Summer 2014

The Multi-Depot Minimum Latency Problem with Inter-Depot Routes

Timothy W. Duket
Purdue University

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Timothy W. Duket

Entitled

The Multi-Depot Minimum Latency Problem with Inter-Depot Routes

For the degree of Master of Science in Industrial Engineering

Is approved by the final examining committee:

Seokcheon Lee

Shimon Nof

Jose Tanchoco

To the best of my knowledge and as understood by the student in the *Thesis/Dissertation Agreement. Publication Delay, and Certification/Disclaimer (Graduate School Form 32)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): Seokcheon Lee

Approved by: Abhi Deshmukh 05/28/2014

Head of the Department Graduate Program Date

THE MULTI-DEPOT MINIMUM LATENCY PROBLEM WITH INTER-DEPOT ROUTES

A Thesis

Submitted to the Faculty

of

Purdue University

by

Timothy W. Duket

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science in Industrial Engineering

August 2014

Purdue University

West Lafayette, Indiana

ACKNOWLEDGEMENTS

My sincerest thanks go to everyone who has played a part in helping me successfully complete this endeavor. This thesis truly represents the work and dedication of many individuals.

A special thank you to Dr. Seokcheon Lee for his abundant guidance and support as my academic advisor throughout this process, and for all that I have learned from his instruction and example. Many thanks to Dr. Shimon Nof and Dr. Jose Tanchoco for investing in me and my research by serving on my advisory committee. Thanks also to my fellow students and lab members. Special thanks to MJ for helping me learn optimization software and develop my model.

I wish to thank my lovely wife, Laura, my parents, sisters, and all of my family and friends, who have made this work possible with years of support of all kinds. Thanks also to the Purdue IE community, including students, faculty, and staff, that has supported me through two degrees and given me a wonderful collection of friendships and memories.

Most importantly, my deepest gratitude and praise is to the Lord Jesus Christ, without whom none of this would be possible.

TABLE OF CONTENTS

Page

LIST OF TABLES vi

LIST OF FIGURES viii

LIST OF SYMBOLS ix

ABSTRACT x

CHAPTER 1: INTRODUCTION 1

1.1 Routing Problems 1

1.2 Motivation to Natural Disasters 5

1.3 The Role of Fairness 6

1.4 Research Objectives 7

1.5 Research Overview 9

CHAPTER 2: LITERATURE SURVEY 12

2.1 The Minimum Latency Problem 12

2.2 Multi-Depot and Multi-Vehicle Routing Problems 12

2.3 Disaster Relief Operations 14

2.4 Job Scheduling 16

CHAPTER 3: MODEL FORMULATION 19

3.1 Performance Measures 19

3.2 System Qualities 19

3.3 Assumptions 20

Page

3.4 Mathematical Model 20

CHAPTER 4: HEURISTIC APPROACHES 25

4.1 Nearest Neighbor Benchmark 25

4.2 Initial Heuristics 26

4.2.1 Auction Policy 26

4.2.2 Insertion 30

4.3 Second-Phase Adjustment 35

CHAPTER 5: PRELIMINARY EVALUATION 38

5.1 Test Instances 38

5.2 Validation on Small Instance Subsets 39

5.2.1 Minimize Latency 40

5.2.2 Minimize Weighted Latency 42

5.3 Validation on Single Vehicle Instances 44

CHAPTER 6: PERFORMANCE EVALUATION 52

6.1 Previous VRP Solutions 52

6.2 Minimize Latency 53

6.2.1 Evaluation of Latencies 54

6.2.2 Evaluation of Fairness 59

6.3 Minimize Weighted Latency 62

6.3.1 Evaluation of Latencies 62

6.3.2 Evaluation of Fairness 66

6.4 Computation Times 68

Page

CHAPTER 7: CONCLUSIONS AND FUTURE WORK 70

7.1 Summary of Results 70

7.2 Contributions 71

7.3 Future Research Directions 72

7.3.1 Multiple Objectives 72

7.3.2 Corporate Involvement 74

7.3.3 Improved Heuristics 74

7.3.4 Collaborative Depots 75

7.3.5 Job Scheduling 76

LIST OF REFERENCES 77

APPENDICES

Appendix A: Mathematical Model of the MDMLPI 82

Appendix B: Pseudocode for Auction Policy 84

Appendix C: Pseudocode for Insertion Technique 85

Appendix D: Pseudocode for Depot Assignment 86

Appendix E: Pseudocode for Second-Phase ADJUSTMENT 87

Appendix F: Small Test Instances 88

LIST OF TABLES

Table Page

2.1: Characteristics of the above MLP literature....13

5.1: Characteristics of the 10 data sets used for testing....38

5.2: Optimal, benchmark, and heuristic results for small problem instances....40

5.3: Ratio of each heuristic and NN benchmark over optimality for small instances....41

5.4: Optimal, heuristic, and benchmark results on small weighted instances....42

5.5: Ratio of new heuristics over optimality for small weighted instances....43

5.6: Performance of new heuristics with rearranging on TSPLIB instances....45

5.7: Ratios of each heuristic over best known solutions for TSPLIB instances....46

5.8: Results of insertion technique using every other ordered node....49

6.1: Results and calculated latency values from previous VRP solutions....52

6.2: Latencies of benchmarking solutions....54

6.3: Latency performance of initial heuristics....55

6.4: Latency performance of the auction policy with the second-phase adjustment applied 57

6.5: Latency performance of the insertion heuristic with the second-phase adjustment applied....58

6.6: Standard deviations of benchmarks and heuristics with second-phase adjustment....59

Table Page

6.7: Percentage improvements in standard deviation of heuristics from the benchmark solutions 60

6.8: Weighted latencies of benchmarking solutions 62

6.9: Weighted latency performance of initial heuristics 63

6.10: Weighted latency performance of the auction policy with second-phase adjustment 65

6.11: Weighted latency improvements of the insertion heuristic with second-phase adjustment 66

6.12: Weighted standard deviations of benchmarks and heuristics after second-phase adjustment 67

6.13: Percentage improvements of best weighted standard deviations over benchmarks .68

6.14: Average computation time (s) for each heuristic 69

LIST OF FIGURES

Figure Page

1.1: Sample routing solution 1 3

1.2: Sample routing solution 2 4

1.3: CDFs for two hypothetical routing solutions 7

4.1: Example network for auction heuristic 28

4.2: Bidding parameters for first bidding customer 29

4.3: Bidding parameters for second bidding customer 29

4.4: Partial routing example for insertion heuristic 31

4.5: Possible position for customer 8 in partial solution 31

4.6: Alternative position for customer 8 in partial solution 32

4.7: Illustration of depot assignment for insertion heuristic 34

4.8: Example complete schedule for second-phase adjustment 36

4.9: Possible repositioning using second-phase adjustment 36

5.1: Routing solution for kroC100 using insertion 47

5.2: Routing solution for kroC100 using insertion with every other ordered node 50

6.1: Ordered latencies for instance *a* solutions from auction and NN 56

6.2: Comparison of CDF's for solutions to instance *a* from auction and NN 61

6.3: Routing solution for instance *a* using auction 64

6.4: Routing solution for instance *a* using insertion 64

LIST OF SYMBOLS

Indices	**Significance**
i,j	Customer indices
k	Vehicle index
Sets	
V	Set of all entities in the system
V_C	Set of all customer nodes
V_D	Set containing the main depot node
V_I	Set of all intermediate depot nodes
R	Set of vehicles
Parameters	
c_{ij}	Travel time between nodes i and j
q_i	Demand of customer i
Q	Homogenous vehicle capacity
Variables	
x_{ijk}	1 if the arc from node i to node j is traversed by vehicle k, 0 else
π_{ik}	Arrival time (latency) of vehicle k at node i
μ_{ik}	Cumulative demand required from vehicle k up to and including node i

ABSTRACT

Duket, Timothy W., M.S.I.E., Purdue University, August 2014. The Multi-Depot Minimum Latency Problem with Inter-Depot Routes, Major Professor: Dr. Seokcheon Lee.

The Minimum Latency Problem (MLP) is a class of routing problems that seeks to minimize the wait times (latencies) of a set of customers in a system. Similar to its counterparts in the Traveling Salesman Problem (TSP) and Vehicle Routing Problem (VRP), the MLP is NP-hard. Unlike these other problem classes, however, the MLP is customer-oriented and thus has impactful potential for better serving customers in settings where they are the highest priority. While the VRP is very widely researched and applied to many industry settings to reduce travel times and costs for service-providers, the MLP is a more recent problem and does not have nearly the body of literature supporting it as found in the VRP. However, it is gaining significant attention recently because of its application to such areas as disaster relief logistics, which are a growing problem area in a global context and have potential for meaningful improvements that translate into reduced suffering and saved lives. An effective combination of MLP's and route minimizing objectives can help relief agencies provide aid efficiently and within a manageable cost.

To further the body of literature on the MLP and its applications to such settings, a new variant is introduced here called the Multi-Depot Minimum Latency Problem with Inter-Depot Routes (MDMLPI). This problem seeks to minimize the cumulative arrival times at all customers in a system being serviced by multiple vehicles and depots. Vehicles depart from one central depot and have the option of refilling their supply at a number of intermediate depots. While the equivalent problem has been studied using a VRP objective function, this is a new variant of the MLP. As such, a mathematical model is introduced along with several heuristics to provide the first solution approaches to solving it. Two objectives are considered in this work: minimizing latency, or arrival times at each customer, and minimizing weighted latency, which is the product of customer need and arrival time at that customer. The case of weighted latency carries additional significance as it may correspond to a larger number of customers at one location, thus adding emphasis to the speed with which they are serviced. Additionally, a discussion on fairness and application to disaster relief settings is maintained throughout. To reflect this, standard deviation among latencies is also evaluated as a measure of fairness in each of the solution approaches.

Two heuristic approaches, as well as a second-phase adjustment to be applied to each, are introduced. The first is based on an auction policy in which customers bid to be the next stop on a vehicle's tour. The second uses a procedure, referred to as an insertion technique, in which customers are inserted one-by-one into a partial routing solution such that each addition minimizes the (weighted) latency impact of that single customer. The second-phase modification takes the initial solutions achieved in the first two heuristics and considers the (weighted) latency impact of repositioning nodes one at a time. This is implemented to remove potential inefficient routing placements from the original solutions that can have compounding effects for all ensuing stops on the tour. Each of these is implemented on ten test instances. A nearest neighbor (greedy) policy and previous solutions to these instances with a VRP objective function are used as benchmarks.

Both heuristics perform well in comparison to these benchmarks. Neither heuristic appears to perform clearly better than the other, although the auction policy achieves slightly better averages for the performance measures. When applying the second-phase adjustment, improvements are achieved and lead to even greater reductions in latency and standard deviation for both objectives. The value of these latency reductions is thoroughly demonstrated and a call for further research regarding customer-oriented objectives and evaluation of fairness in routing solutions is discussed.

Finally, upon conclusion of the results presented in this work, several promising areas for future work and existing gaps in the literature are highlighted. As the body of literature surrounding the MLP is small yet growing, these areas constitute strong directions with important relevance to Operations Research, Humanitarian Logistics, Production Systems, and more.

CHAPTER 1: INTRODUCTION

1.1 Routing Problems

Routing problems form a substantial area of research with myriad applications to industry as well as humanitarian settings. The fundamental goal of routing problems in general is to optimize the routes by which a set of nodes, or customers, are visited and serviced by one or more vehicles. A wide variety of objective functions can be implemented to meet the goals of minimizing travel time, cost, customer wait time, and more. In addition to the abundance of objective functions, many variants of these problems can also be studied to fit different application settings. These variants modify the characteristics of vehicles (single or multiple, homogenous or unique, capacitated or not), customer demand (single or multiple commodity, split delivery or single visit, deterministic or stochastic arrivals), supply depots (single or multiple, capacitated or not, independent or connected), and more. New variants are introduced to accurately model the application of results to different settings.

Perhaps the most basic routing problem is the Traveling Salesman Problem (TSP) which uses a single vehicle to service a set of customers such that route length for the service provider is minimized. When multiple vehicles are used, this becomes the Vehicle Routing Problem (VRP), which is perhaps the most thoroughly studied and widely applied routing problem. Both the TSP and VRP are server-oriented problems that have valuable applications for cost minimization in industry settings such as vehicle fleet logistics. Variants of these problems in the literature are many, including the Time-Dependent TSP (TDTSP), the TSP with Time Windows (TSPTW), the Clustered TSP (CTSP), and the Multi-Depot VRP (MDVRP), among others.

The equivalent problem to the TSP with a customer-oriented objective function is called the Minimum Latency Problem (MLP). The MLP seeks to minimize the cumulative wait time, or latencies, of customers across a system. This objective differs from the TSP and VRP in that the needs of the customer are given priority. On the other hand, the costs that will be incurred by the service provider are not considered in the optimization, which highlights the need for a combination of these two objectives as an extension of research in this field. The other key difference is that the MLP does not consider the cost of the vehicle's return to its starting position, since this does not impact customer wait times. The TSP, on the other hand, minimizes the full routing loop through all customers and back to the starting depot.

This customer-focused objective makes the MLP an important problem, yet under-researched, for settings when customer needs are the highest priority, either within or outside of industry. A crucially important example of such an application that has gained recent attention is in disaster relief efforts, in which quick response and reduced customer wait times are clearly the primary goal.

MLP objectives generate more complex problems than shortest-path variants and have been shown to be NP-hard (Blum et al., 1994; Sahni & Gonzalez, 1976; Sitters, 2002). Research on this class of problem is also found under a variety of names, including the Traveling Repairman Problem (Afrati, Cosmadakis, Papadimitriou, Papageorgiou, & Papakostantinou, 1986), the Traveling Deliveryman Problem (Méndez-Díaz, Zabala, & Lucena, 2008), the TSP with Cumulative Costs (Bianco, Mingozzi, & Ricciardelii, 1993), and the Cumulative Capacitated Vehicle Routing Problem (Ngueveu, Prins, & Wolfler Calvo, 2010). To help organize the literature and highlight promising areas for future research, Moshref-Javadi & Lee (2013) present a taxonomy to the MLP and highlight gaps as well as promising research directions.

A simple illustration of the minimum latency objective is presented to demonstrate the difference from route minimizing objectives as well as its importance in customer-oriented networks. Consider the possible routing solution depicted in Figure 1.1, in which a vehicle departs from the (square) depot and visits a set of five customers along the arcs with associated costs as shown.

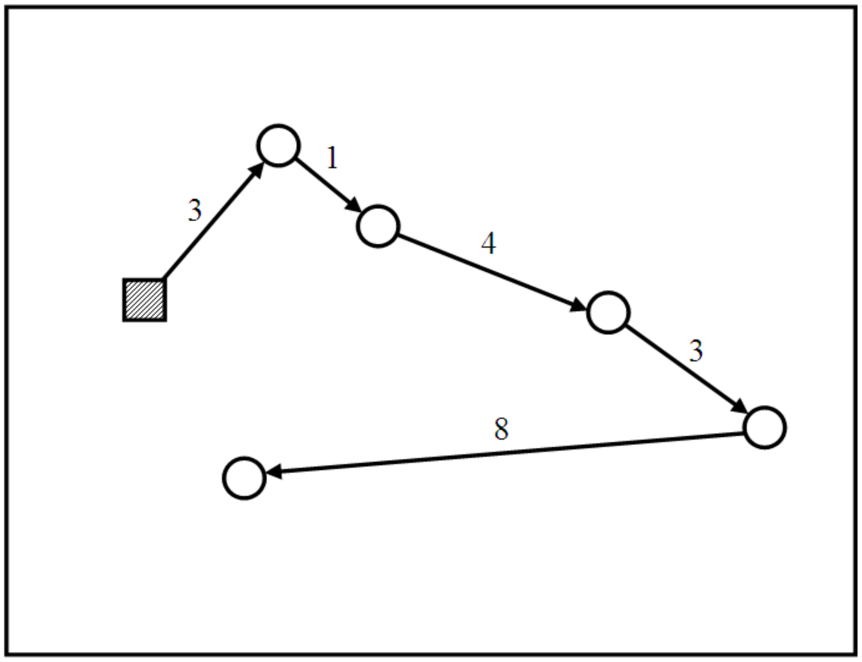


Figure 1.1: Sample routing solution 1

This graphic depicts one possible routing solution for this network, not the only possibility or the optimal solution. If the travel times associated with each arc are added up, it is clear that the time required for the service provider, through to the final customer, is 3 + 1 + 4 + 3 + 8 = 19. The latency associated with this route is the sum of the vehicle's arrivals at each customer, which are calculated as follows:

First customer: 3

Second customer: 3 + 1 = 4

Third customer: 3 + 1 + 4 = 8

Fourth customer: 3 + 1 + 4 + 3 = 11

Fifth customer: 3 + 1 + 4 + 3 + 8 = 19

Thus, the total route latency for this solution is 3 + 4 + 8 + 11 + 19 = 45.

Alternatively, consider another routing possibility that serves the customers in the opposite order. This is shown in Figure 1.2.

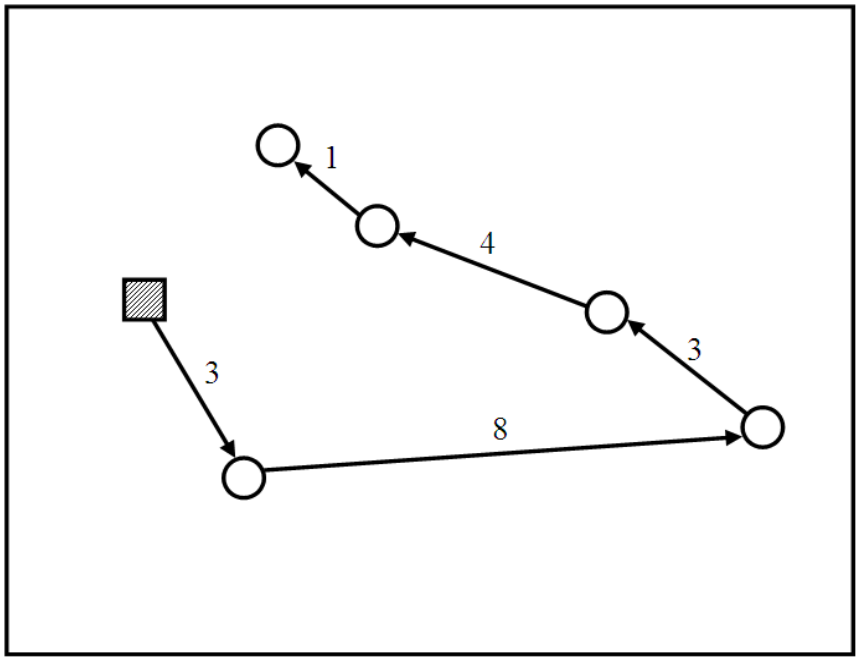


Figure 1.2: Sample routing solution 2

The time requirement of this route for the server is 3 + 8 + 3 + 4 + 1 = 19, which is the same as the previous solution. From a TSP perspective, therefore, the two alternatives are equally good. However, the latency calculations for customers are now:

First customer: 3

Second customer: 3 + 8 = 11

Third customer: 3 + 8 + 3 = 14

Fourth customer: 3 + 8 + 3 + 4 = 18

Fifth customer: 3 + 8 + 3 + 4 + 1 = 19

The total latency in this routing solution is then 3 + 11 + 14 + 18 + 19 = 65, which is significantly increased from the alternative which had total latency of 45. The most prominent cause for this dramatic difference is the placement of the arc associated with travel time 8. Although this arc is traversed in both alternatives, the second solution uses it very early on, causing all remaining customers to absorb this time into their service times. This demonstrates the impactful compounding nature of the MLP. Any inefficient routing assignments, especially if they occur early in the solution, can have dramatic effects on total latency. This stresses the importance of careful routing decisions at every point of the tour. It also highlights the benefit that customer-focused objectives can bring to avoid tremendous customer delays resulting from this compounding effect, especially when delays can mean the difference between life and death in emergency settings.

1.2 Motivation to Natural Disasters

As mentioned, one popular example of applications for the class of MLPs, especially in recent years, is in humanitarian response to natural disasters. Disasters can be devastating to communities and governments, and decision-making in a post-disaster setting needs to be quick and efficient. An effective combination of pre-planning and on-site organization is important for relief efforts to succeed (Altay & Green, 2006). There are numerous areas of disaster relief management that can be studied to improve goals such as communication, supply distribution, and medical response. One area of crucial importance, and thus having potential for impactful improvements, is transportation coordination (Dolinskaya, Shi, & Smilowitz, 2011). This component constitutes a large portion of the cost in managing the supply chain and is vital in meeting customer needs in a timely fashion (Balcik, Beamon, Krejci, Muramatsu, & Ramirez, 2010). Problems in this arena often deal with the additional challenge that certain disasters may be accompanied by limited road networks due to traffic backup and damage that make certain routes impassible, eliciting an even stronger need for quick, adaptive routing solutions.

These post-disaster challenges make the problem of serving all customers as quickly as possible more difficult, but also more important. Timely delivery of medical supplies, efficient clearing of debris from roadways, and emergency response to dangerous environments are all linked to effective, customer-oriented routing for emergency vehicles. With a clear demand for a focus on customer needs, minimum latency objectives have great potential for reducing total customer wait time across the system.

This can translate into quicker response and supply delivery, and consequently, save lives.

1.3 The Role of Fairness

In customer-oriented networks, fairness also becomes a significant goal for scheduling solutions. For example, routing supplies to communities in need as efficiently as possible becomes a high priority in addition to keeping the cost sustained by the service provider to a reasonable or minimum value. This highlights the importance of customer-oriented objective functions. Another similar objective to the MLP is to minimize the maximum wait time among all nodes. Campbell, Vandenbussche, & Hermann (2008) study objectives to minimize average latency (which is the same as total latency divided by the number of customers) as well as the maximum latency at any customer. However, minimizing total latency is preferred in this thesis over minimizing the maximum latency because it takes into account the wait times at all customers.

While the minimum latency objective inherently contains a focus on fair distribution of supplies to customers as discussed, it is also important to evaluate fairness among customers on an individual level. Here, a solution is considered to be fairer if customers are serviced with more consistent, similar wait times. That is, the variance among all latencies is smaller. This is in contrast to a high-variance solution in which some customers are serviced quickly at the expense of others in the system having to absorb very long wait times. The objective of minimizing maximum latency partially addresses this focus, but again only considers the individual needs of the last-served customer. Measuring the standard deviation among customer wait times, on the other hand, provides a strong statistical measure of the spread of wait times. Thus, standard deviation among responses will also be recorded and compared throughout this thesis. A simple illustration of the cumulative distribution functions (CDF) of two solutions shows how these measures translate into quick and fair response in customer-focused applications.

In Figure 1.3, these CDFs are created from two randomly generated datasets with mean and standard deviation close to N(100,25) and N(150,50). If these functions are viewed as emergency response times for supply delivery, for example, it is clear that a larger portion of customers can be serviced more quickly if the first distribution of wait times, ‘Distribution 1,’ is accomplished. In certain emergencies, a natural time constraint may restrict relief operations to a narrow window. If in this hypothetical situation a time constraint of 125 units was in effect, for example, the population serviced by that time by the ‘Distribution 2’ solution appears to be less than half of the whole network, whereas the low mean and standard deviation of ‘Distribution 1’ would allow it to serve the clear majority by that time.

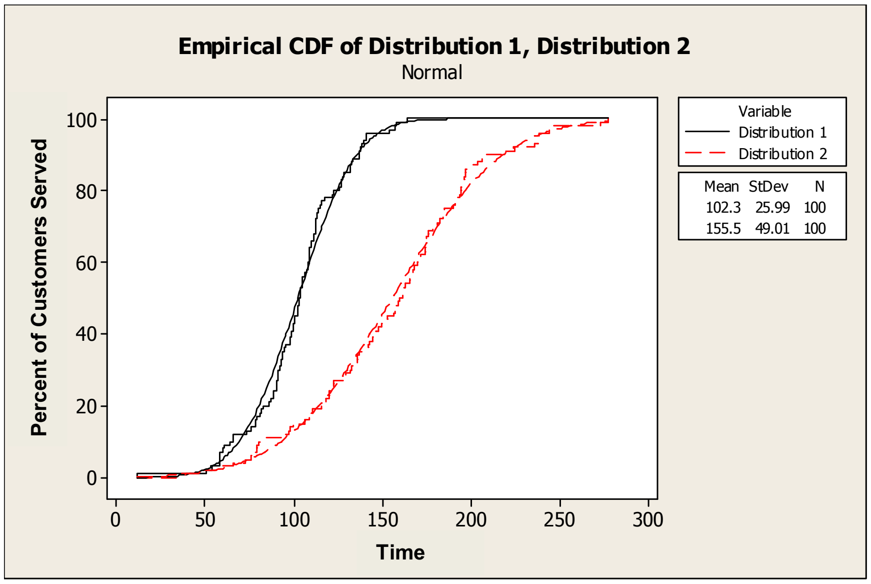


Figure 1.3: CDFs for two hypothetical routing solutions

1.4 Research Objectives

The overarching goal of this thesis is to advance the developments pertaining to the application of routing problems to improve disaster relief operations. This has become a popular topic recently and, while much progress has been made, many interesting problems of value remain untouched. In response to one of these untouched gaps in the current body of literature, an important variant is selected for analysis. This is a version of the MLP that utilizes multiple vehicles and multiple supply depots that can be accessed by any vehicle. The motivation for selecting such a variant stems directly from applicability to disaster relief scenarios. A prominent setup among relief agencies and other entities seeking to provide service involves the prepositioning of supply distribution centers paired with a fleet of emergency and delivery vehicles for transportation to the customers in need.

All decisions in an emergency response setting should seek to provide fair service to those in need. Thus, the most appropriate objective function is to minimize the service times at all customers. Alternative objectives for similar problems often studied in industry logistics and job scheduling settings include minimizing total route length, minimizing maximum service time, and minimizing the number or tardiness of late jobs when time constraints exist. While approaches to many of these objectives are more fully developed, they do not fill the direct need required in emergency networks. Minimizing route length, for example, seeks to ease the costs incurred by the service provider. Minimizing maximum service time, on the other hand, is customer-oriented but does not take into account the service time of every customer but only the one with the maximum wait time. Minimum latency objectives are the ideal alternative because they factor in the service time of every customer. In this work, the basic minimum latency objective is used as well as the objective of minimizing total weighted latency. Weighted latency at a customer is calculated as the product of the wait time and the demand at that node. When applied to post-disaster settings, this weight can correspond to the number of victims at a demand location, so the weighted latency objective gives these high demand nodes some priority in being served as early as possible.

Using these objective functions, the model is developed to closely resemble the organizational setup of a relief provider. Multiple vehicles are used with limited capacity such that they may need to refill their supply at depots in order to fulfill all demand. Capacity constraints are not imposed on supply depots here, as they are entities with much larger storage ability, but imposing such constraints is a potential next step for furthering this research. Upon the completion of routes, vehicles are assumed to return to the main depot from which they started, but this return time is not factored into the objective as it does not directly impact customer latency.

The problem with these qualities is titled the Multi-Depot Minimum Latency Problem with Inter-Depot Routes (MDMLPI). The following research objectives are identified to explore and evaluate this problem:

1. Formulate and validate a mathematical model that is representative of the MDMLPI
2. Develop heuristic algorithms to obtain strong solutions to the MDMLPI in reasonable computation time for the minimum latency and minimum weighted latency objectives
3. Evaluate the power and benefit of these heuristics by comparing to a benchmarking solution and to when a minimum route length objective is used on the same instances
4. Evaluate the performance of these heuristics incorporating the notion of fairness for such networks

1.5 Research Overview

The remainder of the document will be organized as follows:

- **Chapter 2**: Literature Survey
  Relevant literature to the research on this problem is investigated. This includes works pertaining to the variants of the MLP, similar variants of routing problems using multiple vehicles, research directly related to disaster relief logistics, and applications of similar network designs in job scheduling problems. In each case, previously proposed algorithms are highlighted and important learning from the past literature is gathered.

- **Chapter 3**: Model Formulation
  The development of the model for the MDMLPI is documented, including performance measures, system qualities, and assumptions. This chapter also details how each quality or limitation in the network was translated into a

mathematical constraint. The integer programming model is developed in this chapter and is presented concisely in full in Appendix A.

- **Chapter 4**: Heuristic Approaches
  Several heuristic approaches to the MDMLPI, as well as a second-phase reassignment procedure, are introduced. The first algorithm utilizes an auction policy such that vehicles accept bids from customers who seek to be the next stop on a given route. Secondly, an alternative heuristic is presented in the form of a simple insertion technique. This approach assigns nodes sequentially based on proximity to the main depot and positions depot stops once all nodes have been assigned to the route. Lastly, a second-phase adjustment heuristic is presented to improve the results of the first two algorithms, which may be considered as initial routing solutions. This policy evaluates the potential movement of each node to alternative route positions to see if improvements in latency (or weighted latency) can be achieved.

- **Chapter 5**: Preliminary Evaluation
  Given the NP-hardness of a problem as complex as the MDMLPI, the test instances cannot be solved to optimality. Therefore, smaller problems are used to validate that the formulated model is accurate. This is done using two small sets of instances. Firstly, subsets of the larger instances are extracted and solved to optimality using CPlex solver. The heuristics are also run on the subset instances. This assists validation of the defined model and heuristics to ensure that all components are functioning and intuitive routing solutions are being obtained. Secondly, the heuristics are run for some single-vehicle MLP instances to further validate them on problems of larger size. The instances come from the Traveling Salesman Problem Library (TSPLIB) and have been used for the MLP in previous literature.

- **Chapter 6**: Performance Evaluation

  Results are presented for testing of the benchmarking policy and proposed heuristics. Latency and weighted latency values are displayed along with the standard deviations achieved under each objective. The results demonstrate that the proposed heuristics offer significant improvements over the benchmarking policies in terms of both latency and standard deviation, thus carrying valuable implications for fair routing in emergency settings. They offer even greater reductions when compared to the latencies and standard deviations that arise from using a minimum route length objective, exhibiting the key benefits and motivation for using a minimum latency objective in such environments. Lastly, the average computation times of each are presented to show their effectiveness in reasonable time.

- **Chapter 7**: Conclusions and Future Work

  Finally, a summary of what was accomplished and the key lessons from this thesis work are discussed. As the class of MLPs and the field of disaster relief logistics are currently developing, there are many interesting and impactful areas for future research directions. Several of these pertaining to similar work as that presented here are discussed.

CHAPTER 2: LITERATURE SURVEY

2.1 The Minimum Latency Problem

The problem of wait time minimization among customers was first studied by Afrati, Cosmadakis, Papadimfditriou, Papageorgiou, & Papakostantinou (1986) as the Traveling Repairman Problem (TRP). The authors introduce the objective of minimizing the average waiting time of a number of machines in need of repairs by a single repairman. A polynomial time algorithm for the instance in which machines are in a line, called the line-TRP, is proposed. Recently, problems with MLP objectives have attracted attention among researchers as well as for industry applications. Exact algorithms, approximation schemes, and heuristics have been developed to solve or approximate the complex MLP variants in reasonable time (for a few examples, see Abeledo, Fukasawa, Pessoa, & Uchoa, 2010; Goemans & Kleinberg, 1998; Salehipour, Sörensen, Goos, & Bräysy, 2011; Silva, Subramanian, Vidal, & Ochi, 2012; Wu, Huang, & Zhan, 2004; Wu, 2000). For a more complete review, see Moshref-Javadi & Lee, (2013).

2.2 Multi-Depot and Multi-Vehicle Routing Problems

Focusing the scope of search to multiple vehicle instances of the MLP, however, greatly reduces the body of available literature. Those works that deal with such variants, to the extent of our knowledge, fall under the name of the Cumulative Capacitated VRP, or CCVRP. This problem is equivalent to the system of a VRP with the only difference being the altered objective function. Ngueveu, Prins, & Wolfler Calvo (2010) present this problem using homogenous vehicles with limited capacity. The authors introduce methods to formulate the first upper and lower bounds to this problem. A memetic algorithm is used to find the upper bounds. Ribeiro & Laporte (2012) study the same problem and provide improved solutions for the same instances previously used. The authors do this using adaptive large neighborhood search. Ke & Feng (2013) offer further improvements with a two-phase heuristic. This approach employs two perturbation operators on generated solutions at every iteration.

The above literature on the MLP is organized by characteristic in Table 2.1 below, which is adapted from a larger version found in Moshref-Javadi & Lee (2013). The following qualities are tracked for each: whether demand quantity is considered on customer nodes or not, whether demand is deterministic or stochastic, the number of depots, the number of vehicles, whether vehicles have limited capacity or not, and if vehicles are similar in terms of speed, capacity, etc. While other variants of the MLP exist with numerous other characteristics, these works represent the body of MLP literature most closely related to that proposed in this thesis. For example, although all of these referenced papers use deterministic demand, there exist other works not discussed here that use stochastic demand arrivals.

Table 2.1: Characteristics of the above MLP literature

	Demand Quantity	**Demand Type**	**# Dep.**	**# Veh.**	**Veh. Capacity**	**Similar Veh.**
Afrati et al. (1986)		Det.	1	1		✓
Abeledo et al. (2013)		Det.	1	1		✓
Goemans & Kleinberg (1998)		Det.	1	1		✓
Salehipour et al. (2011)		Det.	1	1		✓
Silva et al. (2012)		Det.	1	1		✓
Wu et al. (2004)		Det.	1	1		✓
Wu (2000)		Det.	1	1		✓
Ngueveu et al. (2010)	✓	Det.	1	>1	✓	✓
Ribeiro & Laporte (2012)	✓	Det.	1	>1	✓	✓
Ke & Feng (2013)	✓	Det.	1	>1	✓	✓
MDMLPI	✓	Det.	>1	>1	✓	✓

These works can be grouped into two broader categories designated by the lines in the table above: those that use a single vehicle without capacity constraints or customer demand, and those that consider multiple vehicles with these constraints. It is clear from the table, however, that no previous works use more than one depot. This is also true when considering a larger range of literature; no previous works have been found that use multiple depots in an MLP variant.

Therefore, this work seeks to expand upon these variants in the direction of a multiple vehicle, multiple depot MLP in which vehicles can refill their supply at any depot in the system. The quality of shared supply depots is prevalent in customer-oriented networks like disaster relief operations. This problem is titled the Multi-Depot Minimum Latency Problem with Inter-Depot Routes (MDMLPI). The characteristics of this problem are included in the bottom row of Table 2.1, which clearly identifies the research gap filled by this thesis work.

An equivalent variant of this new problem has been applied to the VRP by Crevier, Cordeau, & Laporte (2007). These authors introduce the MDVRPI as a new variant that contains intermediate depots that can serve vehicles traveling from and ultimately returning to one main depot. Test data sets are developed with randomly generated nodes and depots, and then the main depot is determined as the center point among all intermediate depots. The proposed solution is tested on randomly generated instances with unlimited supply depots serving customers using homogenously capacitated vehicles. The authors develop a two-phase heuristic using a Tabu Search and test it on ten problem instances. As the MDVRPI had not been previously studied, these results are assumed to be the present best-known solutions for their datasets and will be used as a reference for discussion of results in this work.

2.3 Disaster Relief Operations

The relevant literature from disaster relief applications of these problems is also briefly reviewed. Works pertaining to this application area are often found under key phrases such as Humanitarian Logistics / Humanitarian Operations, Disaster Operations Management (DOM), and emergency relief logistics. Different literature in this field deals with varying levels of application to these real-world settings, ranging from theoretical development of new problem variants to case studies of actual implementations of relief effort optimization. It is important to maintain a focus on applicability throughout this work. Thus, while the problem and heuristics presented here are not being applied to a present disaster, it is appropriate to approach them with a careful understanding of how such applications have previously been performed.

Literature applied to this field comes in several forms, including development of new problems and solution approaches, analyses of the field of relief efforts from different perspectives, case studies from groups directly involved in such efforts, and more. Altay & Green (2006) offer a valuable review of applications of Operations Research and the Management Sciences (OR/MS) in this field. The authors note that researchers in this area have not yet produced a thorough body of work to encompass the multitude of impactful questions in this area. The existing literature is reviewed such that unresolved questions and new research directions can be highlighted. Many valuable questions to guide researchers are also discussed, including the ethical role of decision making in such settings and the ideal organizational structures to enhance communication and coordination.

A follow-up to this work is presented by Galindo & Batta (2013), who offer updates of how the field has developed more recently and continue the discussion on such issues as the role of coordination among multiple relief agencies. Other similar reviews can be found in Van Wassenhove (2005), which focuses on the supply chain perspective, as well as Kovács & Spens (2011), Balcik et al. (2010), and McEntire (1999), which include significant discussions on the main issues, gaps, and opportunities in relief.

Campbell et al. (2008) provide a thorough discussion specific to routing efforts in disaster relief by exploring different objectives with bounds on performance between server-oriented and customer-oriented goals. The customer-oriented objectives studied are average latency and maximum latency. The discussion between objectives and regarding fairness in relief routing strongly advance understanding of these topics in the literature.

Another common study in such settings that has direct application value comes in the form of case studies and analyses from relief worker perspectives. Day, Melnyk, Larson, Davis, & Whybark (2012) call for the inclusion of what they refer to as Humanitarian and Disaster Relief Supply Chains (HDRSCs) into the field of supply chain management (SCM). As a concluding component of this argument, they present some of the key needs of this field from the perspective of HDRSC practitioners. Similarly, Howden (2009) places a similar call on the importance of humanitarian operations in disaster relief supply chains. This work is written from the perspective and experience of a humanitarian consultant. Other case studies specific to certain disasters as opposed to types of organizations include analyses of response in Haiti following the 2010 earthquake (Jobe, 2011; Martinez & Wassenhove, 2010) as well as of the response to the 2004 tsunami in Asia (Tabbara, 2008), among others. Costa, Campos, & Bandeira (2012) analyze response efforts to four different disasters affecting Japan in 2011, Brazil in 2011, Pakistan in 2005, and 14 countries bordering the Indian Ocean in 2004.

Lastly, another interesting area involves corporate involvement in response to local disaster relief. Kuo & Means (2012) discuss this involvement and the value of local businesses to serve their local communities in need, as well as the larger players from international relief organizations to larger corporations (e.g. Walmart's presence in recovery from Hurricane Katrina in 2005). Another interesting example of this is presented in Ergun, Heier Stamm, Keskinocak, & Swann (2010) with an analysis of Waffle House Restaurants and the corporations involvements in a number of hurricane relief efforts.

2.4 Job Scheduling

Another application area for which routing problems can provide some insight is job scheduling. Many industry applications require the scheduling of jobs with sequence-dependent setup times such that the setup time for a job depends on the job itself as well as the job preceding it. Setup times are thus represented in matrix form where every entry designates the setup required to transition from job *i* to job *j*. In standard job scheduling notation, sequence-dependent setup times are denoted by s_{ij} in the β field of an $\alpha|\beta|\gamma$ scheduling problem.

Certain variants of these problems can be equated to classes of routing problems such that the setup time between two jobs equates to the travel time between two customers. If a routing problem calls for time windows to service nodes, this is equivalent to the processing time of a job. Otherwise, processing times can be assumed to be zero. For example, the TSP, which minimizes the route time for a single server among n nodes that each need to be visited once, is equivalent to a $1|s_{ij}|c_{max}$ problem. Expanded to the Vehicle Routing Problem (VRP) which has the same objective and m identical servers, this becomes a $P_m|s_{ij}|c_{max}$ problem. Similarly, the Minimum Latency Problem (MLP) seeks to minimize total arrival time at customers and can be equated to a $1|s_{ij}|\sum c_j$ problem, or $P_m|s_{ij}|\sum c_j$ in the multi-server variant.

As these parallel problems may be able to provide insight for each other, it is important to be aware of the research conducted on each of them. Below is a brief look at some similar problems in the field of job scheduling and methods for how they have been studied. T_j used in the notation below corresponds to the tardiness of a job being completed when a deadline is present. This would parallel late arrival of a vehicle to a customer that has a time requirement. Additionally, problems that minimize w_jT_j utilize weighted tardiness of job completion based on the job's importance.

An exact branch-and-bound algorithm for both the $1|s_{ij}|\sum C_j$ and $1|s_{ij}|\sum T_j$ problems, where T_j is the tardiness of a job with a scheduled completion time, is presented by Bianco et al. (1993). Many similar works with the objective of minimizing total tardiness were also found, including a demonstration of lower and upper bounds to the problem (Luo et al., 2005), an ant colony optimization (Liao & Juan, 2007), and a hybrid genetic algorithm (Sioud, Gravel, & Gagné, 2012). Wang & Tang (2010) present a hybrid metaheuristic for the same environment minimizing weighted job completion times. The general single-machine environment with past-sequence-dependent setup times is studied with a variety of objective functions by Koulamas & Kyparisis (2008), who present a sorting procedure to solve instances in O(n log n) time.

Many variants of sequence-dependent processing and assignment problems on parallel machines are well-studied, but approaches specific to the $P_m|s_{ij}|\sum c_j$, which is the MLP with multiple vehicles, are not abundant. Guinet (1993) studies this exact problem and presents an approach adapted from the routing literature. A previous model is adapted for assignment of jobs to machines and fictitious machines are added to allow machine redeployment. The proposed heuristic is tested over a wide variety of instances and performs well.

A similar problem is studied in Lee & Pinedo (1997), where a three-phase heuristic is presented for the problem of $P_m|s_{ij}|\sum w_j T_j$. The heuristic involves a preprocessing phase, a dispatching rule, and simulated annealing and is also tested on a variety of instances. Similarly, Ying (2012) presents an effective iterated greedy heuristic for the objective of minimizing setup times in parallel machines, which is similar to the VRP objective.

Further, a literature review of scheduling problems with sequence-dependent setups was conducted by Zhu & Wilhelm (2006) while Tan, Narasimhan, Rubin, & Ragatz (2000) compare four algorithms in the minimum total tardiness class of problems. These approaches are branch-and-bound, genetic algorithm, simulated annealing, and random-start pair wise interchange. They conclude that both simulated annealing and pair wise interchange are feasible approaches for large datasets, and branch-and-bound is optimal for smaller problems.

CHAPTER 3: MODEL FORMULATION

3.1 Performance Measures

The primary goal is to minimize total latency, or weighted latency, across all customers. As a secondary measure, standard deviation among customer wait times is also sought as an indication of fair routing solutions. These two performance measures will be used in developing and evaluating the routing policies of the MDMLPI.

3.2 System Qualities

The system of the MDMLPI contains the following:

- A set of customers with demand for a single commodity to be fulfilled by one visit of a single vehicle
- A set of identical vehicles with limited capacity
- One main supply depot from which all vehicles begin their routes
- A set of intermediate supply depots at which vehicles can refill their supply en route

One characteristic of the original MDVRPI formulation not implemented with the MDMLPI is a route length maximum on each vehicle. For VRP solutions, it may be preferable for a smaller number of vehicles to service a larger number of customers, since the objective is to simply minimize route length of all vehicles. Thus, this constraint prevents some vehicles from performing large routes while others potentially never leave the main depot and contribute a route length of zero. However, this behavior is clearly suboptimal for problems with an MLP objective. Thus, the restriction on single tour lengths is not applied here, although it is expected to naturally be met regardless, given the nature of the MLP objective. Because the arrival time at all customers should be minimized, a strong solution will virtually always utilize every available vehicle and will inherently assign routes of similar lengths to each one.

3.3 Assumptions

The following assumptions are also made:

- The supply depots have unlimited capacity
- Each stop is assumed to have zero service time such that the time required to service a customer is simply the time required to travel there
- Likewise, time windows for supply replenishment at depots is assumed to be zero
- When calculating latency, only arrival times at customers are considered, not stops at intermediate depots or a return to the main depot, in keeping with the objective function

3.4 Mathematical Model

An original integer programming formulation for this new variant is formed as follows. Note that the nomenclature discussed below is presented concisely as a preliminary section of this document on page *ix*. Consider a set of nodes $V = \{1, 2,\ldots, n\}$, which is the inclusive set composed of customers, a main depot, and multiple intermediate depots. Let the subsets V_C, V_D, and V_I, define the set of customers, main depot, and intermediate depots, respectively. An arc between two nodes i and j in V has an associated travel time of c_{ij}. All arcs are symmetrical, implying that this is the equivalent distance associated with traveling from node j to node i. We also define $R = \{1, 2,\ldots, m\}$ as the set of vehicles servicing these nodes. If the arc from customer i to customer j is traversed by vehicle k, the value of a binary variable x_{ijk} is equal to 1, otherwise it is 0.

To write the objective of this problem, we define π_{ik} as the wait time for customer i being serviced by vehicle k, which is calculated as the sum of all travel times on route k preceding customer i. The objective to minimize latency is thus:

$$min \sum_{k=1}^{m} \sum_{i \in V_C} \pi_{ik} \tag{1}$$

For the objective of minimum weighted latency, we also define the demand for customer i as q_i. The minimum weighted latency objective is then written as:

$$min \sum_{k=1}^{m} \sum_{i \in V_C} q_i \pi_{ik} \tag{2}$$

We now explain the logic behind the formulation of the set of constraints. Firstly, we ensure that each customer is visited by exactly one vehicle. In other words, every customer i requires that the variable x_{ijk} equal 1 for exactly one vehicle and route pair, so:

$$\sum_{k=1}^{m} \sum_{j=1}^{n} x_{ijk} = 1 \qquad \forall i \in V_C \tag{3}$$

A constraint is also imposed that limits each arc to being traversed by at most one vehicle:

$$\sum_{k=1}^{m} x_{ijk} \leq 1 \qquad \forall i \in V, \ \forall j \in V \tag{4}$$

Next, every vehicle is required to begin its route at the main supply depot. In a TSP which has a completed, full-loop tour, this constraint would not always be necessary because a tour would start and end in the same place and have the same route length regardless of starting position. However, this is not the case with a minimum latency objective, and so every vehicle, k, must have $x_{ijk} = 1$ when i is the main depot and j is the one and only first stop on the tour.

$$\sum_{j=1}^{n} x_{ijk} = 1 \qquad \forall i \in V_D, \ \forall k \in R \tag{5}$$

The next constraint is implemented to ensure contiguous routing solutions. So far, the constraints assign a vehicle stop for every node in the system along the available arcs, but do not guarantee that when a vehicle arrives at a node it will necessarily depart from that same node for the next stop. To capture this, the symmetric behavior of the undirected arcs is utilized to define that the number of arrivals at any node j from node i on vehicle k (which is limited to at most one) must equal the number of departures from node j on vehicle k to any other node i.

$$\sum_{i=1}^{n} x_{ijk} = \sum_{i=1}^{n} x_{jik} \qquad \forall j \in V,\ \forall k \in R \tag{6}$$

The next constraint is a very simple limitation to prevent any vehicle from traversing an arc from one node directly to the same node. This accomplishes the same effect as assigning any arc from i to i a sufficiently large value such that it will never be used.

$$x_{iik} = 0 \qquad \forall i \in V,\ \forall k \in R \tag{7}$$

To keep track of the latency at each customer, a constraint must be set on π_{ik} according to the arc lengths that precede it on the route. This is done using the binary variable, x_{ijk}, and a large number operator M. Firstly, if the arc from customer i to j is assigned on route k, this constraint can be written so that the value of π at the ensuing stop j is equal to the previous value of π at stop i plus the arc length between i and j. If this arc is not active on vehicle k, however, the constraint should not calculate any latency. For this the term $(1 - x_{ijk}) * M$ is subtracted from the left hand side of the constraint. Thus, if $x_{ijk} = 0$, the term takes a value of M and the left hand side, which takes on a value of $-M$, is guaranteed to be less than the right hand side of the new latency value. This constraint was adapted from the cumulative wait time formulation used in Ngueveu et al. (2010) and applies to all customers and intermediate depots. Although intermediate depot latency is calculated, it is only used to add the impact of supply refills on the customers who are still waiting. It is not factored into the objective function.

$$\pi_{ik} + c_{ij} - (1 - x_{ijk}) * M \leq \pi_{jk} \qquad \forall i \in V \backslash V_D,\ \forall j \in V,\ \forall k \in R \tag{8}$$

These constraints also eliminate the possibility of subtours. If $x_{ijk} = 1$, then $\pi_{ik} + c_{ij} \leq \pi_{jk}$ must hold. For a subtour to exist, for example from node 1 to 2 to 3 on vehicle 1, then

$\pi_{11} + c_{12} + \pi_{21} + c_{23} + \pi_{31} + c_{31} \leq \pi_{11} + \pi_{21} + \pi_{31}$ and thus $c_{12} + c_{23} + c_{31} \leq 0$ which contradicts the problem definition and cannot occur (Ngueveu et al., 2010).

The next set of constraints performs similarly to the latency calculations and are used to keep track of the running supply a vehicle has available. Let μ_{ik} represent the cumulative amount of stock required from vehicle k when it services customer i. This value should be calculated as the sum of all customer need on the route up to and including customer i. The total need for a routing segment at the next stop, j, will be the amount needed up to the previous stop, i, plus the demand at the next stop, which is denoted by q_j.

$$\mu_{ik} + q_j - (1 - x_{ijk}) * M \leq \mu_{jk} \qquad \forall i \in V,\ \forall j \in V_C,\ \ \forall k \in R \tag{9}$$

To account for supply refills, the following constraint is added to the main depot and set of intermediate depots. At any of these entities, supply required up to that point is zero and all ensuing calculations of μ_{ik} will 'reset' from this starting value.

$$\mu_{ik} = 0 \qquad \forall i \in V/V_C,\ \forall k \in R \tag{10}$$

The final constraint on capacity must ensure that the amount of supply demand for any route segment between depots has to be within the vehicle's carrying capacity, which is denoted Q and is the same for each vehicle.

$$\mu_{ik} \leq Q \qquad \forall i \in V,\ \forall k \in R \tag{11}$$

Finally, x_{ijk} is defined as a binary variable and non-negativity constraints are imposed for variables π_{ik} and μ_{ik}.

$$x_{ijk} \in [0,1] \qquad \forall i \in V,\ \forall j \in V,\ \forall k \in R \tag{12}$$

$$\pi_{ik}, \mu_{ik} \geq 0 \qquad \forall i \in V,\ \forall k \in R \tag{13}$$

For completeness, the model is written in full in Appendix A.

One final note should be made for the implementation of this model into optimization solver software. The formulation precludes the possibility of multiple visits to the same supply depot, including en route stops at the main depot, because of the calculation of latency and supply need. Multiple visits is a quality that should be allowed in the system to account for all possible tours. This can be accomplished by simply adding several hypothetical supply depots, or ‘phantom’ depots, with the same locations, qualities, and associated travel costs as the existing set. This will allow routing solutions to use the resources of a depot multiple times while addressing it by separate indices.

CHAPTER 4: HEURISTIC APPROACHES

The MDMLPI is an NP-hard problem. Because instances of any reasonably large size cannot be solved to optimality, it is beneficial to develop heuristics that can provide strong, near-optimal solutions and that can be solved in a reasonable amount of computation time, even for large problem instances. This motivates the development of two heuristic approaches and a second-phase adjustment that are applied to problem instances containing up to several hundred customers. Additionally, because no optimal solutions exist for the problem instances to be tested, two sets of benchmarking solutions are introduced. The first comes from a nearest neighbor policy that is described in section 4.1. Secondly, because this problem has been studied as a VRP, the best-known solutions to solving with this objective are used. The (weighted) latencies that would result from applying these solutions with the MLP objective are calculated and used for comparison between the two objectives. These calculated values will be introduced in Table 6.2.

4.1 Nearest Neighbor Benchmark

Firstly, a simple nearest neighbor, or greedy, policy was implemented. A nearest neighbor policy typically performs well in routing problems and requires very little computational need. Here, vehicle stops are selected sequentially such that the vehicle with the shortest route length at any time is the next one to select. This allows real-time assignment of stops and balances route lengths among vehicles, which is a desirable property when wait times are to be minimized. The results from this simple policy will be used along with previous VRP solutions as a benchmark for the remaining heuristics.

4.2 Initial Heuristics

4.2.1 Auction Policy

An auction policy is developed in which nodes are assigned to vehicles sequentially as is done in the nearest neighbor benchmark, where the vehicle with the current shortest route at any time accepts bids from all nodes that have not yet been served. Each node bids once at every iteration until it is serviced and the lowest bid of each determines the vehicle's next stop.

Four bidding parameters are included:

- Vehicle's Distance to Bidding Customer: This is the single factor used in the nearest neighbor policy. It gives a more competitive bid to nodes that are close to the vehicle's current location.
- Customer Need: This will affect the number of customers a vehicle is able to serve before requiring a refill stop at an intermediate depot. In the case of weighted latency, it directly relates to the objective function and is paired with a negative calibrating parameter because customers with high need will ideally receive quick service. This high need may correspond to a larger number of people at that node.
- Customer's Distance to Nearest Neighbor: In either the minimum latency or the minimum weighted latency case, a node will be a more attractive option, and thus have a lower bid, if it has a close neighbor that can also be served around the same time.
- Customer's Average Distance to Other Nodes: Similarly, nodes found in small 'clusters' of other nodes with a smaller average distance to its neighbors will have more attractive bids. This adds an element of centrality to the vehicle's decision-making.

These factors led to the bidding function shown below for each node, j, with calibrating parameters and variables defined as follows:

c_{ij}	Distance from vehicle at its current stop i to the bidding node j
c_{il}	Distance from node i to neighbor l

q_i	Supply needed at customer *l*
n	Number of nodes

$$bid(i) = c_{ij} * q_i^{w_1} * \left(\min_{\forall j \in V} c_{il}\right)^{w_2} * \left(\frac{\sum_{j=1}^{n} c_{il}}{n}\right)^{w_3}$$

The values of the three calibration parameters, w_1, w_2, and w_3, were determined as follows. For the supply needed at each customer, which is modified by w_1, a different behavior is expected between the latency and weighted latency objectives. In the case of weighted latency, it is clear that a larger demand should correspond to a more attractive (lower) bid, so the calibration should take on a negative value. With the minimum latency objective, however, this is less clear because visiting customers with large demand will result in earlier, and probably more frequent, stops at intermediate depots. Because this may be suboptimal behavior, positive values were also considered for w_1. For w_2 and w_3, however, both parameters should take on positive values for both objectives. This will give lower bids to customers with smaller distances to neighbors, which is preferable because it gives priority to clusters of nodes.

With these directions established, experimentation was done to find appropriate ranges for these values that lead to strong results. It was determined to vary w_1 from -0.2 to 0.2 by steps of 0.05, w_2 from 0 to 0.5 by 0.05, and w_3 from 0 to 1 by steps of 0.2. These parameter calibrations are uniformly applied to the minimum latency and minimum weighted latency objectives alike.

Stops at intermediate depots must also be considered to ensure that capacity constraints are met. Heuristics must therefore keep track of how much is demanded from each vehicle's route and add a stop at a nearby depot before capacity is violated. For the auction heuristic, this was implemented with the following rule: if at any point bids are calculated and the current vehicle does not have sufficient supply to fulfill the need of the lowest-bidding customer, it is routed to refill at the depot it is closest to at that point.

Pseudocode for the auction policy is included in Appendix B.

As an illustration of how the bidding function works, consider a section of a routing network as in Figure 4.1. The solid arrow shows the last arc traversed by the servicing vehicle and the vehicle's current location is at that lower left node. Although all customers who have not been served will place bids at this point, let's consider the bids of just two nodes as an example. These bidding nodes are identified by connection to the current location by dashed arrows.

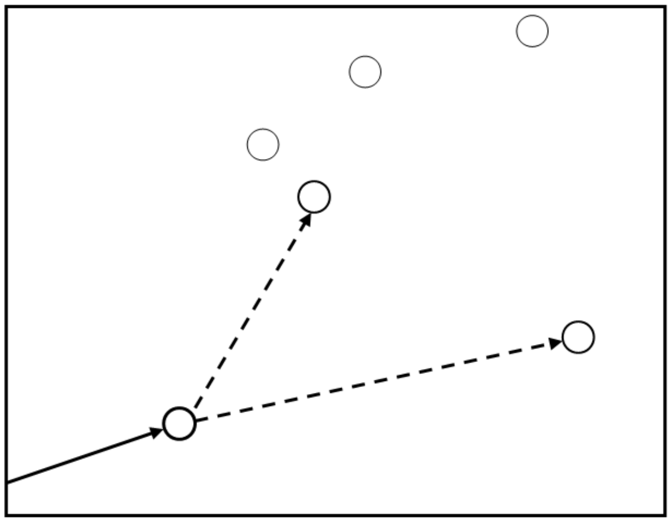
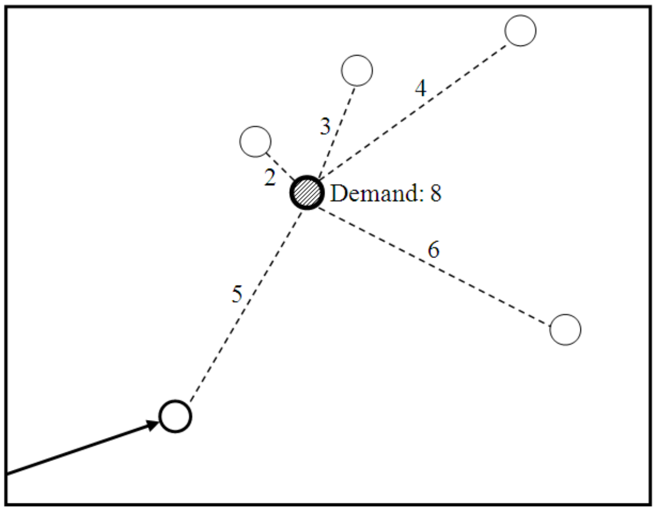


Figure 4.1: Example network for auction heuristic

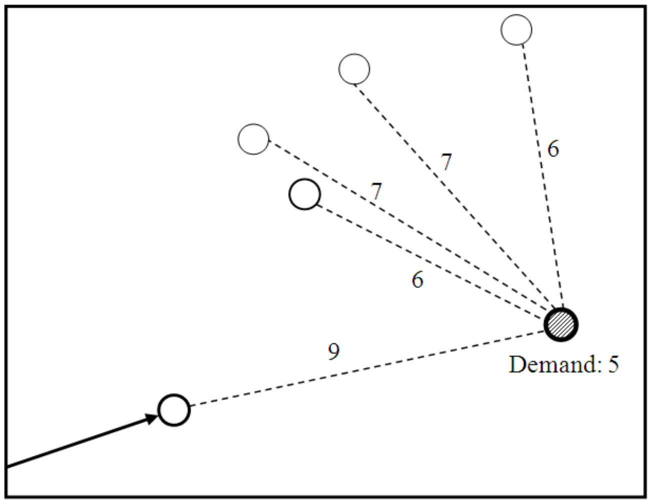
First, we break down the bidding parameters for the node on the left. Figure 4.2 contains further information about arc lengths and demand at this customer. The values of each of the four bidding parameters can be clearly inferred using this information. The distance to the bidding node from the vehicle's current position is 5 units and it has a demand of 8 for the needed commodity. Additionally, the node's nearest neighbor is 2 units away. Lastly, the average travel cost associated between this node and any possible neighbor in the system is calculated to be 4.



$$c_{ij} = 5$$
$$d_j = 8$$
$$min\, c_{il} = 2$$
$$mean\, c_{lj} = \frac{5+2+3+4+6}{5} = 4$$

Figure 4.2: Bidding parameters for first bidding customer

Next, consider the bidding parameters for the second node in the lower right. This has the demand and travel costs detailed in Figure 4.3.



$$c_{ij} = 9$$
$$d_j = 5$$
$$min\, c_{il} = 6$$
$$mean\, c_{il} = \frac{9+6+7+7+6}{5} = 7$$

Figure 4.3: Bidding parameters for second bidding customer

This customer's bidding parameters are a demand of 5, a distance from the vehicle of 9, a nearest neighbor at a distance of 6, and an average neighbor distance of 7. Without calculating the final bids from each node, it is trivial to see that the first customer we considered will have a more attractive bid than the second. It is closer to the vehicle, has higher need, and exists in a cluster of nodes that can be serviced together for efficient performance.

4.2.2 Insertion

The second algorithm developed for this problem is a variation of a well-known approach referred to as insertion (e.g. Campbell et al., 2008; Ke & Feng, 2013). The basic structure runs according to the following steps:

1. Order all customers based on proximity to the main (starting) depot from least to greatest
2. Considering customers in this order, begin placing them one by one into a partial routing solution:
    a. For each customer, consider placing them in every possible position in the current partial solution
    b. Among these, select the position that generates the least increase in overall latency
3. Store the top H partial solutions that have the shortest latency
4. Continue until all nodes have been assigned
5. For each of these top H solutions, that are now complete with all customers, add any necessary stops at intermediate depots (this procedure is detailed below)
6. Select from the final group of the top complete schedules the one that has the minimum total latency

See Appendix C for the pseudocode of the insertion technique.

This technique is illustrated as follows. Consider the partial routing solution in Figure 4.4. At present, the first 7 nodes that are considered have been placed in the solution and node 8 is now considered. Let each row in the figure represent the tour of a single vehicle in the system, i.e. this example has four vehicles. Further, the width of each entry designates the travel cost associated with traveling to that node from its predecessor. Therefore, the width will vary based on the node's predecessor, which determines where the vehicle is traveling from. The first nodes in each route are being visited directly from the main depot.

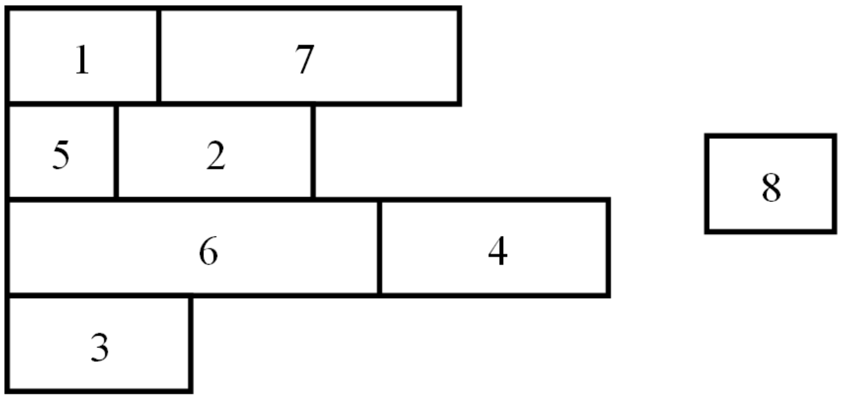


Figure 4.4: Partial routing example for insertion heuristic

The algorithm will consider placing node 8 in every possible position on the partial tour. At each possibility, the increase in overall latency as a result of this assignment can be determined and the position with the minimum latency increase will be stored. Consider placing node 8 in the first position of the first vehicle, as in Figure 4.5.

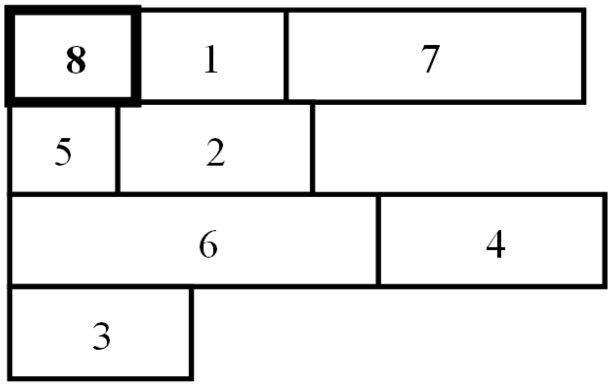


Figure 4.5: Possible position for customer 8 in partial solution

The impact of this addition will come from three things: 1) the travel cost from the main depot (node 0) to node 8 will be added to the overall latency, 2) node 8 will push back the service time of nodes 1 and 7 by this amount, and 3) the new travel cost from node 8 to node 1 will replace the previous cost which was from the main depot to node 1. The calculation of how much latency is added to the solution becomes:

$$\text{Additional latency} = c_{08} + 2c_{08} + (c_{81} - c_{01})$$

As another example, consider Figure 4.6, in which node 8 is placed between nodes 6 and 4 on the third vehicle's route.

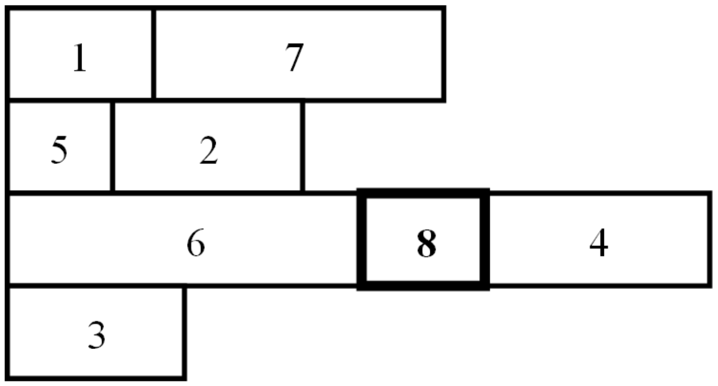


Figure 4.6: Alternative position for customer 8 in partial solution

In addition to the latency increases noted above, there will be another increase of the travel time to node 6, since this (and any other predecessors that could exist) impacts the latency of all following nodes. Thus, additional latency for placing node 8 here can be written:

$$\text{Additional latency} = c_{06} + c_{68} + 1c_{68} + (c_{84} - c_{64})$$

Note that the coefficient on c_{68} in this calculation is 1 because it only has one successor in this case, as opposed to the two successors in the previous example.

The key advantages to this approach are that it is simple to perform and that it builds a set of promising routes, maintaining several strong candidates as opposed to one single partial solution. It does, however, only consider one customer at a time, similar to the nearest neighbor policy, and therefore does not consider the value of things like node clusters with short travel costs among them, which the auction policy takes into account.

For this thesis, a value of $H = 5$ was utilized to aim for a balance of strong results within reasonable computation time. Higher values of H would be able to provide improved solutions, but the time expense of such runs is outside the bounds of this research.

Depot assignment

Following the initial assignment of nodes to routes and sequencing of the tours, it is also necessary to consider how stops at supply depots will be assigned in such cases. The simplest way to do this is to keep a running sum of all node demands and add a depot stop at the nearest depot one stop before that demand will exceed capacity. However, the impact of routing to refill supply can be alleviated if the vehicle considers stopping more than one stop in advance of running out of supply. For example, the vehicle may be much closer to a depot two or three stops before refilling is absolutely necessary, and it may be optimal to stop early instead of incurring later inconvenience.

As a result, whenever a capacity limitation is identified, the algorithm adds an intermediate depot stop at the point among the most recent D stops at which the vehicle was closest to a depot. Because a value of D that is too large can also incur negative effects from too many unnecessary stops, this value was varied between 1 and 10 and the best objective among the results was kept. The pseudocode for this depot assignment procedure is found in Appendix D.

For example, consider the partial route depicted in panel (a) of Figure 4.7. The arrows show the path of the vehicle as assigned by the insertion algorithm without yet considering depots. Consider the case now that the top node in black will not be able to be served due to the vehicle's capacity limit, so a stop at a depot must be added before traveling to this node. Two possible depots are shown as squares in the figure.

Panel (b) of the figure represents what the route may look like if the vehicle were routed to its nearest depot when the demand shortage occurred. It is clear that this requires a significant deviation from the current route trajectory. If the vehicle were to refill several stops in advance of this shortage, such an inefficiency can likely be eliminated.

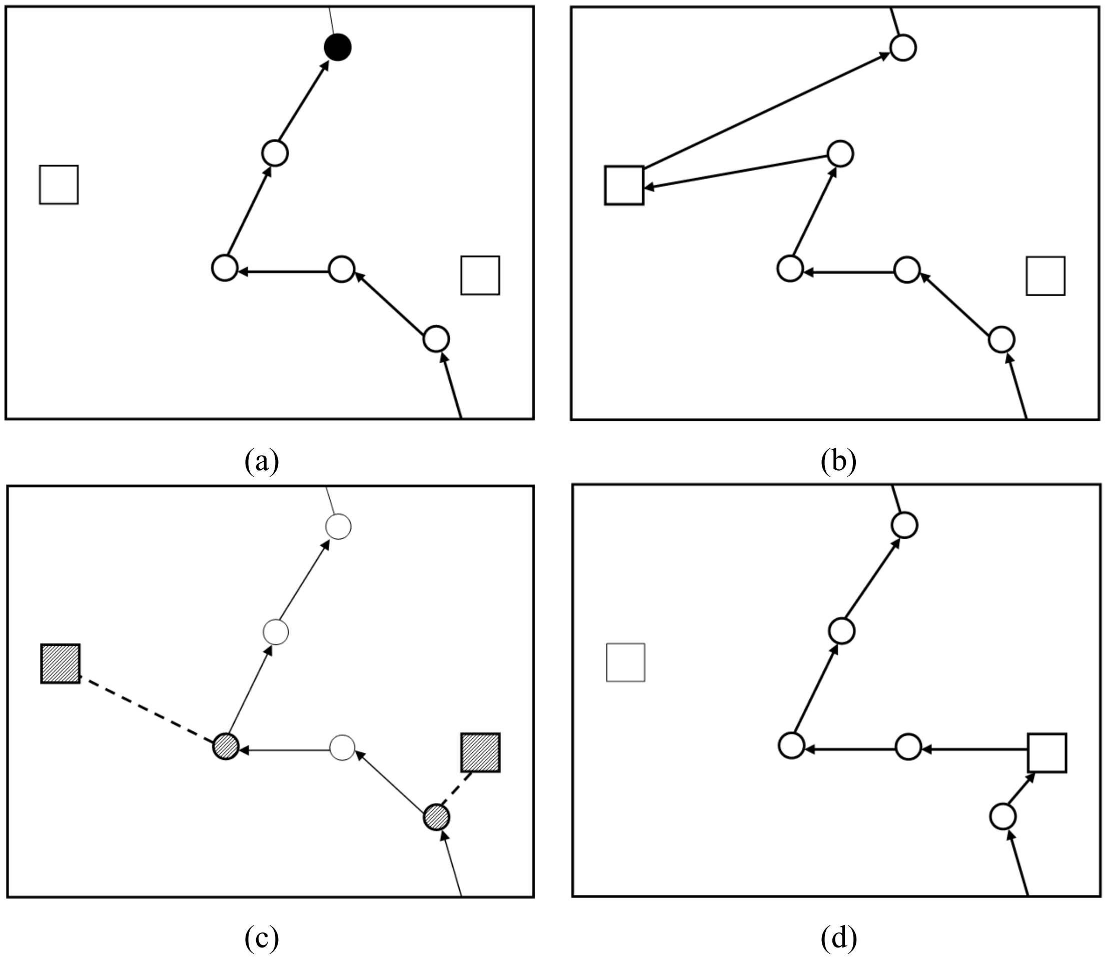


Figure 4.7: Illustration of depot assignment for insertion heuristic

In an effort to capitalize on this potential for reducing latency, the depot assignment algorithm considers the closest point that the route gets to a depot in the last several stops. Panel (c) of the figure shows the two strong candidates for depot stops, which are the minimum distance this section of the route gets from each of the two depots in the figure. The minimum of all candidates, which in this case is the node-depot pair in the lower right of the graph, is selected. Panel (d) then displays this section of the routing solution with the efficiently added depot visit.

Depot assignment is done on all of the top H completed solutions from the initial step. The full route with stops at depots with the best overall (weighted) latency is selected among these as the final routing solution.

4.3 Second-Phase Adjustment

Finally, a second-phase modification to the above solutions is introduced to achieve additional improvements. Due to the cumulative nature of the MLP, an inefficiently-positioned node impacts the routing solution for every stop following it, and thus the effect of any inefficiency is compounded. To help alleviate this effect, this adjustment takes the previous routing solutions and considers moving each customer to any alternate stop on the route to see if an improvement in total latency can be made. With each possibility, it also must be ensured that capacity constraints are still satisfied. Any time an improvement is achieved, the new positioning is checked to ensure that no capacity constraints have been violated. If they have, the improvement is not allowed to occur, since it is no longer a feasible solution. With this structure, results from the second-phase adjustment are guaranteed to be at least as good as the original solution.

The order in which nodes are considered for a position change (either on its current route or being assigned to a different vehicle) impacts the available changes of subsequent nodes. Thus, nodes were considered in a random order, checking every position for the current node before considering the next. As a result, each instance was run multiple times using this adjustment. The number of repetitions was selected to be ten as this is expected to provide a good estimation of the true mean of the distribution. The best results as well as averages are reported. See Appendix E for the pseudocode of this adjustment.

To illustrate this procedure, consider the following example shown in Figure 4.8. Here, assume that the routing solution below is complete for a system of 11 nodes and 4 vehicles. If the total latency (sum of wait times for each unit represented below) is calculated, the solution currently has a latency of 73 time units.

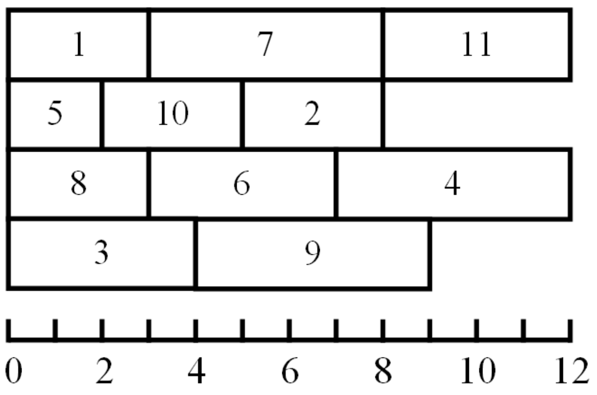


Figure 4.8: Example complete schedule for second-phase adjustment

When the second-phase heuristic is run, it considers moving each node to a new position to see if total latency can be improved. To explore the calculation behind this decision, consider Figure 4.9, which moves node 6 from the middle position for the third vehicle (third row) to the middle position on the last vehicle's route, between nodes 3 and 9.

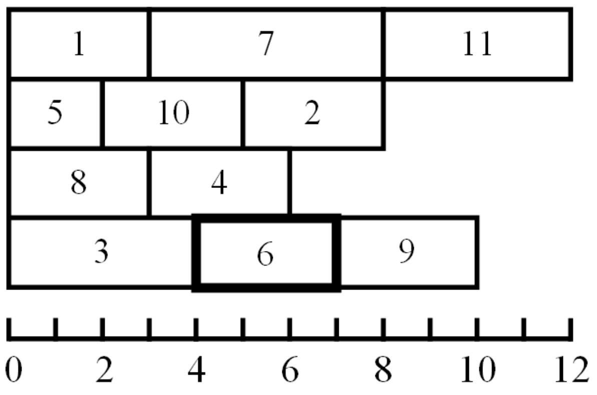


Figure 4.9: Possible repositioning using second-phase adjustment

This adjustment will be implemented and stored if the reductions in latencies outweigh the additional latency values that are incurred. Reductions in this case will result from removing node 6 from the third vehicle's route and moving node 4 to be serviced earlier in the schedule. New latency costs will be incurred as a result of adding node 6 to the last route and pushing back the position of customer 9. Because travel costs depend on both the departure and arrival nodes, changes in the time it takes to service a customer may also change, either in the positive or negative direction. Therefore, this potential movement will have the impact calculated below.

	Impact on total latency
Removal of node 6 latency in row 3:	-7
Reduction in wait time before vehicle departs for node 4 (originally 7, now 3):	-4
Change in travel cost to node 4 (originally 5, now 3):	-2
Wait time before vehicle departs for node 6 in new position:	4
New travel cost to node 6:	3
Additional wait time before vehicle departs for node 9 (originally 4, now 7):	3
Change in travel cost to node 9 (originally 5, now 3):	-2
Total system latency change:	-5

Repositioning node 6 between nodes 3 and 9 is, therefore, a beneficial move in terms of total latency and the new solution will have a cumulative latency of 68 time units, which can be verified by adding the waiting times of all customers shown in the previous graphic.

CHAPTER 5: PRELIMINARY EVALUATION

The proposed model was implemented using CPlex solver to validate all system components. Next, both of the heuristic approaches as well as the second-phase adjustment were coded and implemented in C++. Because the problem instances being solved are too large to be solved to optimality, two methods were used to test the proposed algorithms on similar, smaller datasets for validation.

5.1 Test Instances

The qualities of the ten instances to be tested on, as generated in Crevier et al. (2007) and labeled a through j, are summarized in Table 5.1. The full results are available online at http://neumann.hec.ca/chairedistributique/data.

Table 5.1: Characteristics of the 10 data sets used for testing

Instance	# Depots	# Customers	# Vehicles	Vehicle Capacity
a	5	48	4	150
b	5	96	4	200
c	5	144	4	250
d	5	192	3	300
e	5	240	3	350
f	5	288	3	400
g	7	72	4	175
h	7	144	4	250
i	7	216	3	325
j	7	288	3	400

Note that these details do not include the constraint on route maximum duration. As was mentioned in the problem definition, a strong minimum latency solution will utilize all available vehicles such that the route length of each is relatively similar. This negates the need for the maximum route duration constraint. Such a constraint was used with the minimum route length objective because a strong solution may involve some vehicles not being used at all while others serve extremely long routes. Take, for example, a pair of nodes that are 1 unit away from each other and both 10 units from the starting depot. If the goal is to minimize route length, one vehicle will optimally service both nodes, which requires 11 units of travel, instead of each vehicle traveling 10 units to get to one of the nodes. However, it is always optimal to use every available vehicle when minimizing latencies. In the same example, the single vehicle solution will have latencies of 10 at the first node and 11 units at the second, totaling 21. However, if each vehicle travels to one of the nodes, the latencies at each node will be 10 units, totaling to 20.

5.2 Validation on Small Instance Subsets

Firstly, subsets of the problem instances were created so that they could be solved to optimality using CPlex solver and approximated with the proposed heuristics in order to 1) ensure that all qualities and constraints are properly being implemented in the full complexity of the problem and 2) indicate the performance of the algorithms compared to an optimal solution. Although results from such an approach are not necessarily expandable to the larger problems (for example the smaller problems generate tighter capacity constraints, adding an emphasis on this component that is not as crucial in the larger problems), they provide an initial indication of result quality and ensure validity of the approaches. Additionally, small modifications were made to the heuristics as necessary to fit the smaller problems, such as an adjusted calibration of the bidding parameters used in the auction policy. The details and full problem instances of these test data are included in Appendix F.

Because of the expensive computation of the MDMLPI, very small subsets were used for each problem instance. Each one uses the main depot, two intermediate depots, and six nodes from the original instance. Additionally, a 'phantom' depot was created for the main depot so the option of revisiting en route was possible for the optimization. With these selected, each problem contained ten entities among customer nodes and depots. To select the subsets, the first six nodes in the order in which they are originally numbered were chosen. Additionally, the first and third intermediate depots were selected in order to increase the likelihood of the intermediate depots being on either side of the main depot rather than close to each other in a shared area. The original demand at each of these customers was also retained.

Based on the proportion of these instances to the full ones, it was decided to use a constant vehicle capacity of 30 and to use 2 vehicles in each instance. These characteristics were selected so that vehicles would likely have to each make a stop at an intermediate depot in most of the solutions. This way, the validity and effectiveness of depot assignments could also be tested.

5.2.1 Minimize Latency

The performance of heuristics on the small instances using the minimum latency objective is presented in Table 5.2. Here, optimal solutions are presented along with results from the two heuristics and the nearest neighbor benchmark.

Table 5.2: Optimal, benchmark, and heuristic results for small problem instances

Instance	Optimal	Auction	Insertion	Nearest Neighbor
a	451.78	451.78	505.20	595.35
b	583.05	583.05	586.71	649.28
c	648.73	682.95	648.73	682.95
d	578.06	579.23	578.06	743.04
e	514.61	528.30	514.61	616.24
f	779.45	854.08	779.45	933.63
g	835.05	835.05	909.60	928.57
h	819.20	843.87	1,006.60	971.78
i	885.73	1,011.35	1,037.70	1,053.10
j	389.84	431.58	450.80	589.73

The initial results are as expected. Both heuristics seem to perform generally well when compared to optimality. Between the two of them, the optimal solution is acquired for 7 out of the ten instances. The auction heuristic does this for 3 instances, while insertion accomplishes optimality for 4 of them. It is also easy to see that they outperform the nearest neighbor benchmark in general.

In order to make comparison between the approaches more clear, Table 5.3 presents the results from each policy as a ratio over the optimal value.

Table 5.3: Ratio of each heuristic and NN benchmark over optimality for small instances

Instance	Auction	Insertion	Nearest Neighbor
a	1.0000	1.1182	1.3178
b	1.0000	1.0063	1.1136
c	1.0527	1.0000	1.0527
d	1.0020	1.0000	1.2854
e	1.0266	1.0000	1.1975
f	1.0957	1.0000	1.1978
g	1.0000	1.0893	1.1120
h	1.0301	1.2288	1.1863
i	1.1418	1.1716	1.1890
j	1.1071	1.1564	1.5127
Average	**1.0456**	**1.0771**	**1.2165**

These ratios reveal that the auction policy achieves a slightly better average optimality ratio than insertion, despite achieving the optimal solution for fewer instances. This can be attributed to higher variance in the solutions obtained by insertion, which has several instances with ratios above 1.10 and even 1.20. Given the small size of the instances, these large gaps may simply be the result of one inefficient routing assignment.

For example, the route obtained by the insertion heuristic for instance *i* is very similar to the optimal route. The optimal solution uses the routes:

$$0 \rightarrow 5 \rightarrow 9 \rightarrow 3 \rightarrow 2 \rightarrow 1 \text{ and } 0 \rightarrow 4 \rightarrow 6$$

Note that in this notation, the main depot is denoted 0, the customers are nodes 1 through 6, and the intermediate depots are labeled 8 and 9. For this same instance, the insertion heuristic developed the following route, which is almost identical except for the placement of node 1 and a corresponding depot stop to fulfill it:

$$0 \rightarrow 5 \rightarrow 9 \rightarrow 3 \rightarrow 2 \text{ and } 0 \rightarrow 4 \rightarrow 6 \rightarrow 9 \rightarrow 1$$

Despite the near-optimal route developed with the insertion technique, the solution still had 15.64% longer latency. This effect, however, is not expected to be nearly as extreme in the full instances. The presence of many more nodes will allow more near optimal paths and be able to dilute the impacts of being slightly off from optimality.

5.2.2 Minimize Weighted Latency

Next, each of the ten instances was re-optimized and run for both heuristics with the objective of minimizing weighted latency. The results shown in Table 5.4 are similar to what was seen above. Both heuristics perform well in general, achieving optimal solutions for several cases and very close approximations for most others. A few outlying solutions again exist as a result of the small network in which any suboptimal decision has a proportionally large impact on the overall weighted latency.

Table 5.4: Optimal, heuristic, and benchmark results on small weighted instances

Instance	Optimal	Auction	Insertion	Nearest Neighbor
a	4,412.78	4,412.80	5,031.40	5,875.10
b	6,095.57	6,102.20	6,095.60	7,034.10
c	9,010.67	9,734.50	9,253.00	9,800.60
d	6,424.93	6,440.70	6,424.93	8,679.30
e	6,476.03	6,629.10	6,629.10	7,882.30
f	10,964.80	11,977.00	11,556.00	12,860.00
g	9,740.29	9,772.90	10,096.00	14,397.00
h	8,859.00	9,030.20	9,307.40	10,365.00
i	9,066.23	10,309.00	9,066.23	10,899.00
j	5,744.04	6,183.20	6,275.00	9,960.40

As is shown above, the two heuristic approaches attain the optimal routes for instances *a*, *b*, *d*, and *i* between them. In general, both average a strong approximation to the optimality in this small network. Table 5.5 displays each policy's ratio over the optimal objective value.

Table 5.5: Ratio of new heuristics over optimality for small weighted instances

Instance	**Auction**	**Insertion**	**Nearest Neighbor**
a	1.0000	1.1402	1.3314
b	1.0011	1.0000	1.1540
c	1.0803	1.0269	1.0877
d	1.0025	1.0000	1.3509
e	1.0236	1.0236	1.2172
f	1.0923	1.0539	1.1728
g	1.0033	1.0365	1.4781
h	1.0193	1.0506	1.1700
i	1.1371	1.0000	1.2022
j	1.0765	1.0924	1.7340
Average	**1.0436**	**1.0424**	**1.2898**

Although in the previous case the auction technique slightly outperformed insertion for these tests, both perform about equally in these ten weighted latency instances. The average ratio over optimality is strong for both of the heuristics, hovering just over 1.04 for each.

Using small subsets of the problem instances has thus helped to validate the legitimacy of each heuristic. These small instances contain all components that will be included in the larger implementation, including the use of multiple vehicles and intermediate depot refills. For both the minimum latency and minimum weighty latency objective functions, the algorithms provide strong, near-optimal results. The strength of these solutions is also expected to increase when tested on the full instances for which they were developed. This is because with more sizeable instances, a larger population of nodes will allow additional near-optimal routing possibilities and reduce the consequence of any suboptimal assignments.

For further validation, the heuristics were also tested on single-vehicle instances of the MLP. While optimality is not verified for these instances, strong upper bounds exist from previous literature and will serve as a strong benchmark.

5.3 Validation on Single Vehicle Instances

The second method will test the algorithms on a set of larger, single-depot problems for which strong upper bounds exist. These test instances will allow for further validation of the algorithms and of their promising performance. Note that only the minimum latency case without weighting by demand is considered here since the instances used here do not include customer demand.

Several previous works test approximation algorithms and heuristics for the MLP on instances from the Traveling Salesman Problem Library (TSPLIB). Archer, Levin, & Williamson (2008) apply an approximation algorithm for the MLP that offers an improved approximation ratio from previous works. They present the results of running the algorithm on a variety of instances from the TSPLIB, ranging in size from 51 to 1084.

Abeledo et al. (2010) use a branch-cut-and-price algorithm on similar instances, including improvements on some of the instances from the previous work as well as presenting new upper bounds to others from TSPLIB. The instances used are generally smaller problems. Similarly, Salehipour et al. (2011) improve and expand on these results using a metaheuristic on a variety of instances. This metahueristic utilizes Greedy Randomized Adaptive Search Procedure (GRASP) as well as Variable Neighborhood Descent (VND) and Variable Neighborhood Search (VNS). Finally, Silva et al. (2012) further develop these results by introducing a three-phase metaheuristic. The presented approach uses a combination of GRASP, VNS, and Iterated Local Search (ILS) and is run on instances used in both Abeledo et al. (2010) and Salehipour et al. (2011). Several upper bounds are improved upon and notable reductions in computation requirements are demonstrated.

These works collectively present the best-known upper bounds (i.e. best solutions achieved so far) to many instances adapted from TSPLIB with an MLP objective function. Several of these instances are selected from the previous literature for validation and comparison of the algorithms presented herein. Those that are selected range in size from 42 to 107 nodes. The MDMLPI instances that are used for testing range from 48 to 288 nodes serviced among either 3 or 4 vehicles. Thus, these single-depot instances are comparable in size to the individual routes that the MDMLPI will be divided into.

Table 5.6 presents the best-known results from previous literature for each problem as well as the results of each heuristic adapted to the characteristics of the single-depot problem. Note that the following table presents results from the heuristics with the second-phase adjustment applied.

Table 5.6: Performance of new heuristics with rearranging on TSPLIB instances

Data	**Best Known Solution**	**Auction**	**Insertion**
berlin52	143,721	143,420	135,253
dantzig42	12,528	11,690	10,999
eil101	27,513	29,806	29,822
eil51	10,178	10,023	10,239
eil76	17,976	18,331	19,933
kroA100	983,128	1,051,500	1,136,803
kroB100	986,008	1,066,500	1,133,872
kroC100	961,324	1,015,100	1,196,915
kroD100	976,965	1,081,100	1,160,789
kroE100	971,266	1,020,300	1,048,637
lin105	585,823	670,970	761,501
pr107	1,980,767	2,224,100	2,091,237
pr76	3,455,242	3,504,300	3,922,911
rat99	54,984	60,790	63,274
rd100	340,047	370,400	374,433
st70	19,215	21,591	21,408

The heuristics perform well in general and offer improved solutions for three problem instances: berlin52, dantzig42, and eil51. Under these characteristics, the quality of each approach seems to be stronger in smaller problem instances. The results are more clearly demonstrated in Table 5.7, which presents the ratio of each heuristic solution over the previous best-known results.

Table 5.7: Ratios of each heuristic over best known solutions for TSPLIB instances

Data	**Auction**	**Insertion**
berlin52	0.9979	0.9411
dantzig42	0.9331	0.8780
eil101	1.0833	1.0839
eil51	0.9848	1.0060
eil76	1.0197	1.1088
kroA100	1.0695	1.1563
kroB100	1.0816	1.1500
kroC100	1.0559	1.2451
kroD100	1.1066	1.1882
kroE100	1.0505	1.0797
lin105	1.1453	1.2999
pr107	1.1228	1.0558
pr76	1.0142	1.1354
rat99	1.1056	1.1508
rd100	1.0893	1.1011
st70	1.1237	1.1141
Average	**1.0615**	**1.1059**

The auction policy seems to perform slightly better in these cases than the insertion technique, with an average ratio of 1.0615 over previous best-known solutions as compared to 1.1059 from the insertion results. A couple noteworthy observations stem from considering these results. First, while each heuristic performs generally well in these single-depot instances, it is anticipated that the results when applied to the full MDMLPI will be closer to optimality. This is because both heuristics were developed for the multiple vehicle, multiple depot problem and have particular qualities that cater to carefully assigning nodes to individual routes and efficiently sequencing customers as well as stops at depots.

Secondly, further analysis of the insertion technique demonstrates how a small adjustment can more accurately fit select cases. As described above, the order in which nodes are assigned in the insertion technique is based on proximity to the starting depot. Nodes that have a smaller Euclidean distance to the main depot are assigned first. This performs well in general, but strong performance is particularly noted for the single-vehicle instances of smaller size (e.g. berlin52, dantzig 42, and eil51). This observation also indicates that the approach may perform better in problem instances of the multi-vehicle case in which vehicles each serve around this number of nodes.

However, a weaker performance is detected here when the single vehicle serves a larger number of customers. To see this effect and why ordering nodes by main depot proximity may contribute to this inefficient behavior, consider Figure 5.1. This depicts the routing solution of the insertion technique for the kroC100 instance. Here, each '+' symbol represents a customer and a line is an arc that is traversed between two of them.

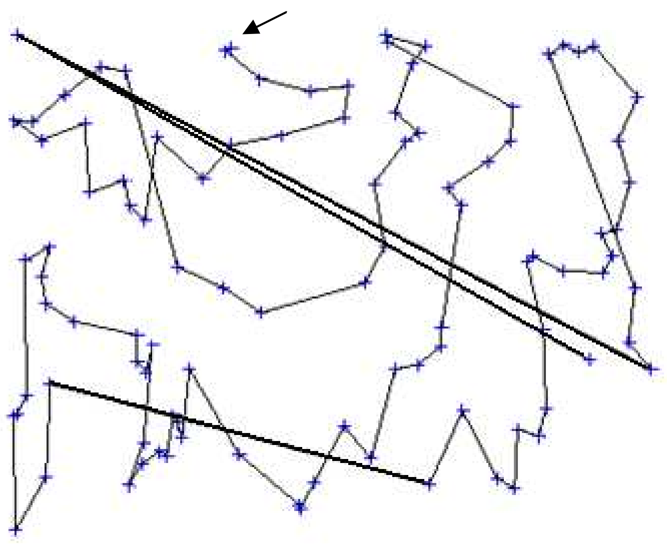


Figure 5.1: Routing solution for kroC100 using insertion

The starting position for the vehicle (indicated by the arrow on the graph) is near the top middle of the service area. As the vehicle progresses along its route, the nature of the algorithm causes it to stay close to the starting position early before expanding outward to the furthest regions of the service area. This initial behavior on the graph resembles concentric circles (or arcs) expanding outwardly from the starting position. While this initial behavior does not appear to be inefficient itself, it causes significant later delays.

The clearest inefficiency comes with the last two nodes serviced. It can be identified by a pair of long arcs (in bold on the diagram) that cut diagonally across a large section of the routes. Because the second-to-last node served (which is in the top left of the routing area) was considered relatively early in the process, it appears to have gotten locked into a position that later became inefficient.

In addition to this clear inefficiency, there is another lengthy arc (also in bold along the bottom-left of the diagram) that has significant implications. This is found in the lower-left area of the graph, where the vehicle is heading from left to right to serve the final group of customers around the rightmost edge of the area. The difference between this and the previously cited inefficiency is that the cost of traversing this lengthy arc impacts the wait times of numerous customers that follow it, unlike the last one which only impacted the final two customers. There are still 27 nodes serviced after this point, which means the long arc is contributing a large amount to system wait times.

To alleviate such an effect, an alternative to this ordering for larger instances is suggested. Once the ordering of nodes has been determined by main depot proximity, the heuristic could start considering every other node at first, then make a second run through the ordered list and assign those that were skipped. For example, the first pass would consider nodes 1, 3, 5, … , n-1 (assuming n is even) and the second will assign nodes 2, 4, 6, … , n. With this technique, the general structure of routes can be formed from start to finish, effectively using n/2 nodes. On the second pass, nodes can be conveniently placed into the already existing routing structure. Dividing into two steps will alleviate the pressure of a single vehicle being constrained closer to the main depot early on, which may not always be efficient behavior.

The 16 single-vehicle instances from above were run with this adjustment. To examine its effects, Table 5.8 presents these results and their ratio when compared to the previous best-known solutions.

Table 5.8: Results of insertion technique using every other ordered node

Data	**Result**	**Ratio**
berlin52	136,500	0.9498
dantzig42	11,761	0.9388
eil101	30,392	1.1046
eil51	10,097	0.9921
eil76	19,244	1.0706
kroA100	1,022,293	1.0398
kroB100	1,116,976	1.1328
kroC100	1,077,629	1.1210
kroD100	1,071,704	1.0970
kroE100	1,017,427	1.0475
lin105	675,848	1.1537
pr107	2,161,080	1.0910
pr76	3,416,496	0.9888
rat99	60,549	1.1012
rd100	358,945	1.0556
st70	20,462	1.0649
Average		**1.0593**

While some solutions are improved and a few are worsened, the above results demonstrate that the adjusted ordering of nodes generally performs better in these instances, with an average ratio of 1.0593 from previous best-known solutions. This is especially true for problems of larger size, such as kroA100, kroC100, and lin105, which have some of the most significant reductions.

The new routing solution achieved in the kroC100 instance is shown in Figure 5.2. The new solution does not show the behavior of first servicing nodes closer to the starting point as strongly, which contributes to improving overall latency by about 12.5%. It is noteworthy that the longest arcs traversed in the solution occur in the last several steps of the route, and are therefore not compounded for many subsequent customers.

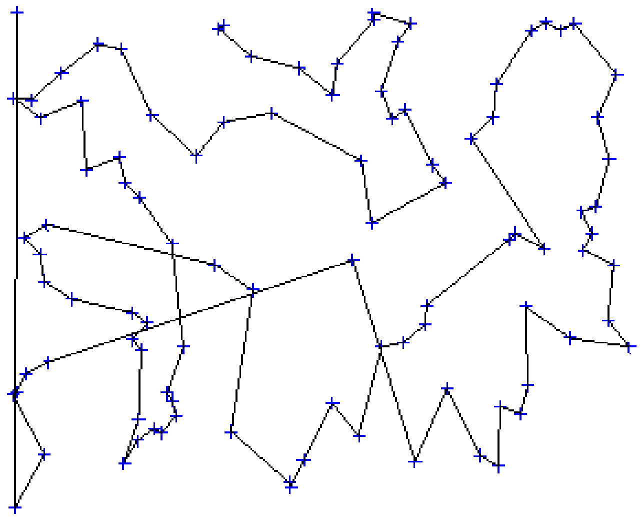


Figure 5.2: Routing solution for kroC100 using insertion with every other ordered node

As a result of this demonstrated impact and proposed modification, the following rule is applied to the insertion heuristic for the MDMLPI: if the number of nodes in the problem instance divided by the number of vehicles available is 60 or fewer, the original ordering of nodes for consideration is used. If this number is more than 60, then the two-pass approach considering every other ordered node is implemented. With this in effect, the original ordering will be used for the five smallest instances (*a*, *b*, *c*, *g*, and *h*) while the modification of using every other sorted node will be applied to the five larger instances (*d*, *e*, *f*, *i*, and *j*).

The value of 60 used as the threshold for the above rule was selected for two reasons. Firstly, calculations were done to evaluate what threshold value would achieve the best average performance on the above data. The range from approximately 50 to 60 customers showed the best results. Using a cutoff in this area seems to capitalize on the benefits of each approach and, if this rule were applied above, would achieve a smaller percentage error of around 5.5%. Thus, this is used as an indicator for the performance of the full problems to be tested next. The number of nodes divided by the number of vehicles, which is used for sorting in comparison to this threshold value, is as follows for each of the ten test instances: 12, 24, 36, 64, 80, 96, 18, 36, 72, and 96. Therefore, any threshold within the discussed range will achieve identical results and the value of 60 was selected arbitrarily among this range.

CHAPTER 6: PERFORMANCE EVALUATION

6.1 Previous VRP Solutions

For the ten test cases, the latency and weighted latency have been calculated from the best-known solutions available online at http://neumann.hec.ca/chairedistributique/data and are shown in Table 6.1. These solutions come from solving the identical network of nodes, depots, and intermediate depots that is presented here, but with an objective to minimize the total route length among all vehicles. The calculated latencies will be useful in comparing the results of the MDMLPI to demonstrate the benefits of using a minimum latency objective.

Table 6.1: Results and calculated latency values from previous VRP solutions

Instance	Best Route Length	Latency of Best Solution	Weighted Latency of Solution
a	997.94	6,101	86,447
b	1,307.28	17,297	231,570
c	1,747.61	42,083	533,030
d	1,871.42	63,291	803,650
e	1,942.85	75,877	1,048,900
f	2,284.35	129,260	1,653,200
g	1,162.58	13,880	187,810
h	1,587.37	39,215	566,720
i	1,972.00	76,514	952,700
j	2,294.06	124,890	1,621,000

6.2 Minimize Latency

We first examine the results from the MDMLPI using the objective of minimizing latency. This will be done by evaluating the latencies achieved in the benchmark, initial, and adjusted solutions, followed by an evaluation of the standard deviations in each. The same organization will be used in the following section for the objective of minimizing weighted latency.

For the tables presented below, the following notation for results is defined:

VRP	Solutions from the previous solutions using a VRP objective
NN	Solutions from the nearest neighbor benchmark
A	Solutions from the auction heuristic
A2	Solutions from the auction heuristic with the second-phase adjustment applied
I	Solutions from the insertion heuristic
I2	Solutions from the insertion heuristic with the second-phase adjustment applied
$L(i)$	Sum of latencies for solution using approach i
$\sigma(i)$	Standard deviation of latencies for solution using approach i
$L_W(i)$	Sum of weighted latencies for solution using approach i
$\sigma_W(i)$	Standard deviation of weighted latencies for solution using approach i

For example, $L(I2)$ will be used to refer to the total latency for an instance when the insertion technique with second-phase adjustment is used. Similarly, let

$$\frac{L_W(NN) - L_W(A)}{L_W(A)}\%$$

be used to denote the percentage improvement of the total weighted latency when using the auction heuristic over the nearest neighbor benchmark.

6.2.1 Evaluation of Latencies

Benchmark Instances

Firstly, the results from the benchmarking solutions are shown in Table 6.2. As expected, the nearest neighbor policy performs reasonably well, improving upon the results of using a VRP objective in each of the ten instances.

Table 6.2: Latencies of benchmarking solutions

Instance	***L(NN)***	***L(VRP)***
a	6,096	6,101
b	13,245	17,297
c	27,273	42,083
d	52,616	63,291
e	65,971	75,877
f	100,630	129,260
g	10,935	13,880
h	25,822	39,215
i	71,405	76,514
j	104,380	124,890

The average percent reduction of the simple greedy heuristic over the best-known solutions using a VRP objective is 18.92%. Without yet viewing the results of the proposed heuristics, this fact already makes a noteworthy argument for the benefits of MLP objectives. In this case, using one of the simplest and quickest policies to minimize latency can reduce the average customer wait time by up to 35%, which is the reduction in latency for instance *c*, compared to server-oriented objectives like the VRP. This observation does not carry any weight if the problem being solved is, for example, cost minimization for a company's logistics, but is of note in appropriate setting like disaster relief operations. We will next observe the performance of the proposed heuristics that attempt to offer further improvements over the nearest neighbor solutions.

Initial Heuristics

For each of the two initial heuristics in the Table 6.3, the first column presents the latency results and the second calculates the percentage improvement of each over the nearest neighbor (NN) benchmark.

Table 6.3: Latency performance of initial heuristics

Instance	***L(A)***	$\frac{L(NN) - L(A)}{L(NN)}\%$	***L(I)***	$\frac{L(NN) - L(I)}{L(NN)}\%$
a	4,860	20.28	4,676	23.30
b	11,850	10.53	12,306	7.09
c	27,113	0.59	27,029	0.89
d	48,986	6.90	51,478	2.16
e	61,643	6.56	64,993	1.48
f	88,931	11.63	87,668	12.88
g	8,293	24.17	8,286	24.22
h	25,078	2.88	27,128	-5.06
i	61,695	13.60	60,727	14.95
J	94,044	9.90	92,941	10.96
Average		**10.70**		**9.29**

Both heuristics perform well and improve the solutions obtained from NN. Between the two, neither one demonstrates itself as clearly dominating the other. The auction policy obtains a higher average improvement in latency, but the difference is nominal, comparing at 10.70% for auction to 9.29% for insertion. It also reliably improves upon every instances, whereas the insertion heuristic falls short of the NN result for instance h. On the other hand, insertion achieves a better latency than auctioning for 6 out of the 10 instances. These instances are a, c, f, g, i, and j, so from these data there does not seem to be a pattern based on problem size or other characteristics that implies when one approach may be ideal. Therefore, this first set of results shows strong performance from each heuristic and shows that either one will generally be able to result in good latency reductions from the NN benchmark. Further analyses using the second-phase heuristic, comparing standard deviations, and using the weighted latency objective will help distinguish the benefits and drawbacks of each approach.

A visualization of the implications of such latency reductions is presented in Figure 6.1 below. This example considers instance *a* and compares the auction policy to the NN benchmark, although a similar graph could be presented for either heuristic or any problem instance.

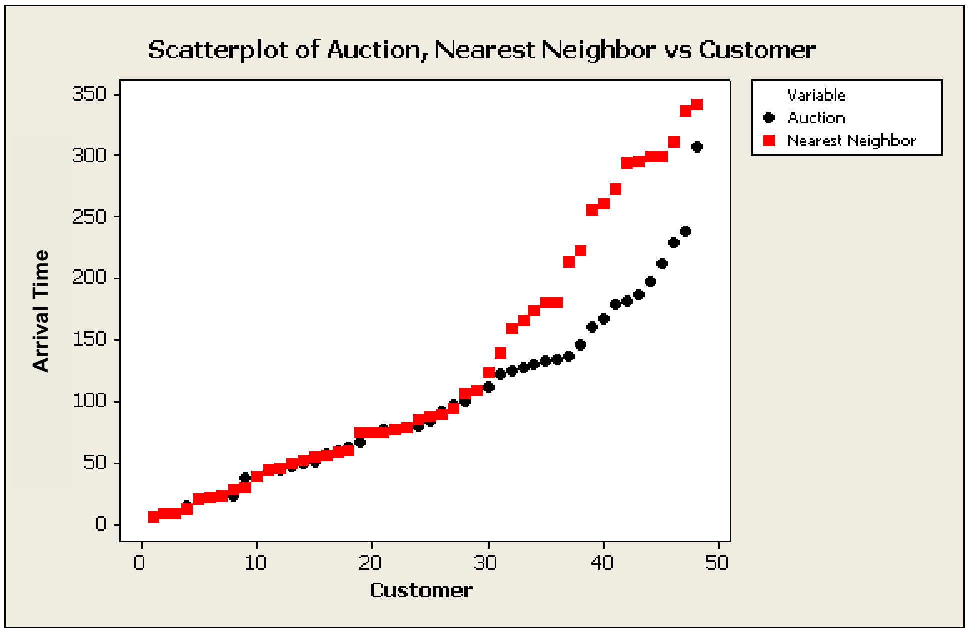


Figure 6.1: Ordered latencies for instance *a* solutions from auction and NN

The figure plots the ordered arrival times at customers from shortest to longest wait time for instance *a*. Both the auction and nearest neighbor policy solutions service about 30 out of 48 customers with very similar wait times. However, the impact of the compounding nature of MLPs is seen as the wait times for the nearest neighbor solution increase quickly after this point. It is among these customers, those that are served later in the routes, that the savings of the auction policy are clearly seen.

Comparing these results to those obtained from the best-known solutions to the MDVRPI for the same instances also demonstrates the benefits of MLP objectives in customer-oriented networks. The improvements that the auction policy shows over the VRP benchmarks, *L(VRP)*, average 28.03% across the ten instances, and for the insertion heuristic this average is 27.05%. Thus, while VRP objectives are valuable in many industry applications, these initial results exhibit that the benefits of MLP objectives in disaster relief and other customer-oriented networks are great.

Second-Phase Adjustment

The second-phase heuristic was applied to the above solutions ten times each to gather information on best and average performance. These results are presented in Tables 6.4 and 6.5 as well as the percentage improvements of the best result over the NN policy.

Table 6.4: Latency performance of the auction policy with the second-phase adjustment applied

Instance	**Best $L(A2)$**	$\frac{L(NN) - \textbf{Best}}{L(NN)}\%$	**Ave $L(A2)$**	$\frac{L(NN) - \textbf{Ave}}{L(NN)}\%$
a	4,781	21.58	4,801	21.24
b	11,542	12.86	11,547	12.82
c	26,044	4.51	26,133	4.18
d	47,800	9.15	48,132	8.52
e	60,363	8.50	60,467	8.34
f	86,462	14.08	86,771	13.77
g	8,180	25.20	8,242	24.63
h	24,567	4.86	24,632	4.61
i	60,301	15.55	60,606	15.12
j	92,641	11.25	92,869	11.03
Average		**12.75**		**12.43**

Table 6.5: Latency performance of the insertion heuristic with the second-phase adjustment applied

Instance	**Best $L(I2)$**	$\frac{L(NN) - \textbf{Best}}{L(NN)}\%$	**Ave $L(I2)$**	$\frac{L(NN) - \textbf{Ave}}{L(NN)}\%$
a	4,598	24.58	4,644	23.81
b	11,379	14.09	11,536	12.90
c	26,068	4.42	26,207	3.91
d	51,283	2.53	51,373	2.36
e	62,999	4.51	63,238	4.14
f	86,472	14.07	86,669	13.87
g	8,154	25.43	8,217	24.85
h	25,602	0.85	25,655	0.65
i	59,747	16.33	59,855	16.17
j	91,318	12.51	91,582	12.26
Average		**11.93**		**11.49**

The average latency reduction from NN for these instances is improved by several percent when the second-phase modification is used to eliminate certain inefficiencies. The adjustment also seems to perform consistently, as the average values across the ten runs for each instance are close in value to the best of the ten runs.

Comparison of results between the two heuristics yields similar discussion as before. When paired with the modification algorithm, both the auction and insertion heuristics are affected similarly. Auctioning maintains a slight margin over insertion in terms of average overall improvements, and both heuristics achieve the better result over the other one for 5 of the 10 instances. The second-phase heuristic was able to improve upon the insertion technique in instance *h* sufficiently such that it now offers improvement in latency (0.65%) over the NN benchmark.

Overall, both heuristics offer great improvements in routing solutions that are further improved when paired with the second-phase modification. Again, this is stronger when compared to previous VRP solutions for these instances. Average improvements for the auction and insertion heuristics over these VRP benchmarks are, respectively, 29.65% and 29.07%.

6.2.2 Evaluation of Fairness

Per the above discussion on fairness, standard deviations were also recorded for benchmarks and heuristic results. Smaller standard deviation among wait times is a strongly desirable property of a routing solution as it increases fairness in the distribution of customer service times. The results in Table 6.6 are presented for the two heuristics when paired with the second-phase adjustment.

Table 6.6: Standard deviations of benchmarks and heuristics with second-phase adjustment

Instance	$\boldsymbol{\sigma(VRP)}$	$\boldsymbol{\sigma(NN)}$	$\boldsymbol{\sigma(A2)}$	$\boldsymbol{\sigma(I2)}$
a	81.79	103.24	69.15	63.81
b	129.74	122.83	93.21	88.09
c	188.47	138.74	126.69	134.11
d	222.23	211.97	188.48	187.52
e	223.03	219.98	202.94	205.17
f	310.31	256.67	226.71	216.68
g	116.39	120.01	77.32	79.33
h	172.01	119.70	112.93	125.87
i	232.03	220.03	206.40	183.95
j	296.37	264.14	239.51	227.81

Both heuristics improve standard deviations in all ten instances from the benchmarks. Interestingly, while the nearest neighbor policy had improved latencies from the VRP solutions in all ten instances, there are a few instances for which it presents a higher standard deviation.

In order to more clearly evaluate the improvements of each heuristic, Table 6.7 presents these same results in terms of percentage improvements from the two benchmarks.

Table 6.7: Percentage improvements in standard deviation of heuristics from the benchmark solutions

	Auction		**Insertion**	
Instance	$\frac{\sigma(NN)-\sigma(A2)}{\sigma(NN)}\%$	$\frac{\sigma(VRP)-\sigma(A2)}{\sigma(VRP)}\%$	$\frac{\sigma(NN)-\sigma(I2)}{\sigma(NN)}\%$	$\frac{\sigma(VRP)-\sigma(I2)}{\sigma(VRP)}\%$
a	32.11	15.45	36.90	21.98
b	24.11	28.16	28.28	32.10
c	8.68	32.78	3.33	28.84
d	11.08	15.19	11.53	15.62
e	7.75	9.01	6.73	8.01
f	11.67	26.94	15.58	30.17
g	35.57	33.57	33.90	31.85
h	5.66	34.35	-5.16	26.82
i	6.20	11.05	16.40	20.72
j	9.32	19.19	13.75	23.13
Average	**15.22**	**22.57**	**16.12**	**23.92**

As with the latency results, these percentages show that both algorithms perform very well. They show strong reductions in standard deviation of wait times averaging around 15% compared to NN and over 20% from VRP solutions. The following graphic in Figure 6.2 demonstrates another impact of this variance in wait times. It again utilizes the case of instance *a* for the auction policy compared to the NN benchmark.

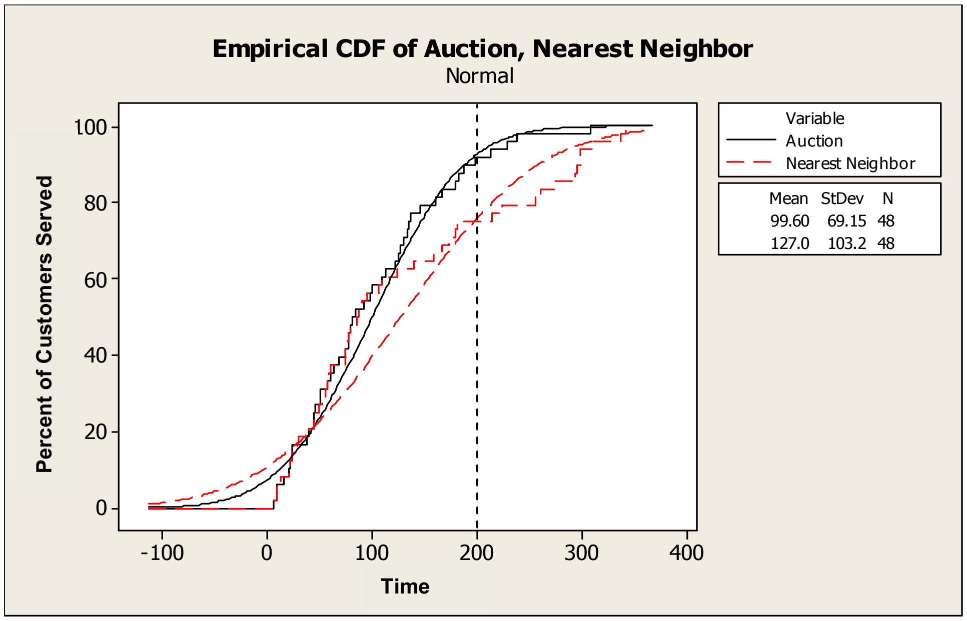


Figure 6.2: Comparison of CDF's for solutions to instance *a* from auction and NN

Figure 6.2 depicts the empirical Cumulative Distribution Functions (CDFs) of the two routing solutions. As was seen in the earlier graph of ordered latencies for these two solutions, the curves look very similar for the first portion of customers in the problem instance. However, there is a sharp separation between the two curves for the latter half of the graph.

Consider customer-focused operations that have some time constraint imposed. In such situations, the CDFs above can be interpreted as the proportion of customers serviced by a certain amount of time, which is along the x-axis. If these two solutions are compared in a setting in which a time constraint of 200 time units exists (as depicted by the vertical line), the solution found using the NN heuristic would be able to serve somewhere around 70% of customers by this time. On the other hand, over 90% of customers would be able to receive service within 200 time units in the solution from the auction policy.

6.3 Minimize Weighted Latency

6.3.1 Evaluation of Latencies

Next, the weighted latency results for benchmarks and heuristics are shown. The weighted latency of a node is the product of the amount of time before it is served and the demand at that node. Weighted latency objectives are of crucial importance when the demand of a node corresponds to the number of people present there. The weighted latency values from the two benchmarking solutions are first shown in Table 6.8.

Table 6.8: Weighted latencies of benchmarking solutions

Instance	$L_W(NN)$	$L_W(VRP)$
a	82,798	86,447
b	158,310	231,573
c	313,870	533,031
d	664,830	803,646
e	927,750	1,048,927
f	1,259,000	1,653,188
g	149,570	187,814
h	351,260	566,722
i	935,570	952,698
j	1,371,900	1,621,018

Similar to the previous case, the NN policy again performs generally well. It averages a 20.52% reduction in weighted latency from the previous VRP solutions, which is comparable to the case without weights (18.92%).

Initial Heuristics

The results of the auction and insertion heuristics are presented next in Table 6.9 along with the percentage reduction in latency from the NN policy, which is already a significant improvement from the VRP benchmark.

Table 6.9: Weighted latency performance of initial heuristics

Instance	$L_W(A)$	$\frac{L_W(NN) - L_W(A)}{L_W(NN)}\%$	WL(I)	$\frac{L_W(NN) - L_W(I)}{L_W(NN)}\%$
a	67,744	18.18	65,503	20.89
b	144,191	8.92	137,281	13.28
c	300,354	4.31	343,464	-9.43
d	603,806	9.18	614,741	7.53
e	848,359	8.56	869,876	6.24
f	1,069,279	15.07	1,124,314	10.70
g	109,060	27.08	111,006	25.78
h	334,586	4.75	339,654	3.30
i	759,817	18.79	767,494	17.97
j	1,180,411	13.96	1,277,805	6.86
Average		**12.88**		**10.31**

The results are similar when compared to the NN benchmark as in the case when latency weights were not considered. The percentage improvements are slightly stronger in this case and, once again, the auction policy has a slightly better overall performance in these instances than insertion. The average improvement across the ten instances for auction is 12.88%. Interestingly, the insertion policy, although averaging 10.31% improvement in latency from NN solutions, again has a noticeably inefficient solution, this time in instance *c*. Here, the result of using this heuristic is 9.43% worse than what is achieved using a simple greedy policy.

The only instances for which the insertion heuristic achieves a stronger solution than auctioning are a, b, and c, three of the smallest. This is not observed, however, in the smallest of the 7-depot instances (which are g through j). The routing solutions for the auction and insertion heuristics in instance a are depicted on the following page. It is interesting to note that this is one of the strongest improvements achieved and both heuristics result in similar weighted latency values, yet the routing solutions differ quite clearly, as shown in Figures 6.3 and 6.4.

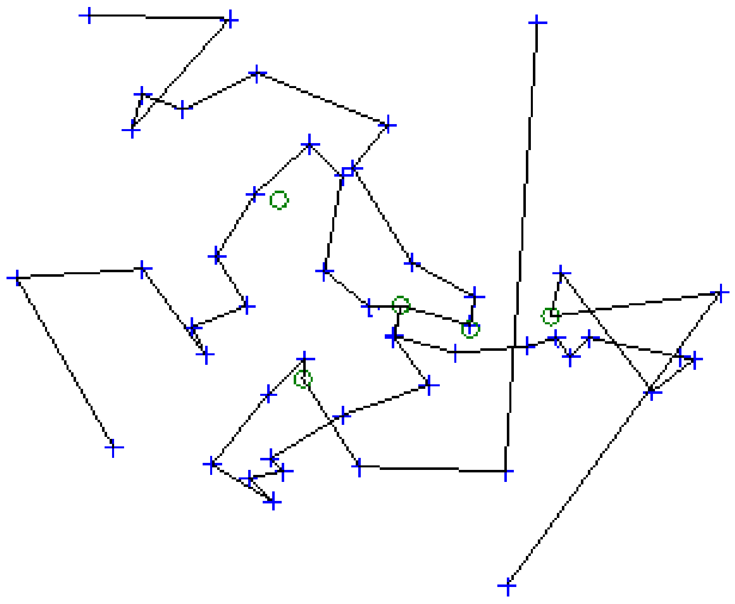


Figure 6.3: Routing solution for instance *a* using auction

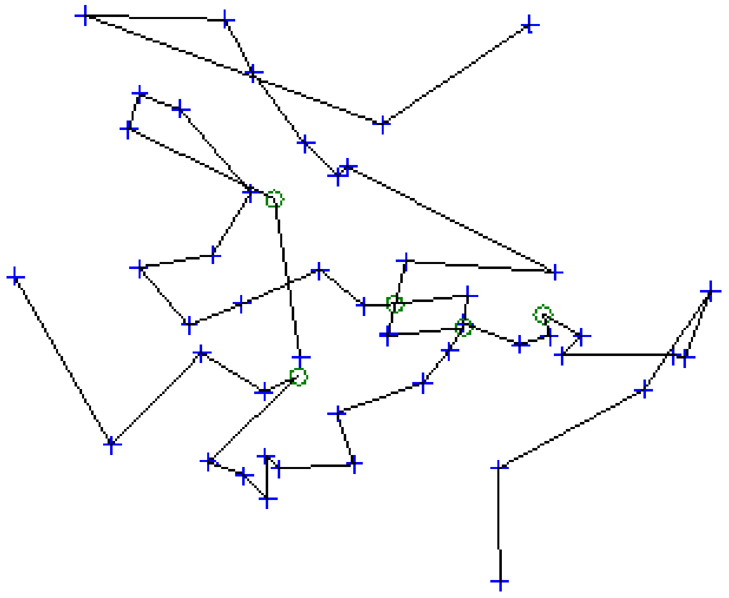


Figure 6.4: Routing solution for instance *a* using insertion

Second-Phase Adjustment

Next, the second-phase heuristic was run on these results in the same manner as before, recording the best and average results among ten runs. The weighted latencies and percentage improvements from NN are presented below for each.

In keeping with previous observations so far, improvements are made for the average latencies across all ten instances by several percent. The results presented in Tables 6.10 and 6.11 show that the adjustment procedure seems to affect both the auction and insertion heuristics about equally. Additionally, it is noteworthy that the best and average weighted latencies in all cases are very close to each other, highlighting the consistent performance of the second-phase heuristic.

Table 6.10: Weighted latency performance of the auction policy with second-phase adjustment

Instance	**Best $L_W(A2)$**	$\frac{L_W(NN) - \text{Best}}{L_W(NN)}\%$	**Ave $L_W(A2)$**	$\frac{L_W(NN) - \text{Ave}}{L_W(NN)}\%$
a	65,996	20.29	66,575	19.59
b	140,726	11.11	141,063	10.89
c	292,767	6.72	292,920	6.67
d	583,271	12.27	585,014	12.01
e	801,862	13.57	804,192	13.32
f	1,018,783	19.08	1,030,664	18.14
g	104,055	30.43	104,315	30.26
h	318,142	9.43	320,255	8.83
i	735,971	21.33	737,234	21.20
j	1,130,438	17.60	1,134,011	17.34
Average		**16.18**		**15.82**

Table 6.11: Weighted latency improvements of the insertion heuristic with second-phase adjustment

Instance	**Best $L_W(I2)$**	$\frac{L_W(NN) - \textbf{Best}}{L_W(NN)}\%$	**Ave $L_W(I2)$**	$\frac{L_W(NN) - \textbf{Ave}}{L_W(NN)}\%$
a	62,383	24.66	63,235	23.63
b	129,591	18.14	129,882	17.96
c	315,351	-0.47	317,635	-1.20
d	587,146	11.68	589,149	11.38
e	837,019	9.78	839,628	9.50
f	1,061,714	15.67	1,069,217	15.07
g	110,325	26.24	110,878	25.87
h	321,881	8.36	324,213	7.70
i	723,654	22.65	728,533	22.13
j	1,167,436	14.90	1,183,927	13.70
Average		**15.16**		**14.57**

Previously, when the insertion heuristic performed worse than the NN benchmark in instance *h* for the case without weights, the second-phase adjustment was able to offer sufficient improvement on the solution to surpass the NN results. In this case, however, the best result achieved from insertion with rearranging for instance *c* still does not achieve reduced latency from NN. This instance aside, both the auction and insertion heuristics seem to perform comparably, but auctioning may have the upper hand in terms of variance and consistency.

6.3.2 Evaluation of Fairness

When comparing fairness in weighted latencies, customer demand is also considered in latency standard deviations. Table 6.12 shows the weighted standard deviations of both benchmarks and the best result from each heuristic with the second-phase adjustment already applied.

Table 6.12: Weighted standard deviations of benchmarks and heuristics after second-phase adjustment

Instance	$\sigma_W(VRP)$	$\sigma_W(NN)$	$\sigma_W(A2)$	$\sigma_W(I2)$
a	1,543.95	1,646.27	1,132.73	1,027.50
b	2,628.41	1,864.47	1,456.68	1,267.73
c	3,611.63	2,032.83	1,725.81	1,948.73
d	3,945.48	3,570.58	2,762.71	2,682.72
e	4,189.62	3,995.88	3,326.16	3,367.62
f	5,680.81	4,439.71	3,012.52	3,340.01
g	2,331.25	2,194.12	1,242.36	1,271.74
h	3,463.54	2,208.18	1,756.79	1,790.98
i	3,949.43	4,244.21	3,173.83	3,076.77
j	5,282.86	4,773.79	3,760.21	3,895.04

Once again, the heuristics perform more or less equally. Weighted standard deviations are clearly improved upon by each for all instances, and neither seems to be clearly fairer than the other. Insertion also improves weighted standard deviation in instance *c* when compared to NN, despite previously having failed to surpass the NN benchmark in terms of weighted latency. The percentage improvements of each heuristic are shown in Table 6.13.

Table 6.13: Percentage improvements of best weighted standard deviations over benchmarks

Instance	**Auction** $\frac{\sigma_W(\mathbf{NN}) - \sigma_W(\mathbf{A2})}{\sigma_W(\mathbf{NN})}\%$	$\frac{\sigma_W(\mathbf{VRP}) - \sigma_W(\mathbf{A2})}{\sigma_W(\mathbf{VRP})}\%$	**Insertion** $\frac{\sigma_W(\mathbf{NN}) - \sigma_W(\mathbf{I2})}{\sigma_W(\mathbf{NN})}\%$	$\frac{\sigma_W(\mathbf{VRP}) - \sigma_W(\mathbf{I2})}{\sigma_W(\mathbf{VRP})}\%$
a	31.19	26.63	37.59	33.45
b	21.87	44.58	32.01	51.77
c	15.10	52.22	4.14	46.04
d	22.63	29.98	24.87	32.01
e	16.76	20.61	15.72	19.62
f	32.15	46.97	24.77	41.21
g	43.38	46.71	42.04	45.45
h	20.44	49.28	18.89	48.29
i	25.22	19.64	27.51	22.10
j	21.23	28.82	18.41	26.27
Average	**25.00**	**36.54**	**24.59**	**36.62**

6.4 Computation Times

Finally, the average computation time of each of these heuristics is presented in Table 6.14. Each heuristic performs in reasonable time, with the auction policy being faster than the insertion technique under the settings selected for this research.

Table 6.14: Average computation time (s) for each heuristic

	Minimize Latency			**Minimize Weighted Latency**		
Instance	**Auction**	**Insertion**	**Rearranging**	**Auction**	**Insertion**	**Rearranging**
a	21.0	29.3	3.8	32.6	30.8	7.0
b	87.8	110.6	25.6	136.6	116.5	24.8
c	214.0	306.4	79.4	329.5	327.1	76.3
d	414.2	699.0	189.6	638.1	737.3	183.1
e	712.1	1,437.5	395.1	1,086.5	1,431.9	453.6
f	1,095.8	2,662.4	740.4	1,695.6	2,579.5	724.8
g	46.9	51.3	11.9	75.2	51.6	11.4
h	213.1	340.1	82.2	337.2	334.4	74.5
i	531.7	1,073.4	312.3	845.3	1,126.0	267.3
j	1,055.3	2,937.4	808.5	1,733.4	2,798.5	678.0

CHAPTER 7: CONCLUSIONS AND FUTURE WORK

7.1 Summary of Results

This thesis introduces a new problem to the routing literature called the Multi-Depot Minimum Latency Problem with Inter-Depot Routes (MDMLPI). Similar variants have been studied previously, including a VRP equivalent of this system as well as related multi-depot versions of the MLP. However, this is the first presentation of a multi-depot MLP that allows vehicles to refill at a selection of intermediate supply depots along their routes. This is, in fact, one of the literature gaps and future research recommendations identified in Moshref-Javadi & Lee (2013).

To begin discussion on this problem, an original mathematical model for it is formulated and two heuristic approaches are proposed. When validated on small problem instances and run compared to two benchmarking solution sets, both heuristic algorithms show strong performance and significantly improved objective function values. The improvements from each are comparable and both heuristics seem appropriate for obtaining strong routing solutions in such settings and show promise for future expansion. These improvements are consistent for both minimum latency and minimum weighted latency objectives as well as for standard deviation of service times, which is recorded to evaluate fairness in routing solutions.

A regular focus is maintained throughout on applications of such routing problems and solution approaches to disaster relief routing networks. The ability of these approaches to greatly reduce average latency and variance in latency carries strong implications for such applications. As the number of natural disasters continues to climb, a subset of literature focuses more and more on improving and applying effective routing solutions to these settings. This work helps to sharpen this focus with a demonstration of what great improvements are possible and with a step forward to advance further development of relevant problem variants.

7.2 Contributions

In short, the following contributions are made to further discussion on the class of MLPs and their application to disaster relief operations:

1. A new problem, the MDMLPI, is developed in which multiple vehicles service an area and are able to stop at a selection of intermediate supply depots along the way
2. An IP formulation is proposed for the MDMLPI
3. A heuristic based on an auction policy in which customers bid to be the next stop on a vehicle's route is proposed
4. A second heuristic, referred to as insertion, is presented as a centralized alternative in which customers are added one-by-one to minimize impact on current latency
5. A simple second-phase adjustment is also introduced to be run on either of the proposed algorithms to identify and remove some inefficient routing results
6. Both algorithms are paired with the second-phase adjustment and run on test problem instances for comparison with benchmarking solutions
7. All solutions attained via these heuristics result in great reductions in latency for both minimum latency and minimum weighted latency objectives
    a. Compared to a nearest neighbor (NN) benchmark, improvements of up to 12% average reduction in latency and up to 16% average reduction for weighted latency are achieved
    b. Compared to previous solutions for the VRP equivalent to this problem, average reductions of around 30% are achieved for both objectives
8. Solutions also greatly reduce standard deviations among customer service times in the instances tested

   a. Compared to the NN benchmark, average reductions in standard deviation for the minimum latency and minimum weighted latency objectives, respectively, are up to 15% and 25%
   b. Compared to previous VRP solutions, they are around 22% and 36%, respectively

9. Discussion on fairness in disaster relief routing and evaluation of standard deviation as a measure of fairness is also included throughout to advance this component of the literature and urge fellow researchers to continue developing such evaluation methods
10. Finally, future research directions that arise from similar problems to the MDMLPI are recommended based on the results herein

7.3 Future Research Directions

Efficient routing of supplies after natural disasters is one of the most prominent application areas for this work. The presented algorithms expand this customer-focused discussion that has garnered recent attention among routing problem and operations research literature. This work emphasizes the importance of evaluating a measure of fairness in addition to a latency objective and urges the research community to continue developing solution approaches for new variants of the MLP that will help improve the practice of serving customer needs.

Several interesting directions for future research are apparent through this study, including improved approaches to the problem presented herein, introduction of other variants that accurately reflect what is done in practice in customer-oriented networks, and more. A few select directions are discussed below.

7.3.1 Multiple Objectives

Firstly, one of the most important directions for continuation of this work is the use of multiple objectives. The problem studied in this thesis seeks to minimize (weighted) latencies for customers without considering the cost to the service provider. However, this may provide unrealistic solutions for real-world applications in which organizations are limited in spending ability. Thus, this research is a first step for such a problem network that can be expanded to consider the multiple objective cases that can balance customer focus and reasonable expense.

This may involve defining one primary and one secondary objective function or a weighted combination of multiple objectives. Averbakh & Berman (1995) used this idea with the introduction of the Sales-Delivery Man Problem (SDMP), which has a TSP primary objective and a Delivery Man Problem (i.e. MLP) secondary objective. In other words, a set of optimal solutions to the TSP objective are presented, and the one among those with the minimum latency is selected. The study applies this problem to tree and cactus networks. A similar approach would be to weight two objectives according to the particular needs of the situation and acquire a set of Pareto-optimal solutions from which to choose. Nolz, Doerner, & Hartl (2010) study a similar multi-objective model for distribution of water in relief efforts.

Multiple-objective problems can provide meaningful insights for organizations involved in relief efforts who desire to make the most impact with the money they have available. In some cases, a group's budget may be defined by limited funding, grants, or donations, and optimal MLP solutions may not be feasible if costs are too high. As an illustration, imagine the logistics of a small relief contributor that aims to minimize wait times while easing the financial burden. A multi-objective balance between customer- and server-oriented routing may use an objective function such as:

$$min(\theta_1 MLP + \theta_2 TSP)$$

where θ_1 and θ_2 are user-selected weights of each objective such that they sum to 1. This gives the organization flexibility in the relative weights of each objective to cater to the unique qualities of the environment. It also allows them to consider multiple alternatives and select among Pareto optimal solutions when they are presented, as opposed to requiring an advanced decision. Using this objective, there is sound justification for routing decisions that seek to best serve a balance of quick response and effective use of funding.

7.3.2 Corporate Involvement

A similar objective may also assist for-profit companies seeking to use their resources to alleviate local impacts without compromising their own financial stability. Kuo & Means (2012) discuss the role of companies in such positions. Other analyses and case studies highlight examples of this setting (e.g. Ergun, Heier Stamm, Keskinocak, & Swann, 2010).

Corporations may benefit from further developed approaches using a combination of objective functions. Here, the issue might not come from limited funding as may be seen in the case of relief agencies and nonprofits, but rather from the need to maintain a profitable business model. This will undoubtedly put a constraint on how much companies are willing to spend in total aid.

7.3.3 Improved Heuristics

In this work, an auction policy and insertion technique are proposed to provide strong solutions to the MDMLPI. As this problem has not been previously studied, benchmarking solutions are used to evaluate the strength of the results. While latency reductions are achieved from the benchmarks and the benefits of minimum latency objectives are clearly demonstrated, different heuristic and approximation approaches can be introduced to further improve result quality.

For example, depot assignment in the auction policy is one potential change. Instead of allowing all available customers to bid, a variation could allow only those customers whose demand would not exceed capacity constraints to bid. The difficulty with this is determining when such a bid should be serviced and when a depot stop would be better. For example, if there is only one node that has small enough demand to be satisfied and it is very far away from the vehicle, it would make sense to refill supply rather than traversing a long, inefficient arc to serve just one more customer before refilling. Relative weights between customer and depot bids would need to be developed to achieve best performance in such situations.

Varying the characteristics of the insertion technique and depot assignment may also lead to further reduced latencies. Additionally, depot assignment in both cases is based on proximity between the vehicle-depot pair alone with little foresight. This is also a potential direction for future improvements on these heuristics.

Literature on similar problems in the classes of Traveling Salesman Problems and Vehicle Routing Problems is more fully developed than the MLP and points to many potential approaches to the MDMLPI based on success with route-minimizing objectives. Some examples, while many more exist, include approaches based on tabu search (Aras, Aksen, & Tuğrul Tekin, 2011; Crevier et al., 2007; Renaud, Laporte, & Boctor, 1996), genetic algorithms (Ho, Ho, Ji, & Lau, 2008), dynamic programming (Gromicho, van Hoorn, Kok, & Schutten, 2012; Held & Karp, 1962; Tatarakis & Minis, 2009), and large neighborhood search (Ribeiro & Laporte, 2012).

Alternatively, the approaches used here may also be able to provide strong results if adapted for route minimizing objectives. Several of the previous best-known results to the MDVRPI that were used for benchmarking were inadvertently improved upon for several instances. Adapting these models with the objective of minimizing travel time could be able to contribute to the VRP literature as well, although that body of work is much more developed and has different applications than those focused on here.

7.3.4 Collaborative Depots

This research also gives rise to discussion of several possible characteristics of supply depots that may have interesting effects and applications. For example, in disaster relief settings where multiple organizations may be present, effective collaboration can be represented by a variant of the MLP in which multiple depots each start their own set of vehicles and allow them to refill at the supply depots of other organizations. Such a system would look identical to the one used in this research with two differences: 1) there would be no main depot designated and 2) each depot would be able to be the starting point for one or more routes.

Collaboration between the different entities involved in relief operations is an ideal characteristic that is difficult to achieve in practice. The diversity of the groups involved presents many challenges that need to be overcome for effective collaboration (Dolinskaya et al., 2011; Van Wassenhove, 2005). Simulation and mathematical modeling of such environments with the involvement of multiple organizations have potential to demonstrate what effective collaboration can look like and to set expectations for what relief workers can hope to accomplish if improved cooperation is achieved.

Modeling the routing networks of these groups with shared supply depots is one step that can be taken in that direction. Introductory tests were run for this variant using the heuristics that were proposed in this work and show promising results comparable to those achieved herein.

7.3.5 Job Scheduling

Another application area, as alluded to in the literature survey, is job scheduling problems, which can equate to certain classes of routing problems. The results achieved here may also carry implications for job scheduling equivalents. The property that routing problems have of variable travel time to nodes depending on starting location is paralleled by sequence dependent setup times among jobs that need to be processed on machines. If all qualities of the MDMLPI are maintained except for capacity constraints and the presence of depots, it equates to the problem of having multiple machines that collectively need to process a set of jobs where the setup time of a job depends also on the job that precedes it on the machine. An interesting research question would be to see how the auction and insertion heuristics would compare to other developed algorithms when applied to such a setting.

LIST OF REFERENCES

LIST OF REFERENCES

Abeledo, H., Fukasawa, R., Pessoa, A., & Uchoa, E. (2013). The Time Dependent Traveling Salesman Problem: Polyhedra and Algorithm. *Mathematical Programming Computation*, *5*(1), 27–55.

Afrati, F., Cosmadakis, S., Papadimitriou, C. H., Papageorgiou, G., & Papakostantinou, N. (1986). The Complexity of the Traveling Repairman Problem. *Informatique Theorique et Applications*, *20*(1), 79–87.

Altay, N., & Green, W. G. (2006). OR/MS Research in Disaster Operations Management. *European Journal of Operational Research*, *175*(1), 475–493.

Aras, N., Aksen, D., & Tuğrul Tekin, M. (2011). Selective Multi-Depot Vehicle Routing Problem with Pricing. *Transportation Research Part C: Emerging Technologies*, *19*(5), 866–884.

Archer, A., Levin, A., & Williamson, D. P. (2008). A Faster, Better Approximation Algorithm for the Minimum Latency Problem. *Society for Industrial and Applied Mathematics Journal on Computing*, *37*(5), 1472–1498.

Averbakh, I., & Berman, O. (1995). Sales-Delivery Man Problems on Treelike Networks. *Networks*, *25*(2), 45–58.

Balcik, B., Beamon, B. M., Krejci, C. C., Muramatsu, K. M., & Ramirez, M. (2010). Coordination in Humanitarian Relief Chains: Practices, Challenges and Opportunities. *International Journal of Production Economics*, *126*(1), 22–34.

Bianco, L., Mingozzi, A., & Ricciardelii, S. (1993). The Traveling Salesman Problem with Cumulative Costs. *Networks*, *23*, 81–91.

Blum, A., Chalasani, P., Coppersmith, D., Pulleyblank, B., Raghavan, P., & Sudan, M. (1994). The Minimum Latency Problem. *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing - STOC '94*, 163–171.

Campbell, A. M., Vandenbussche, D., & Hermann, W. (2008). Routing for Relief Efforts. *Transportation Science*, *42*(2), 127–145.

Costa, S. R. A. Da, Campos, V. B. G., & Bandeira, R. A. D. M. (2012). Supply Chains in Humanitarian Operations: Cases and Analysis. *Procedia - Social and Behavioral Sciences*, *54*, 598–607.

Crevier, B., Cordeau, J.-F., & Laporte, G. (2007). The Multi-Depot Vehicle Routing Problem with Inter-Depot Routes. *European Journal of Operational Research*, *176*(2), 756–773.

Day, J. M., Melnyk, S. A., Larson, P. D., Davis, E. W., & Whybark, D. C. (2012). Humanitarian and Disaster Relief Supply Chains: A Matter of Life and Death. *Journal of Supply Chain Management*, *48(2)*, 21–36.

Dolinskaya, I. S., Shi, Z. E., & Smilowitz, K. R. (2011). Decentralized Approaches to Logistics Coordination in Humanitarian Relief. *Proceedings of the 2011 Industrial Engineering Research Conference*.

Ergun, Ö., Heier Stamm, J. L., Keskinocak, P., & Swann, J. L. (2010). Waffle House Restaurants Hurricane Response: A Case Study. *International Journal of Production Economics*, *126*(1), 111–120.

Galindo, G., & Batta, R. (2013). Review of Recent Developments in OR/MS Research in Disaster Operations Management. *European Journal of Operational Research*, *230*(2), 201–211.

Goemans, M., & Kleinberg, J. (1998). An Improved Approximation Ratio for the Minimum Latency Problem. *Mathematical Programming*, *82*(1-2), 111–124.

Gromicho, J., van Hoorn, J. J., Kok, A. L., & Schutten, J. M. J. (2012). Restricted Dynamic Programming: A Flexible Framework for Solving Realistic VRPs. *Computers & Operations Research*, *39*(5), 902–909.

Guinet, A. (1993). Scheduling Sequence-Dependent Jobs on Identical Parallel Machines to Minimize Completion Time Criteria. *International Journal of Production Research*, *31*(7), 1579–1594.

Held, M., & Karp, R. M. (1962). A Dynamic Programming Approach to Sequencing Problems. *Journal of the Society for Industrial and Applied Mathematics*, *10*(1), 196–210.

Ho, W., Ho, G. T. S., Ji, P., & Lau, H. C. W. (2008). A Hybrid Genetic Algorithm for the Multi-Depot Vehicle Routing Problem. *Engineering Applications of Artificial Intelligence*, *21*(4), 548–557.

Howden, M. (2009). How Humanitarian Logistics Information Systems Can Improve Humanitarian Supply Chains: A View from the Field. In *Proceedings of the 6th International ISCRAM Conference*. Gothenburg, Sweden.

Jobe, K. (2011). Disaster Relief in Post-Earthquake Haiti: Unintended Consequences of Humanitarian Volunteerism. *Travel medicine and infectious disease*, *9*(1), 1–5.

Ke, L., & Feng, Z. (2013). A Two-Phase Metaheuristic for the Cumulative Capacitated Vehicle Routing Problem. *Computers & Operations Research*, *40*(2), 633–638.

Koulamas, C., & Kyparisis, G. J. (2008). Single-Machine Scheduling Problems with Past-Sequence-Dependent Setup Times. *European Journal of Operational Research*, *187*(3), 1045–1049.

Kovács, G., & Spens, K. M. (2011). Trends and Developments in Humanitarian Logistics – A Gap Analysis. *International Journal of Physical Distribution & Logistics Management*, *41*(1), 32–45.

Kuo, S. S., & Means, B. (2012). Corporate Social Responsibility After Disaster. *Washington University Law Review*, 1–47.

Lee, Y. H., & Pinedo, M. (1997). Scheduling Jobs on Parallel Machines with Sequence-Dependent Setup Times. *European Journal of Operational Research*, *100*, 464–474.

Liao, C.-J., & Juan, H.-C. (2007). An Ant Colony Optimization for Single-Machine Tardiness Scheduling with Sequence-Dependent Setups. *Computers & Operations Research*, *34*(7), 1899–1909.

Luo, X., Wang, C., Liu, Z., Qu, R., Shu, Q., & Notation, A. (2005). Lower and Upper Bounds for Single Machine Problem with Sequence Dependent Setup to Minimize Maximum Tardiness. In *International Conference on Service Systems and Service Management* (pp. 375–378). Chongqing, China.

Martinez, A. J. P., & Wassenhove, L. N. V. A. N. (2010). Using OR to Support Humanitarian Operations: Learning from the Haiti Earthquake. *INSEAD Working Paper Series*.

McEntire, D. A. (1999). Issues in Disaster Relief: Progress, Perpetual Problems and Prospective Solutions. *Disaster Prevention and Management*, *8*, 351–361.

Méndez-Díaz, I., Zabala, P., & Lucena, A. (2008). A New Formulation for the Traveling Deliveryman Problem. *Discrete Applied Mathematics*, *156*(17), 3223–3237.

Moshref-Javadi, M., & Lee, S. (2013). A Taxonomy to the Class of Minimum Latency Problems. In *Proceedings of the 2013 Industrial and Systems Engineering Research Conference*. San Juan, Puerto Rico.

Ngueveu, S. U., Prins, C., & Wolfler Calvo, R. (2010). An Effective Memetic Algorithm for the Cumulative Capacitated Vehicle Routing Problem. *Computers & Operations Research*, *37*(11), 1877–1885.

Nolz, P. C., Doerner, K. F., & Hartl, R. F. (2010). Water Distribution in Disaster Relief. *International Journal of Physical Distribution & Logistics Management*, *40*(8/9), 693–708.

Renaud, J., Laporte, G., & Boctor, F. F. (1996). A Tabu Search Heuristic for the Multi-Depot Vehicle Routing Problem. *Computers & Operations Research*, *23*(3), 229–235.

Ribeiro, G. M., & Laporte, G. (2012). An Adaptive Large Neighborhood Search Heuristic for the Cumulative Capacitated Vehicle Routing Problem. *Computers & Operations Research*, *39*, 728–735.

Sahni, S., & Gonzalez, T. (1976). P-Complete Approximation Problems. *Journal of the ACM*, *23*(3), 555–565.

Salehipour, A., Sörensen, K., Goos, P., & Bräysy, O. (2011). Efficient GRASP+VND and GRASP+VNS metaheuristics for the traveling repairman problem. *4OR*, *9*(2), 189–209.

Silva, M. M., Subramanian, A., Vidal, T., & Ochi, L. S. (2012). A Simple and Effective Metaheuristic for the Minimum Latency Problem. *European Journal of Operational Research*, *221*(3), 513–520.

Sioud, A., Gravel, M., & Gagné, C. (2012). A Hybrid Genetic Algorithm for the Single Machine Scheduling Problem with Sequence-Dependent Setup Times. *Computers & Operations Research*, *39*(10), 2415–2424.

Sitters, R. (2002). The Minimum Latency Problem is NP-Hard for Weighted Trees. *Lecture Notes in Computer Science*, *2337*, 230–239.

Sitters, R. (2013). Polynomial Time Approximation Schemes for the Traveling Repairman and Other Minimum Latency Problems, 1–23.

Tabbara, L. (2008). *Emergency Relief Logistics: Evaluation of Disaster Response Models Based on Asian Tsunami Logistics Response*. Oxford Brookes University.

Tan, K.-C., Narasimhan, R., Rubin, P. a, & Ragatz, G. L. (2000). A comparison of four methods for minimizing total tardiness on a single processor with sequence dependent setup times. *Omega*, *28*(5), 609.

Tatarakis, a., & Minis, I. (2009). Stochastic Single Vehicle Routing with a Predefined Customer Sequence and Multiple Depot Returns. *European Journal of Operational Research*, *197*(2), 557–571.

Van Wassenhove, L. N. (2005). Humanitarian Aid Logistics: Supply Chain Management in High Gear. *Journal of the Operational Research Society*, *57*(5), 475–489.

Wang, X., & Tang, L. (2010). A Hybrid Metaheuristic for the Prize-Collecting Single Machine Scheduling Problem with Sequence-Dependent Setup Times. *Computers & Operations Research*, *37*(9), 1624–1640.

Wu, B. Y. (2000). Polynomial Time Algorithms for Some Minimum Latency Problems. *Information Processing Letters*, *75*(5), 225–229.

Wu, B. Y., Huang, Z.-N., & Zhan, F.-J. (2004). Exact Algorithms for the Minimum Latency Problem. *Information Processing Letters*, *92*(6), 303–309.

Ying, K.-C. (2012). Scheduling Identical Wafer Sorting Parallel Machines with Sequence-Dependent Setup Times Using an Iterated Greedy Heuristic. *International Journal of Production Research*, *50*(10), 2710–2719.

Zhu, X., & Wilhelm, W. E. (2006). Scheduling and Lot Sizing with Sequence-Dependent Setup: A Literature Review. *IIE Transactions*, *38*(11), 987–1007.

APPENDICES

Appendix A: Mathematical Model of the MDMLPI

Consider a set of nodes $V = \{1, 2,\ldots, n\}$, which is the inclusive set composed of customers, a main depot, and multiple intermediate depots. Let the subsets V_C, V_D, and V_I, define the set of customers, main depot, and intermediate depots, respectively. An arc between two nodes i and j in V has an associated travel time of c_{ij}. All arcs are symmetrical, implying that this is the equivalent distance associated with traveling from node j to node i. We also define $R = \{1, 2,\ldots, m\}$ as the set of vehicles servicing these nodes. If the arc from customer i to customer j is traversed by vehicle k, the value of a binary variable x_{ijk} is equal to 1, otherwise it is 0.

All vehicles have a uniform maximum carrying capacity, designated Q, and the need of this commodity required at a given customer i is denoted q_i. Let π_{ik} represent the wait time for customer i being serviced by vehicle k, which is calculated as the sum of all travel times on route k that precede customer i. Similarly, let μ_{ik} represent the cumulative amount of stock required from vehicle k when it services customer i, which is the sum of all customer need on the route up to and including customer i. The model and constraints are then defines as follows:

$$\textit{minimize} \quad \sum_{k=1}^{m} \sum_{i \in V_C} \pi_{ik} \quad or \quad \sum_{k=1}^{m} \sum_{i \in V_C} q_i \pi_{ik} \tag{1}$$

$$\textit{subject to} \quad \sum_{k=1}^{m} \sum_{j=1}^{n} x_{ijk} = 1 \qquad \forall i \in V_C \tag{2}$$

$$\sum_{k=1}^{m} x_{ijk} \leq 1 \qquad \forall i \in V, \ \forall j \in V \tag{3}$$

$$\sum_{j=1}^{n} x_{ijk} = 1 \qquad \forall i \in V_D, \ \forall k \in R \tag{4}$$

$$\sum_{i=1}^{n} x_{ijk} = \sum_{i=1}^{n} x_{jik} \qquad \forall j \in V, \ \forall k \in R \tag{5}$$

$$x_{iik} = 0 \qquad \forall i \in V,\ \ \forall k \in R \tag{6}$$

$$\pi_{ik} + c_{ij} - \left(1 - x_{ijk}\right) * M \leq \pi_{jk} \qquad \forall i \in V \backslash V_D,\ \ \forall j \in V,\ \ \forall k \in R \tag{7}$$

$$\mu_{ik} + q_j - \left(1 - x_{ijk}\right) * M \leq \mu_{jk} \qquad \forall i \in V,\ \ \forall j \in V_C,\ \ \forall k \in R \tag{8}$$

$$\mu_{ik} = 0 \qquad \forall i \in V/V_C,\ \ \forall k \in R \tag{9}$$

$$\mu_{ik} \leq Q \qquad \forall i \in V,\ \ \forall k \in R \tag{10}$$

$$x_{ijk} \in [0,1] \qquad \forall i \in V,\ \ \forall j \in V,\ \ \forall k \in R \tag{11}$$

$$\pi_{ik}, \mu_{ik} \geq 0 \qquad \forall i \in V,\ \ \forall k \in R \tag{12}$$

The objective function (1) minimizes cumulative wait times across the system for all customers, not including wait times at the main or any intermediate depots. Constraints (2) ensure that each customer is visited by exactly one vehicle while constraints (3) state that every arc between customers can only be traversed by one vehicle. Constraints (4) guarantee that every vehicle begins at the main depot. Constraints (5) ensure that if a vehicle arrives at a node, it also departs from that node. Constraints (6) state that no arc can be traversed from one entity to itself.

Constraints (7) are adapted from the cumulative wait time formulation used in Ngueveu et al. (2010) and assign wait times at each entity. If the arc from i to j is traversed by vehicle k, the wait time at j equals the wait time at i plus the travel time between them. These constraints also eliminate subtours. If $x_{ijk} = 0$, they are always satisfied because a large number operator, M, is always greater than $\pi_{ik} + c_{ij} - \pi_{jk}$. If $x_{ijk} = 1$, then $\pi_{ik} + c_{ij} \leq \pi_{jk}$ must hold. For a subtour to exist, for example from node 1 to 2 to 3 on vehicle 1, then $\pi_{11} + c_{12} + \pi_{21} + c_{23} + \pi_{31} + c_{31} \leq \pi_{11} + \pi_{21} + \pi_{31}$ and thus $c_{12} + c_{23} + c_{31} \leq 0$ which contradicts the problem definition and cannot occur (Ngueveu et al., 2010). Constraints (8) perform similarly to constraints (7) and calculate the vehicle's load at each stop on its route based on its previous stops at customers. Constraints (9) define the required vehicle load at the intermediate depots as zero. Constraints (10) ensure that these vehicle load values are always less than the vehicle capacity Q. Finally, constraints (11) and (12) define the variable types.

Appendix B: Pseudocode for Auction Policy

```
**Comment:** run for all calibrating parameter ($w_1$, $w_2$, $w_3$) combinations
current best solution ← inf;
for $w_1$ ← -0.2 to 0.2 by 0.05
  for $w_2$ ← 0 to 0.5 by 0.05
    for $w_3$ ← 0 to 1 by 1
      counter ← 1;
      while all customers are not yet assigned
        k ← vehicle with shortest current route;
        for all unassigned nodes
          bid(i) ← $(c_{ij})(q_i)^{w1}[\min_{\forall j}(c_{ij})]^{w2}[(\sum_j c_{ij})/n]^{w3}$;
        end
        if need of customer with winning bid < vehicle supply
          routes(k,counter) ← min(bids);
          vehicle supply ← vehicle supply – need of winning customer;
        else
          routes(k,counter) ← $\min_{j \in depot\ set}(c_{ij})$;
          vehicle supply ← full capacity
        end
      counter ← counter + 1;
      end
      if this solution < current best solution
        current best solution ← this solution;
        Record best combination of calibrating parameters;
      end
    end
  end
```

Appendix C: Pseudocode for Insertion Technique

```
**Define** *all_routes* and *best_latencies* to store partial routes and associated latencies
Set best_latencies(1:$H$) ← inf;
sorted ← customers sorted in ascending order by proximity to main depot
**if** n_customers / n_routes > 60
  order = [sorted(1:2:end)  sorted(2:2:end)];
**else**
  order = sorted;
**end**
**for** counter ← 1 to n_customers
   i = order(counter);
   **for** route_number ← 1 to $H$
      Set routes ← all_routes(route_number,:,:);
      **for** j ← 1 to n_vehicles
         Set current_route ← routes(j,:);
         **for** k ← 1 to number of elements in current_route
            try_new_route ← [current_route(1:k)  i  current_route(k+1:end)];
            **if** latency of try_new_route is less than min(best_latencies)
               **Comment:** the next if statement prevents multiple instances of the same
                   partial route (beyond the first step when counter = 1)
               **if** latency value is not already stored in best_latencies or counter = 1
                  Replace min(best_latencies) with latency of try_new_route;
                  Replace partial route corresponding to this latency with new route;
               **end**
            **end**
         **end**
      **end**
   **end**
**end**
```

Appendix D: Pseudocode for Depot Assignment

```
Set current_best ← inf;
for $D$ ← 1 to 10
    for $h$ ← 1 to $H$
        for i ← 1 to the number of vehicles
            Set route_need ← 0;
            for j ← 1 to the number of nodes assigned to vehicle $i$
                if vehicle has enough supply to meet demand at stop $j$
                    Add demand at stop $j$ to route_need;
                else
                    Comment: assign a depot stop among the most recent $D$ stops
                    Set depot_best ← inf;
                    for k ← $\max(j - D, 0)$ to $j - 1$
                        for l ← 1 to the number of depots
                            if latency of the route with depot $l$ inserted at position $k$ < depot_best
                                Store route as the temporary best, set depot_best to this latency;
                            end
                        end
                    end
                end
            end
        end
        if depot_best < current_best
            Set current best to depot_best and store this route as the temporary best;
        end
    end
end
```

Appendix E: Pseudocode for Second-Phase ADJUSTMENT

```
**Comment:** Begin with variable 'routes' from auction solution;
current best solution ← inf;
infeasible ← 0;
order ← rand(number of customers);
for each customer in randomly generated order
    i ← current customer;
    Remove customer i from current position;
    for h ← all routes
        for j ← all stops on current route
            routes(h,j) ← i;
            for the number of stops on the current route
                if current stop is a depot
                    vehicle supply ← full capacity;
                else
                    vehicle supply ← vehicle supply – need of customer i;
                    if vehicle supply < 0
                        infeasible ← 1
                    end
                end
            end
            if this solution < current best solution
                if infeasible = 0
                    current best solution ← this solution;
                end
            end
        end
    end
end
```

Appendix F: Small Test Instances

The instances used for small preliminary evaluation, which are subsets of the larger problems, are shown in full here. For all instances, let x and y denote the coordinates of each numbered node and q represent the customer demand. The depot information is listed after all customer data and can be identified by demand values of 0. The main depot is listed first, denoted D, followed by two intermediate depots, I1 and I2. Note also that for these tests a 'ghost' depot was implemented to allow secondary stops at the main depot if needed. This was denoted as node 10 and contained the exact same qualities as the main depot node.

All small instances used six customers and two intermediate depots. Two vehicles were allowed, each with a capacity of 30. The coordinates of all entities for each instance are detailed below.

Instance *a*:

Node	x	y	q
1	-29.73	64.136	12
2	-30.664	5.463	8
3	51.642	5.469	16
4	-13.171	69.336	5
5	-67.413	68.323	12
6	48.907	6.274	5
D	5.243	22.26	0
I1	-10.442	19.999	0
I2	21.387	17.105	0

Instance *b*:

Node	x	y	q
1	33.588	30.75	4
2	48.828	65.314	12
3	86.176	59.344	3
4	39.27	-33.057	15
5	-23.37	86.853	13
6	48.132	95.593	20
D	-16.357	93.311	0
I1	30.22	24.786	0
I2	32.663	44.73	0

Instance *c*:

Node	x	y	q
1	-55.28	-24.371	9
2	-48.297	53.314	22
3	-49.072	-38.489	10
4	25.311	-18.561	24
5	-24.469	-3.815	25
6	24.591	-17.896	6
D	-10.419	60.364	0
I1	-18.391	1.121	0
I2	24.292	-27.704	0

Instance *d*:

Node	x	y	q
1	-44.629	-55.64	11
2	36.096	64.935	11
3	-25.47	44.58	23
4	-33.954	-73.059	16
5	45.654	35.73	16
6	14.954	-36.719	2
D	-1.477	47.205	0
I1	-12.284	9.158	0
I2	-23.138	48.45	0

Instance *e*:

Node	x	y	q
1	65.991	-49.829	10
2	-36.938	-36.743	25
3	-2.734	18.774	18
4	31.116	-35.907	16
5	2.789	8.008	14
6	31.152	43.665	7
D	-36.304	-21.307	0
I1	8.682	-10.438	0
I2	0.269	-8.154	0

Instance *h*:

Node	x	y	q
1	-40.289	-42.303	20
2	-64.709	-17.389	10
3	5.06	-14.349	8
4	72.095	20.233	6
5	2.594	-15.002	1
6	-24.176	-72.894	19
D	-13.19	66.498	0
I1	-12.167	-0.086	0
I2	14.233	21.173	0

Instance *f*:

Node	x	y	q
1	-66.174	-37.811	19
2	2.673	-35.223	21
3	38.751	8.618	17
4	62.653	20.667	6
5	60.974	-6.11	6
6	-98.535	-39.532	19
D	8.411	-81.274	0
I1	-2.676	-9.467	0
I2	32.883	-1.779	0

Instance *i*:

Node	x	y	q
1	-41.235	-66.357	3
2	34.064	-59.357	16
3	20.917	-52.582	8
4	-38.538	-37.396	15
5	41.058	22.931	12
6	-54.034	-53.131	13
D	8.099	80.725	0
I1	3.871	-2.597	0
I2	38.791	22.443	0

Instance *g*:

Node	x	y	q
1	-92.7	-59.18	20
2	71.179	12.543	6
3	31.537	66.638	19
4	-4.694	25.537	10
5	-30.194	67.773	18
6	12.677	-57.471	1
D	-32.355	-20.966	0
I1	10.629	9.326	0
I2	-42.175	-14.554	0

Instance *j*:

Node	x	y	q
1	12.805	1.886	10
2	18.213	1.373	21
3	57.947	-48.779	8
4	-52.429	-84.088	23
5	60.797	-32.593	10
6	23.151	4.205	9
D	25.385	22.986	0
I1	-14.367	-20.341	0
I2	-40.488	-35.864	0