman [20] used variable neighborhood descent and defined an increasing parameter to accept worsening solutions that are incremented according to the incumbent solution objective value. Also, Ajam et al. [23] used VNS as an improvement step. Outside of the VNS structure, a shaking procedure is used that removes a percentage of the solution and inserts it back. VNS procedure starts back with the first neighborhood structure when found a better solution. VNS is used in different combinatorial optimization problems and stochastic

optimization problems as well. For a general picture of VNS see [86] and for the applications of VNS in stochastic optimization see [58].

So far, we have discussed the model structures and objectives of the deterministic post-disaster debris clearance problem and uncertainty-related disaster operations management studies and the TS, SA, and VNS metaheuristics applied to stochastic problems. As discussed above, the few models that consider shortest path-based accessibility measures mostly concentrate on maximizing partial network accessibility that is related only to critical nodes in the network such as relief demand points and depots. As discussed above, the few models that consider shortest path-based accessibility measures mostly concentrate on maximizing partial network accessibility that is related only to critical nodes in the network such as relief demand points and depots. The only previous study similar to the one proposed here is that of Ozdamar et al.'s [6] where the accessibility of the whole network is maximized. The limitations of their study are that repair times are deterministic and only basic heuristics are developed. Our contributions to Ozdamar et al. [6] are the stochastic setting of the problem and the metaheuristics that are adapted as solution methods. We note that in a post-disaster situation, considering the whole network may also alleviate relief distribution tasks, because many survivors may wish to evacuate the affected region due to lack of basic utility services and water sanitation issues. Survivor evacuation leads to a reduction of relief demand and therefore, it is beneficial to maximize the whole network's accessibility for both evacuation and relief distribution purposes [87]. In the literature, many researchers also focus on demand satisfaction-based objectives, and this results mainly in prioritizing route clearance between supply and demand nodes, thereby neglecting other parts of the network. Eventually, as relief efforts intensify, these routes become congested and other routes are required to be cleared. Due to these reasons, here, we propose a more holistic view of the problem by considering the accessibility between all pairs of nodes over the entire planning horizon. This strategy prioritizes clearing the critical blocked links that belong to most of the shortest paths and improves the overall network accessibility, rather than accessibility between certain nodes. The stochastic model proposed in this study targets the overall network accessibility as a function of time and offers a realistic representation of the disaster recovery operation in the way that it considers multiple scenarios with varying degrees of the network. Finally, the results from the proposed model (i.e., clearance order of blocked links) are very simple and actionable. Therefore, the approach proposed in this study

can be implemented easily to prioritize repair activities in real-time during actual disasters and provide valuable contributions to the recovery efforts.

The rest of this thesis is organized as follows. First, we introduce the mathematical model for the debris clearance problem (DCP) and illustrate the solution to the model with a small-scale example. Then, we present metaheuristics proposed for DCP and we report the computational results of the metaheuristics and discuss their performance. Next, we present a sample average approximation approach to determine the number of scenarios required and share the related computational results. Finally, we provide managerial insights in the Conclusion Section.

3. MATHEMATICAL MODEL

The mathematical model consists of a two-stage stochastic multi-period network flow problem that also involves crew scheduling constraints. The goal is to minimize the cumulative expected lengths of all shortest paths between each pair of nodes in the network over the planning horizon. Debris-blocked links in the network are closed to traffic and cannot take part on any shortest path until they are cleared by a cleaning crew. The number of cleaning crews is limited, and a crew can clear one road segment at a time. Here, we assume that the specific road segments that are blocked are determined by satellite imagery and known with certainty. However, the clearance duration of a road segment is stochastic, and the uncertainty is represented by a discrete number of different repair time scenarios. The model identifies the optimal unique sequence of links to be cleared and this sequence is valid for all scenarios. On the other hand, crew assignments to blocked links differ across scenarios since task completion times of each crew depend on the clearance durations for the given scenario.

We define our model on a bi-directional network $D = (N, A)$ where N represents the node set and A represents the link (arc) set. Each bi-directional link is represented by two opposite directed links between the same pair of end nodes. In the all-pairs shortest path minimization problem, both source nodes s and sink nodes v are elements of N. Disrupted links (arcs) in the network are denoted by the set $BA \subset A$. The bi-directional partner link of a disrupted link b is denoted by b^r, i.e., links b and b^r have the same pair of end nodes and they are in opposite directions. In order to reduce the number of binary variables, only $b \in BA$, that is, disrupted links in BA are represented in a single direction, and $b^r \in A \setminus BA$. If b is cleared, then, b^r is also cleared. Furthermore, assume that BA is an (arbitrarily) ordered set and let $|b|$ represent the arbitrary order of link b in set BA. The length of the planning horizon, T is calculated as $\mathrm{T} = \max_{\omega} \sum_{\mathrm{b} \in \mathrm{BA}} \tilde{\mathrm{m}}_{\mathrm{b}}^{\omega}$ where $\tilde{\mathrm{m}}_{\mathrm{b}}^{\omega}$ is an uncertain clearance duration of a disrupted link b in a scenario ω. Below the notation required for the mathematical formulation is summarized.

Notation:

Sets and indices:

T: length of the planning horizon, $t \in T$.

A: set of links (arcs) in the network, $a \in A$.

BA: set of disrupted links, $b \in BA \subset A$.

N: the set of nodes in the network, $n \in N$.

C: the set of available crews, $c \in C$.

E_n: set of links entering node n.

L_n: the set of links leaving node n.

S: the set of source nodes, $(S = N)$, $s \in S$.

V: the set of sink nodes $(V = N)$, $v \in V$.

b^r: disrupted link in the opposite direction of disrupted link $b \in BA$ $(b^r \notin BA)$.

SV: set of (s, v) source-sink pairs, $(s, v) \in SV$, and $s \neq v$.

Ω: set of repair time scenarios, $\omega \in \Omega$.

Parameters:

$\tilde{m}_b^\omega$: Clearance duration of a closed link $b \in BA$ for a scenario $\omega \in \Omega$, $m_b^\omega = m_{b^r}^\omega$

d_a: Travel distance of link a, $a \in A$.

BIG: a very large positive number.

p_ω: Probability of occurrence of a scenario $\omega \in \Omega$.

Decision Variables:

$x_{sv}^{at\omega}$: Positive variable indicating if arc a is on the shortest path between source s and sink v in period t in scenario ω, $\omega \in \Omega$, $a \in A$, $(s, v) \in SV$.

$y_b^{t\omega}$: Binary variable indicating whether disrupted link $b \in BA$ is traversable in period t in scenario ω.

$order_{bb'}$: Binary variable indicating if disrupted link $b \in BA$ is cleared before the link $b' \in BA$ by any crew (Note that $b' \neq b^r$).

k_b: Integer variable indicating the unique position of precedence of a disrupted link $b \in BA$.

τ_b^ω: Clearance start time of a disrupted link $b \in BA$ in scenario ω, $\omega \in \Omega$.

$z_{cb}^{t\omega}$: Binary variable indicating if disrupted link $b \in BA$ has been cleared by crew c in period t, in scenario ω, $\omega \in \Omega$.

Given the above notation, Lines (3.1) -(3.15) display the deterministic equivalent of the two-stage stochastic programming model under a finite set of scenarios $\omega \in \Omega$. In this model, the first stage decisions involve the sequencing of the repair of disrupted links. $\mathrm{order}_{bb'}$ and k_b are the first stage variables that represent, respectively, the precedence relationship between the disrupted links and the position of each disrupted link in the repair sequence. Based on the first stage solution, namely the repair sequence given by the values of the variables $\mathrm{order}_{bb'}$ and k_b, the second stage determines the crew schedules that minimize the expected value of the total length of shortest paths between all node pairs over the planning horizon under each scenario $\omega \in \Omega$. In this stage, $z_{cb}^{t\omega}$ represent the assignment of crews to repair task of disrupted links and τ_b^{ω} denote the scheduled start time of repair tasks under each scenario $\omega \in \Omega$. In other words, variables $z_{cb}^{t\omega}$ and τ_b^{ω} determine the crew schedule to be implemented for scenario $\omega \in \Omega$. The remaining two recourse variables in the second stage, i.e., $x_{sv}^{at\omega}$ and $y_b^{t\omega}$ construct the network flow for each scenario $\omega \in \Omega$. These two variables are used to compute the length of the shortest paths between all node pairs based on the crew schedule denoted by variables $z_{cb}^{t\omega}$ and τ_b^{ω}.

$$\text{Minimize } F(x_{sv}^{at\omega}) = \sum_{(s,v)\in SV} \sum_{a\in A} \sum_{t\in T} \sum_{\omega} p_{\omega}\ d_a\ x_{sv}^{at\omega} \tag{3.1}$$

s.t.

$$\sum_{a\in E_s} x_{sv}^{at\omega} - \sum_{a\in L_s} x_{sv}^{at\omega} = -1 \qquad \forall (s,v) \in SV\ , \forall t \in T, \forall \omega \in \Omega \tag{3.2}$$

$$\sum_{a\in E_n} x_{sv}^{at\omega} - \sum_{a\in L_n} x_{sv}^{at\omega} = 0$$

$$\forall n \backslash \{s,v\} \in N, \forall t \in T, \forall (s,v) \in SV, \forall \omega \in \Omega \tag{3.3}$$

$$\sum_{a\in E_v} x_{sv}^{at\omega} - \sum_{a\in L_v} x_{sv}^{at\omega} = 1 \qquad \forall (s,v) \in SV\ , \forall t \in T, \forall \omega \in \Omega \tag{3.4}$$

$$y_b^{t\omega} \geq x_{sv}^{bt\omega} \qquad \forall (s,v) \in SV, \forall b \in BA, \forall t \in T, \forall \omega \in \Omega \tag{3.5}$$

$$y_b^{t\omega} \geq x_{sv}^{b'^{t\omega}} \qquad \forall (s,v) \in SV, \forall b \in BA, \forall t \in T, \forall \omega \in \Omega \tag{3.6}$$

$$\tau_b^{\omega} + \widetilde{m_b^{\omega}} \leq \tau_{b'}^{\omega} + \mathrm{BIG}\ (1 - \mathrm{order}_{bb'}) + \mathrm{BIG}\ \left(2 - \sum_t z_{cb}^{t\omega} - \sum_t z_{cb'}^{t\omega}\right)$$

$$\forall c \in C, \forall (b,b') \in \mathrm{BA}: |b| < |b'|, \forall \omega \in \Omega \tag{3.7}$$

$$\tau_{b'}^{\omega} + \widetilde{m_{b'}^{\omega}} \leq \tau_{b}^{\omega} + \text{BIG}\,(\text{order}_{bb'}) + \text{BIG}\left(2 - \sum_t z_{cb}^{t\omega} - \sum_t z_{cb'}^{t\omega}\right)$$

$$\forall c \in C, \forall (b, b') \in BA: |b| < |b'|, \forall \omega \in \Omega \qquad (3.8)$$

$$\tau_{b}^{\omega} \leq \tau_{b'}^{\omega} + \text{BIG}\,(1 - \text{order}_{bb'}) \qquad \forall (b, b') \in \text{BA}: |b| < |b'|, \forall \omega \in \Omega \qquad (3.9)$$

$$k_b + 1 \leq k_{b'} + \text{BIG}\,(1 - \text{order}_{bb'}) \qquad \forall (b, b') \in \text{BA}: |b| < |b'| \qquad (3.10)$$

$$k_{b'} + 1 \leq k_b + \text{BIG}\,(\text{order}_{bb'}) \qquad \forall (b, b') \in \text{BA}: |b| < |b'| \qquad (3.11)$$

$$\sum_c \sum_t z_{cb}^{t\omega} = 1 \qquad \forall b \in \text{BA}, \forall \omega \in \Omega \qquad (3.12)$$

$$\tau_{b}^{\omega} + \widetilde{m_{b}^{\omega}} = \sum_t \sum_c z_{cb}^{t\omega} . t \qquad \forall b \in \text{BA}, \forall \omega \in \Omega \qquad (3.13)$$

$$y_b^{t\omega} - y_b^{t-1,\omega} \geq \sum_c z_{cb}^{t\omega} \qquad \forall b \in BA, \forall t \in T, \forall \omega \in \Omega \qquad (3.14)$$

$$k_b \ \ integer; \quad y_b^{t\omega}, \text{order}_{bb'}, z_{cb}^{t\omega} \ binary; \quad \tau_b^{\omega}, x_{sv}^{at\omega} \geq 0 \qquad (3.15)$$

The objective function $F(x_{sv}^{at\omega})$ in Equation (3.1) minimizes the cumulative sum of the expected lengths of all-pairs shortest paths throughout the planning horizon over all scenarios. To reach this goal, the model identifies the best order in which closed roads are cleared by each crew across all scenarios. The objective function enables maximum network accessibility throughout the network during the restoration period. Equations (3.2), (3.3), (3.4) represent the network flow balance for the shortest path problem for every source-sink pair (s, v) in each period over every scenario. Equations (3.2) ensure that only one link exits a source node on the shortest path between (s, v) node pair. Equations (3.3) balance the shortest path link flow at each node on the path except for the associated source and sink nodes. Equations (3.4) enforce that exactly one link enters a sink on the shortest path between (s, v) node pair. Equations (3.5) ensure that a disrupted link can only be a member of a shortest path after it has been cleared in a scenario. Equations (3.6) make sure that a link is either cleared in both directions or closed in both directions in a scenario, i.e., a disrupted link's directional partner b^r has the same status as link *b*. This constraint is necessary because we do not include directional partners in set *BA*. Equations (3.7) - (3.8) decide on the order of clearance between two disrupted links in *BA* (b and b'), whenever they are both cleared

by the same crew. If they are not addressed by the same crew, then, these equations do not impose an order of clearance between the two. If the same crew works on links b' and b, then these equations create a precedence relationship between their start and completion times in each scenario. Note that Equations (3.7), (3.8) address only $b \in BA$ and not b^r. The order variable that decides on clearance schedules of crews is not linked to any scenario. Since Equations (3.7), (3.8) are loose equations, Equations (3.9) are necessary to prevent cycling. Equations (3.9) impose that if a clearance precedence exists between b' and b, then starting time of b' should be less than that of b in any scenario, regardless of the crew it is assigned. Equations (3.10), (3.11) impose one unique precedence relationship between broken arcs in the clearance sequence. Equations (3.10) state that if b is cleared before b', then in the overall precedence of broken links, the position variable k_b should be prior to $k_{b'}$. Otherwise, Equations (3.11) hold. The unique position k_b assigned to each broken link b indicates the clearance priority of this broken link based on all disaster scenarios. It is designated to produce schedules considering the inherent uncertainty in multiple stochastic scenarios. Note that in the event of an actual disaster, position variables k_b can serve as a clearance priority according to which the clearance crews are assigned to broken links in real-time as they complete their current tasks and free up for reassignment. Equations (3.12) ensure that a link is cleared only once by one crew during the planning horizon in each scenario. Equations (3.13) calculate the start time of a link's clearance operation in each scenario. Equations (3.14) maintain that a link becomes traversable as soon as it has been cleared by any crew. Note that according to Equations (3.14), once a link becomes traversable, it stays traversable until the end of the planning horizon. Equations (3.15) represent variable domains. Note that the variable $x_{sv}^{at\omega}$ does not have to be defined as binary due to the special structure of the coefficient matrix.

In the model, we have $|BA|(T\,|\Omega| + T|C||\Omega| + |BA|)$ binary variables where $|BA|$ is the number of links to be cleared, $|\Omega|$ is the number of scenarios and $|C|$ is the number of crews, and T is the number of discrete time periods.

3.1. AN ILLUSTRATIVE EXAMPLE

To illustrate the model's outputs, a small instance with ten nodes and 34 links is solved (see Figure 3.1). All links are bidirectional and links #3, #4, #5, #6, #13, #14 #15, #16, #19, #20, #23 #24, #29 and #30 are disrupted. These links are shown in dashed lines in Figure 3.1.

In Table 3.1, the shortest paths between all pairs of source and sink nodes are presented. The lengths of the shortest paths are calculated both with and without the impact of disruption. Table 3.1 illustrates the impact of blockages in the network upon the lengths of all-pair shortest paths.

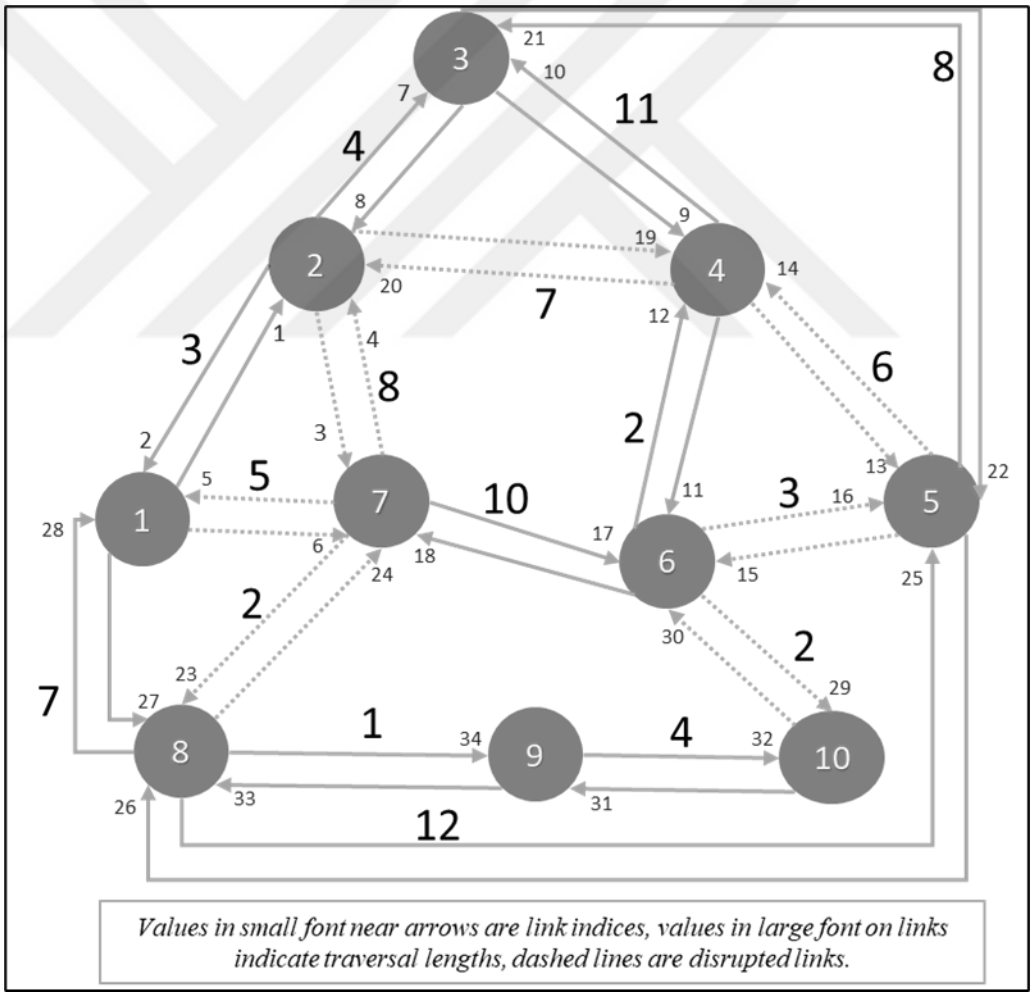

Figure 3.1 Illustrative Example

Table 3.1 Shortest paths between all pairs (s, v) without and with disrupted links in the example network using deterministic repair durations

Shortest paths in undisrupted n.				Shortest paths in disrupted n.				Shortest paths in undisrupted n.				Shortest paths in disrupted n.			
s	**v**	**Links #**	**Length**	**s**	**v**	**Links #**	**Length**	**s**	**v**	**Links #**	**Length**	**s**	**v**	**Links #**	**L**
1	2	1	3	1	2	1	3	**3**	**10**	**15-22-29**	**13**	**3**	**10**	**2-8-27-32-34**	**19**
1	3	1-7	7	1	3	1-7	7	**4**	**5**	**11-16**	**5**	**4**	**5**	**10-22**	**19**
1	**4**	**1-19**	**10**	**1**	**4**	**1-7-9**	**18**	4	6	11	2	4	6	11	2
1	**5**	**1-11-16-19**	**15**	**1**	**5**	**1-7-21**	**15**	**4**	**7**	**11-23-29-31-**	**11**	**4**	**7**	**11-18**	**12**
1	**6**	**1-11-19**	**12**	**1**	**6**	**1-7-9-11**	**20**	**4**	**8**	**11-29-31-33**	**9**	**4**	**8**	**2-8-10-27**	**25**
1	**7**	**5**	**5**	**1**	**7**	**1-7-9-11-18**	**30**	**4**	**9**	**11-29-31**	**8**	**4**	**9**	**2-8-10-27-34**	**26**
1	8	27	7	1	8	27	7	**4**	**10**	**11-29**	**4**	**4**	**10**	**2-8-10-27-32-34**	**30**
1	9	5-24-34	8	1	9	27-34	8	**5**	**6**	**15**	**3**	**5**	**6**	**9-11-22**	**21**
1	**10**	**5-24-32-34**	**12**	1	10	**27-32-34**	**12**	**5**	**7**	**15-23-29-31-**	**12**	**5**	**7**	**9-11-18-21**	**31**
2	3	7	4	2	3	7	4	**5**	**8**	**15-29-31-33**	**10**	**5**	**8**	**26**	**12**
2	**4**	**19**	**7**	**2**	**4**	**7-9**	**15**	**5**	**9**	**15-29-31**	**9**	**5**	**9**	**26-34**	**13**
2	5	11-16-19	12	2	5	7-22	12	**5**	**10**	**15-29**	**5**	**5**	**10**	**26-32-34**	**17**
2	**6**	**11-19**	**9**	**2**	**6**	**7-9-11**	**17**	**6**	**7**	**23-29-31-33**	**9**	**6**	**7**	**18**	**10**
2	**7**	**2-5**	**8**	**2**	**7**	**7-9-11-18**	**27**	**6**	**8**	**29-31-33**	**7**	**6**	**8**	**2-8-10-12-27**	**27**
2	8	2-27	10	2	8	2-27	10	**6**	**9**	**29-31**	**6**	**6**	**9**	**2-8-10-12-27-34**	**28**
2	**9**	**3-24-34**	**11**	**2**	**9**	**2-27-34**	**11**	**6**	**10**	**29**	**2**	**6**	**10**	**2-8-10-12-27-32-34**	**32**
2	**10**	**11-19-29**	**11**	**2**	**10**	**2-27-32-34**	**15**	**7**	**8**	**24**	**2**	**7**	**8**	**2-8-10-12-17-27**	**37**
3	**4**	**8-19**	**11**	**3**	**4**	**9**	**11**	**7**	**9**	**24-34**	**3**	**7**	**9**	**2-8-10-12-17-27-34**	**38**
3	5	21	8	3	5	21	8	**7**	**10**	**24-32-34**	**7**	**7**	**10**	**2-8-10-12-17-27-32-**	**42**
3	**6**	**15-22**	**11**	**3**	**6**	**9-11**	**13**	8	9	34	1	8	9	34	1
3	**7**	**2-5-8**	**12**	**3**	**7**	**9-11-18**	**23**	8	10	32-34	5	8	10	32-34	5
3	8	2-8-27	14	3	8	2-8-27	14	9	10	32	4	9	10	32	4
3	**9**	**2-5-8-24-34**	**15**	**3**	**9**	**2-8-27-34**	**15**								

We assume that there are two clearance crews to handle closed links. All disrupted links are assigned random repair times within a uniform distribution in the interval [1,5]. We create a three-scenario case for each link's repair time where each scenario has an equal realization probability. The repair time information for the three scenarios is given in Table 3.2.

Table 3.2 Repair times of broken links under scenarios ω_1-ω_3 for the illustrative example

	#3	**#5**	**#13**	**#15**	**#20**	**#23**	**#29**
ω_1	4	4	1	3	3	2	4
ω_2	5	2	2	2	2	4	3
ω_3	5	5	1	5	2	3	5

We implement the rolling horizon approach depicted in Figure 1.1 to solve this instance. The DCP models created at each iteration are solved using CPLEX version 12 within a CPU time limit of 7200 secs on a laptop with an i7 9th generation Intel CPU with 16 GB RAM. Initially, the DCP model is solved using the repair time data for the three scenarios in Table 1 and the resulting clearance sequence is #13<#23<#29<#20<#15<#5<#3. The solution has a relative gap of 0.01 percent. We also tested a five-scenario instance; however, the solution has a 10 percent relative gap from the best possible solution. We also experimented with solving larger instances with 90 directed links and 32 nodes using CPLEX, but CPLEX was not able to proceed beyond the preprocessing phase and failed to produce a feasible solution. Therefore, we were not able to produce optimal or near optimal solutions for realistic size problems. Gannt charts of the crew schedules resulting from to the rolling horizon approach are depicted in Figure 3.2, where the first section titled "**t=0**" displays the clearance schedules created by the initial sequence #13<#23<#29<#20<#15<#5<#3 under the three repair time scenarios in Table 3.2. For each scenario ω, rows titled c1 and c2 display the schedules for crews 1 and 2, respectively. In each Gannt chart, blocks shaded dark and light gray represent the planned and executed part of the schedule as of the time period indicated at the chart heading.

As observed in Figure 3.2, in the initial solution, we observe that the optimal sequence of tasks is the same for all three scenarios as imposed by constraints (9) and (10) in the model, however, crew assignments and the start and end times of tasks differ due to the lengths of

repair times in each scenario. Based on this sequence, jobs #13 and #23 are assigned to the two available crews at the beginning of the recovery effort, i.e., t=0. Once the two crews start their work, they report new repair time estimates of 3 and 8 for jobs #13 and #23, respectively. Based on the updated estimates, the first job completed is job #13 at t=3. Therefore, we solve the DCP again at t=3 to revise the clearance sequence based on the updated repair times. In the updated DCP, the repair time for job #23 is taken as 5, i.e., the remaining time to complete the job at t=3, and job #23 is fixed as the first job on the sequence. (In this example, we assume that the repair time estimates become available as soon as the crew starts the job and rerun DCP at that time. In practice, DCP would be run again when the crews report updated repair times.) The clearance schedule for the DCP with updated repair times is computed as #20<#29<#15<#5<#3. Note that the order of jobs #20 and #29 are reversed compared to the original sequence and the freed-up crew is assigned to the first job on the updated sequence, i.e., job #20. Next, the repair time for job #20 is communicated by the clearance crew as 2. Therefore, DCP is solved again at t=5 when job #20 is completed to yield the sequence #29<#5<#15<#3, which indicates that the freed-up crew should be assigned to job #29. Once the repair work is underway for job #29, its repair time estimate is updated to be 11. The next job to complete is #23 at t=8, when the DCP yields the sequence #15<#5<#3 and the available crew is now assigned to the first job on the sequence, i.e., job #15. Again, note that the order of jobs #15 and #5 is reversed compared to the previous sequence. Finally, at t=12, DCP is solved once more upon the completion of job #15 and produces the sequence #5<#3, indicating the freed-up crew should now be assigned to job #5, which leaves job #3 to be the last job to be completed. Based on the five DCP solutions throughout the recovery effort, the clearance sequence implemented by the rolling horizon approach is #13<#23<#20<#29<#15<#5<#3 and the resulting objective function value is 22,642.

Table 3.3 compares the performance of the rolling horizon approach with implementing the original sequence created based on the initial repair time estimates obtained immediately after the disaster. When the sequence is not revised based on the updated repair time estimates from the field, the objective function value increases to 22,804. On the other hand, the sequence created using the deterministic DCP using the updated information i.e., the solution with perfect information yields an objective function value of 22,320. These results demonstrate the merit of using the rolling horizon approach to revise the clearance sequence

t=0 **Clearance Sequence: #13<#23<#29<#20<#15<#5<#3**

t		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
ω_1	c1		23			20			15													
	c2	13			29				5			3										
ω_2	c1		13			29		15			3											
	c2				23		20		5													
ω_3	c1	13				29				5												
	c2			23			20				15			3								

t=3 **Clearance Sequence: #20<#29<#15<#5<#3**

t		*1*	*2*	*3*	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
ω_1	c1			*13*			20											29				
	c2			*23*					23			15			5				3			
ω_2	c1			*13*			20										29					
	c2			*23*					23		15		5			3						
ω_3	c1			*23*					23					15				5				
	c2			*13*			20										29					3

t=5 **Clearance Sequence: #29<#5<#15<#3**

t		*1*	*2*	*3*	*4*	*5*	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
ω_1	c1			*13*		*20*											29					
	c2					*23*			23				5				15				3	
ω_2	c1			*13*		*20*											29					
	c2					*23*			23		5				15						3	
ω_3	c1					*23*			23				5					15				
	c2			*13*		*20*											29					3

t=8 **Clearance Sequence: #15<#5<#3**

t		*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	9	10	11	12	13	14	15	16	17	18	19	20	21
ω_1	c1			*13*		*20*			*29*								29				3	
	c2								*23*				15								5	
ω_2	c1			*13*		*20*			*29*								29					3
	c2								*23*				15								5	
ω_3	c1								*23*				15								5	
	c2			*13*		*20*			*29*								**29**					3

t=12		**Clearance Sequence: #5<#3**																			
t		*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	13	14	15	16	17	18	19	20	21
ω_1	c1			*13*		*20*							*29*				29				3	
	c2								*23*				*15*								5	
ω_2	c1			*13*		*20*							*29*				29				3	
	c2								*23*				*15*								5	
ω_3	c1								*23*				*15*								5	
	c2			*13*		*20*							*29*				29				3	

Figure 3.1 Solution of the Illustrative Example by the Rolling Horizon Approach

as data becomes available throughout the recovery process.

Table 3.3 Summary of results for the illustrative example

	Objective Function Value	Final Sequence
Solution with Initial Estimates	22,804	#13<#23<#29<#20<#15<#5<#3
Rolling Horizon Solution	22,642	#13<#23<#20<#29<#15<#5<#3
Solution with Perfect Information	22,320	#29<#20<#15<#23<#3<#5<#13

We observe that CPLEX takes a long time to obtain an optimal solution even though the size of this example is quite small. This poor performance is likely to be due to the additional complexity resulting from the objective function that aims to maximize network accessibility throughout the planning horizon. In order to find high-quality solutions for this complicated planning problem within an acceptable time, heuristics seem to be a viable option. In the next section, we propose two metaheuristics for the debris clearance problem and identify the best neighborhood structures that yield the best performance among a wide variety of candidate neighborhoods.

4. PROPOSED METAHEURISTIC APPROACHES

In order to deal with larger-scale networks, we propose two metaheuristics: Variable Neighborhood Search (VNS) and a hybrid algorithm that combines Tabu Search (TS) with Simulated Annealing (SA) which we denote as HTSSA. We encode the solution to this problem in terms of the unique sequence of tasks to be completed by crews. Both VNS and HTSSA perturb this sequence according to a specified neighborhood definition or several neighborhoods applied in a cyclic manner. We propose several novel permutation-based swap mechanisms in this study. They are utilized in both HTSSA and VNS algorithms. It is also important to start these algorithms from a good initial solution. To this end, we design a new random constructive algorithm that is based on Biased Random Sampling (BRS). This approach enables faster convergence to good-quality solutions.

First introduced by Mladenović [85], VNS is an algorithm that systematically alters neighborhood operators throughout the search. VNS takes an initial solution, creates a neighboring one, and accepts it if it improves the objective function. Whenever an inferior solution is encountered, the neighborhood generation operator is changed to the next in line on a list of predefined neighborhood operators. Hansen et al. [86] and Bianchi et al. [58] review various VNS applications and algorithms in both stochastic and deterministic problems. In the debris clearance literature, VNS is used by [20] and [23].

Salman and Yücel [72] use TS to locate emergency facilities in the event of random network damage after a disaster. TS is a good choice for optimizing discrete problems because it avoids the problem of becoming trapped in a local optimum by keeping a tabu list of last visited solutions. TS can also deal with problems concerning uncertainty [69]. SA has also been applied in various stochastic problems, see the reviews [58, 59]. SA starts from an initial solution, creates a neighboring solution, and accepts the new solution if it improves the objective function; it may also accept an inferior solution probabilistically, but the probability of acceptance diminishes according to the inferiority of the new solution and also according to the cooling temperature. In this study, we implement a hybrid version of TS and SA (HTSSA) that combines the main features of both algorithms to address the stochastic scheduling problem on hand. Furthermore, we embed different neighborhood schemes into HTSSA and also implement them cyclically as in VNS.

Below we first describe the two approaches we use to create an initial solution for these metaheuristics and then discuss the implementation details of each metaheuristic.

4.1. INITIAL SOLUTION GENERATION

Starting from a promising solution has its advantages in reducing the solution time to reach a local optimum or hopefully a global one. In the initial solution generation algorithms and the metaheuristics, we encode a solution to our problem as a permutation of the disrupted links indicating the order in which they are cleared. One possible way of creating an initial solution is to form this permutation randomly, giving each broken link the same probability of appearing in each position in the permutation. We call this method Uniform Random Sampling (URS). Alternatively, we can create a permutation in which broken links that contribute more to network accessibility are more likely to appear earlier in the permutation. This approach is called Biased Random Sampling (BRS) in the literature [88]. To understand whether there is any benefit in biasing the probability space in this manner, we conducted a computational experiment to compare the quality of the initial solution obtained from the URS and BRS approaches.

Table 4.1 Comparison between BRS and URS approaches with P-30-3 Instance

	BRS		URS	
	Obj F	Execution T.(min)	Obj F	Execution T.(min)
1	44,619,016.65	6.51	45,940,854.81	6.71
2	43,626,193.79	6.52	45,226,735.44	6.55
3	43,626,193.79	6.59	45,951,367.29	6.58
4	44,482,141.89	6.62	45,775,704.98	6.60
5	44,702,278.96	6.53	43,958,590.93	6.74
6	44,868,039.28	6.54	46,267,587.16	6.60
7	44,320,884.08	6.53	46,260,572.68	6.54
8	43,885,252.38	6.58	46,846,450.74	6.58
9	43,865,996.22	6.56	46,188,284.86	6.74
10	43,419,081.83	6.55	45,390,679.44	6.59
AVG	44,141,507.89	6.56	45,780,682.83	6.62

The instances used in the study are introduced in Section 5.1 in detail. Among these instances we conducted the experimentation on the initial solution generation approaches and neighborhood structures using the P-30-3 instance due to its high network complexity and large problem size.

Table 4.1 demonstrates the results of the experiments, we conducted on the P-30-3 instance to compare the performance of the BRS and URS approaches. The rows demonstrate the results of individual runs for BRS and URS, namely the best objective value obtained from the run, and corresponding execution time in minutes. Each run consists of 500 execution of the BRS / URS method, creating 500 initial solutions. The column "Obj. F." displays the best (lowest) objective function value among these 500 solutions, whereas the column titled "Execution T." provides the total execution time for creating the 500 solutions. (We should note that we also tried running BRS and URS for 12 hours and we observed that the best result obtained was only 0.4 percent better after 12 hours.) The computational results summarized in Table 4.1 indicate that BRS outperforms URS in terms of solution quality. We also observe that the objective function value is higher on average for URS and the computational times for both methods are comparable. The results in Table 4.1 indicate that BRS produces a considerably better initial solution than URS without requiring any significant computational burden.

In our BRS approach, each broken link's effect on the overall network accessibility is considered. In BRS, the priority of each link $b \in BA$ is calculated as follows. We first assume that only link b is blocked and all remaining links are open. Then, we calculate obj_b which is the sum of the lengths of all shortest paths between all node pairs in the network under the latter condition. Calculated values for the example can be seen in Table 4.2.

Then, we rank all broken links b in descending order of their obj_b, such that the link with maximum obj_b has $Rank_b = 1$ and the link with minimum obj_b has $Rank_b = |BA|$. In Eq. (4.1) we calculate each link's priority (P_b).

$$P_b = \frac{1}{Rank_b + 1} \qquad \forall b \in BA \qquad (4.1)$$

Then, we normalize each P_b according to Eq. (4.2) to obtain biased probabilities NP_b and sample the initial solution permutation using these probabilities. All relevant calculations

are demonstrated in Table 4.3.

$$NP_b = \frac{P_b}{\sum_{b \in BA} P_b} \qquad \forall b \in BA \qquad (4.2)$$

Table 4.2 Objective Function Value Calculations for All Disrupted Links

$b \in BA$	Obj_b	$b \in BA$	Obj_b
#1	442574	#17	456122
#2	450896	#18	439106
#3	442730	#19	441256
#4	441658	#20	442478
#5	446534	#21	459912
#6	438562	#22	469476
#7	459180	#23	487892
#8	461722	#24	465690
#9	452946	#25	441484
#10	440628	#26	441576
#11	443410	#27	442740
#12	449020	#28	456028
#13	445182	#29	438992
#14	444316	#30	444016
#15	450740	#31	448356
#16	450104		

4.2. CALCULATING THE FITNESS OF A SOLUTION

The fitness value of a permutation of tasks is the value of $F(x_{sv}^{at\omega})$ in Eq. (3.1). If we arrange $order_{bb'}$ and k_b variables according to the current permutation and solve the mathematical model, then, we can find the value of $F(x_{sv}^{at\omega})$. Unfortunately, this method proves to be computationally expensive. Therefore, instead of solving the mathematical model, we use a more direct approach. Under each scenario, we simulate the execution of the corresponding permutation schedule dictated by k_b variables by assigning the next repair task in the permutation to the first idle crew. After all repair tasks are assigned, in each time period, we re-calculate the sum of all shortest paths by running Dijkstra's [89] shortest path algorithm that considers the links that are currently broken in that time period. We sum up Dijkstra's results overall time periods until the completion of the planning period. This process is

Table 4.3 Normalized Probabilities of Disrupted Links and Probability Intervals

$b \in BA$	Obj_b	$Rank_b$	P_b	NP_b	Biased Probability Interval
#23	487892	1	0.500	0.164	0.000 - 0.164
#22	469476	2	0.333	0.109	0.164 - 0.272
#24	465690	3	0.250	0.082	0.272 - 0.354
#8	461722	4	0.200	0.065	0.354 - 0.420
#21	459912	5	0.167	0.055	0.420 - 0.474
#7	459180	6	0.143	0.047	0.474 - 0.521
#17	456122	7	0.125	0.041	0.521 - 0.562
#28	456028	8	0.111	0.036	0.562 - 0.598
#9	452946	9	0.100	0.033	0.598 - 0.631
#2	450896	10	0.091	0.030	0.631 - 0.661
#15	450740	11	0.083	0.027	0.661 - 0.688
#16	450104	12	0.077	0.025	0.688 - 0.713
#12	449020	13	0.071	0.023	0.713 - 0.573
#31	448356	14	0.067	0.022	0.573 - 0.758
#5	446534	15	0.062	0.020	0.758 - 0.779
#13	445182	16	0.059	0.019	0.779 - 0.798
#14	444316	17	0.056	0.018	0.798 - 0.816
#30	444016	18	0.053	0.017	0.816 - 0.833
#11	443410	19	0.050	0.016	0.833 - 0.850
#27	442740	20	0.048	0.016	0.850 - 0.866
#3	442730	21	0.045	0.015	0.866 - 0.880
#1	442574	22	0.043	0.014	0.880 - 0.894
#20	442478	23	0.042	0.014	0.894 - 0.908
#4	441658	24	0.040	0.013	0.908 - 0.921
#26	441576	25	0.038	0.012	0.921 - 0.934
#25	441484	26	0.037	0.012	0.934 - 0.946
#19	441256	27	0.036	0.012	0.946 - 0.957
#10	440628	28	0.034	0.011	0.957 - 0.969
#18	439106	29	0.033	0.011	0.969 - 0.979
#29	438992	30	0.032	0.010	0.979 - 0.990
#6	438562	31	0.031	0.010	0.990 - 1.000

carried out for every scenario and each scenario's objective function is multiplied with its corresponding probability and added up to compute the value of $F(x_{sv}^{at\omega})$.

4.3. GENERAL DESCRIPTION OF PROPOSED ALGORITHMS

In this dissertation we have three proposed algorithms. These algorithms are Variable Neighborhood Search (VNS), Hybrid Tabu Search-Simulated Annealing (HTSSA) and Hybrid Tabu Search-Simulated Annealing with Variable Neighborhood Search (HTSSA-VNS). First, we would like to explain general logic of the proposed algorithms with pseudocodes. Algorithm 4.1 provides the pseudocode of VNS. As depicted in Algorithm 4.1 VNS starts from an initial solution seeks to improve this solution using the first neighborhood. The search of the first neighborhood stops when the current solution can no longer be improved. Then the algorithm continues with the same hill-climbing strategy for the next neighborhood in the neighborhood list. In VNS, only solutions that improve the current solution are accepted, otherwise, the neighborhood operator is changed. Once the algorithm reaches the end of the neighborhood list, it restarts from the first neighborhood. The VNS algorithm terminates after computational time limit t_{max} is reached. Neighborhood operators and neighborhood lists used in the VNS algorithm are explained in detail in Section 4.4.

Algorithm 4.1. Variable Neighborhood Search (VNS)

Initialize
k
k'
k_{max}
t
t_{max}
Generate/Get initial solution s(0)
Set f(x) = s(0)
While $t < t_{max}$
Generate new solution Neighborhood (k, f(x))
    If objective value of f(x) < objective value of best solution
        Set best solution = f(x)
    Else:
        Function NeighborhoodChange (k, k')
            if $k' = k_{max}$
                k'=0
            $k \leftarrow k'$ +1 // Next neighborhood

Algorithm 4.2. provides the pseudocode of the HTSSA. In this algorithm a single neighborhood structure is used within a hybrid tabu search and simulated annealing search structure. In HTSSA, in any iteration i, a neighbor solution is accepted, and the subset of broken links placed in a tabu list, if its fitness value is smaller or equal to that of the incumbent solution, h. If the neighbor has a worse fitness value, then, its probability of acceptance $PA(\Delta, t_i)$ is calculated. The calculations of $PA(\Delta, t_i)$ and further details are given in Section 4.5. the broken links in an accepted move are declared tabu such that the same set of broken links are not used in next L iterations. Note that in the HTSSA algorithm, accepting solutions with inferior objective function values is permitted with a certain probability as opposed to the hill-climbing strategy employed by the VNS algorithm.

Algorithm 4.2. Hybrid Tabu Search and Simulated Annealing (HTSSA)

Initialize:
$beta$
t
t_{max}
Generate/Get initial solution s(0)
Set best_solution = s(0)
Set accepted_solution = s(0)
While $t < t_{max}$
    Generate new solution f(x)
    If objective value of f(x) ≤ objective value of best_solution
        Set best_solution = f(x)
        Set accepted_solution = f(x)
        Set tabu links
    Else if objective value of f(x) < objective value of accepted_solution
        Set accepted_solution = f(x)
    Else
        If t < 0.01:
            1. t = 1
        Else:
            2. t = t / (1 + $beta$ * t)
        Set $delta$ = objective value of f(x) - objective value of accepted_solution
        Set acceptance_probability = $e^{(-\text{delta / objective value of f(x)} * \text{t})}$
        Generate a random number $alpha$.
        If $alpha$ < acceptance_probability:
            Set accepted_solution = f(x)
        Else:
            Reject f(x)

We also implement a hybrid metaheuristic that uses the strategies of both HTSSA and VNS. In this algorithm the HTSSA metaheuristic run is performed for each neighborhood structure in the order that appears in the neighborhood list. In these algorithm, tabu list holds visited solutions only for current the neighborhood. This algorithm uses two types of diversification. First it iterates through various neighborhood structures and second it allows inferior solutions encountered within HTSSA. Algorithm 4.3. illustrates the pseudocode of HTSSA-VNS.

Algorithm 4.3. Hybrid Tabu Search and Simulated Annealing with Variable Neighborhood Search (HTSSA-VNS)

Initialize:
k
k'
k_{max}
$beta$
t
t_{max}
Generate/Get initial solution s(0)
Set best solution = s(0)
Set accepted solution = s(0)
While $t < t_{max}$
Generate new solution Neighborhood (k, f(x))
    If objective value of f(x) ≤ objective value of best solution
        Set best solution = f(x)
        Set accepted solution = f(x)
        Set tabu links
    Else if objective value of f(x) < objective value of accepted solution
        Set accepted solution = f(x)
    Else
        If t < 0.01:
            1. t = 1
        Else:
            2. $t = t$ / (1 + $beta$ * t)
        Set delta = objective value of f(x) - objective value of accepted solution
        Set acceptance_probability = $e^{(-\text{delta} / \text{objective value of f(x)} * t)}$
        Generate a random number $alpha$.
        If $alpha$ < acceptance probability:
            Set accepted solution = f(x)
        Else:
            Reject f(x)

Function NeighborhoodChange (k, k')
if $k' = k_{max}$
$k' = 0$
$k \leftarrow k' + 1$ // Next neighborhood

4.4. VARIABLE NEIGHBORHOOD SEARCH ALGORITHM (VNS)

In VNS, only solutions that improve the current solution are accepted, otherwise, the neighborhood operator is changed. The key to success in VNS is the neighborhood operators used in the algorithm. We describe below the neighborhood structures we have implemented in this study. We use nine different neighborhoods in our VNS algorithm. Each neighborhood is denoted by $N_m, m = 1,2,3,4,5,6,10,15,20$. As discussed below neighborhoods N_m where $m = 1,2,10,15,20$ correspond to the *Insert-m (N_m)* neighborhood in which m broken links are inserted to a new position at a time. The remaining neighborhoods N_3, N_4, N_5, N_6 represent completely different neighborhood structures such that the subscript value does not indicate the number of broken links involved in the operator. Note that all broken links and all insert positions have an equal probability of being selected in the neighborhoods described below.

Insert-m (N_m): Randomly pick m broken links (m =1,2,10,15,20) and assign each selected link to a new position.

Two-link-string insert reversed (N_3): In this neighborhood, we remove a randomly selected string of two broken links, reverse the string and assign it to a new position in reverse order. To determine the starting position of the string to be removed, we generate a random number between [1, ($|BA|$-1)] and remove the string starting at this position.

Three-link-string insert reversed (N_4): In this neighborhood, we remove a string of three broken links. We then choose randomly among three operations: (1) reverse the string; (2) exchange the position of the first two broken links with the third one; (3) exchange the position of the last two broken links with the first one. We then implement the selected operator. To determine the starting position of the string to be used, we generate a random number between [1, ($|BA|$-2)].

Swap two to one (N_5): In this neighborhood we remove a string of two broken links and swap the position of the pair with the position of a third broken link. To determine the starting position of the string to be removed, we generate a random number between [1, ($|BA|$-1)] and remove the string starting at this position. The third broken link is selected randomly among the remaining broken links.

Swap two links (N_6): In this neighborhood we randomly pick two broken links and swap their positions.

Below we illustrate each neighborhood using the permutation of broken links **#1, #2, #3, #4, #5, #6, #7, #8, #9, #10** as the current solution.

Neighborhood N_1:

- Create a random number between (0, 100): 50→ 50 corresponds to broken link #5
- Renormalize, pick a number from (1,2,3,4,6,7,8,9,10) (excluding the current position)
- Create a random number: 10→ 10 corresponds to position 1
- Insert link #5 at position 1

The new permutation is #5, #1, #2, #3, #4, #6, #7, #8, #9, #10

Neighborhood N_2:

- Create a random number between (0, 100): 50→ 50 corresponds to broken link #5
- Renormalize, pick a number from (1,2,3,4,6,7,8,9,10) (excluding the current position)
- Create a random number between (0, 100): 10→ 10 corresponds to position 1
- Create a random number between (0, 100): 67→ 67 corresponds to broken link #6
- Renormalize, pick a number from (2,3,4,5,7,8,9,10) (excluding position 1 and the current position)

- Create a random number between (0, 100): 38→ 38 corresponds to position 3

The new permutation is #5, #1, #6, #2, #3, #4, #7, #8, #9, #10

Neighborhood N_3:

- Create a random number between (1,9) for the starting point of the string → 8 corresponds to position 8
- Compose the two broken link length string starting from 8: #8, #9 and reverse the string

The new permutation is #1, #2, #3, #4, #5, #6, #7, #9, #8, #10

Neighborhood N_4:

- Here we have 3 combinations of a three-broken-link-length-string i.e., (#4, #5, #6)
- ORDERINGS:
  i. Reverse the string - #6, #5, #4
  ii. Exchange the position of the first two broken links with the third one - #6, #4, #5
  iii. Exchange the position of the last two broken links with the first one - #5, #6, #4
- Create a random number between (1,8) for the starting point of the string → 8 corresponds to position 8
- Compose the three-broken-link-length-string starting from 8: #8, #9, #10
- Create a random number (1,2,3) → 1 corresponds to ordering 1

The new permutation is #1, #2, #3, #4, #5, #6, #7, #10, #9, #8

Neighborhood N_5:

- Create a random number between (1,9) for the starting point of the string → 8 corresponds to position 8
- Compose the two broken arc length string starting from 8: #8, #9
- Renormalize, remaining broken links**:** #1, #2, #3, #4, #5, #6 #7, #10

- Create a random number between (0,100): 50→ 50 corresponds to link #4
- Switch the positions

The new permutation is #1, #2, #3, #8, #9, #5, #6, #7, #4, #10

Neighborhood N_6:

- Create a random number: 50→ 50 corresponds to broken link #5
- Renormalize, remaining broken links: #1, #2, #3, #4, #6 #7, #8, #9, #10
- Create a random number between (0,100): 11→ 11 corresponds to link #1
- Switch the positions

The new permutation is #5, #2, #3, #4, #1, #6, #7, #8, #9, #10

As discussed previously, in VNS, neighborhood generation schemes are applied sequentially and cyclically. The cyclic schemes that we have experimented are listed below:

K_1: N_1, N_2, N_6

K_2: N_1, N_6, N_2

K_3: N_1, N_2, N_6, N_3

K_4: N_1, N_6, N_2, N_3

K_5: N_6, N_1, N_2, N_3, N_4, N_5

K_6: N_1, N_2, N_6, N_3, N_4, N_5

K_7: N_1, N_2, N_6, N_3, N_4, N_5, N_{10}

K_8: N_1, N_2, N_6, N_3, N_4, N_5, N_{20}

K_9: N_1, N_2, N_6, N_3, N_4, N_5, N_{15}

Note that we include different permutations of the same set of neighborhoods (such as K_1 – K_2 pair, K_3 – K_4, and K_5 – K_6 pairs) to see if the order of neighborhoods significantly affects the quality of the solution. K_5 – K_6 have a larger portfolio of neighborhood generation strategies. We also included K_7, K_8, and K_9 which have the same first 6 neighborhoods as in K_6, but they include respectively N_{10}, N_{20}, N_{15}, as their 7th neighborhood. Since these last

neighborhoods relocate 10, 20, and 15 broken links, they cause a significant change in the solution and are intended as a shaking mechanism.

4.5. HYBRID TABU SEARCH-SIMULATED ANNEALING ALGORITHM (HTSSA)

In HTSSA, in any iteration *i*, a neighbor solution is accepted, and the subset of broken links placed in a tabu list, if its fitness value is smaller or equal to that of the incumbent solution, *h*. If the neighbor has a worse fitness value, then, its probability of acceptance $PA(\Delta, t_i)$ is calculated according to Eq. (4.3)

$$PA(\Delta, t_i) = e^{\frac{\Delta}{F_h * t_i}} \tag{4.3}$$

where Δ is the difference between the fitness values of the two neighboring solutions, F_h is the incumbent solution's fitness value and t_i is the temperature value in iteration *i*. We reduce the temperature t_i as in Eq. (4.4) each time a neighboring solution worsens. β is a parameter in the range [0,1]. When t_i drops below 0.01, it is reset to 1, such that re-annealing begins.

$$t_{i+1} = \frac{t_i}{1+\beta * t_i} \tag{4.4}$$

For each accepted solution, the subset of broken links whose positions are modified to create that solution is placed in a tabu list. To avoid cycling back to previously visited solutions, a tabu list of length *L* is used to hold a number of visited solutions. Once, the tabu list is full, a new subset of broken links replaces the oldest member of the list. Hence, revisiting is prohibited during the next *L* iterations. We also implement HTSSA with neighborhood lists and, in these versions, tabu list holds visited solutions only when the neighborhood remains in the incumbent operator. Unlike VNS, by accepting worse solutions, HTSSA implements a natural diversification method. The tabu list also allows diversification by avoiding visits to the same attraction basin. While in typical TS, the best solution in a neighborhood is always accepted even if it is inferior to the incumbent solution, in HTSSA we accept inferior solutions according to their inferiority level. This is the main difference of our hybrid algorithm from pure TS or pure SA.

4.6. IDENTIFYING THE NUMBER OF SCENARIOS REQUIRED USING SAMPLE AVERAGE APPROXIMATION

Approximating the expected value of an objective function is a computational handicap in stochastic models because the number of scenarios that provide a good approximation needs to be large. Hoyos et al. [90] discuss different scenario reduction techniques when dealing with large feasible spaces. In debris clearance literature, both Aslan and Çelik [28] and Sancı and Daskin [29] use SAA to approximate the expected objective function in two-stage stochastic programming models.

In our model, we need to identify the number of different repair time scenarios necessary to obtain a clearance schedule within a specified optimality gap. In SAA, a random sample of repair times $m_b^1, m_b^2, \ldots, m_b^N$ for link b are generated to create N independent scenarios. The objective function $F(x_{sv}^{at\omega})$ given in Eq. (3.1) is approximated by the sample average function $\hat{V}_N$ defined in Eq. (4.5). Therefore, during a VNS or HTSSA run, we calculate the fitness function of a solution according to Eq. (4.5) and try to minimize $\hat{V}_N$ in each iteration.

$$Minimize\ \hat{V}_N = \frac{1}{N} \sum_{\omega=1}^{N} \sum_{(s,v)\in SV} \sum_{a\in A} \sum_{t\in T} d_a\, x_{sv}^{at\omega} \qquad (4.5)$$

Once a number of scenarios N is decided for a given network, we have to create as many N-scenario replicas as possible to represent the problem better. The number of N-scenario replications is denoted by M. An estimator of a lower bound for the objective function value of the true problem is provided in Eq. (4.6) [91]. Under mild regularity conditions, Norkin et al. [92] show that as N increases the value of Eq. (4.6) converges to the true solution of the problem.

$$\bar{V}_N^M = \frac{1}{M} \sum_{m=1}^{M} \hat{V}_N^m \qquad (4.6)$$

$\bar{V}_N^M$ is a conservative estimate and provides a statistical lower bound to the true problem.

We utilize the SAA scheme found in [5] to calculate respective upper and lower bounds and the optimality gap. We create M independent random replications of m_b^N for j = 1,…, M. For each replication, we solve the corresponding SAA problem using a VNS or HTSSA algorithm minimizing Eq. (4.5). We then calculate $\bar{V}_N^M$ as an estimator of the lower bound

for the true objective function and calculate the variance of this estimator as in [5]. We start by identifying the permutation of tasks $(\hat{x})$ giving the best $\hat{V}_N$ among M scenario replications. We then insert $(\hat{x})$ into the scenario setting of each of the $M-1$ replications and calculate corresponding objective function values, $g_N^m(\hat{x})$. The average of all such $g_N^m(\hat{x})$ constitutes an upper bound $\bar{g}_N^M(\hat{x})$ for the true problem [5]. We then compute the optimality gap gap_{NM} between the upper and lower bounds as in Eq. (4.7). The standard deviation of gap_{NM} is calculated according to Eq. (4.8) where $g_N^m(\hat{x})$ is the objective function value obtained when $\hat{x}$ is inserted into the solution of the scenario setting of m^{th} replication, and $\hat{V}_N^m$ is the objective function value calculated in Eq. (4.5).

$$gap_{NM} = \bar{g}_N^M(\hat{x}) - \bar{V}_N^M \tag{4.7}$$

$$\sigma_{gap}^2 = \frac{1}{M(M-1)} \sum_{m=1}^{M} \left[\left(g_N^m(\hat{x}) - \hat{V}_N^m \right) - \left(\bar{g}_N^M(\hat{x}) - \bar{V}_N^M \right) \right]^2 \tag{4.8}$$

Using Kleywegt et al's. [5] iterative algorithm, we increase the number of scenarios N and/or the number replications M until gap_{NM} and σ_{gap}^2 both reach some small, predetermined level. In such a case, we assume that the parameters N and M are sufficient to represent the true problem. Otherwise, we increase N and M and re-start the process to calculate the new gap_{NM} and σ_{gap}^2.

5. COMPUTATIONAL RESULTS

The following sections provide results of metaheuristic experimentations and discuss obtained results.

5.1. THE DATA

We implement our heuristics on two real networks, the road map of a residential district in Pendik and the road map of Caddebostan district in Istanbul (see Figures 5.1 and 5.2). The first district is selected because the network does not adhere to the Manhattan grid structure and has an irregular road pattern which may pose a challenge and the second one looks like a Manhattan square. Both districts are coastal to the Marmara Sea where the earthquake faults are expected to give way to an earthquake larger than 7.0 Richters according to [93].

The lengths of the streets are measured in meters using ArcGIS 10.0 software. Since anticipating the location of the disrupted links and the amount of damage to the area is beyond the scope of this study, the broken links are assigned randomly. The clearance times of disrupted links are generated randomly between [1,5] time periods according to a uniform distribution. Pendik network has 90 directed links and 32 nodes while Caddebostan has 198 links and 69 nodes. In order to test the performance of the heuristics, 15 instances each are generated for both Caddebostan and Pendik networks. The instances are created by setting different percentages of blocked links in each network. The percentage of blocked links to the number of total links in each network is set to three levels: 10 percent, 20 percent, and 30 percent. For each level of blockage, five different instances are created by randomly selecting the locations of the blocked links in each level of blockage. The nomenclature for the instances is designed as follows: C-10-3 indicates Caddebostan network with 10 percent broken links and instance 3.

5.2. COMPARISON OF INITIAL SOLUTION GENERATION PERFORMANCES

To compare the performance of the initial solution generation methods, BRS and URS, we designed an experiment where we conducted 10 replications on problem instance P-30-3,

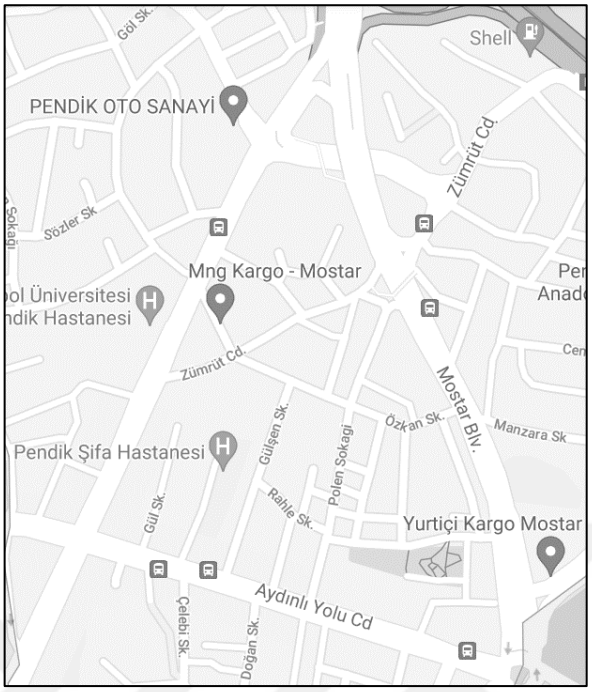

Figure 5.1 Map of Bahçelievler, Pendik/İstanbul

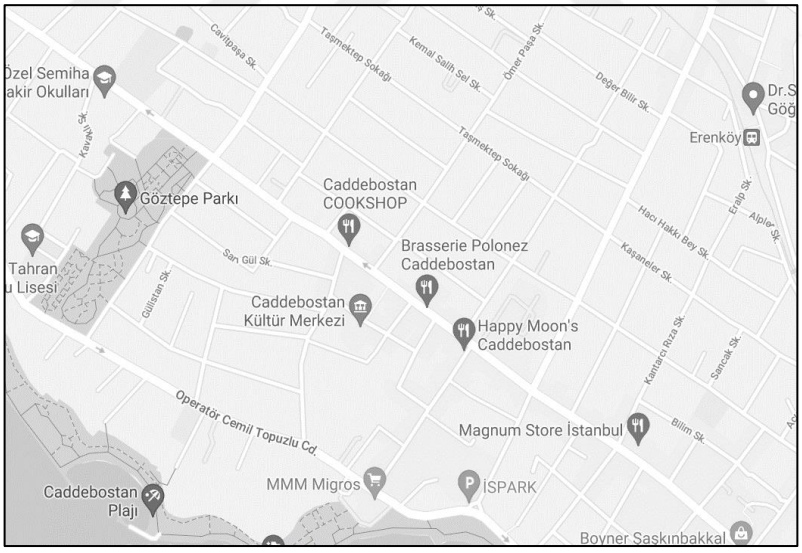

Figure 5.2 Map of Caddebostan, Kadıköy/İstanbul

where each replication generates 500 random solutions for P-30-3 using both BRS and URS methods, creating a total of 5000 initial solutions for each method. The results indicate that BRS outperforms URS in all 10 replications in terms of the solution quality: the minimum objective function value over 500 executions is 1.2 percent higher for URS on average, and

the computational times for both methods are comparable (both around 6.5 CPU minutes). Moreover, at a 95 percent confidence level, the difference between the objective function values of the two methods is statistically significant. These results indicate that BRS produces a considerably better initial solution than URS without requiring any significant additional computational burden. Therefore, we opt for initializing both VNS and HTSSA algorithms using the BRS method.

5.3. COMPARING HTSSA, VNS, AND DIFFERENT NEIGHBORHOOD OPERATORS

All experiments given in this table are conducted with the same 3-scenario data set of the Pendik network instance, P-30-3. We apply all neighborhood operators described previously to both VNS and HTSSA and compare them in terms of $F(x_{sv}^{at\omega})$ values. We provide the results in Table 5.1. In Table 5.1, different rows indicate the results obtained by different neighborhood schemes embedded in either HTSSA or VNS. Each scheme's run starts from two different initial solutions: The column titled "BRS*" summarizes the results when the metaheuristic starts from the best solution among the 5000 randomly generated solutions using the BRS technique explained above, whereas the results in the column titled "BRS" are generated by initializing the algorithms from a single solution randomly generated by BRS. Furthermore, 5 metaheuristic replications are conducted under each neighborhood scheme under VNS and HTSSA. (Note that we conducted a preliminary experiment with 5, 10, and 20 replications. The results of this experiment are given in Appendix C. The pairwise Welch's t-tests we performed on these results indicated that the means from the 5, 10, and 20 replication experiments were not significantly different at a 95 percent confidence level. Therefore, we continued our experiments with 5 replications.) BEST* indicates the minimum objective function value achieved among all methods found in Table 5.1. Under each column in Table 5.1, the average percentage deviation gap from BEST* over 5 replications are provided for each neighborhood/metaheuristic combination. The execution time limit for each run is 3600 CPU secs on the PC specified previously in CPLEX runs.

Table 5.1, the first column indicates the metaheuristic (VNS or HTSSA) and the neighborhood scheme used. Note that VNS is run using one of the neighborhood lists K_1-K_9 described above. On the other hand, HTSSA is run with either a single neighborhood or a

neighborhood list. When a neighborhood list is used, HTSSA switches to the next neighborhood operator on the list whenever a solution is rejected. The subscript of the metaheuristic indicates the neighborhood (N#) or neighborhood list (K#) where # indicates the type of neighborhood or neighborhood list used (e.g., $HTSSA_{N_1}$ or VNS_{K_2}). When the subscript contains a neighborhood list, the metaheuristic switches to the next neighborhood as soon as a non-improving solution is rejected. We also experiment with longer non-improving solution streaks to explore the current part of the solution space more thoroughly. When the neighborhood or neighborhood list is followed by a suffix #NI this notation implies that a streak of # consecutive non-improving iterations are allowed before the incumbent neighborhood scheme is changed. We also implement a version where the search restarts from a new random BRS or BRS* solution after # consecutive non-improving iterations. This version is indicated with the subscript ending –S#. Note that we experimented with these additional versions only for the neighborhood lists that produce superior results, namely VNS_{K_6}, which yielded the solution with the best objective function value (BEST*) and VNS_{K_2}, which provided the lowest average percent deviation from BEST* over 5 replications. Since prior experimentation revealed no superiority between different tabu list lengths L and β values, all experiments with HTSSA are conducted using L=3 and β=0.1.

Table 5.1 Comparison between HTSSA and VNS neighborhood generation versions on P-30-3

	Average % Deviation from BEST* over 5 replications			Average % Deviation from BEST* over 5 replications	
Metaheuristic/ Neighborhood Scheme	Starting from the Best Initial Solution (BRS*)	Starting from a Random Initial Solution (BRS)	Metaheuristic/ Neighborhood Scheme	Starting from the Best Initial Solution (BRS*)	Starting from a Random Initial Solution (BRS)
$HTSSA_{N_1}$	6.35	5.72	-	-	-
$HTSSA_{N_2}$	5.29	6.09	-	-	-
$HTSSA_{N_3}$	6.78	9.00	-	-	-
$HTSSA_{N_4}$	6.86	8.98	-	-	-
$HTSSA_{N_5}$	5.57	6.04	-	-	-
$HTSSA_{N_6}$	6.40	6.39	-	-	-
$HTSSA_{K_1}$	6.48	6.51	VNS_{K_1}	0.24	0.26
$HTSSA_{K_2}$	6.09	6.37	VNS_{K_2}	0.08	0.37
$HTSSA_{K_{2\text{-}5NI}}$	6.16	6.36	$VNS_{K_{2\text{-}5NI}}$	0.11	0.55
$HTSSA_{K_{2\text{-}10NI}}$	6.08	6.16	$VNS_{K_{2\text{-}10NI}}$	0.10	0.38
$HTSSA_{K_{2\text{-}20NI}}$	6.00	6.65	$VNS_{K_{2\text{-}20NI}}$	0.11	0.17
$HTSSA_{K_{2\text{-}S200}}$	5.60	9.61	$VNS_{K_{2\text{-}S200}}$	0.27	0.60
$HTSSA_{K_{2\text{-}S300}}$	6.19	7.42	$VNS_{K_{2\text{-}S300}}$	0.21	0.24
$HTSSA_{K_3}$	6.68	7.33	VNS_{K_3}	0.13	0.43
$HTSSA_{K_4}$	6.66	6.66	VNS_{K_4}	0.12	0.49
$HTSSA_{K_5}$	6.72	6.76	VNS_{K_5}	0.21	0.13
$HTSSA_{K_6}$	7.07	8.03	VNS_{K_6}	0.32	0.55
$HTSSA_{K_{6\text{-}5NI}}$	6.77	6.63	$VNS_{K_{6\text{-}5NI}}$	0.12	0.34
$HTSSA_{K_{6\text{-}10NI}}$	6.11	6.59	$VNS_{K_{6\text{-}10NI}}$	0.12	0.37
$HTSSA_{K_{6\text{-}20NI}}$	5.58	5.66	$VNS_{K_{6\text{-}20NI}}$	0.23	0.48
$HTSSA_{K_{6\text{-}S200}}$	5.75	9.50	$VNS_{K_{6\text{-}S200}}$	0.21	0.54
$HTSSA_{K_{6\text{-}S300}}$	6.12	7.44	$VNS_{K_{6\text{-}S300}}$	0.35	0.35
$HTSSA_{K_7}$	6.94	6.63	VNS_{K_7}	0.08	0.22
$HTSSA_{K_8}$	5.62	5.26	VNS_{K_8}	0.18	0.26
$HTSSA_{K_9}$	6.96	6.28	VNS_{K_9}	0.44	0.39

Among all implemented versions of HTSSA, $HTSSA_{N_2}$ outperforms all other versions when the search starts from the best BRS* solution. $HTSSA_{K_8}$ is the best version when the search starts from any BRS solution created for that run. Both versions have a deviation of about 5.2 percent from BEST*. It is interesting to note that while the N_2 scheme produces a slight perturbation of an incumbent solution, K_8 executes occasional large size shakes as a strategy. However, they produce comparable results. Pairwise t-tests performed on the results of HTSSA versions indicate that the difference between different neighborhood schemes is mostly insignificant, except for some versions who are initiated by a different type of solutions, BRS and BRS* (See Appendix A for the results of the pairwise t-tests for the metaheuristic versions in Table 5.1). In HTSSA, BRS* produces 7.93 percent better results on average as compared to BRS.

VNS results demonstrate that when we start the search from BRS^*, VNS_{K_2} and VNS_{K_7} yield the best outcome with a 0.08 percent deviation. However, when the starting solution is initialized with random initial solutions, VNS_{K_5} outperforms all other versions. In the literature, Paula et al. [94] recommend using three neighborhood operators in VNS design. We observe that our results are partly consistent with this recommendation, since K_2 list that has three neighborhoods is among the best performers. However, we should note that K_7 having seven neighborhoods on their lists, respectively also produces comparable results. Nevertheless, the best solution among all metaheuristic versions (BEST*), is obtained by VNS_{K_6} with 6 neighborhoods. Thus, experimenting with up to seven neighborhood operators was not a useless effort. We also observe that starting from BRS* rather than BRS significantly enhances the performance of VNS in general, BRS* -VNS deviations being 41 percent better than BRS -VNS on average. Pairwise comparison tests of percentile deviations performed at 95 percent confidence level given in Appendix A also support this observation. However, paired tests among VNS versions with the same initialization procedure indicate that only a few VNS pairs exhibit statistically different performance. Based on the results from both HTSSA and VNS versions, we can state that using the BRS* initialization provides better overall performance for both metaheuristics. Therefore, investing more time into creating a high-quality initial solution using the BRS* is worthwhile.

When we compare the VNS variants with HTSSA variants, we see that VNS variants consistently provide superior results to HTSSA variants. Initialized by BRS*, the average

deviation obtained by VNS variants is 6.08 percent less than that of HTSSA variants, and the performance difference between the two metaheuristics increases slightly to 6.59 percent when BRS initialization is used. Furthermore, the results in Appendix A indicate that the difference between the VNS and HTSSA metaheuristics is significant regardless of the neighborhood scheme used. This difference may be due to the hill-climbing approach used in the VNS variants as opposed to the HTSSA variants where inferior solutions are allowed probabilistically. Given the computationally expensive objective function evaluation to be executed in each iteration, a hill-climbing approach has the advantage of achieving high-quality solutions within a limited time.

Both HTSSA and VNS execute around an average of 4800 iterations in a 3600 CPU sec run. In Figure 5.3, we illustrate the percentage improvement of the top three performing VNS variants with respect to the BRS* initial solution. We observe that after the first 1000 iterations, the rates of improvement of VNS_{K_2} and VNS_{K_7} slow down significantly while $VNS_{K_{2\text{-}10NI}}$ converges relatively more slowly. In fact, the quality of solutions improves exponentially for all variants within the first 500 iterations. This implies that good solutions are obtained within the first 10 minutes of a run. Fast convergence to a high-quality solution is crucial in a disaster setting where decisions need to be made quickly to deliver timely assistance to those in need.

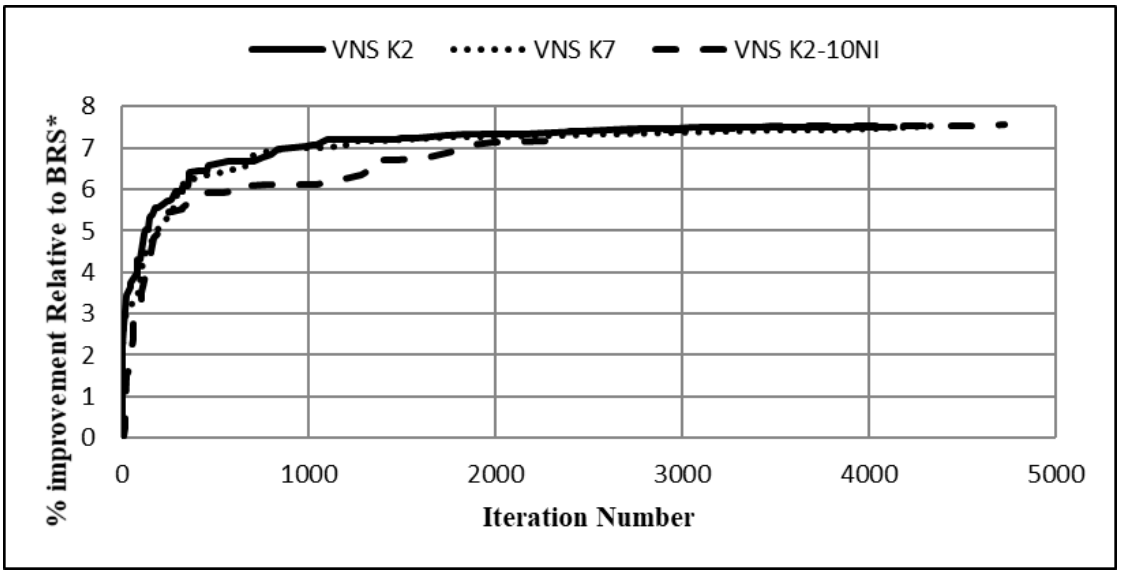

Figure 5.3 The percentage improvement of the three best-performing algorithms over BRS*

Next, we introduce the 17 instances to test the top three performing metaheuristic schemes in prior experimentation (P-30-3). For each instance (C: Caddebostan, P: Pendik) two geographical location is chosen as explained in Section 5.1. The degree of damage to the area is implemented in three levels (10: 10 percent of the entire links, 20: 20 percent of the entire links, 30: 30 percent of the entire links in the network). Additionally in each instance, the position of the broken links differs (i.e., P-30-1/2/3). For each instance, the BRS mechanism is run for 5000 executions and the resulting permutation is used as an initial solution. Table 5.2 demonstrates average improvement over BRS^* for each instance. Each row corresponds to a different instance while the columns indicate the average improvement provided for that instance by each scheme.

Table 5.2 Average improvement of neighborhood schemes over BRS^* in 18 instances

	Metaheuristic/ Neighborhood Scheme					
Instance	VNS_{K_2}	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
C-10-1	0.37	0.37	0.37	0.00	0.03	0.03
C-10-2	0.55	0.55	0.56	0.01	0.01	0.08
C-10-3	0.48	0.48	0.49	0.00	0.01	0.00
C-20-1	41.72	39.26	41.61	25.45	25.40	5.45
C-20-2	46.92	45.14	46.91	29.77	27.24	9.74
C-20-3	50.80	49.44	52.00	53.82	46.52	34.56
C-30-1	49.94	52.81	50.44	40.41	37.69	15.02
C-30-2	52.18	51.84	47.97	37.25	35.56	12.24
C-30-3	47.74	42.09	42.09	37.17	25.70	14.73
P-10-1	0.00	-	0.00	0.00	0.00	0.00
P-10-2	0.00	-	0.00	0.00	0.00	0.00
P-10-3	0.01	-	0.01	0.00	0.00	0.00
P-20-1	1.23	-	1.23	0.18	0.18	0.19
P-20-2	1.10	-	1.10	0.23	0.07	0.16
P-20-3	1.14	-	1.13	0.18	0.08	0.10
P-30-1	7.34	7.31	7.25	1.45	1.05	1.15
P-30-2	8.58	8.33	8.67	2.44	1.85	0.72
P-30-3	7.22	7.22	7.21	2.10	1.82	1.80

When we compare the average improvement percentage relative to the initial solution of VNS to HTSSA (Table 5.1 and Appendix A), we see that the results are consistent with prior findings meaning that in almost all instances VNS beats HTSSA in terms of performance. All pairwise t-tests of the experimentation demonstrated in Table 5.2 are presented in

Appendix B. VNS schemes result in the same percentage improvement in the C-10-1 instance. In C-10-2/3 $VNS_{K_{2\text{-}10NI}}$ performs best with a 0.01 percent difference, with no significant difference between sample means of VNS schemes at a 95 percent level. On the other hand, HTSSA and VNS results are proved significantly different. In C-20-1/2 instances, the best performing scheme is VNS_{K_2}, but no significant difference was observed between VNS variants. $HTSSA_{K_{6\text{-}20NI}}$ have non-equal means with HTSSA and VNS versions. $HTSSA_{N_2}$ and $HTSSA_{N_5}$ have no significant difference in pairwise comparison. VNS and HTSSA versions have a significant difference when compared in terms of equal means for the samples. C-20-3 performed incompatible with the rest of the experimentation. $HTSSA_{N_2}$ outperforms all other schemes with respect to average improvement relative to the initial solution beating the best VNS version, $VNS_{K_{2\text{-}10NI}}$, with a 0.02 percent. Additionally, no significant difference between the means of all schemes is present at the 95 percent significance level.

In C-30-1, $VNS_{K_{2\text{-}10NI}}$ and VNS_{K_7} perform better than VNS_{K_2}, VNS_{K_7} outperforms all. No significant difference is observed between VNS variants. VNS_{K_2} performs worst among VNS variants having no significant difference with $HTSSA_{N_2}$, the variant that is superior to other HTSSA variants. $HTSSA_{N_5}$ is the second top-performing variant in HTSSA variants but has a significant difference with top-performing VNS variants, $VNS_{K_{2\text{-}10NI}}$ and VNS_{K_7}. Far worse performance is obtained by $HTSSA_{K_{6\text{-}20NI}}$ and the pairwise t-test results show significant differences with all other variants. C-30-2 and C-30-3 results are compliant with the prior results in Table 5.1. In C-30-2 $VNS_{K_{2\text{-}10NI}}$ and $HTSSA_{N_2}$ results in no significant difference and in C-30-3 $HTSSA_{N_2}$ shows difference with $HTSSA_{K_{6\text{-}20NI}}$.

For P-10-1/2/3 instances no improvement over BRS* the solution is obtained except for P-10-3 with a weak improvement when VNS_{K_2} and $VNS_{K_{2\text{-}10NI}}$ is employed. All variants of metaheuristics schemes are compared pairwise and have no significant difference. In VNS_{K_7}, K_7 list has N_{10} neighborhood that randomly picks 10 broken links and assigns each selected link to a new position. VNS_{K_7} is not available for P-10 and P-20 instances since the total number of broken links is not sufficient to iterate through one or more steps. P-20-1/2/3 and P-30-1/2 instances have resulted in exactly the same as in Table 5.1 and Appendix A which

means VNS beats HTSSA. P-30-2 resulted in significantly different means for $HTSSA_{K_{6\text{-}20NI}}$ and $HTSSA_{N_2}$.

In the above experiments, we limited the number of scenarios per problem instance to three scenarios. While using a large number of scenarios improve representation, our experiments indicate that increasing the number of scenarios in a problem instance significantly increases the computational time required to solve it. Figure 5.4 demonstrates that the computational effort for executing each VNS iteration increases exponentially with the number of scenarios. In this figure, the horizontal axis indicates the number of scenarios used, whereas the vertical axis displays the number of iterations performed within 3600 CPU seconds for problem instance P-30-3 using VNS_{K_2}. Given this increased computational requirement, we opted to use a limited number of scenarios in order to allow each heuristic version to perform more iterations so that we can better analyze the performance difference among VNS and HTSSA versions. However, in practice, it is important to use a sufficient number of scenarios to represent the uncertainty in the problem. In the following section, we report the results of the SAA technique that we apply to find the optimal number of scenarios to be used.

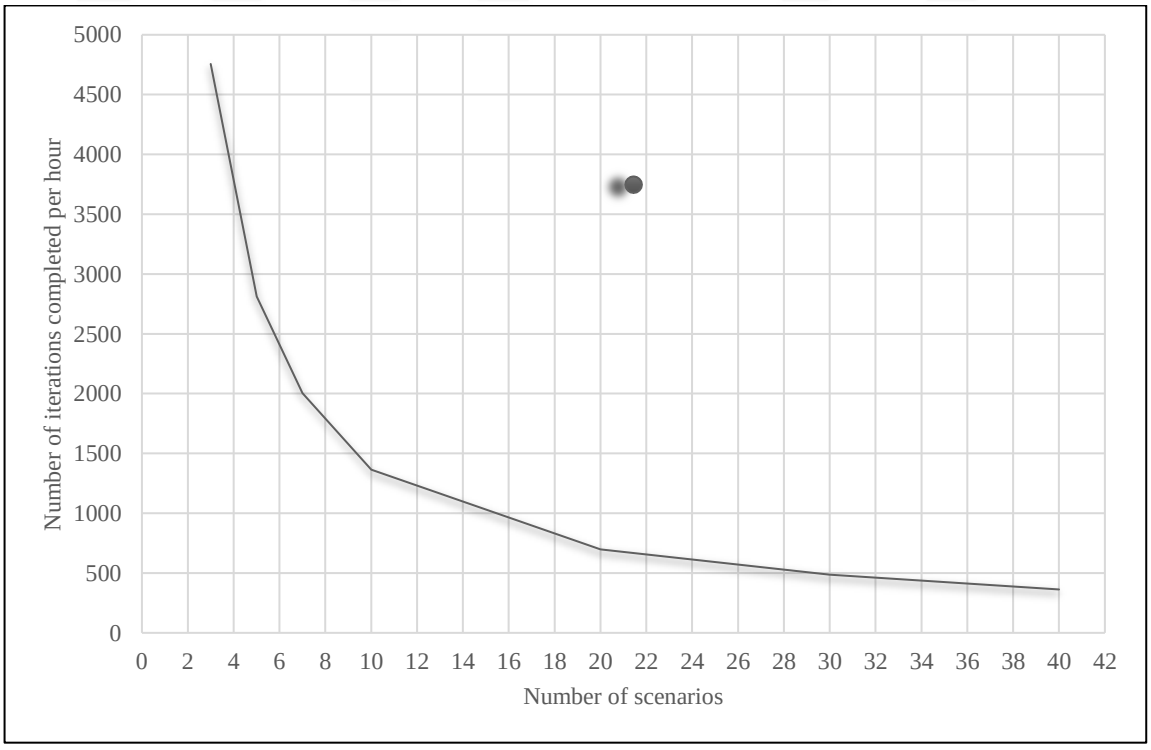

Figure 5.4 Impact of the number of scenarios used on the computational time for the VNS heuristic

We also used the VNS_{K_2} heuristic to solve the illustrative example in Section 3.1. Note that solution of the stochastic model using CPLEX takes 7200 secs to obtain an optimal solution. For comparison, we performed 20 runs of VNS_{K_2} starting from a single BRS solution. The heuristic reached the optimal solution in an average of 62 iterations within the first minute. We should also note that VNS_{K_2} is able to perform an average of 185355 iterations in an hour for this small illustrative example.

To summarize, computational experiments demonstrate that the VNS initialized with BRS* offers the best performance. However, the performance difference among variants with different neighborhood schemes is not very pronounced. Nevertheless, we have decided to use the variant that provided the best average deviation from BEST*, namely, VNS_{K_2} while implementing SAA to validate the solution obtained.

5.4. PERFORMANCE OF THE ROLLING HORIZON APPROACH

We implemented the rolling horizon approach on instance P-10-1 using the VNS_{K_2} variant, which provides the smallest average deviation from BEST*. Table 5.3 compares the performance of this approach with the results of implementing the initial sequence as is (without any updates) and applying the clearance sequence created based on perfect information. The results in this table demonstrate that the rolling horizon approach provides a significant improvement over the DCP solution created using the initial estimates.

Table 5.3 Repair times of broken links under scenarios $\boldsymbol{\omega_1}$-$\boldsymbol{\omega_3}$

	Job	**#19**	**#33**	**#37**	**#47**	**#51**	**#57**	**#79**	**#85**	**#89**
Repair Time	$\boldsymbol{\omega_1}$	2	5	5	1	2	2	2	2	2
	$\boldsymbol{\omega_2}$	4	3	4	2	5	1	2	3	5
	$\boldsymbol{\omega_3}$	3	3	2	3	2	3	2	4	4

We implement the rolling horizon approach depicted in Figure 1.1 to solve P-10-1. The DCP models created at each iteration are solved using BRS and VNS_{K_2} duo. At each DCP iteration, BRS is run for 5000 iterations to acquire BRS* and the search is run for an hour

with VNS_{K_2}. Initially, the clearance sequence is #79 < #47 < #85 < #33 < #19 < #37 < #57 < #89 < #51 at t=0. Detailed information regarding the rolling horizon approach implementation for instance P-10-1 is demonstrated in Table 5.4. In the table values in bold indicate updated information. Based on this sequence, jobs #47 and #79 are assigned to the two available crews at the beginning of the recovery effort, i.e., t=0. Once the two crews start their work, they report new repair time estimates of 5 and 8 for jobs #47 and #79, respectively. Based on the updated estimates, the first job completed is job #47 at t=5. Therefore, we solve the DCP again at t=5 to revise the clearance sequence based on the updated repair times. In the updated DCP, the repair time for job #89 is taken as 3, i.e., the remaining time to complete the job at t=5, and job #89 is fixed as the first job on the sequence. (In this example, we assume that the repair time estimates become available as soon as the crew starts the job and rerun DCP at that time. In practice, DCP would be run again when the crews report updated repair times.) The clearance schedule for the DCP with updated repair times is computed as #33<#19<#37<#85<#57<#51. Note that the order of jobs #57 and #85 is reversed compared to the original sequence and the freed-up crew is assigned to the first job on the updated sequence, i.e., job #89. Next, the repair time for job #19 and job #33 is communicated by the clearance crews as 1 and 4. Therefore, DCP is solved again at t=8 to yield sequence #37<#85<#57<#51, which indicates that the freed-up crew should be assigned to job #37. The next job to complete is #37 at t=9, when the DCP yields the sequence #57<#85<#51 and the available crew is now assigned to the first job on the sequence, i.e., job #37. Again, note that the order of jobs #57 and #85 is reversed compared to the previous sequence. At t=12 job #57 repair time is updated to 1 and DCP is solved once more. Resulting sequence #85<#51. Finally, at t=13, DCP is solved once more upon the completion of job #57, indicating the freed-up crew should now be assigned to job #51, which leaves job #51 to be the last job to be completed. Based on the five DCP solutions throughout the recovery effort, the clearance sequence implemented by the rolling horizon approach is #79<#47<#89<#19<#33<#37<#57<#85<#51 and the resulting objective function value is 27,903,026.37.

Table 5.5 compares the performance of the rolling horizon approach with implementing the original sequence created based on the initial repair time estimates obtained immediately after the disaster. When the sequence is not revised based on the updated repair time estimates from the field, the objective function value increases to 28,133,056.84. On the

other hand, the sequence created using the deterministic DCP using the updated information i.e., the solution with perfect information yields an objective function value of 27,448,071.85. These results demonstrate the merit of using the rolling horizon approach to revise the clearance sequence as data becomes available throughout the recovery process.

Table 5.4 Summary of results of the DCPs solved according to the rolling horizon approach

t	Repair Time Update	Link Repair Times (Realization/Remaining)									Clearance Sequence
		#19	#33	#37	#47	#51	#57	#79	#85	#89	
0	#79, #47	- /-	- /-	- /-	**5/5**	- /-	- /-	**8/8**	- /-	- /-	#85<#33<#19<#37<#57<#89<#51
5	#89	- /-	- /-	- /-	5/**0**	- /-	- /-	8/**3**	- /-	**3/3**	#33<#19<#37<#85<#57<#51
8	#19, #33	**1/1**	**4/4**	- /-	5/0	- /-	- /-	8/**0**	- /-	3/**0**	#37<#85<#57<#51
9	#37	1/**0**	4/**3**	**9/9**	5/0	- /-	- /-	8/0	- /-	3/0	#57<#85<#51
12	#57	1/0	4/**0**	9/**6**	5/0	- /-	**1/1**	8/0	- /-	3/0	#85<#51
13	#85	1/0	4/0	9/**5**	5/0	- /-	1/**0**	8/0	**6/6**	3/0	#51
Summary											
Repair Time Realizations									Sequence Resulting from The Rolling Horizon		
#19:1, #33:4, #37:9, #47: 5, #51:7, #57:1, #79:8, #85:6, #89:3									#79<#47<#89<#19<#33<#37<#57<#85<#51		

Table 5.5 Summary of results for the P-10-1 instance

	Objective Function Value	Final Sequence
Solution with Initial Estimates	28,133,056.84	#79<#47<#85<#33<#19<#37<#57<#89<#51
Rolling Horizon Solution	27,903,026.37	#79<#47<#89<#19<#33<#37<#57<#85<#51
Solution with Perfect Information	27,448,071.85	#33<#47<#89<#19<#79<#57<#37<#85<#51

5.5. COMPUTATIONAL PERFORMANCE WITH RESPECT TO PROBLEM SIZE

For the experimentation on the computational performance of the algorithm VNS_{K_2} eight random instances are introduced. The instance with the smallest size has a total of 142 arcs and arcs are gradually added to this instance to create larger instances with up to 1700 arcs. Total number of arcs for instances 1-8 are: 142, 152, 164, 218, 374, 638, 970, 1700. To observe the impact of level of network damage on the computational performance, four versions of each of these eight instances were created with respectively 10, 20, 30, 40 percent of the arcs in the network are broken.

Figure 5.5 displays the number of iterations completed versus broken link percentage levels for all instances. As the number of broken arcs in the network increases, the number of iterations completed decreases. On the other hand, an increase in the network size impacts the number of iterations completed more dramatically. Moreover, for small size networks, an increase in broken link percentage leads to a more pronounced decline in the number of completed iterations. As the network gets larger, the impact of broken link percentage on the number of iterations decreases gradually.

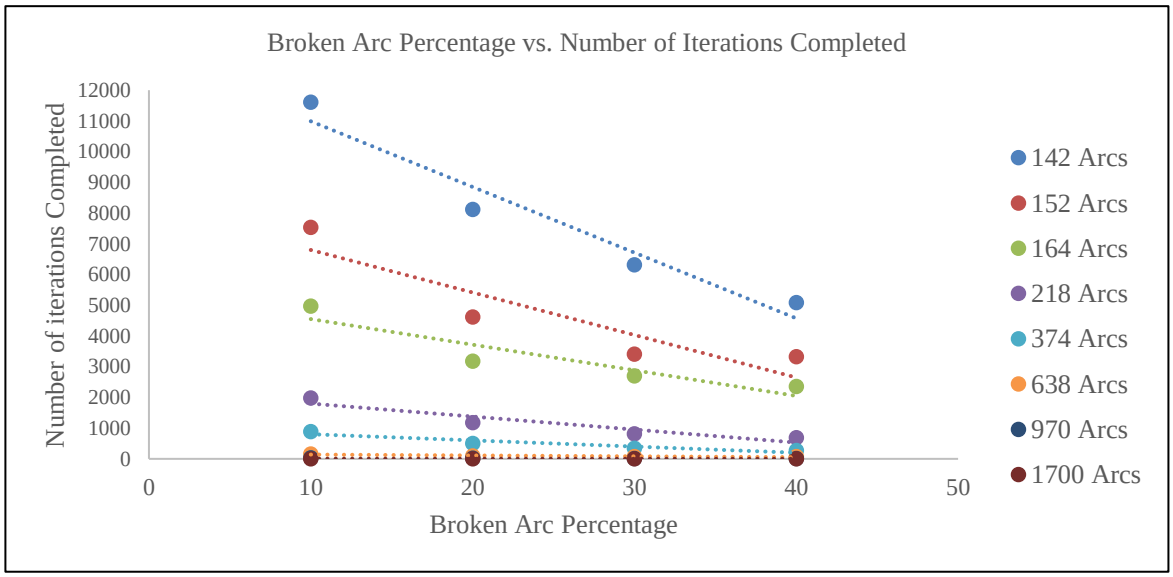

Figure 5.5 Iterations completed versus percentage levels for the VNS heuristic

5.6. SAMPLE AVERAGE APPROXIMATION (SAA) RESULTS

The SAA technique is used to find out the number of stochastic repair duration scenarios that would result in a robust clearance schedule. SAA experimentations are initialized with the BRS^* solution and VNS_{K_2} runs are conducted to minimize the objective function value in Eq. (4.5). [91] prove that Eq. (4.7) overestimates the optimality gap, and, as N increases, this bias decreases. For computationally expensive problems, [5] suggest increasing the number of replications, M. We prefer to increase M rather than N. The results obtained for the iterations of Kleywegt et al.'s [5] algorithm are provided in Table 5.3. We have tested 3, 5, and 7 scenario instances with 50, 100, 150, and 200 replications in each scenario setting. We report the percentage ratio of the upper bound $\bar{g}_N^M(\hat{x})$ to the lower bound $\bar{V}_N^M$ which we denote as % gap_{NM} in Table 5.3. We observe that gap_{NM} is not reduced at every step when the number of scenarios is as small as 3. Therefore, we increase N to five and then, to 7, to ensure that the optimality gap is reduced or remains the same each time M increases. The upper bounds of the 95 percent confidence intervals for % gap_{NM} are also indicated in Table 5.3, also as percentages. That is when N equals 7 and M is 200, the upper bound of % gap_{NM} is 2.6 percent. Note that when N equals 7 and M is 50, % gap_{NM} becomes 2.5 percent, which is our pre-specified stopping criterion level for % gap_{NM}. The corresponding 95 percent confidence interval's upper bound is also very close to % gap_{NM} at that level and gets further closer as M increases. At the five and seven scenario cases the gap decreases as further replications are added. This is not the case as in scenario three scenario level. As Kleywegt et al. [5] assert that, the scenario solution objective function values have a discrete distribution and the probability that a new replication would produce a better result is less than 1/(M+1). Thus, in the three scenario cases, the results are reasonable considering the probability that the new replication produces a better result is decreasing with the replication number increase.

Table 5.6 Sample Average Approximation Results

	N=3		N=5		N=7	
	% gap_{NM}	95% upper bound for % gap_{NM}	% gap_{NM}	95% upper bound for % gap_{NM}	% gap_{NM}	95% upper bound for % gap_{NM}
M=50	5.5	6.1	4	4.5	2.5	2.8
M =100	5.7	6.1	4	4.3	2.5	2.8
M =150	5.9	6.2	3.9	4.2	2.5	2.7
M =200	6.1	6.4	3.9	4.1	2.5	2.6

We now validate the solutions obtained in the seven-scenario, 200 replications experiment. The best three objective function values among 200 replications are provided in the second column of Table 5.4. The permutations corresponding to these three solutions are each evaluated with a 1000-scenario instance. The objective function values calculated in the 1000-scenario setting are indicated in the third column. We observe that in the 1000-scenario case, the objective function values of these three solutions do not differ substantially from each other. Thus, they could be used safely as solutions to the true problem.

Table 5.7 Sample Average Approximation Validation

Best 3 solutions	$\hat{V}_7^{200}$	$\hat{V}(\hat{x})_{1000}$
Solution $\hat{x}_1$	40112358.36	43475781.11
Solution $\hat{x}_2$	39966200.64	43487224.69
Solution $\hat{x}_3$	40167022.08	43625370.32

6. CONCLUSION

In this study, we introduce a new stochastic debris clearance problem that focuses on the accessibility of the entire road network. We propose a novel MIP formulation for this problem in the form of a stochastic multi-period network flow model with discrete variables for crew assignment and work prioritization. The model tracks the accessibility for all pairs of nodes in the network during the progress of the clearance operations. Due to the complexity of the resulting MIP model, we develop two metaheuristics for this problem, one Variable Neighborhood Search Algorithm (VNS) and another hybrid algorithm that combines Tabu Search and Simulated Annealing (HTSSA). Computational results demonstrate that the VNS algorithm, which employs a hill-climbing strategy offers better solution quality than HTSSA. We attribute this performance difference to the acceptance of inferior solutions in HTSSA, which leads the algorithm to spend time in unpromising parts of the solution space. Due to the time-consuming evaluation of the objective function value, a hill-climbing approach that concentrates on improving the solution quickly appears to be more appropriate for the debris clearance problem. Along the same lines, we see that the quality of the initial solution used at the beginning of the algorithm is instrumental in achieving better solutions. Our experiments indicate that creating many alternative solutions using a biased random sampling approach (BRS) and initializing the metaheuristic with the best one produces good results. We also note that the improvement achieved by the VNS algorithm over the best starting solution BRS^* increases with the level of blockage in the network.

Given the stochastic nature of the problem, representing the true problem using a limited number of scenarios is a challenge. To address this issue, we utilize a sample average approximation technique to determine the minimum number of scenarios required for the results to sufficiently represent the true objective function. Computational experiments show that a 2 percent gap can be achieved using 7 scenarios, which is a reasonable value for the operational problem at hand.

To address the online nature of the debris clearance problem, we propose a rolling horizon approach that continuously revises the clearance schedule during the recovery effort based

on the latest repair time estimates from the clearance crews. We illustrate the implementation of this rolling horizon approach on example problems and demonstrate the benefit of revising the clearance based on the latest information.

In conclusion, we would also like to briefly discuss the implementation aspects of the methodology proposed here. For an operational problem especially in a post-disaster setting, the decision support must be provided quickly, and the recommendations are simple and easy to follow. The outcome of the proposed approach is a sequence of the road segments to be cleared to maximize overall accessibility after estimates of clearance durations are made after the disaster takes place. It is straightforward to use this sequence as a priority order and assign road clearance teams to blocked roads in the order they appear in the sequence. Therefore, the solution to the debris clearance problem introduced here can be communicated and implemented easily after real disasters. Also, the metaheuristic algorithms proposed to solve this problem can create high-quality solutions in as low as 10 minutes, which will allow the solutions to be communicated to the disaster management authorities on time. We should also note that the success of the proposed approach is highly dependent on the clearance duration estimates used. Immediately after a disaster, it is critical to allocate resources to produce precise estimates using GIS-based debris assessment and/or helicopter surveys in order to ensure a successful disaster recovery operation. On-going communication with the clearance crews in the field is also crucial to update these initial estimates so that the clearance sequence can be revised based on the latest and most accurate information. Given the latest estimates, we believe that the proposed methodology offers a holistic and practical approach to address the work prioritization issues in disaster recovery.

REFERENCES

1. Özdamar L, Ertem MA. Models, solutions and enabling technologies in humanitarian logistics. *European Journal of Operational Research*. 2015; 244: 55–65.

2. Çelik M. Network restoration and recovery in humanitarian operations: Framework, literature review, and research directions. *Surveys in Operations Research and Management Science*. 2016; 21: 47–61.

3. Karimzadeh S, Miyajima M, Hassanzadeh R, et al. A GIS-based seismic hazard, building vulnerability and human loss assessment for the earthquake scenario in Tabriz. *Soil Dynamics and Earthquake Engineering* 2014; 66: 263–280.

4. Tuzun Aksu D, Ozdamar L. A mathematical model for post-disaster road restoration: Enabling accessibility and evacuation. *Transportation Research Part E: Logistics and Transportation Review*. 2014; 61: 56–67.

5. Kleywegt AJ, Shapiro A, Homem-De-Mello T. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*. 2002; 12: 479–502.

6. Özdamar L, Tüzün Aksu D, Ergüneş B. Coordinating debris cleanup operations in post disaster road networks. *Socio-Economic Planning Sciences*. 2014; 48: 249–262.

7. Averbakh I, Pereira J. The flowtime network construction problem. *IIE Transactions (Institute of Industrial Engineers)*. 2012; 44: 681–694.

8. Sakuraba CS, Santos AC, Prins C, et al. Road network emergency accessibility planning after a major earthquake. *EURO Journal on Computational Optimization*. 2016; 4: 381–402.

9. Maya Duque PA, Dolinskaya IS, Sörensen K. Network repair crew scheduling and routing for emergency relief distribution problem. *European Journal of Operational Research*. 2016; 248: 272–285.

10. Duque PM, Sörensen K. A GRASP metaheuristic to improve accessibility after a

disaster. *OR Spectrum*. 2011; 33: 525–542.

11. Yan S, Shih YL. A time-space network model for work team scheduling after a major disaster. *Journal of the Chinese Institute of Engineers, Transactions of the Chinese Institute of Engineers,Series A/Chung-kuo Kung Ch'eng Hsuch K'an*. 2007; 30: 63–75.

12. Yan S, Shih YL. An ant colony system-based hybrid algorithm for an emergency roadway repair time-space network flow problem. *Transportmetrica*. 2012; 8: 361–386.

13. Çelik M, Ergun Ö, Keskinocak P. The post-disaster debris clearance problem under incomplete information. *Operations Research*. 2015; 63: 65–85.

14. Moreno A, Munari P, Alem D. A branch-and-Benders-cut algorithm for the Crew Scheduling and Routing Problem in road restoration. *European Journal of Operational Research*. 2019; 275: 16–34.

15. Ulusan A, Ergun O. Restoration of services in disrupted infrastructure systems: A network science approach. *PLoS ONE*. 2018; 13: 1–29.

16. Morshedlou N, González AD, Barker K. Work crew routing problem for infrastructure network restoration. *Transportation Research Part B: Methodological*. 2018; 118: 66–89.

17. Furuta H, Ishibashi K, Tsu KN, et al. Optimal restoration scheduling of damaged networks under uncertain environment by using improved genetic algorithm. *Tsinghua Science and Technology*. 2008; 13: 400–405.

18. Tang CH, Yan S, Chang CW. Short-term work team scheduling models for effective road repair and management. *Transportation Planning and Technology*. 2009; 32: 289–311.

19. Sahin H, Kara BY, Karasan OE. Debris removal during disaster response: A case for Turkey. *Socio-Economic Planning Sciences*. 2016; 53: 49–59.

20. Kasaei M, Salman FS. Arc routing problems to restore connectivity of a road network.

Transportation Research Part E: Logistics and Transportation Review. 2016; 95: 177–206.

21. Akbari V, Salman FS. Multi-vehicle prize collecting arc routing for connectivity problem. *Computers and Operations Research*. 2017; 82: 52–68.

22. Akbari V, Salman FS. Multi-vehicle synchronized arc routing problem to restore post-disaster network connectivity. *European Journal of Operational Research*. 2017; 257: 625–640.

23. Ajam M, Akbari V, Salman FS. Minimizing latency in post-disaster road clearance operations. *European Journal of Operational Research*. 2019; 277: 1098–1112.

24. Berktaş N, Kara BY, Karaşan OE. Solution methodologies for debris removal in disaster response. *EURO Journal on Computational Optimization*. 2016; 4: 403–445.

25. Caunhye AM, Aydin NY, Duzgun HS. Robust post-disaster route restoration. *OR Spectrum*. 2020; 42: 1055–1087.

26. Ransikarbum K, Mason SJ. Multiple-objective analysis of integrated relief supply and network restoration in humanitarian logistics operations. *International Journal of Production Research*. 2016; 54: 49–68.

27. Li S, Teo KL. Post-disaster multi-period road network repair: work scheduling and relief logistics optimization. *Annals of Operations Research*. 2019; 283: 1345–1385.

28. Aslan E, Çelik M. Pre-positioning of relief items under road/facility vulnerability with concurrent restoration and relief transportation. *IISE Transactions*. 2019; 51: 847–868.

29. Sanci E, Daskin MS. Integrating location and network restoration decisions in relief networks under uncertainty. *European Journal of Operational Research*. 2019; 279: 335–350.

30. Shin Y, Kim S, Moon I. Integrated optimal scheduling of repair crew and relief vehicle after disaster. *Computers and Operations Research*. 2019; 105: 237–247.

31. Rezaei H, Khademi Zare H, Bashiri M, et al. A humanitarian reconfiguration and rehabilitation model for preparedness and response to earthquakes using a scheduled reopening of links. *Journal of Industrial and Systems Engineering*. 2018; 11: 96–115.

32. Hoyos MC, Morales RS, Akhavan-Tabatabaei R. OR models with stochastic components in disaster operations management: A literature survey. *Computers and Industrial Engineering*. 2015; 82: 183–197.

33. Liberatore F. Preface. *Decision Aid Models for Disaster Management and Emergencies*. 2013; 7: vii–x.

34. Balcik B, Bozkir CDC, Kundakcioglu OE. A literature review on inventory management in humanitarian supply chains. *Surveys in Operations Research and Management Science*. 2016; 21: 101–116.

35. Jia H, Ordóñez F, Dessouky MM. Solution approaches for facility location of medical supplies for large-scale emergencies. *Computers and Industrial Engineering*. 2007; 52: 257–276.

36. Balcik B, Beamon BM. Facility location in humanitarian relief. *International Journal of Logistics Research and Applications*. 2008; 11: 101–121.

37. Rawls CG, Turnquist MA. Pre-positioning of emergency supplies for disaster response. *Transportation Research Part B: Methodological*. 2010; 44: 521–534.

38. Rawls CG, Turnquist MA. Pre-positioning and dynamic delivery planning for short-term response following a natural disaster. *Socio-Economic Planning Sciences*. 2012; 46: 46–54.

39. Salmerón J, Apte A. Stochastic optimization for natural disaster asset prepositioning. *Production and Operations Management*. 2010; 19: 561–574.

40. Falasca M, Zobel CW. A two-stage procurement model for humanitarian relief supply chains. *Journal of Humanitarian Logistics and Supply Chain Management*. 2011; 1: 151–169.

41. Döyen A, Aras N, Barbarosoğlu G. A two-echelon stochastic facility location model

for humanitarian relief logistics. *Optimization Letters*. 2012; 6: 1123–1145.

42. Wang J, Yang H, Zhu J. A Two-stage Stochastic Programming Model for Emergency Resources Storage Region Division. *Systems Engineering Procedia*. 2012; 5: 125–130.

43. Peeta S, Sibel Salman F, Gunnec D, et al. Pre-disaster investment decisions for strengthening a highway network. *Computers and Operations Research*. 2010; 37: 1708–1719.

44. Tzeng GH, Cheng HJ, Huang TD. Multi-objective optimal planning for designing relief delivery systems. *Transportation Research Part E: Logistics and Transportation Review*. 2007; 43: 673–686.

45. Yan S, Shih YL. Optimal scheduling of emergency roadway repair and subsequent relief distribution. *Computers and Operations Research*. 2009; 36: 2049–2065.

46. Stepanov A, Smith JM. Multi-objective evacuation routing in transportation networks. *European Journal of Operational Research*. 2009; 198: 435–446.

47. Zhang JH, Li J, Liu ZP. Multiple-resource and multiple-depot emergency response problem considering secondary disasters. *Expert Systems with Applications*. 2012; 39: 11066–11071.

48. Barbarosoğlu G, Arda Y. A two-stage stochastic programming framework for transportation planning in disaster response. *Journal of the Operational Research Society*. 2004; 55: 43–53.

49. Mete HO, Zabinsky ZB. Stochastic optimization of medical supply location and distribution in disaster management. *International Journal of Production Economics*. 2010; 126: 76–84.

50. Bozorgi-Amiri A, Jabalameli MS, Alinaghian M, et al. A modified particle swarm optimization for disaster relief logistics under uncertain environment. *International Journal of Advanced Manufacturing Technology*. 2012; 60: 357–371.

51. Bozorgi-Amiri A, Jabalameli MS, Mirzapour Al-e-Hashem SMJ. A multi-objective

robust stochastic programming model for disaster relief logistics under uncertainty. *OR Spectrum*. 2013; 35: 905–933.

52. Rennemo SJ, Rø KF, Hvattum LM, et al. A three-stage stochastic facility routing model for disaster response planning. *Transportation Research Part E: Logistics and Transportation Review*. 2014; 62: 116–135.

53. Noyan N, Balcik B, Atakan S. A stochastic optimization model for designing last mile relief networks. *Transportation Science*. 2016; 50: 1092–1113.

54. Najafi M, Eshghi K, Dullaert W. A multi-objective robust optimization model for logistics planning in the earthquake response phase. *Transportation Research Part E: Logistics and Transportation Review*. 2013; 49: 217–249.

55. Ozdamar L. Disaster Relief Logistics. *Wiley StatsRef: Statistics Reference Online*. 2017; 1–11.

56. Grass E, Fischer K. Two-stage stochastic programming in disaster management: A literature survey. *Surveys in Operations Research and Management Science*. 2016; 21: 85–100.

57. Gutjahr W. Recent trends in metaheuristics for stochastic combinatorial optimization. *Open Computer Science* 2011; 1: 58–66.

58. Bianchi L, Dorigo M, Gambardella LM, et al. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*. 2009; 8: 239–287.

59. Juan AA, Faulin J, Grasman SE, et al. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives*. 2015; 2: 62–72.

60. Fred Glover. Tabu Search - Part I. *Orsa Journal on Computing*. 1989; 1: 190–206.

61. Glover F. Tabu Search -Part I and Part II. *Journal on Computing*. 1989; 1 and 2: 190–206, 4-32,.

62. Birge JR, Dempster MAH. Stochastic programming approaches to stochastic

scheduling. *Journal of Global Optimization*. 1996; 9: 417–451.

63. Wilson JM, Birge J, Louveaux F. Stochastic Programming. *The Journal of the Operational Research Society*. 1998; 49: 897.

64. Gendreau M, Laporte G, Seguin R. A Tabu Search Heuristic for the Vehicle Routing Problem with Stochastic Demands and Customers. *Operations Research*. 1996; 44: 469–477.

65. Aringhieri R. Solving chance-constrained programs combining tabu search and simulation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2004; 3059: 30–41.

66. Bianchic L, Birattari M, Chiarandini M, et al. Metaheuristics for the vehicle routing problem with stochastic demands. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2004; 3242: 450–460.

67. Haugland D, Ho SC, Laporte G. Designing delivery districts for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*. 2007; 180: 997–1010.

68. Sungur I, Ren Y, Ordóñez F, et al. A model and algorithm for the courier delivery problem with uncertainty. *Transportation Science*. 2010; 44: 193–205.

69. Shukla N, Choudhary AK, Prakash PKS, et al. Algorithm portfolios for logistics optimization considering stochastic demands and mobility allowance. *International Journal of Production Economics*. 2013; 141: 146–166.

70. Teng S, Ong HL, Huang HC. A comparative study of metaheuristics for vehicle routing problem with stochastic demands. *Asia-Pacific Journal of Operational Research*. 2003; 20: 103–119.

71. Costa D, Silver EA. Tabu Search When Noise is Present: An Illustration in the Context of Cause and Effect Analysis. *Journal of Heuristics*. 1998; 4: 5–23.

72. Salman FS, Yücel E. Emergency facility location under random network damage:

Insights from the Istanbul case. *Computers and Operations Research*. 2015; 62: 266–281.

73. Şeker M, Noyan N. Stochastic optimization models for the airport gate assignment problem. *Transportation Research Part E: Logistics and Transportation Review*. 2012; 48: 438–459.

74. Liu F, Dong M, Hou F, et al. Facility Layout Optimization with Stochastic Logistic Flows. 2007: 534–539.

75. Marchand A, Gendreau M, Blais M, et al. Optimized operating rules for short-term hydropower planning in a stochastic environment. *Computational Management Science*. 2019; 16: 501–519.

76. Schepler X, Absi N, Feillet D, et al. The stochastic discrete berth allocation problem. *EURO Journal on Transportation and Logistics*. 2019; 8: 363–396.

77. Chiyoshi FY, Morabito R. A Tabu Search algorithm for solving the extended maximal availability location problem. *International Transactions in Operational Research*. 2011; 18: 663–678.

78. Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. *Science*. 1983; 220: 671–680.

79. Gutjahr WJ. Two metaheuristics for multiobjective stochastic combinatorial optimization. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. 2005; 3777 LNCS: 116–125.

80. Mohammadi M, Forghani K. Solving a stochastic berth allocation problem using a hybrid sequence pair-based simulated annealing algorithm. *Engineering Optimization*. 2019; 51: 1810–1828.

81. Tang Q, Fu Z, Zhang D, et al. Addressing the Bike Repositioning Problem in Bike Sharing System: A Two-Stage Stochastic Programming Model. *Scientific Programming*.; 2020; 2020.

82. Fattahi M, Govindan K. Integrated forward/reverse logistics network design under uncertainty with pricing for collection of used products. *Annals of Operations Research*. 2017; 253: 193–225.

83. Conradie DG, Morison LE, Joubert JW. Scheduling at coal handling facilities using Simulated Annealing. *Mathematical Methods of Operations Research*. 2008; 68: 277–293.

84. Ritt M, Costa AM, Miralles C. The assembly line worker assignment and balancing problem with stochastic worker availability. *International Journal of Production Research*. 2016; 54: 907–922.

85. Mladenović N, Hansen P. Variable neighborhood search. *Computers and Operations Research*. 1997; 24: 1097–1100.

86. Hansen P, Mladenović N, Moreno Pérez JA. Variable neighbourhood search: Methods and applications. *Annals of Operations Research*. 2010; 175: 367–407.

87. Ozdamar L, Aksu DT, Yasa E, et al. Disaster relief routing in limited capacity road networks with heterogeneous flows. *Journal of Industrial and Management Optimization*. 2018; 14: 1367–1380.

88. Glover F, Kochenberger GA. *Handbook of Metaheuristics International Series in Operations Research & Management Science*. 2003.

89. Dijkstra EW. A note on two problems in connexion with graphs. *Numerische Mathematik*. 1959; 1: 269–271.

90. Hoyos MC, Morales RS, Akhavan-Tabatabaei R. OR models with stochastic components in disaster operations management: A literature survey. *Computers and Industrial Engineering*. 2015; 82: 183–197.

91. Mak WK, Morton DP, Wood RK. Monte Carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*. 1999; 24: 47–56.

92. Norkin VI, Pflug GC, Ruszczyński A. A branch and bound method for stochastic

global optimization. *Mathematical Programming, Series B*. 1998; 83: 425–450.

93. Bohnhoff M, Bulut F, Dresen G, et al. An earthquake gap south of Istanbul. *Nature communications*.; 2013;4:1999.

94. De Paula MR, Ravetti MG, Mateus GR, et al. Solving parallel machines scheduling problems with sequence-dependent setup times using variable neighbourhood search. *IMA Journal of Management Mathematics*. 2007; 18: 101–115.

APPENDIX A: T-TEST RESULTS FOR PROPOSED HEURISTICS WITH P-30-3 INSTANCE

Results of the pairwise t-test between the average percentage deviation from BEST* values of metaheuristic variants. An “X” in a cell indicates that the performance of the variants in the row and column of that cell are statistically different at a 95percent confidence level. No rows or columns are available unless there is significant difference between metaheuristic variants.

Table A.1 T-test Results for P-30-3 BRS/BRS* of VNS and BRS*/BRS* of VNS and VNS BRS*/BRS* and VNS and HTSSA BRS*/BRS*

		BRS*								
		VNS_{K_2}	$VNS_{K_{2\text{-}5NI}}$	$VNS_{K_{2\text{-}10NI}}$	$VNS_{K_{2\text{-}20NI}}$	VNS_{K_3}	VNS_{K_4}	$VNS_{K_{6\text{-}5NI}}$	$VNS_{K_{6\text{-}10NI}}$	VNS_{K_7}
BRS	VNS_{K_1}	X								X
	$VNS_{K_{6\text{-}10NI}}$	X	X	X	X	X	X	X	X	X
	$VNS_{K_{6\text{-}S200}}$	X		X						
		BRS*								
BRS*	$VNS_{K_{6\text{-}S200}}$	X								X
		BRS*								
		All VNS Variants								
BRS*	All HTSSA Variants	X								

Table A.2 T-test Results for P-30-3 BRS/BRS*

		BRS*																								
		$HTSSA_{N_1}$	$HTSSA_{N_2}$	$HTSSA_{N_3}$	$HTSSA_{N_4}$	$HTSSA_{N_5}$	$HTSSA_{N_6}$	$HTSSA_{K_1}$	$HTSSA_{K_2}$	$HTSSA_{K_{2\text{-}5NI}}$	$HTSSA_{K_{2\text{-}10NI}}$	$HTSSA_{K_{2\text{-}20NI}}$	$HTSSA_{K_{2\text{-}S200}}$	$HTSSA_{K_{2\text{-}S300}}$	$HTSSA_{K_3}$	$HTSSA_{K_4}$	$HTSSA_{K_5}$	$HTSSA_{K_6}$	$HTSSA_{K_{6\text{-}5NI}}$	$HTSSA_{K_{6\text{-}10NI}}$	$HTSSA_{K_{6\text{-}20NI}}$	$HTSSA_{K_{6\text{-}S200}}$	$HTSSA_{K_{6\text{-}S300}}$	$HTSSA_{K_7}$	$HTSSA_{K_8}$	$HTSSA_{K_9}$
BRS	$HTSSA_{N_1}$				X													X								X
	$HTSSA_{N_5}$																	X					X			
	$HTSSA_{K_{2\text{-}S200}}$	X	X			X	X	X	X	X	X	X	X	X	X		X			X	X	X	X		X	
	$HTSSA_{K_{2\text{-}S300}}$					X						X	X									X	X			
	$HTSSA_{K_3}$												X													
	$HTSSA_{K_4}$												X													
	$HTSSA_{K_5}$																									
	$HTSSA_{K_6}$		X			X					X	X	X							X	X	X	X		X	
	$HTSSA_{K_{6\text{-}20NI}}$																	X								
	$HTSSA_{K_{6\text{-}S200}}$	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X	X	X	X	X	X	X	X	X
	$HTSSA_{K_{6\text{-}S300}}$												X													
	$HTSSA_{K_8}$																	X								

APPENDIX B: T-TEST RESULTS FOR ADDITIONAL EXPERIMENTS

Results of the pairwise t-test between the average percentage improvement from BRS* values of metaheuristic variants. An “X” in a cell indicates that the performance of the variants in the row and column of that cell are statistically different at a 95percent confidence level.

Table B.1 C-10-1 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					
$HTSSA_{N_5}$					

Table B.2 C-10-2 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					
$HTSSA_{N_5}$					

Table B.3 C-10-3 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					
$HTSSA_{N_5}$					

Table B.4 C-20-1 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					X
$HTSSA_{N_5}$					X

Table B.5 C-20-2 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					X
$HTSSA_{N_5}$					X

Table B.6 C-30-1 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}					X
VNS_{K_7}				X	X
$VNS_{K_{2\text{-}10NI}}$				X	X
$HTSSA_{N_2}$					X
$HTSSA_{N_5}$					X

Table B.7 C-30-2 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$				X	X
$HTSSA_{N_2}$					X
$HTSSA_{N_5}$					X

Table B.8 C-30-3 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}				X	X
VNS_{K_7}				X	X
$VNS_{K_{2\text{-}10NI}}$				X	X
$HTSSA_{N_2}$					X
$HTSSA_{N_5}$					

Table B.9 P-20-1 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}	-		X	X	X
VNS_{K_7}		-	-	-	-
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					
$HTSSA_{N_5}$					

Table B.10 P-20-2 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}	-		X	X	X
VNS_{K_7}		-	-	-	-
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					
$HTSSA_{N_5}$					

Table B.11 P-20-3 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}	-		X	X	X
VNS_{K_7}		-	-	-	-
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					
$HTSSA_{N_5}$					

Table B.12 P-30-1 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					
$HTSSA_{N_5}$					

Table B.13 P-30-2 t-test results

	VNS_{K_7}	$VNS_{K_{2\text{-}10NI}}$	$HTSSA_{N_2}$	$HTSSA_{N_5}$	$HTSSA_{K_{6\text{-}20NI}}$
VNS_{K_2}			X	X	X
VNS_{K_7}			X	X	X
$VNS_{K_{2\text{-}10NI}}$			X	X	X
$HTSSA_{N_2}$					X
$HTSSA_{N_5}$					

APPENDIX C: T-TEST RESULTS FOR INCREASED SAMPLE SIZE WITH P-30-3

Tables C.1-C.6 display the computational results of VNS_{K_2} version with 5, 10 and 20 replications. Tables C.1-C.3 include results when the heuristic is initialized with BRS* and Tables C.4-C.6 include results when the heuristic is initialized with BRS.

Table C.1 VNS_{K_2} results with 5 replications initialized with BRS*

	Obj. F.	% Imp Rel to Initial	% Rel to BEST*
1	40,266,331.66	0.07	0.000
2	40,321,000.78	0.07	0.002
3	40,270,287.04	0.07	0.001
4	40,272,464.38	0.07	0.001
5	40,281,412.00	0.07	0.001
Average	**40,282,299.17**	**0.07**	**0.0008**

Table C.2 VNS_{K_2} results with 10 replications initialized with BRS*

	Obj. F.	% Imp Rel to Initial	% Rel to BEST*
1	40,266,634.60	0.07	0.000
2	40,306,825.96	0.07	0.001
3	40,307,198.20	0.07	0.001
4	40,274,830.48	0.07	0.001
5	40,393,155.29	0.07	0.004
6	40,272,506.62	0.07	0.001
7	40,470,187.19	0.07	0.005
8	40,274,941.36	0.07	0.001
9	40,396,275.77	0.07	0.004
10	40,262,541.94	0.07	0.000
Average	**40,322,509.74**	**0.07**	**0.0018**

Table C.3 VNS_{K_2} results with 20 replications initialized with BRS*

	Obj. F.	% Imp Rel to Initial	% Rel to BEST*
1	40,483,289.51	0.07	0.006
2	40,278,818.86	0.07	0.001
3	40,277,281.06	0.07	0.001
4	40,399,860.23	0.07	0.004
5	40,459,539.41	0.07	0.005
6	40,293,868.84	0.07	0.001
7	40,377,156.88	0.07	0.003
8	40,361,071.36	0.07	0.003
9	40,298,399.74	0.07	0.001
10	40,382,401.90	0.07	0.003
11	40,350,603.10	0.07	0.003
12	40,284,514.00	0.07	0.001
13	40,275,693.76	0.07	0.001
14	40,289,203.30	0.07	0.001
15	40,273,468.90	0.07	0.001
16	40,279,680.82	0.07	0.001
17	40,280,951.32	0.07	0.001
18	40,279,163.38	0.07	0.001
19	40,297,879.66	0.07	0.001
20	40,249,186.18	0.07	0.000
Average	**40,323,601.61**	**0.07**	**0.0018**

Table C.4 VNS_{K_2} results with 5 replications initialized with BRS

	Obj. F.	% Imp Rel to Initial	% Rel to BEST*
1	40,255,622.50	0.13	0.000
2	40,260,471.52	0.13	0.000
3	40,795,091.34	0.17	0.013
4	40,270,428.28	0.19	0.001
5	40,426,478.69	0.20	0.004
Average	**40,401,618.47**	**0.16**	**0.0037**

Table C.5 VNS_{K_2} results with 10 replications initialized with BRS

	Obj. F.	% Imp Rel to Initial	% Rel to BEST*
1	40,270,589.98	0.16	0.001
2	40,319,513.80	0.20	0.002
3	40,293,607.48	0.14	0.001
4	40,274,536.78	0.18	0.001
5	40,380,625.84	0.18	0.003
6	40,491,766.55	0.17	0.006
7	40,266,331.66	0.16	0.000
8	40,451,450.45	0.16	0.005
9	40,416,061.25	0.20	0.004
10	40,650,337.50	0.19	0.010
Average	**40,381,482.13**	**0.17**	**0.0033**

Table C.6 VNS_{K_2} results with 20 replications initialized with BRS

	Obj. F.	% Imp Rel to Initial	% Rel to BEST*
1	40,475,625.59	0.18	0.006
2	40,287,432.52	0.17	0.001
3	40,321,870.00	0.19	0.002
4	40,411,963.31	0.14	0.004
5	40,272,723.10	0.18	0.001
6	40,270,428.28	0.16	0.001
7	40,300,538.80	0.15	0.001
8	40,249,186.18	0.21	0.000
9	40,287,432.52	0.21	0.001
10	40,383,216.34	0.18	0.003
11	40,270,428.28	0.15	0.001
12	40,350,603.10	0.17	0.003
13	40,465,016.75	0.22	0.005
14	40,385,751.40	0.23	0.003
15	40,282,406.62	0.18	0.001
16	40,396,110.77	0.18	0.004
17	40,287,432.52	0.14	0.001
18	40,304,383.30	0.19	0.001
19	40,249,186.18	0.14	0.000
20	40,305,265.72	0.15	0.001
Average	**40,327,850.06**	**0.18**	**0.0019**