Single-Depot Vehicle Routing Problem for Home Delivery of Pharmaceutical Products

Harms

Follow this and additional works at: https://scholarworks.wmich.edu/masters_theses

Part of the Industrial Engineering Commons, and the Operational Research Commons

Recommended Citation
https://scholarworks.wmich.edu/masters_theses/434
SINGLE-DEPOT VEHICLE ROUTING PROBLEM FOR HOME DELIVERY OF PHARMACEUTICAL PRODUCTS

by

Kimberly Starr Harms

A Thesis submitted to the Graduate College in partial fulfillment of the requirements for the degree of Master of Science in Engineering (Industrial) Engineering
Industrial and Manufacturing Engineering
Western Michigan University
December 2013

Thesis Committee:

Steven Butt, Ph.D., Chair
Azim Houshyar, Ph.D.
Laila Cure, Ph.D.
SINGLE-DEPOT VEHICLE ROUTING PROBLEM FOR HOME DELIVERY OF PHARMACEUTICAL PRODUCTS

Kimberly Starr Harms, M.S.E.

Western Michigan University, 2013

Some retail pharmacies offer home delivery services to establish a competitive advantage in the growing healthcare market. The handling of orders for home delivery includes several internal and external processing and transportation activities. In this work, models were developed for use in a pharmacy delivery system to assist personnel in their daily activities. This work includes standard operating procedures for picking, packing and loading of delivery orders and the assignment and routing of vehicles to deliver customer orders. Results from simulated delivery areas identified interesting tradeoffs when comparing densely populated areas to sparsely populated areas. In addition, recommendations for order handling, transport and stowage are presented in an effort to increase product safety and reduce personnel movements. The resulting spreadsheet tool and data storage system were constructed for operation by pharmacy personnel and integrate with current data management systems. Opportunities for future work were also identified.
Copyright by
Kimberly Starr Harms
2013
ACKNOWLEDGMENTS

The greatest acknowledgement is to my adviser, Dr. Steven Butt. This thesis would not have been possible without his guidance and inspiration. A significant acknowledgement is to Dr. Charles Kinnison for editing. To mention also, the faculty and staff in the Industrial and Manufacturing Engineering Department at Western Michigan University who offer students the commitment and support to each their own goals and aspirations. Finally, I would like to personally acknowledge Tia, Nicole, and Ashley for their endless motivation.

Kimberly Starr Harms
# TABLE OF CONTENTS

ACKNOWLEDGMENTS .................................................................................................................. ii
LIST OF TABLES ......................................................................................................................... vi
LIST OF FIGURES ....................................................................................................................... vii

CHAPTER

I. INTRODUCTION .................................................................................................................... 1
   1.1 Background ..................................................................................................................... 2
   1.2 Product Delivery .......................................................................................................... 3
   1.3 Research Objectives .................................................................................................... 3
   1.4 Case Study .................................................................................................................. 4
   1.5 Summary ..................................................................................................................... 5

II. LITERATURE REVIEW ....................................................................................................... 6
   2.1 Internal Order Handling ............................................................................................. 6
      2.1.1 Picking .................................................................................................................. 7
      2.1.2 Packing ............................................................................................................... 8
      2.1.3 Vehicle Loading ................................................................................................. 9
   2.2 Vehicle Routing .......................................................................................................... 12
      2.2.1 Traveling Salesman Problem .............................................................................. 13
      2.2.2 Multiple Traveling Salesman Problem ............................................................. 17
      2.2.3 Vehicle Routing Problem .................................................................................. 17
   2.3 Summary ..................................................................................................................... 23
<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>III. METHODOLOGY</td>
<td>25</td>
</tr>
<tr>
<td>3.1 Order Processing</td>
<td>26</td>
</tr>
<tr>
<td>3.1.1 Order Entry to Filled Prescriptions</td>
<td>26</td>
</tr>
<tr>
<td>3.1.2 Delivery Personnel Activities</td>
<td>28</td>
</tr>
<tr>
<td>3.2 Routing Methodology</td>
<td>29</td>
</tr>
<tr>
<td>3.2.1 Program Component #1 – “Customer Database”</td>
<td>30</td>
</tr>
<tr>
<td>3.2.2 Program Component #2 – “VRP Petal”</td>
<td>33</td>
</tr>
<tr>
<td>3.3 Summary</td>
<td>36</td>
</tr>
<tr>
<td>IV. ROUTING PROGRAM LOGIC</td>
<td>38</td>
</tr>
<tr>
<td>4.1 Generate Matrix</td>
<td>38</td>
</tr>
<tr>
<td>4.2 VRP Petal Heuristic</td>
<td>44</td>
</tr>
<tr>
<td>4.3 TSP Post-Optimization Heuristic</td>
<td>45</td>
</tr>
<tr>
<td>4.4 Summary</td>
<td>47</td>
</tr>
<tr>
<td>V. ANALYSIS AND RESULTS</td>
<td>48</td>
</tr>
<tr>
<td>5.1 Pick, Pack, Load</td>
<td>48</td>
</tr>
<tr>
<td>5.1.1 Picking and Packing</td>
<td>48</td>
</tr>
<tr>
<td>5.1.2 Loading</td>
<td>54</td>
</tr>
<tr>
<td>5.1.3 Residual Processing</td>
<td>55</td>
</tr>
<tr>
<td>5.2 Routing Program</td>
<td>55</td>
</tr>
<tr>
<td>5.2.1 Code Validation</td>
<td>56</td>
</tr>
<tr>
<td>5.2.2 Processing Time</td>
<td>56</td>
</tr>
</tbody>
</table>
Table of Contents—continued

CHAPTER

5.2.3 Program Comparison (Version 1 vs. Version 2) ............... 57

VI. SUMMARY AND CONCLUSIONS .................................................. 63

6.1 Pick, Pack, and Load .................................................................. 64

6.2 Routing Program ........................................................................ 64

6.3 Review of Research Objectives .................................................. 66

VII. FUTURE RESEARCH .............................................................. 67

APPENDICES

A. Order Flow Chart ......................................................................... 71

B. MICROSOFT EXCEL® VBA CODE – Customer Database .......... 73

C. MICROSOFT EXCEL® VBA CODE – Routing Program .......... 84

D. Delivery Personnel Standard Operating Procedure .................. 122

E. Sample Address Data .................................................................. 124

F. Delivery Picking System ............................................................... 131

BIBLIOGRAPHY .............................................................................. 133
LIST OF TABLES

1. Original TSP Heuristic (Farthest Insertion Algorithm) (Golden et al., 1980) ........................................................................................................ 16
2. Classical Heuristic Methods (Cordeau et al., 2002) .......................... 18
3. Original VRP Heuristic (Petal Algorithm) (Foster and Ryan, 1976) .. 23
4. Matrix Generation Variables and Logic.............................................. 39
5. Master Logic, Calculations, and Bearing Array (SECTION [1]) ........ 41
6. Master Travel Matrix (SECTION [1]) ............................................. 42
7. Routing Matrix (SECTION [2]) .................................................... 43
8. Travel Sub-Matrix (SECTION [3]) ................................................ 44
9. Adapted VRP Heuristic (Petal Algorithm) ....................................... 45
10. Adapted TSP Heuristic (Farthest Insertion Algorithm) ................... 46
11. HangUP® Product Options........................................................... 52
12. Sample Processing Times (Minutes) ............................................. 57
13. Experimental Design..................................................................... 58
14. Exponential Equation Coefficients and R2 Values......................... 60
15. Total Travel Time (Minutes) ....................................................... 60
16. Percent Difference in Total Travel Time (Version 1 vs. Version 2) .... 61
17. Example Spanning Petal Set ........................................................ 66
18. Recognized Cost Factors of Delivery Services............................... 69
# LIST OF FIGURES

1. Life Expectancy (The World Bank Group, 2013) ........................................ 1
3. Research Objectives ......................................................................................... 3
4. Truck Loading Problem (Yüceer and Özakça, 2010) ..................................... 11
5. TSP Model (Golden et al., 1980) ................................................................ 15
6. Subtour-Breaking Constraints (Golden et al., 1980) .................................. 15
7. Linear Program of Vehicle Scheduling Problem (Foster and Ryan, 1976) ......................................................... 19
8. VRP Model (Bodin et al., 1983) ................................................................. 21
9. Order Entry to Filled Prescriptions .............................................................. 27
10. Delivery Personnel Activities ........................................................................ 29
11. Routing Program Information Flow .............................................................. 30
12. Customer Database Summary ....................................................................... 31
13. “New Customer” Worksheet (Customer Database User Interface) ............ 32
14. VRP Petal Summary ...................................................................................... 35
15. “Input” Worksheet (VRP Petal User Interface) ............................................ 36
16. Delivery Personnel SOP (Part 1) .................................................................. 50
17. Delivery Personnel SOP (Part 2) .................................................................. 51
18. Cart Picking/Packing System ...................................................................... 51
19. Delivery Personnel SOP (Part 3) .................................................................. 53
20. Vehicle Loading System ............................................................................... 54
21. Program Processing Times ........................................................................... 59
CHAPTER I

INTRODUCTION

According to the United States Census Bureau (2013), the total population of the World exceeds seven billion people, and is growing. The different populations of the World are each experiencing a positive trend in life expectancy (Figure 1) (World Bank Group, 2013). Bunker (2001) suggests that the growth in life expectancies is in part, due to medical care contributions. The scale of the healthcare system within the United States has grown to more than 16% of the Gross Domestic Product (GDP) (World Bank Group, 2013). Figure 2 shows the healthcare proportion of GDP for the United States and several other regions of the World. Continued growth is projected for the healthcare system as members of the “Baby Boomer” generation increase demand for healthcare products and services (The Brookings Institution, 2013).

![Life Expectancy Graph](image-url)

*Figure 1: Life Expectancy (The World Bank Group, 2013)*
1.1 Background

The healthcare system is broad. To generalize, healthcare includes proactive and reactive activities by skilled caregivers (Merriam-Webster, 2013). In a healthcare system, the preparation and distribution of pharmaceutical products is the responsibility of a pharmacy, where pharmacy operations are managed by a pharmacist (Merriam-Webster, 2013). Pharmacies are present in hospitals, independent retail and retail chain facilities (McKesson Corporation, 2013). The pharmacy sector of the United States healthcare system accounted for more than 270,000 pharmacist positions in 2010 (Bureau of Labor Statistics, 2012). Approximately 43% of these positions were held in pharmacies and drug stores. As the healthcare system grows, pharmacies should anticipate increased demands for products and services.
1.2 Product Delivery

In addition to supplying products, retail companies sometimes offer product delivery services to achieve competitive advantage. Delivery of retail products caters convenience to the customer. Certain customers may want to avoid taking time to retrieve their order while other customers may face limited mobility. The latter is especially important for the sick and elderly, a significant group for retail pharmacy considerations. The delivery of pharmaceutical products is a demanded service that can differentiate a retail pharmacy from competitors.

1.3 Research Objectives

The scale of the healthcare system is extensive globally. As population grows, the healthcare system has the opportunity to contribute to continued improvements in life expectancy. Retail pharmacies constitute one type of pharmacy within the healthcare system. One method for a retail pharmacy to establish competitive advantage in the growing healthcare system is to provide home delivery of pharmaceutical products. Figure 3 presents the specific objectives of this research.

1. Define a generalized model which optimizes delivery service efficiencies.
2. Design a tool that permits delivery personnel to apply theoretical approaches, routinely.
3. Test model results in scenario environments.
4. Acknowledge perspective dimensions for future model development.

*Figure 3: Research Objectives*
Under the assumption that activities related to pharmaceutical product delivery are completed by an internal delivery personnel position, an evaluation of product handling for delivery preparation and execution may provide fundamental insights to design effective delivery protocols. These protocols are instrumental for a company to accommodate market growth. Theoretical approaches can be organized into a general structure which supports delivery operations at a retail pharmacy.

Research objectives remained focused on delivery personnel activities at a retail pharmacy. The parameters of research are defined by a single-depot system with multiple route/vehicle assignments. A sample system was used as a model for development, but the generalized procedure is mindful for adaptation to applications in a variety of retail pharmacies offering product delivery services.

1.4 Case Study

A case study of an existing retail pharmacy in southern Indiana was incorporated into this research to assist with protocol development. The case study pharmacy evaluated currently offers delivery. Products available for delivery include prescriptions, infusions, compounded medications, over the counter (OTC) medications, oxygen and convalescent aid products and devices. The delivery region includes sixteen adjacent counties. In-house product inventory concerns at the facility are negligible as product refill requests are processed and received within one business day. Inventory areas for completed customer orders are Pharmacy bins and refrigerator, Clean Room refrigerator, OTC, and Warehouse.
It is reasonable to expect pharmacy delivery systems to have unique characteristics. Yet, a generalized delivery protocol is proposed which is flexible to customization to satisfy varying system applications. An outline of order flow through the observed pharmacy system is given in Appendix A. *Order Flow Chart*. Time is a standardized metric used to quantify each stage of order processing.

1.5 Summary

Healthcare systems around the World are significant to society, both economically and with regards to society member well-being. As healthcare system demands continue to grow, pharmacies must adapt and prepare for increased demand and competition. This research addresses the general operations associated with retail pharmacy delivery services. Providing standard operating procedures and resources to delivery personnel may avoid or reduce the development of operational inefficiencies.

Research considerations begin with a specific review of existing literature in Chapter 2. The research methodology and routing program logic are discussed in Chapters 3 and 4, respectively. Analysis and results of the proposed systems in retail pharmacy scenarios are detailed in Chapter 5. Finally, research findings and opportunities for future research are acknowledged in Chapters 6 and 7.
CHAPTER II

LITERATURE REVIEW

Consideration of internal and external pharmacy operations presents potential opportunities for delivery system improvements. This literature review is divided into two major sections. The first section of literature considers the system structure for order picking, packing, and loading. Current research is extensive and thorough for these internal operations. The second section presents existing vehicle routing techniques and methodologies. Key techniques relevant to this work are discussed in detail.

2.1 Internal Order Handling

As companies strive to eliminate waste, efficiency measures are applied to internal order handling operations. Bartholdi and Hackman (2011) define warehouse operations by inbound and outbound processes. Inbound processes involve the stages of receiving and handling the product into storage. Outbound processes are the activities of “picking” customer orders from storage for packing and shipping. Heuristics and system technologies are used in solving problems related to inbound and outbound processes. Warehouse processing objectives include maximizing efficiencies and minimizing associated costs (Hostetler and Masel, 2009; Hostetler, 2010; and Bartholdi and Hackman, 2011).

Several large-scale, warehouse-based, retail companies have improved product supply chains by creating more efficient order handling procedures. As with retail, healthcare product handling is part of the overall healthcare supply chain. In an article
published by The McKesson Corporation (2013), an estimate is cited that the healthcare supply chain lags retail supply chains by 10 to 15 years in efficiency measures. This suggests opportunities for improvement in the healthcare supply chain.

For the pharmacy component of the healthcare supply chain, product perishability becomes a concern for both inventory management and product handling. Nahmias (1982) outlines special considerations for perishable products in inventory management. Vila-Parrish et al. (2008) investigated pharmacy inventory policies within hospitals through simulation modeling. The research considered both raw material and finished good perishability of pharmaceutical products. Though important, the inventory policies of a general retail pharmacy are not considered in this research. At the observed pharmacy, inventory replenishment lead time was one business day. A discussion of perishable product handling is found in the following sections.

2.1.1 Picking

Due to travel of personnel, warehouse picking is the most demanding operation of internal order handling (Bartholdi and Hackman, 2011). Order picking can be organized by the scale of distribution; low-volume and high-volume (Bartholdi and Hackman, 2011). In the case of low-volume distribution, management of the system often focuses on finding a route that minimizes travel. High-volume distribution, instead, typically has a focus on the work flow and alleviating bottlenecks.

Large-scale picking operations have driven design and innovation. Andel (2000) points to Amazon.com gains by empowering order pickers with knowledge of picking fundamentals, while incorporating advanced picking technology to achieve further
efficiency. Amazon.com has constructed a business model to support order shipment the same day of order receipt and accommodate the more frequent ordering habits of its growing customer population (McCue, 2012). As with Amazon.com, other warehouse-based retailers, such as Landsend.com, include order picking among the operations contributing to better customer service (DSN Retailing, 2000). At the time, picking operations at Landsend.com involved a process of matching warehouse shelving labels to picking-and-packing lists.

2.1.2 Packing

The process of order packing can be another labor-intensive stage of the warehouse operations (Bartholdi and Hackman, 2011). The customer order is checked for completeness and accuracy during packing. A fundamental component of packing is to unitize customer orders for shipping or vehicle loading. In the process of unitizing, special handling considerations may be required for certain products.

A successful integration of technology into the packaging process is demonstrated by the warehouse-based online retail store, Overstock.com (Hartman, 2009). Overstock.com implemented a system called PriorityPak, which reduced packing operation requirements from eight full-time workers to one worker. The PriorityPak technology also considers product dimensions in the packing process.

The United States Postal Service (USPS) has developed an integrated solution for customers to ship products (USPS.com, 2013). A program, termed Priority Mail Flat Rate®, is a system of providing packaging and shipping services for a set fee based on
package size (restrictions apply). The USPS has a competitive advantage by simplifying the cost structure and standardizing shipments (e.g., dimensionally, service features).

Product packaging for perishable products must be sufficient to uphold product integrity through all transportation events. For the food industry, the United States Department of Agriculture (USDA) provides guidelines for the transportation of perishable products (USDA, 2006). The guidelines include handling considerations, temperature management, and vehicle upkeep. Pharmaceuticals are another perishable product requiring special consideration for packaging and handling. The Therapeutic Research Center (2008) provides a guideline to pharmacy personnel as to the handling of temperature-sensitive and other stability-sensitive pharmaceutical products. Improper handling can compromise the medicinal benefits to the patient, and wasted product can present large costs to the pharmacy. Ohkawara et al. (2012) present an innovative approach to improve the environmental fluctuations in temperature while transporting living tissue.

2.1.3 Vehicle Loading

Vehicle loading is the last stage of internal order handling. The retail company may either use an external shipping company or provide in-house delivery services to the customer. In either case, the company providing the delivery to the customer is assumed to be tasked with vehicle loading. The process of vehicle loading involves the handling of finished packaged products into a specific location within the delivery vehicle. Vehicle organization and dimensional limitations are considered factors during loading.
Considering only ground deliveries, such as those completed by United Parcel Service (UPS) motor vehicles, vehicle loading is the preceding activity to customer delivery. Total delivery capacity of a company is directly related to the scale of the company, but is comprised of individual resource capacities. Again considering UPS, the delivery fleet consists of 70,000 drivers (Brewin, 2003). Each driver follows a company-defined standardized procedure. The result then is a driver/vehicle capacity, which is a factored element of the company’s overall capacity. Therefore, the standardized procedures for a single vehicle have the ability to impact fleet efficiency, regardless of size.

The Truck Loading Problem is a theoretical formulation that manages vehicle capacities and product assignments according to demand at delivery locations (Yüceer and Özakça, 2010). Model results allocate vehicle capacity to each of the product types that best satisfies the demand of every destination. The objective of the Truck Loading Problem is to maximize the replenishment time. Replenishment time is the route cycle time, representing a frequency to return to a base location for more supply of product. Truck Loading Problems are an application of a Mixed Integer Linear Program. In general form, a Truck Loading Problem is defined by a number of compartments to hold a certain number of product varieties (i.e., dimensions), that are to be delivered to a number of demand locations from a source location.

The model for the Truck Loading Problem is presented in Figure 4 (Yüceer and Özakça, 2010). The model is formed by three index sets. Compartments are tracked by \( I = \{1, 2, \ldots, m\} \), destinations by \( J = \{1, 2, \ldots, n\} \), and products by \( K = \{1, 2, \ldots, q\} \). The three indices are used by a decision variable, \( x_{ijk} \), for the number of product \( k \in K \) going
to destination \( j \in J \), which is stored in compartment \( i \in I \). Modeled constraints are vehicle capacity constraints and destination demand constraints. The constant \( C_i \) represents the capacity of compartment \( i \in I \). The coefficient \( p_k \) represents packaging size of product \( k \in K \). The product \( k \in K \) demand rate, for destination \( j \in J \), is kept by variable \( d_{jk} \). Delivery quantity is then, \( D_{jk} = \sum_{i=1}^{m} x_{ijk} \), \( \forall j \in J \) and \( k \in K \). Replenishment time \((t)\) is the minimum ratio, among compartments, of delivery quantity to the demand rate, seen in equation (1).

\[
t = \min_{j \in J, k \in K} \left\{ \frac{\sum_{i=1}^{m} x_{ijk}}{d_{jk}} \right\} \quad (1)
\]

**OBJECTIVE:**

Maximize \( t \) \hspace{1cm} (2)

**SUBJECT TO:**

\[
\sum_{j=1}^{n} \sum_{k=1}^{q} p_k x_{ijk} \leq C_i \hspace{1cm} i \in I \quad (3)
\]

\[
t d_{jk} - \sum_{i=1}^{m} x_{ijk} \leq 0 \hspace{1cm} j \in J, k \in K \quad (4)
\]

\[
x_{ijk} \geq 0 \hspace{2cm} \text{integer for } i \in I, j \in J, k \in K \quad (5)
\]

\[
t \geq 0 \quad (6)
\]

*Figure 4: Truck Loading Problem (Yüceer and Özakça, 2010)*

Statement (2) of the Truck Loading Problem (Figure 4) defines the objective of maximizing the minimum replenishment time, \( t \). Constraint (3) defines the capacity constraints for the different vehicle compartments. Constraint (4) represents the demand constraints. Finally, constraint (5) restricts the quantity decision variable to integer values and constraint (6) limits replenishment time to positive values. The model terminates once demand for a product at the next destination will not be satisfied.

Yüceer and Özakça (2010) presented a heuristic approach to solve the truck loading problem with significantly reduced processing time. Results of randomly selected
problems were optimal in 82.2% of the tested iterations, with prompt solution processing time.

Within a vehicle, resource utilization may present an opportunity to further increase efficiencies (Hostetler, 2010). Hostetler (2010) evaluated packing sequence, storage location, and storage equipment within a vehicle to improve the delivery process. The system assumed unique delivery destinations for each loaded unit, given that routing is known. The research assessed the varieties of loading problem attributes. Units may have homogeneous or differing degrees of heterogeneous packaging, also fixed or variable container dimensions and problem objectives can vary. The authors discuss how a rearrangement cost is incurred if a package is not available at the time of delivery.

Iori and Martello (2010) extend a relatively recent research approach to solving both a vehicle loading problem and vehicle routing problem in one effort. The research considers both two- and three-dimensional packing into a vehicle. Heuristic and exact approaches are evaluated. Combining problems into one solution procedure is complex, yet may offer an opportunity for modeling a delivery system more thoroughly.

2.2 Vehicle Routing

Managers of delivery services are set with the task of finding efficient routing for vehicles and personnel. A diverse collection of applications for routing problem solution methodologies has spurred intense and comprehensive research. The field of routing is broad. Thus, the following literature review is specific and narrow to guide the reader through solution development that is relevant to the current work.
Allotted processing time, accuracy of results, flexibility of the model and simplicity of use are key considerations for the best type of solution procedure (Cordeau et al., 2002). Cordeau et al. (2002) suggest an acceptable processing time range of 10 to 20 minutes for routing programs run daily. The routing program discussed herein utilizes heuristics. Heuristics feature acceptable levels of accuracy with lower processing times when compared with exact approaches. It is important to note that accuracy is defined as a percent difference of a heuristic solution from the exact solution.

2.2.1 Traveling Salesman Problem

Traveling salesmen problems (TSP) require the construction of a single, minimum cost cycle through a set of nodes (Bodin et al., 1983; Applegate et al., 2006). Coined the Traveling Salesman Problem in the 1900s, the history of the TSP spans back to at least the late 1800s (Applegate et al., 2006). The intuitive nature of the problem has prompted broad research and application. Applegate et al. (2006) give an interesting historical synopsis of the TSP for the curious reader.

Transportation applications of the TSP utilize one delivery vehicle/person. Solutions to the TSP may be achieved through optimal or heuristic approaches. Golden et al. (1980) summarize several heuristic approaches. The approaches can be classified as tour construction, tour improvement, or composite procedure.

Selection of a TSP heuristic depends on the demands of the application. Composite procedures should be selected when a high degree of accuracy is required, thereby accepting higher processing cost (Golden et al., 1980). On the other hand, quick tour construction procedures are best when an approximate solution is acceptable. Tour
construction procedures often result in an accuracy range within 3% to 8% of the optimal solution. Noted by Golden et al. (1980), the Farthest Insertion Algorithm is one tour construction algorithm that performs reasonably well within stated limitations. Accuracy results from five test problems, containing 100 nodes each, had a mean and standard deviation of 4.93% and 2.35%, respectively. Running a 3-opt procedure following the implementation of the Farthest Insertion Algorithm greatly improves accuracy results, but at significant cost in processing time.

Golden et al. (1980) provides a standard mathematical model (Figure 5) for the TSP. Consider a network \( G \), consisting of a node set \( N \) and branch set \( A \). The cost \( C \) is a matrix that contains the travel cost between each pair of nodes along the respective branches. Travel goes from node \( i \) to node \( j \). The objective (7) of the TSP is to minimize the total travel costs while maintaining a single route that travels from a depot location through every other node exactly once. Travel flow through each node is controlled by constraints (8) and (9). Only one branch variable is active entering and leaving a given location. Constraint (10) is a placeholder for a subtour-breaking constraint. Finally, constraint (11) restricts the variable, \( x_{ij} \), to be binary. If the branch between locations \( i \) and \( j \) is used in the TSP route, \( x_{ij} \) is set to a value of 1 and 0, otherwise. Applying this mathematical formulation enables the modeler to solve the route to optimality.

To detail constraint (10), a subtour occurs when all other constraints are satisfied, but the nodes are separated into subsets with one cycle among each subset. The resulting solution then has an incomplete travel flow through all nodes. Depending on the scenario modeled, one of three subtour-breaking constraints (equations 12 – 14, in Figure 6) can be applied in place of constraint (10). Constraint (12) requires that a subset is connected
to the remaining nodes of the network. Constraint (13) prohibits a cycle amongst a selection of arcs. The last subtour-breaking constraint, constraint (14), works through contradictory results when a subtour exists. The constraint prevents subtours thereby forcing the route to be continuous through all nodes, beginning and ending at the same node.

**OBJECTIVE:**

\[
\text{Minimize } \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}
\]

**SUBJECT TO:**

\[
\sum_{j=1}^{n} x_{ij} = b_j = 1 \quad (j = 1, \ldots, n) \tag{8}
\]

\[
\sum_{i=1}^{n} x_{ij} = a_i = 1 \quad (i = 1, \ldots, n) \tag{9}
\]

\[X = (x_{ij}) \in S\tag{10}\]

\[x_{ij} = 0 \text{ or } 1 \quad (i, j = 1, \ldots, n) \tag{11}\]

*Figure 5: TSP Model (Golden et al., 1980)*

\[
S = \{(x_{ij}) : \sum_{i\in Q} \sum_{j\in Q} x_{ij} \geq 1 \text{ for every nonempty proper subset } Q \text{ of } N\}
\]

\[
S = \{(x_{ij}) : \sum_{i\in R} \sum_{j\in R} x_{ij} \leq |R| - 1 \text{ for every nonempty subset } R \text{ of } \{2, 3, \ldots, n\}\}
\]

\[
S = \{(x_{ij}) : y_i - y_j + nx_{ij} \leq n - 1 \text{ for } 2 \leq i \neq j \leq n \text{ for some real numbers } y_i\}
\]

*Figure 6: Subtour-Breaking Constraints (Golden et al., 1980)*

A balance between optimality and program processing time must be considered. Due to the daily runs of the routing program, the use of an heuristic expedites processing time with acceptance of the sacrifice in exact accuracy. The original Farthest Insertion Algorithm (Golden et al., 1980) is shown in Table 1. For use in this work, this procedure was coded into Microsoft Excel® Visual Basic for Applications (VBA) and called as part of the overall routing routine. The corresponding code can be found in Appendix C.

Butt and Ryan (1999) discuss a tour storage feature for improved program management. In their research, only new tours are evaluated with the TSP heuristic,
calling on stored values for previously evaluated tours. The purpose of the feature is to reduce overall processing time. Ryan et al. (1993) presents an approach for attaining accuracy improvements by completing the TSP procedure following the addition of each delivery location into a route assignment when there are multiple routes. Implementing this change improves the solution, but requires a considerably higher number of iterations 

\[ [(number \ of \ locations)^2 - number \ of \ locations] \] to the solution process. The added program processing time may compromise the feasibility of program implementation, when solution speed is critical. A more efficient version of a TSP remains an opportunity for future development.

<table>
<thead>
<tr>
<th>Table 1: Original TSP Heuristic (Farthest Insertion Algorithm) (Golden et al., 1980)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Let ( c_{ij} ) be the cost to travel from node ( i ) to node ( j ).</td>
</tr>
<tr>
<td>1) Start with a subgraph consisting of node ( i ) only.</td>
</tr>
<tr>
<td>2) Find node ( j ) such that ( c_{ij} ) is maximal and form the subtour ( i-j-i ).</td>
</tr>
<tr>
<td>3) Selection step. Given a subtour, find node ( k ) not in the subtour farthest from any node in the subtour.</td>
</tr>
<tr>
<td>4) Insertion step. Find the arc ((i,j)) in the subtour which minimizes ( c_{ik} + c_{kj} - c_{ij} ). Insert ( k ) between ( i ) and ( j ).</td>
</tr>
<tr>
<td>5) Go to step 3) unless we have a Hamiltonian cycle.</td>
</tr>
</tbody>
</table>

Applegate et al. (2006) define and discuss the current methods for attaining TSP solutions. The solution procedures include both exact and heuristic approaches. As new solution methodologies are proposed, and existing methods are improved, quicker and more accurate results are expected. An online resource of computer code for the implementation of TSP procedures (i.e., Concorde) is discussed within the TSP book by Applegate et al. (2006).
2.2.2 Multiple Traveling Salesman Problem

An expansion of the TSP is the multiple traveling salesman problem (MTSP) (Bodin et al., 1983). Multiple fleet vehicles are directed through separate routes, leaving and returning to the same depot. Each demand node is visited exactly once, by one of the fleet vehicles. Routes are designed to reduce total travel time. This problem requires the assignment of nodes to routes in addition to finding the optimal minimum cost route through the selected nodes for each vehicle. This is not a trivial addition which can vastly increase the complexity of the problem and the solution time.

2.2.3 Vehicle Routing Problem

The Vehicle Routing Problem (VRP), recognized as the single depot, multiple vehicle, node routing problem, is a classic approach used in transportation and resource allocation applications (Bodin et al., 1983; Laporte, 2009). The standard VRP approach differs from a MTSP approach in that the vehicle capacities are known and limited, and each node hosts a deterministic demand on the route. The standard VRP can be expanded to support maximum route time for each fleet vehicle. Exact approaches have been defined for the VRP. However, as with applications of TSP approaches, often the permitted processing time restricts the type of solution methodology used.

Bodin et al. (1983) summarize VRP approaches into seven categories. The route first-cluster second method is a reversal in processing of the cluster first-route second procedure. Both methods work by processing through collections of nodes and/or arcs. Savings/insertion methods build routes incrementally based on the least cost of adding locations to the existing route. The improvement/exchange approach, on the other hand,
begins with a feasible solution and incrementally transitions to other feasible solutions that are more optimal. Newer methodologies include both mathematical-programming-based and interactive optimization approaches. The first implements existing formulations for the problem components. The second methodology engages subjective interaction of the decision-maker. The decision-maker’s expertise and knowledge assist the program towards optimality, while resulting in a high degree of solution implementation. Finally, exact procedures apply specialized algorithms to identify optimal values.

Cordeau et al. (2002) completed a comparison of classical VRP heuristic methods. Heuristic results were tested against the best known solutions for fourteen problem sets, ranging from 50 to 199 locations and under either capacity or capacity and demand restrictions. The solution value and processing time, in seconds, were recorded. Table 2 displays a summary of processing time and accuracy results for the classical VRP methods.

<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Time (sec)</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clarke and Wright</td>
<td>0.13</td>
<td>6.71%</td>
</tr>
<tr>
<td>Two-matching</td>
<td>13,371.42</td>
<td>0.63%</td>
</tr>
<tr>
<td>Sweep</td>
<td>105.6</td>
<td>7.09%</td>
</tr>
<tr>
<td>1-Petal</td>
<td>15.6</td>
<td>5.85%</td>
</tr>
<tr>
<td>2-Petal</td>
<td>208.8</td>
<td>2.38%</td>
</tr>
</tbody>
</table>

*Table 2: Classical Heuristic Methods (Cordeau et al., 2002)*

The research of Courdeau et al. (2002), showed that Clarke and Wright Savings heuristic excels in speed, but accuracy is lower than most. The heuristic is simple to program, but is not flexible. For example, the heuristic is not adequate for use with time windows. Therefore, using this approach may limit the scope of a routing program. The
results from the Two-Matching heuristic are nearest to the optimal solution, but the severity in processing time requirements may be excessive for daily use.

The Sweep, 1-Petal, and 2-Petal algorithms operate similarly through a rotating ray about the depot (Cordeau et al., 2002). Routes are built according to capacity and duration constraints. The 1-Petal heuristic is best with respect to processing time, and its accuracy is in between the accuracies of the Sweep and 2-Petal approaches. The algorithms are flexible and can support a constrained VRP with time windows.

The Sweep algorithm, originally proposed by Gillett and Miller (1974), builds routes incrementally by considering locations in radial order. Adapted from the Sweep algorithm, Foster and Ryan (1976) defined a general LP (Figure 7) for solving the vehicle scheduling problem. The index values are $i$ for delivery and $j \in J$ for route. The LP objective statement (15) is to reduce the sum of the associated travel costs for each route $j$. The variable $x_j$ is equal to 1 when route $j$ is present in the schedule and 0, otherwise. Value $V$ is the vehicle cost, in units of miles, while term $m_j$ is the mileage of route $j$. The term $a_{ij}$ is equal to 1 when a delivery $i$ is made on route $j$. Constraint (16) requires each delivery to be assigned to a route. The research by Foster and Ryan (1976) elaborates on several dimensions of model expansion for specific applications.

\[
\text{OBJECTIVE:} \\
\text{Minimize } \sum_{j \in J} (V + m_j) x_j \\
\text{SUBJECT TO:} \\
\sum_{j \in J} a_{ij} x_j = 1 \quad (i = 1, ..., n)
\]

*Figure 7: Linear Program of Vehicle Scheduling Problem (Foster and Ryan, 1976)*
Ryan et al. (1993) provides a development on the integer programming methods previously defined by Foster and Ryan (1976). The development implements the concept of a generalized petal set that allows for improvements in efficiency and solution accuracy while maintaining integer solution integrity of the LP formulation. The petal routing approach minimizes the number of vehicles required for deliveries, and the total distance travelled by those vehicles. The approach requires a TSP methodology to execute a post-optimization on each configured route set. Renaud et al. (1996) extends the Petal heuristic, and therefore the Sweep algorithm, further by relaxing the structure of petals created during routing. The adapted heuristic approaches the results of a presented tabu search method in terms of accuracy, while significantly reducing processing time.

The standard VRP mathematical model (Bodin et al., 1983) is shown in Figure 8. VRP construction supports either delivery or pickup demand. The variable \( n \) is the number of nodes, with \( n = 1 \) used to represent the depot location. The number of vehicles, \( NV \), consist of a heterogeneous fleet with capacity \( K_v \) and time restriction \( T_v \) for vehicle \( v \). Node \( i \) has a demand of \( d_i \) and service time \( t_i^v \) (based on vehicle \( v \)), where \( d_1 = 0 \) and \( t_1^v = 0 \). The travel time of vehicle \( v \) from node \( i \) to node \( j \) is \( t_{ij}^v \), with \( t_{ii}^v = \infty \). The factor of travel time is used to determine final cost of travel between two locations, notated \( c_{ij} \). The binary variable \( x_{ij}^v \) is 1 when the arc from node \( i \) to node \( j \) is traveled by vehicle \( v \) and 0, otherwise. Finally, \( X \) is the matrix of \( x_{ij}^v \) values, regardless of vehicle type.
**OBJECTIVE:**

\[
\text{Minimize } \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}^v
\]  

(17)

**SUBJECT TO:**

\[
\sum_{i=1}^{n} \sum_{v=1}^{NV} x_{ij}^v = 1 \quad (j = 2, ..., n)
\]  

(18)

\[
\sum_{i=1}^{n} x_{ij}^v - \sum_{j=1}^{n} x_{pj}^v = 0 \quad (v = 1, ..., NV; p = 1, ..., n)
\]  

(19)

\[
\sum_{i=1}^{n} d_i (\sum_{j=1}^{n} x_{ij}^v) \leq K_v \quad (v = 1, ..., NV)
\]  

(20)

\[
\sum_{i=1}^{n} t_i^v \sum_{j=1}^{n} x_{ij}^v + \sum_{i=1}^{n} \sum_{j=1}^{n} t_{ij}^v x_{ij}^v \leq T_v \quad (v = 1, ..., NV)
\]  

(21)

\[
\sum_{j=2}^{n} x_{ij}^v \leq 1 \quad (v = 1, ..., NV)
\]  

(22)

\[
X \in S
\]  

(23)

\[
x_{ij}^v = 0 \text{ or } 1 \quad \text{for all } i, j, v
\]  

(24)

*Figure 8: VRP Model (Bodin et al., 1983)*

The objective (17) of the VRP is to minimize total travel costs of a fleet of NV vehicles. Constraint (18) ensures that a demand node is met once, by a single vehicle. Equation (19) requires the vehicle \( v \) that enters node \( p \) to also exit node \( p \). Vehicle capacity is managed by constraint (20). Note that it is assumed that demand at each node does not exceed the capacity of any truck (\( \max d_i < \min K_v \)). Elapsed route time is modeled in constraint (21). Route time is comprised of two components: travel time and service time. The route time for any vehicle is not permitted to exceed the time restriction for said vehicle. In constraint (22), availability of vehicle \( v \) to leave the depot to start a route is limited to 1 occurrence. A more detailed discussion of subtour-breaking constraints (Figure 6) available in place of constraint (23) is included in the preceding TSP section. Constraint (24) restricts the variable, \( x_{ij} \), to be binary. If the branch between locations \( i \) and \( j \) are used in the route of vehicle \( v \), \( x_{ij}^v \) is set to a value of 1 and 0, otherwise.

The standard VRP model is well-suited to the routing problem under consideration. However, there are differences between the standard model and the
approach followed in this work. The assumption of \( \max d_i < \min K_v \) is not held as a binding constraint in this application. Instead, if \( \max d_i > \min K_v \) or \( t_{i,j}^v x_{i,j}^v + t_{j,1}^v x_{j,1}^v > T_v \), then route \( v \) is assigned between node 1 (depot) and node \( j \). The service time \( t_{i,j}^v \) at a location is assumed to be 0. Only deliveries are made, no pickups. Also, the fleet is assumed homogeneous. Although the fleet consists of different vehicle makes and classes, there is virtually an infinite (non-binding) capacity associated with each vehicle.

One variable within the considered scenario is the number of vehicles in the fleet. As a result, the model does not constrain the number of vehicles used and instead constructs routes strictly based on capacity and time constraints. Foster and Ryan (1976) define a straightforward calculation for the number of vehicles, shown in equation (25). The formula outputs the minimum number of routes/vehicles for the total demand requirement of all locations, \( \sum_{i=1}^{n} q_i \), and the capacity, \( K \), of the vehicle.

\[
v = \frac{\sum_{i=1}^{n} q_i}{K} \tag{25}
\]

Applying the VRP model in Figure 8 would solve the routes to optimality, but processing time is a crucial concern of routing program development. Thus, this work instead implements a 1-Petal heuristic to solve for an improved routing solution, in less time. Table 3 presents the steps of the Petal algorithm presented by Foster and Ryan (1976). The heuristic works to define the nodes of an individual subset (“petal”) of deliveries to one route.
1. Label locations.
   - Depot: \( i = 1 \)
   - Assign \( i = 2 \) to an arbitrary delivery location
   - Number the remaining delivery locations in radial order \( (i = 3, \ldots, n) \), either clockwise or counter-clockwise. Break ties arbitrarily.

2. Build petals incrementally according to the cyclic order.
   - During petal construction, order is cyclic in that subsets beginning at higher order values wrap around to include the deliveries at the beginning of the order.
   - Define subsets (“petals”) of delivery locations. Configurations of petals are deemed feasible if the totaled capacity does not exceed the route capacity.

3. Process subset configurations with a TSP heuristic to determine total distance traveled. *
   - The petal set is feasible if the route mileage limit is not exceeded.
   - Objective: Find the minimum (optimal) cost spanning petal set.

*Ryan et al. suggest that an efficient TSP heuristic may be applied following the addition of each node to a route for an improved final solution. (1993)*

*Table 3: Original VRP Heuristic (Petal Algorithm) (Foster and Ryan, 1976)*

2.3 Summary

Order picking, packing, and loading are internal operations that can introduce costly waste for a company. Both picking and loading operations can be labor-intensive. Therefore, it is important to reduce the travel of personnel by incorporating literature-defined principles and techniques. In loading, vehicle equipment and organization provides yet another potential for improvements in process efficiency. An important aspect of product handling is the use of solution procedures by companies seeking to improve operations, regardless of size. Retail companies have embraced handling fundamentals and technology to eliminate waste and increase capacity.

Certain products require special handling throughout the internal and external delivery processes. Improper handling can compromise the quality of the delivered
product. Pharmaceuticals requiring special handling are temperature-sensitive and must remain in a specific temperature range to maintain medicinal effectiveness.

Vehicle routing is an extensive area of research where solution procedures have been developed to accommodate problems with either processing time or accuracy priorities. The objective of routing problem modeling is to minimize the travel over routes. The TSP solves for the shortest path of a single-route problem. The VRP is a composite of multiple TSP routes. Heuristics permit a solution to be found quickly, while accepting a loss in solution exactness. Currently, research is being extended to consider vehicle routing and vehicle loading in a common model. Research in vehicle routing has significance to a variety of home-delivery service providers.

Standard protocols equip a retail pharmacy with research-defined methodologies to potentially gain operational efficiency. The Farthest Insertion and Petal algorithms were jointly selected for adaptation of a delivery personnel support tool in this work. A delivery support tool, referred to as “routing program,” is created to objectively define a daily routing schedule. Theoretical models provide the foundation to achieve desired competitive advantage of providing product delivery services.
CHAPTER III

METHODOLOGY

Management of a retail pharmacy delivery system must consider the associated personnel and operations required to provide the delivery service. As a result, this research was completed to support the management and operation of home deliveries from a retail pharmacy. This research offers procedures to accommodate service expansion and discusses a proposed routing program that is designed to reduce costs of travel (i.e., within facility and between delivery points) and the costs of labor time that would otherwise be used to schedule delivery routes.

A substantial process is required for delivery personnel to prepare and deliver orders to customers. This work considered three independent stages of order processing in the development of a solution process. The discussion is guided by observations from a case study pharmacy to establish a basis for system design standards. In terms of home delivery, a customer order may require more items than just prescriptions; therefore, a general “order” term is used to refer to the collection of items a customer has demanded.

The first stage of processing includes the preparation of orders from throughout the pharmacy. The first stage is not a portion of delivery personnel responsibilities, and therefore was not a focus of this research, but it is described to better define the overall system. This work proposed no changes to the existing order preparation stages.

After an order is complete and ready for delivery, the second stage of processing considers the activities of order picking, packing, and loading. The objective is to minimize travel and handling during order retrieval within a pharmacy facility. The last
stage includes the routing of the vehicles for delivery. A secondary objective of this work was to define standard operating procedures for the delivery personnel. Conceptually, the proposed systems are adaptable to suit unique pharmacy operations and configurations.

3.1 Order Processing

A complete process map of the order flow through the case study pharmacy system is provided in Appendix A. In the following discussion, the order flow through the pharmacy is discussed in two distinct sections. The first section considers the stages between order entry and filled prescriptions. The second section considers the activities completed by delivery personnel.

3.1.1 Order Entry to Filled Prescriptions

The process between accepting orders into the system and filling prescriptions is the first stage in order processing (Figure 9). The two pharmacy inputs are orders and products. Orders can enter the system via phone, fax, email, or patient drop-offs. Entering products include pharmaceuticals, OTC medications, and convalescent aid products and devices. Orders are processed through the internal computer database system while products are placed into the respective inventory locations.
Entering orders, which include infusions, are routed through a verification stage. An infusion is a solution prepared for intravenous medicine delivery (Merriam-Webster, 2013). Infusions are a costly item, thus first requiring a consult with the covering insurance company and a follow-up with the patient. Once verification measures are complete, the pharmacist or pharmacy technician chooses to accept or reject the order. Rejected orders exit the system, while accepted orders rejoin the processing with all other pending customer orders to have the prescription filled. It is important to mention that a clean room is necessary to prepare infusions; therefore, a separate Clean Room location is defined within the pharmacy facility.

The routing of filled prescriptions, or “scripts”, splits based on the classification of customers. Scripts from “walk-up” customers are immediately processed, as “walk-up” customers are those who deliver the script and wait for it to be filled. The other two
classifications, “will-call” and “deliver,” are placed in the appropriate finished goods receptacle; located in the Pharmacy bin system or refrigerator, or the Clean Room refrigerator. Regardless of the finished good storage location, every customer order receives a place card in the bin system. Identifying an order as special handling, an orange ticket is attached to the place card if part or all of the order is stored in either of two refrigerator locations. The pharmacist or pharmacy tech will also avoid closing the final package with a staple to indicate the entire order is not complete without further attention.

A customer classified as “will-call” is considered to retrieve their order through one of two methods. They may either pick up their order at the pharmacy (“will-call (pickup)”) or notify the pharmacy that they would prefer to have their order delivered (“will-call (deliver)”). Customers classified as “deliver” are known to desire delivery at initial order entry to the system. Any “will-call (pickup)” order remains stored until the customer arrives to the pharmacy to retrieve their order. All remaining orders, “will-call (deliver)” and “deliver” enter the control and responsibility of the delivery personnel.

3.1.2 Delivery Personnel Activities

The primary steps of the process completed by delivery personnel include order picking, packing, loading, and delivery (Figure 10). The orders handled during this stage of processing consist only of “will-call (deliver)” and “deliver” identifications. The next section discusses the routing of deliveries. Delivery personnel are currently trained through mentorship and have no formal medical background. Delivery personnel do have skilled training in convalescent aid functionality and setup.
3.2 Routing Methodology

Theoretical routing models are the basis for the routing methodology used in this work. However, the following discussion will introduce the organization of the resulting program interface to set the premise for the theoretical model review in Chapter 4.

The routing program platform was developed in Microsoft Excel®. This platform was chosen since a Microsoft application is assumed to be familiar, easy to use and possibly more financially feasible than more sophisticated software packages. In the following discussion, the fundamental structure of the routing program is introduced.

Input for the routing program can be either manually entered or retrieved from an existing database in the form of a text file (".txt"). Information required is a customer identifier (e.g., last name) and delivery address. Figure 11 shows the flow of input information into the routing program. The routing program was organized according to two separate files; “Customer Database” and “VRP Petal”. The next two sections are dedicated to defining the purpose of the program files.
3.2.1 Program Component #1 – “Customer Database”

The routing program is broken into two primary components, differentiated by two separate Microsoft Excel® workbooks; “Customer Database” and “VRP Petal.” The Customer Database file is the source of customer location and travel time information. The file has one user interface worksheet (“New Customer”) and two database worksheets (“Customer Database” and “Travel Matrix Database”). The purpose of each worksheet is summarized in Figure 12. Refer to Appendix B for a full copy of the code used in program development of the Customer Database. The “New Customer” worksheet is displayed in Figure 13. The first section allows the user to either add or delete customer information in the “Customer Database.” In adding information manually, users enter the customer’s last name; internal reference number; delivery address; and other relevant information that is desired for final routing output (e.g. phone number). The other option available for adding information is to select the “Customer Database” worksheet and copy in a pre-generated list of customers and corresponding information. If a customer no longer wishes to receive deliveries, that customer’s information is deleted from the database to support quicker processing times in matrix
construction. To delete customer records from the system, either the customer’s last name or internal reference number may be entered in the “Deletion Reference” cell. The information for the deleted customer is removed from both the “Customer Database” and “Travel Matrix Database.”

**Figure 12: Customer Database Summary**
Both customer addition and deletion activities prepare the “Customer Database” worksheet for matrix generation operations. Two forms of information are desired from matrix generation operations. The first set of information required is the travel information between each pair of locations; depot and delivery demand locations. Travel time and travel distance are two potential parameters for the travel information collected. The second set of information is the latitude and longitude between the depot and all delivery locations. The four values are inputs to calculate the bearing between the depot and the delivery location. The travel matrix is dimensionally $n \times n$. The bearing array contains only the final bearing values and is $(n - 1) \times 1$.

Once input information in the “Customer Database” is complete, the travel time matrix and bearing array are created through one of two options. The first, most useful option, is to update an existing matrix. This option takes an existing bearing array and travel time matrix and adds new information without altering existing data. Two values are included on the “New Customer” worksheet to inform the user of the status of the
workbook (circled on Figure 13). For the routing program to work correctly, the number of customers in the “Customer Database” should match the number of customers in the “Travel Matrix Database.” Updating of an existing matrix takes \(((\text{customers in database})^2 - (\text{customers in matrix})^2)\) iterations to make the travel time matrix and bearing array complete.

The second option is to regenerate an entirely new travel time matrix and bearing array. The purpose of this option is to refresh all travel values for accuracy dealing with alterations to the road system. Generating all-new information is time consuming since \((\text{number of locations})^2\) iterations must be performed.

The output of this program includes both an array for bearings and a travel time matrix. A bearing is calculated between the latitude and longitude of a delivery location and the latitude and longitude of the depot. The measurement gives a radial value used in the routing process. A zero degree bearing corresponds to North, increasing in a clockwise direction. Travel time is taken between any two locations (depot and delivery locations included). The array and matrix are referenced by the VRP Petal file, as needed.

3.2.2 Program Component #2 – “VRP Petal”

The second program component is contained in the VRP Petal file (Figure 14). The workbook is organized by “Input” and “Output” worksheets. In the “Input” worksheet, users enter a customer reference value (last name or internal reference number) and the customer demand in terms of number of units. There is not a limit placed on the number of locations that can be routed. However, more locations will increase processing time in an exponential manner. To define limitations on each route,
the user must also enter the maximum duration, in minutes, and vehicle capacity, in units, for each route. Due to small volume requirements of prescriptions, compounded medications and infusions, fleet vehicles are assumed to be homogenous with non-binding capacity constraints. Therefore, only the duration constraint is binding. The program interprets user input and writes a reference number that refers to the line of data in the Customer Database file. Following run completion of the routing sequence, a route summary is provided in the “Input” worksheet. This allows the user to be informed promptly of the number of routes, the total duration of all routes, and the longest route required.

The first stage of the routing routine is to generate a travel matrix and bearing array for the locations listed for routing. Calling on values from the Customer Database, the array and matrix are configured into two separate worksheets. These worksheets provide the input information for the second stage, “Create Routes” function. In creating routes, the program steps through both VRP and TSP modules to find the ‘best’ route possible. The macro buttons for each stage are circled in Figure 15. Following each stage, the user is informed with a prompt box that the program has completed the routine.
The “Output” worksheet is configured to handle the display of twenty-five individual routes, although expansion would be effortless. Included in the output display are the expected route duration, number of deliveries, and an ordered list of deliveries. Worksheet anchor cells make the display easily customizable to reflect desired route information. For example, a route report may include a phone number and warehouse picking instructions in addition to the customer reference value and address. An assumption of the program output is that the delivery personnel would employ a form of Global Positioning System (GPS) during route travel, or lookup driving instructions online before departure. The VRP Petal code is available in Appendix C.
3.3 Summary

Order processing includes every stage of order handling throughout the pharmacy. A process map was discussed to exhibit the interface between order preparation and delivery activities. Three types of orders enter the pharmacy system. Two of these types are routed into the delivery portion of processing. Delivery processing is defined by internal and external operations. Internal operations include order picking, packing and loading. The primary external operation is delivering to the customer.

To assist the development of route schedules, this chapter also introduced a fundamental construct for a routing program. The routing program was organized into two components. The first component, Customer Database, provides storage of customer, master travel, and master bearing information. The second component, VRP Petal, contains the structure to execute the routing heuristic. Required inputs for the second component include a customer reference value, customer demand, and route constraint.
parameters. Both component files feature a user interface. Generated routing is written to a worksheet for reference by delivery personnel.

The routing program operates through several Microsoft Excel® VBA macros. The next chapter thoroughly reviews the routing program logic, developed from the theoretical routing models previously discussed in Chapter 2.
CHAPTER IV

ROUTING PROGRAM LOGIC

There are three primary routines orchestrated within the Microsoft Excel® VBA routing program. The routines include the matrix generation, VRP petal heuristic, and TSP post-optimization heuristic. Matrix generation is used in both the Customer Database and VRP Petal program components. The joint VRP and TSP solution procedures are implemented in the VRP Petal file. Each routine is discussed in the following sections. Refer to Appendix B and Appendix C for a copy of the associated VBA code.

4.1 Generate Matrix

Table 4 summarizes the variables and logic used in creating cost matrices of travel time and an array of travel bearings. Travel time is a standard metric for comparison and was used instead of travel distance to eliminate the factor of variable traffic flow. The matrix generation structure is called on three occasions in the program, denoted by the call commands of “SECTION” in the generic logic. The matrix and array created in the Customer Database file are referred to as the “Master Travel Matrix” and the “Master Bearing Array.” Information for routing is retrieved from the Customer Database file to construct the “Routing Travel Matrix” and “Petal Bearing Array.” The last implementation of travel matrix generation is the “Travel Sub-Matrix” for the TSP approach. Each generated travel matrix is dimensionally n rows by n columns. Travel
between two locations is interpreted as moving from location \( j \), in row \( j \), to location \( i \), in column \( i \). Bearing arrays are dimensionally \( n \) rows by 1 column.

<table>
<thead>
<tr>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n = \text{number of locations} )</td>
</tr>
<tr>
<td>( x = \text{location counter (Depot = 1, customer = 2, ..., n)} )</td>
</tr>
<tr>
<td>( D2R = \text{degree to radian conversion} = \frac{\pi}{180} )</td>
</tr>
<tr>
<td>( i = \text{T0 counter (Depot = 1, customer = 2, ..., n)} )</td>
</tr>
<tr>
<td>( j = \text{FROM counter (Depot = 1, customer = 2, ..., n)} )</td>
</tr>
<tr>
<td>( \text{Location}(x) = \text{postal address of location } x (\forall x) )</td>
</tr>
<tr>
<td>( \text{Lat}(x) = \text{location } x \text{ latitude} )</td>
</tr>
<tr>
<td>( \text{Long}(x) = \text{location } x \text{ longitude} )</td>
</tr>
<tr>
<td>( \text{Reference}(x) = \text{VLOOKUP of customer address} )</td>
</tr>
<tr>
<td>( \text{Bearing}(x) = \text{array of bearings from Depot (origin), measured} )</td>
</tr>
<tr>
<td>( \text{counter} \rightarrow \text{clockwise from East axis } (x = 2, ..., n) )</td>
</tr>
<tr>
<td>( \text{Time}(j, i) = \text{travel time from node } j \text{ to node } i )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Generic Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>( DO \ i = 1 \ to \ n )</td>
</tr>
<tr>
<td>( DO \ j = 1 \ to \ n )</td>
</tr>
<tr>
<td>( IF \ i \neq \ j, THEN )</td>
</tr>
<tr>
<td>( \rightarrow \text{Master Travel Matrix &amp; Bearing Array} )</td>
</tr>
<tr>
<td>( \rightarrow \text{Routing Travel Matrix} )</td>
</tr>
<tr>
<td>( \rightarrow \text{Travel Sub-Matrix} )</td>
</tr>
<tr>
<td>( END \ IF )</td>
</tr>
<tr>
<td>( END\ DO )</td>
</tr>
<tr>
<td>( END\ DO )</td>
</tr>
</tbody>
</table>

| SECTION [2b.] |
|\( \rightarrow \text{Sort Petal Bearing Array} \) |

Table 4: Matrix Generation Variables and Logic

One iteration of matrix generation is required in the **Customer Database**. Delivery addresses maintained in the database are used to determine a bearing from the depot to the delivery destination and create the matrix of travel times. Table 5 and Table 6 define the subroutines used to complete both tasks in database management.
The calculation of bearing uses inputs of depot and delivery location latitude and longitude values. The bearing calculation is required later in solving the VRP. In an effort to keep the daily routing processing times low, the “Master Travel Matrix” and “Master Bearing Array” are the source for operations completed in the VRP Petal file. The generation of this matrix and array is time-consuming, but only required if the Customer Database is altered; either by adding new customers or an update is desired of the existing values in the matrix. Asymmetry in travel times is permitted in the current program configuration. To determine the travel time, the two locations are entered into the online TomTom® Live Traffic maps system. For ease of program adaptation to other applications, travel distance and time are captured. Only travel time is recorded in the current matrix. Program run time savings are apparent by separating matrix generation with online data capture from the daily routing matrix generation.
SECTION [1] – Master Travel Matrix & Bearing Array

\[
\begin{align*}
    a &= \text{Location}(j) \rightarrow \text{From} \\
    b &= \text{Location}(i) \rightarrow \text{To} \\
    \text{IF } i &= 1, \text{THEN} \\
    \text{Call } \text{LongitudeAndLatitude}(j) \\
    \text{Call } \text{MasterBearingArray}(a,b) \\
    \text{END IF} \\
    \text{Call } \text{MasterTravelMatrix}(a,b)
\end{align*}
\]

SUB – LongitudeAndLatitude(j)

Open “http://routes.tomtom.com/lbs/services/geocode/1/query/
" & Location(j) & "\}/json/1e2099c7 – eea9 – 476b – aac9 –
b20dc7100af1; language = en; map = basic"

\[
\begin{align*}
    \text{Lat}(j) &= \text{"latitude"} \\
    \text{Long}(j) &= \text{"longitude"}
\end{align*}
\]

SUB – MasterBearingArray(a,b)

\[
\begin{align*}
    \text{lat}(a) &= \text{Lat}(a) \times D2R \\
    \text{lat}(b) &= \text{Lat}(b) \times D2R \\
    \text{long}(a) &= \text{Long}(a) \times D2R \\
    \text{long}(b) &= \text{Long}(b) \times D2R \\
    x &= \cos(\text{lat}(a)) \times \sin(\text{lat}(b)) - \sin(\text{lat}(a)) \times \cos(\text{long}(a)) \times \cos(\text{long}(b)) \\
    y &= \sin(\text{long}(b) - \text{long}(a)) \times \cos(\text{lat}(b)) \\
    z &= \tan^{-1}\left(\frac{x}{y}\right) \\
    D2R \\
    \text{IF } z &< 0, \text{THEN} \\
    z &= z + 360 \\
    \text{END IF}
\end{align*}
\]

\[
\begin{align*}
    \text{Bearing}(j) &= z
\end{align*}
\]

Table 5: Master Logic, Calculations, and Bearing Array (SECTION[1])

\[
\begin{align*}
    \text{Time}(j, i) \text{ and Bearing}(x) \text{ are the two output parameters from the matrix and}
    \text{array generation. The Time}(j, i) \text{ values are read by both SECTIONS [2] and [3]. Values}
    \text{of Bearing}(x) \text{ are only called in SECTION [2].}
\end{align*}
\]
SECTIONS [2] and [3] matrix generation sequences are called during the routing program, in the VRP Petal file. To use the routing program, delivery personnel begin by entering those customers requiring deliveries, noted by a reference value such as a last name or internal reference number. Sourcing information from the Customer Database file, the first phase of the routing program is to generate a travel time matrix (i.e., “Routing Travel Matrix”) consisting of only those customers to be included in the upcoming route. The relevant bearing information of these customers is also collected (i.e., “Petal Bearing Array”). Table 7 outlines the logic of building the matrix and array. Once the travel matrix and bearing array are constructed, locations are sorted according to the bearing measurement in preparation for the routing procedure (see Table 7, SECTION [2b.]).
This program structure evaluates routing specifically in the order of the bearing metric. Ryan et al. (1993) suggested evaluating “generalized petals” which permit deviations from strictly bearing measurement order. By constructively changing the order, the petals are permitted to overlap in a manner that can better utilize capacity and reduce the total travel time. Due to this suggestion, the program has been divided between the routing matrix generation and petal algorithm calculations so that the user may alter the bearing order before forming petal sets.

Table 8 displays SECTION [3] of the matrix generation script. This SECTION is called at the beginning of the Farthest Insertion Algorithm, used for the TSP post-optimization process on constructed petal sets. Generation of this sub-matrix refers to the “Routing Travel Matrix” created in SECTION [2a]. The matrix is re-generated for every delivery route having more than one delivery location. The sub-matrix contains both the depot and delivery locations within the considered route. Each route evaluated is
considered as an independent TSP. Actual total travel time of the petal set is determined by completing the TSP post-optimization analysis.

<table>
<thead>
<tr>
<th>SECTION [3] – Travel Sub-Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a = \text{Reference}(j) )</td>
</tr>
<tr>
<td>( b = \text{Reference}(i) )</td>
</tr>
<tr>
<td>( Time(j, i) = \text{RoutingTravelMatrix.Cells}(a, b) ) → see SECTION[2a.]</td>
</tr>
</tbody>
</table>

Table 8: Travel Sub-Matrix (SECTION [3])

4.2 VRP Petal Heuristic

Table 9 outlines the adapted petal heuristic for the VRP. The heuristic is a new procedure, adapted from the original petal algorithm given previously in Table 3. Note that the TSP approach is called into step 5 or 8 of the adapted heuristic, depending on the “version” tested. The two versions will be discussed in the next section.
1. Label locations.
   - Depot: $i = 1$
   - Assign $i = 2$ to an arbitrary delivery location
   - Number the remaining delivery locations in clockwise radial order ($i = 3, \ldots, n$). Break ties arbitrarily.
2. Begin with the first petal route ($v = 1$) consisting only of the first delivery location ($i = 1$)
   - Feasible petals (a.) satisfy a demand that does not exceed the capacity of the vehicle, and (b.) have a total distance travelled that does not exceed the imposed distance limit.
3. Calculate total route demand ($d_R = d_R + d_{i+1}$) and total route time ($t_R = t_R + t_{i,i+1} + t_{i+1,i} - t_{i,1}$) of adding the next delivery location ($i = i + 1$) to the current route.
   a. If either $d_n > K_v$ or $t_n > T_v$, then delivery location $n$ is assigned to a unique route.
   b. If either $d_R > K_v$ or $t_R > T_v$, the incoming node becomes the first node in the new route.
      - $t_{total} = t_{total} + (t_R - t_{i,i+1} - t_{i+1,1} + t_{1,1})$
      - $d_R = d_{i+1}$
      - $t_R = t_{i,i+1} + t_{i+1,1}$
      - $v = v + 1$
      - $i = i + 1$
   c. Else, the incoming node is added to the current route.
      - $d_R = d_R + d_{i+1}$
      - $t_R = t_R + t_{i,i+1} + t_{i+1,1} - t_{1,1}$
      - $i = i + 1$
4. Repeat step 3 until all delivery locations are assigned to a route.
5. TSP Version 1: Complete post-optimization on each route using the TSP Farthest Insertion Algorithm heuristic approach to determine actual route costs, and thus total cost ($t_{total}$).
6. Repeat steps 2 – 5 for each delivery location, using a different first delivery location each iteration.
7. Select the routing with the best found total cost ($t_{total}$).
8. TSP Version 2: Complete post-optimization on each route using the TSP Farthest Insertion Algorithm heuristic approach.

Table 9: Adapted VRP Heuristic (Petal Algorithm)

4.3 TSP Post-Optimization Heuristic

Table 1 (page 16) and Table 10 (below) display original and adapted TSP heuristics. The five steps of the original heuristic remain unchanged, while three steps are
added in the adapted model for use in the routing program. First, the algorithm repeats with all possible starting locations within the TSP route (Foster and Ryan, 1976). This removes the chance of beginning with a less desirable location, which would limit the final solution to an increased total route travel cost. The best-found solution is kept for the specific route and the entire process is completed for each route of the VRP initial solution.

Let $c_{ij}$ be the cost to travel from node $i$ to node $j$.

1) Start with a subgraph consisting of node $i$ only.
2) Find node $j$ such that $c_{ij}$ is maximal and form the subtour $i$-$j$-$i$.
3) Selection step. Given a subtour, find node $k$ not in the subtour farthest from any node in the subtour.
4) Insertion step. Find the arc $(i, j)$ in the subtour which minimizes $c_{ik} + c_{kj} - c_{ij}$. Insert $k$ between $i$ and $j$.
5) Go to step 3) unless we have a Hamiltonian cycle.

6) Using a different starting node $i$, repeat steps 2) through 5)
7) Apply the best found Hamiltonian cycle to route
8) Repeat steps 1) through 7) for each route

|Table 10: Adapted TSP Heuristic (Farthest Insertion Algorithm)|

The TSP heuristic was implemented at two different locations within the code to test the differences in processing time and accuracy of results. The first version, Version 1, follows the procedure described in literature (Foster and Ryan, 1976) where the TSP heuristic approach is used to evaluate every petal set generated for total travel time. The second version, Version 2, only calls the TSP heuristic approach once for the petal set with the minimum radially ordered travel time among petal sets generated. Version 1 requires ($number of locations$ − 1) more TSP heuristic initializations than the Version 2 counterpart. It is logical to expect the results for Version 1 to provide higher accuracy.
while requiring significantly more processing time when compared to Version 2. A test was formulated to identify the relationship between efficiency (processing time) and accuracy (route duration). The results of this test are presented in the next chapter.

4.4 Summary

Three sets of Microsoft Excel® VBA routines were used in the implementation of the proposed routing methodology. The routine for matrix generation is called in both the Customer Database and VRP Petal files. An adapted Petal heuristic for the VRP, and an adapted Farthest Insertion heuristic for the TSP, is jointly called in the VRP Petal file to identify an optimized routing scheme. The Petal heuristic is an adaptation of the original heuristic for use in the current application. The modification to the Farthest Insertion heuristic was necessary for improved accuracy and fitment with post-optimization prompts by the VRP routine. Two versions of the routing code are created and discussed. These provide an introductory basis for testing the tradeoffs in accuracy for processing time. The code created for all logic segments is included in the Appendices.
CHAPTER V

ANALYSIS AND RESULTS

The organization of this chapter follows the internal and external processing completed by delivery personnel. Internal processing includes the stages of picking and packing orders, followed by loading of the vehicle. Included in analysis of internal operations are suggested equipment to assist the delivery personnel. External processing is managed through the proposed routing methodology described in the previous chapter and includes the daily route assignments for each vehicle. A standardized work procedure was developed for delivery service personnel. Finally, three sample test sets of customer addresses were evaluated using the routing program developed for implementation of this work.

5.1 Pick, Pack, Load

5.1.1 Picking and Packing

The current picking and packing process followed by delivery personnel at the case study pharmacy varied. In most cases, the process followed was to travel through the facility with a bulk container and pick all orders for delivery. Those orders requiring refrigeration were placed in a protective plastic bag before being placed in an ice-cooled cooler. Once order picking was complete, the container and cooler were loaded into the back of the delivery vehicle. At the delivery location, the order was retrieved from the container and cooler (as needed).
Readers should reference Appendix D. Delivery Personnel Standard Operating Procedure for the complete process map of delivery personnel activities. Note that this is a suggested structure, as currently there is no standard operating procedure for delivery personnel at the case study pharmacy.

The first step for personnel is to print the Delivery Orders at a computer workstation using the existing pharmacy database (Figure 16). These orders are from customers classified as either “deliver” or “will-call (deliver).” With the printouts of customer orders, the delivery personnel will access the routing program, VRP Petal, through Microsoft Excel®. Once the file is open and any existing input and output are removed, the last name or pharmacy’s internal customer reference number for each order is entered. If a change is desired to existing route limitations, the delivery personnel may enter the desired capacity and duration limitations for the routes to be scheduled. To finish routing, the delivery personnel selects the “Travel Time & Bearings” button, followed by the “Create Routes” button. The results are summarized in the “Input” worksheet, and detailed in the “Output” worksheet. The delivery personnel may print the route report of each assigned route. At this operation point, the Delivery Orders are separated and organized by route.
Figure 16: Delivery Personnel SOP (Part 1)

Assuming customer orders for one route will be filled by one person, the delivery personnel take the assigned route report and the Delivery Orders for the route (Figure 17). The ordered information acts as a picking ticket to the delivery personnel as they travel through the facility to fill orders using a picking cart (conceptual drawing in Figure 18). Note that the fabrication cost of the picking cart and vehicle retro-fitting is not considered in analysis. Before picking operations can begin, a picking cart is prepped. If there are orange tickets on any of the Delivery Orders, the delivery personnel first retrieves a cooler with ice and stores the cooler in the base shelf of the cart.
In future research, the best cooling method for product transportation should be investigated to better protect perishable products. A report by the USDA (2006) offers a table, entitled “Heat Absorption Characteristics of Various Refrigerator Mediums,” to outline the refrigerator properties of ice, carbon dioxide, and nitrogen.
The second step of preparing the cart includes an expansion of the current picking resources. To assist with picking operations, a product called HangUP® by Monaco LLC is implemented into the operations at the pharmacy (www.hangupbags.com). Refer to Appendix F for product images. The bag products are developed with specific consideration to applications in pharmacies. To prevent undersized bags from slowing the picking process, one sufficiently large bag style is suggested. Three bag size options offered by the company are displayed in Table 11. Current pricing is included. Cost calculations are based on delivery quantities at the observed pharmacy. The quantity of deliveries within one day is approximately 60 delivery orders. The quantity of bags suggested is 80. This will provide sufficient supplies for highest volume days of delivery and replacement of some damaged bags.

<table>
<thead>
<tr>
<th>Bag Style</th>
<th>Width</th>
<th>Height</th>
<th>Unit Price</th>
<th>Quantity</th>
<th>Est. Total (+ shipping)</th>
</tr>
</thead>
<tbody>
<tr>
<td>107</td>
<td>14”</td>
<td>21”</td>
<td>$1.13</td>
<td>80</td>
<td>$90.56</td>
</tr>
<tr>
<td>109</td>
<td>16”</td>
<td>25”</td>
<td>$1.74</td>
<td></td>
<td>$138.80</td>
</tr>
<tr>
<td>111</td>
<td>20”</td>
<td>25”</td>
<td>$1.74</td>
<td></td>
<td>$138.80</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product</th>
<th>Unit Price</th>
<th>Quantity</th>
<th>Est. Total (+ shipping)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bag Caddy</td>
<td>$15</td>
<td>4</td>
<td>$60.00</td>
</tr>
</tbody>
</table>

Table 11: HangUP® Product Options

To organize and prevent damage to empty bags, a “Bag Caddy” is recommended. Each Bag Caddy holds 40 bags. Two caddies are assigned to the preparation area of the picking cart to supply empty bags. More caddies may be useful on-board a vehicle, to retain empty bags after a delivery is made. To prepare the picking cart, bags are removed from the Bag Caddy and a Delivery Order is placed within the bag. The order and bag are then hung on the picking cart in the order of delivery. Once the preparation finishes, orders are ready to be picked.
The delivery personnel begin picking by checking orders to determine if any OTC products are listed. If so, the delivery personnel move the cart to the OTC area and selects the necessary items, placing them in the appropriate order bag. OTC products are assumed to have packaging that is virtually resilient to handling concerns. Therefore it is reasonable to position OTC to bear the most weight products within a unitized order. Following the OTC area, the delivery personnel travel with the picking cart to the Pharmacy. The delivery personnel start by selecting all orders from the Pharmacy bins and placing within the order bags. While placing the orders, the delivery personnel should be aware of any orange ticket attached to an order (Figure 19). Orange tickets indicate that part of a customer’s order is stored within a refrigerator area; the Pharmacy refrigerator and/or the Clean Room refrigerator. If there is an orange ticket for the Pharmacy refrigerator, the delivery personnel retrieve those products first. Next, if there is an orange ticket for the Clean Room refrigerator, the delivery personnel travel with the picking cart to the Clean Room.

*Figure 19: Delivery Personnel SOP (Part 3)*
The picking and packing process ends after either the Pharmacy or Clean Room locations, depending on the presence of an orange ticket(s). Loading of the customer orders follows.

5.1.2 Loading

From the picking and packing operation, the bags are hung on the cart in delivery order. The bag order should be maintained and loaded in the vehicle following a FIFO processing scheme. Bags are loaded using the side door (assuming a standard cargo van). The first delivery should be placed nearest the driver-side door, with the last delivery closest to the side door. To load the bags, the bar holding the bags is un-hooked from the delivery cart frame and hung from the frame suspended inside of the vehicle. The frames within the vehicle are crossbars, parallel to the dashboard, located behind the driver and passenger seats (see Figure 20). If a cooler has been used, the cooler is placed between the driver and passenger seats if space permits, otherwise behind the crossbars.

![Figure 20: Vehicle Loading System](image)
5.1.3 Residual Processing

Delivery personnel are expected to use a GPS to guide the vehicle to the next delivery location. As the delivery personnel arrive to deliver the next order, the order is retrieved from the cross-bar along with any contents stored in the cooler. Once an order has been delivered, the empty HangUP® bag is returned to the on-board Bag Caddy. Prior to returning to the pharmacy, delivery personnel refill the gas tank of their vehicle for the next day of delivery. Empty order bags are returned to the Bag Caddy at the picking cart preparation location, and the empty bars are returned to the picking cart from the vehicle. If a cooler was used, the ice must be dumped in a location away from foot-paths before storing the cooler in a place to air-dry. Following this standardized procedure offers consistency in daily processing.

5.2 Routing Program

At the case study pharmacy, the current vehicle routing was scheduled by experienced delivery personnel. The delivery personnel sectioned deliveries into geographical groups. The number of groups created was determined by the number of vehicles available for delivering orders on that particular day. A delivery person spent an estimated 15 minutes determining the sequence to deliver to customers within their assigned route. Current decisions about route assignments and sequence of travel were subjective.
5.2.1 Code Validation

The routing program was validated in two stages. The first stage was a validation of the Farthest Insertion algorithm used a sample TSP problem featuring asymmetrical travel costs (Syslo, 1983). The second round of validation was completed for the petals created by the VRP section of code. The problem used to validate VRP was taken from the discussion of Ryan et al. (1993).

5.2.2 Processing Time

The computer specifications for testing of the routing program were as follows: HP Pavilion dm4 Notebook PC, Intel® Core™ i5-2410M CPU @ 2.30GHz, 4.00 GB RAM, 64-bit OS. The addresses used for evaluation were taken as an excerpt from a list of 500 postal addresses in the United States (Dunning, 2013). The addresses used are given in Appendix E. The criterion for location selection was to come from the same state. The addresses within a state were sorted by Zip Code. A selection of 100 addresses was taken from the consecutive list of Zip Codes. Duplicate addresses were removed prior to the selection of the sample addresses.

The limitations placed on routing included a route duration of 480 minutes and a delivery capacity of 20 units. Each location was assumed to possess a demand for one unit. Processing time was defined as the time to generate the “Routing Travel Matrix” and “Petal Bearing Array,” and then process all heuristic components of both the Petal and Farthest Insertion algorithms. The code for all components of processing time was linked together to prevent any user delay. Time to create the “Master Travel Matrix” and
“Master Bearing Array” was not considered in the processing time. Table 12 displays the computer processing time results for three different states.

<table>
<thead>
<tr>
<th>Number of Deliveries</th>
<th>Sample</th>
<th>Average</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maryland</td>
<td>Michigan</td>
<td>New York</td>
</tr>
<tr>
<td>10</td>
<td>0.54</td>
<td>0.53</td>
<td>0.51</td>
</tr>
<tr>
<td>20</td>
<td>2.23</td>
<td>2.65</td>
<td>5.66</td>
</tr>
<tr>
<td>30</td>
<td>5.25</td>
<td>6.73</td>
<td>9.30</td>
</tr>
<tr>
<td>40</td>
<td>8.67</td>
<td>10.50</td>
<td>20.28</td>
</tr>
<tr>
<td>50</td>
<td>17.55</td>
<td>17.20</td>
<td>27.72</td>
</tr>
<tr>
<td>60</td>
<td>24.93</td>
<td>20.58</td>
<td>46.76</td>
</tr>
<tr>
<td>70</td>
<td>37.76</td>
<td>28.56</td>
<td>57.83</td>
</tr>
<tr>
<td>80</td>
<td>48.94</td>
<td>34.98</td>
<td>81.40</td>
</tr>
<tr>
<td>90</td>
<td>65.99</td>
<td>46.22</td>
<td>95.45</td>
</tr>
<tr>
<td>100</td>
<td>69.11</td>
<td>45.11</td>
<td>131.41</td>
</tr>
</tbody>
</table>

*Table 12: Sample Processing Times (Minutes)*

5.2.3 Program Comparison (Version 1 vs. Version 2)

The routing program was tested in two versions. The purpose for testing was to quantify the differences in processing time and result accuracy. The first version followed the procedure defined by Foster and Ryan (1976). The second version differs in one regard: the TSP heuristic was only applied on the best-found petal routing solution instead of on every petal routing solution. The methodology followed by Version 1 is most accurate among the two, but due to \( n - 1 \) fewer TSP iterations, Version 2 has shorter processing times. The accuracy measurement was used as an objective value to determine the balance between total travel time and computer processing time.

Though comparison was dependent on the inputted addresses and constraints, three sets of delivery locations were simulated with the routing program. Organized by Zip Code, 10-100 locations were taken from three states (Maryland, “1 (MA)”; Michigan
“2 (MI)”; and New York, “3 (NY)”). Maryland and Michigan were selected to simulate a mixture of rural and suburban setting. All locations within New York State were considered within the urban setting of New York City. The same delivery locations were inputted to each version. Computer testing conditions remained constant. Constraining values on each route included total travel time and vehicle capacity; $T_v = 8 \text{ hours} = 480 \text{ minutes}, K_v = 20 \text{ units}$, respectively. Table 13 summarizes the design of experiment and the data collected.

<table>
<thead>
<tr>
<th></th>
<th>1 (MA)</th>
<th>2 (MI)</th>
<th>3 (NY)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver. 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ver. 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ver. 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ver. 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 13: Experimental Design*

The heuristic procedure was exponentially more time consuming as the number of delivery locations increased. Processing time, shown in Figure 21, assumed the updated “Master Travel Matrix” and “Master Bearing Array” already existed in the *Customer Database* file. The time reflects the daily processing time for generation of the vehicle routes to handle the number of delivery locations.
Each resulting processing time arc was fitted with an exponential curve. Table 14 displays the leading coefficients, $c$ and $k$, which were direct inputs to equation (25). Based on the values of $R^2$, the fitted equations had a strong relationship with the processing times collected.
\[ f(x) = ce^{kx} \quad (25) \]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 (MA)</td>
<td>2 (MI)</td>
</tr>
<tr>
<td></td>
<td>0.8407</td>
<td>1.1277</td>
</tr>
<tr>
<td></td>
<td>0.5048</td>
<td>0.4330</td>
</tr>
<tr>
<td></td>
<td>0.91288</td>
<td>0.84693</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Version 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 (MA)</td>
<td>0.0672</td>
<td>0.0650</td>
</tr>
<tr>
<td></td>
<td>0.4512</td>
<td>0.4506</td>
</tr>
<tr>
<td></td>
<td>0.94950</td>
<td>0.93093</td>
</tr>
</tbody>
</table>

*Table 14: Exponential Equation Coefficients and $R^2$ Values*

Table 15 displays the results for total travel time ($t_{\text{total}}$). A known relationship between the two versions is that *Version 1* will consistently provide either the same or better results than *Version 2* ($t_{\text{total}}^{\text{ver.1}} \geq t_{\text{total}}^{\text{ver.2}}$). Reviewing the final routes from all three location regions, the binding constraints differed. All routes from Maryland and all routes from Michigan were bound by the route duration ($T_r$) constraint. All routes from New York were bound by the capacity ($K_r$) constraint.

<table>
<thead>
<tr>
<th>Number of Deliveries</th>
<th>1 (MA)</th>
<th>2 (MI)</th>
<th>3 (NY)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 (MA)</td>
<td>2 (MI)</td>
<td>3 (NY)</td>
</tr>
<tr>
<td>10</td>
<td>246</td>
<td>246</td>
<td>283</td>
</tr>
<tr>
<td>20</td>
<td>681</td>
<td>688</td>
<td>591</td>
</tr>
<tr>
<td>30</td>
<td>988</td>
<td>1000</td>
<td>923</td>
</tr>
<tr>
<td>40</td>
<td>1163</td>
<td>1169</td>
<td>1292</td>
</tr>
<tr>
<td>50</td>
<td>1310</td>
<td>1316</td>
<td>1792</td>
</tr>
<tr>
<td>60</td>
<td>1532</td>
<td>1572</td>
<td>2473</td>
</tr>
<tr>
<td>70</td>
<td>2054</td>
<td>2101</td>
<td>2800</td>
</tr>
<tr>
<td>80</td>
<td>2194</td>
<td>2195</td>
<td>3655</td>
</tr>
<tr>
<td>90</td>
<td>2396</td>
<td>2444</td>
<td>4005</td>
</tr>
<tr>
<td>100</td>
<td>3144</td>
<td>3148</td>
<td>8357</td>
</tr>
</tbody>
</table>

*Table 15: Total Travel Time (Minutes)*
Using Version 1 program output as a baseline, total travel time output was used to determine a percent difference between the models. The percent difference calculation (26) demonstrated the reduction in total travel time accuracy of Version 2 within the simulated scenario. Table 16 includes shaded trials for runs where only one route was created. Differences were only present between the two methods when more than one route was constructed.

\[
Percent \text{ Difference} = \frac{t_{\text{total}}^{\text{ver.2}} - t_{\text{total}}^{\text{ver.1}}}{t_{\text{total}}^{\text{ver.1}}} \quad (26)
\]

A noticeable difference existed in the percent difference in total travel time of Versions 1 and 2 among the different sampling locations. New York was much higher than that of Maryland and Michigan. A suggested relationship existed for the urban setting, or capacity-constrained results. As the number of delivery locations increased, the percent differences decreased. Results for Maryland and Michigan remained below 2.61% for all but the test point for 20 delivery locations in Michigan.

<table>
<thead>
<tr>
<th>Number of Deliveries</th>
<th>1 (MA)</th>
<th>2 (MI)</th>
<th>3 (NY)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>20</td>
<td>1.03%</td>
<td>6.60%</td>
<td>0.00%</td>
</tr>
<tr>
<td>30</td>
<td>1.21%</td>
<td>0.87%</td>
<td>26.76%</td>
</tr>
<tr>
<td>40</td>
<td>0.52%</td>
<td>0.39%</td>
<td>22.75%</td>
</tr>
<tr>
<td>50</td>
<td>0.46%</td>
<td>0.95%</td>
<td>17.97%</td>
</tr>
<tr>
<td>60</td>
<td>2.61%</td>
<td>0.40%</td>
<td>15.29%</td>
</tr>
<tr>
<td>70</td>
<td>2.29%</td>
<td>0.86%</td>
<td>13.33%</td>
</tr>
<tr>
<td>80</td>
<td>0.05%</td>
<td>0.74%</td>
<td>12.37%</td>
</tr>
<tr>
<td>90</td>
<td>2.00%</td>
<td>0.67%</td>
<td>12.96%</td>
</tr>
<tr>
<td>100</td>
<td>0.13%</td>
<td>0.24%</td>
<td>12.02%</td>
</tr>
</tbody>
</table>

*Table 16: Percent Difference in Total Travel Time (Version 1 vs. Version 2)*
The results of analysis were not definitive, but suggest interesting opportunities for further research. First, it is important to identify the factors leading to a decreasing polynomial trend in the urban setting. There is a visual difference between the first two states (duration constrained), and New York, New York (capacity constrained). The percent difference in total travel time may decrease as the number of delivery locations increase. Defining the exact relationship between the number of delivery locations and the resulting accuracy is vital to determining if a threshold exists in the number of delivery locations where Version 2 may become more economical.

Archetti et al. (2011) researched computational special cases of certain digraph shapes. The configurations of the delivery network may have the ability to impact the accuracy and processing time of modeling. Therefore, it may be advantageous for the program user to evaluate the expected accuracy deficits incurred in the considered delivery region. The evaluation can provide the decision-maker the choice between a program with higher accuracy and higher processing times (Version 1) and a program with reduced accuracy and much lower processing times (Version 2). Thorough cost analysis can lead objective conclusions on the best program for the delivery area.
CHAPTER VI

SUMMARY AND CONCLUSIONS

Product delivery is a valuable service for customers, yet potentially costly for the supplying company. The stages of warehouse picking, packing and loading are non-value added operations in the product supply chain. Therefore, reducing the time delivery personnel spend handling product is an advantageous consideration by management.

This research evaluated the activities associated with preparation and execution of delivery services by delivery personnel at a retail pharmacy. The parameters of this research were a single retail pharmacy location that used multiple vehicle routes to service a varying number of customer demand locations. Delivery activities were organized into internal and external order handling operations. Internal order handling consisted of every stage of customer order picking, packing and loading into a vehicle. External order handling was the classification for the process of delivering orders to all customers. The observed pharmacy had no standard operating procedure established for delivery personnel.

Objectives of this research were focused on improving efficiencies of those operations performed by delivery service personnel. There were four specific objectives of this research. (1) To define a generalized model which optimizes delivery service efficiencies. (2) Design a tool that permits delivery personnel to apply theoretical approaches, routinely. (3) Test model results in scenario environments. (4) Acknowledge perspective dimensions for future model development. Components of this research for the first three objectives will be summarized in the remainder of this chapter. The fourth
objective is discussed in Chapter 7. The proposed system components were designed to be highly adaptable to different retail pharmacy arrangements.

6.1 Pick, Pack, and Load

Appendix D defines the proposed organization for internal delivery operations at a pharmacy. Conceptual drawings are provided for a picking cart and in-vehicle storage system. The suspended hanger bar maintains order of customer products by transferring directly from the picking cart into the in-vehicle storage system. Customer orders are fulfilled simultaneously at each pharmacy inventory location. Each customer order is unitized by utilizing a product such as the HangUP® pharmacy bagging system. Items requiring refrigeration are stored in an ice-cooled insulated cooler until arriving to the delivery location. Customer orders are hung in the sequence of delivery. Orders are loaded into the vehicle to support FIFO flow. Empty bags are stored on a “Bag Caddy.”

6.2 Routing Program

A model and associated tool were constructed to increase route efficiencies, organize Delivery Orders, and reduce route planning time. Processing time was a primary consideration for the likelihood of program implementation. The VRP was coded according to the 1-Petal algorithm with the internally called TSP code following the Farthest Insertion procedure.

The objective of the delivery routing problem was to minimize total travel time among all routes. Existing 1-Petal and Farthest Insertion heuristics were specifically
adapted for routing tool construction. Models were adapted to provide acceptable accuracy of results, with a reasonable processing time (according to literature).

The routing tool was built in Microsoft Excel® VBA and staged in two files. The Customer Database file maintains a database of all potential delivery customers, the “Master Travel Matrix” and “Master Bearing Array.” Travel time between two locations was accessed through the Live Traffic feature on the TomTom® website (www.tomtom.com). Data from the file was read by the VRP Petal file after delivery reference values were entered by the delivery personnel. Route constraints included total vehicle capacity and route duration. Upon completion of the route processing, output included ordered customer addresses for each route. Customer Delivery Orders were ordered and used as the picking ticket.

Two versions of code were tested for processing time and relative accuracy between the models. The first version followed the methodology prescribed by literature. The second version greatly reduced the number of iterations by calling the TSP heuristic only once and applying it only to the “best-found” petal set. Differences in solution values for total travel time indicated that using the TSP heuristic on a petal set with an initially higher total travel time (pre-optimization) may result in a better overall solution. Results of the comparison suggested that there may be a threshold in the number of delivery locations where the costs associated with processing time will exceed the cost of decreased route accuracy.

Technical improvements to the routing program could significantly reduce the processing time. The first improvement to the routing heuristic would be to terminate remaining calculations in the process if all delivery locations on the first routing attempt
can be assigned to one route under the stated duration and capacity constraints. When multiple routes are assigned, the storage mechanism proposed by Butt and Ryan (1999) offers to reduce repetitious TSP calculations for previously configured route sets. Regarding the petal routing scheme, anytime a petal begins at a specific node, the resulting petal will always be of the same size. The nodes in the spanning petal set indicate the starting node for the petal algorithm that will return identical routes. Therefore, the storage mechanism approach can be extended to petal routing. For example, if the starting node is $i = 2$ and the current spanning petal set is defined to be $\{2, 8, 9, 12, 15\}$, we may also conclude the results of Table 17.

<table>
<thead>
<tr>
<th>Starting Node ($i$)</th>
<th>Spanning Petal Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>${2, 8, 9, 12, 15}$</td>
</tr>
<tr>
<td>9</td>
<td>${2, 8, 9, 12, 15}$</td>
</tr>
<tr>
<td>12</td>
<td>${2, 8, 9, 12, 15}$</td>
</tr>
<tr>
<td>15</td>
<td>${2, 8, 9, 12, 15}$</td>
</tr>
</tbody>
</table>

*Table 17: Example Spanning Petal Set*

6.3 Review of Research Objectives

The objective of evaluating delivery personnel operations at a retail pharmacy was completed from the perspective of internal and external operations. The proposed system included a tool for delivery personnel. Theoretical approaches were applied to achieve improved service operations. The program was tested in three simulated scenarios to better understand the impact of delivery location configurations. Finally, two versions of the delivery program were tested to investigate a compromise in accuracy for reduced processing time.
CHAPTER VII

FUTURE RESEARCH

A routing model, programmed in Microsoft Excel®, offers accessibility to retail pharmacies seeking to improve efficiencies in delivery operations. Further program design can customize the developed routing program’s ability to adapt and satisfy specific needs of another company.

There are several opportunities for program expansion and evaluation through future research. The current model is theoretical in construction. One assumption is that fleet size is infinite. The constraints considered include vehicle capacity and route duration. A desirable feature of construction would instead constrain the number of vehicles to the number actually available for use. The objective of minimizing total travel time would remain unchanged. Management could use the model and select the number of routes to generate, where route duration is then unconstrained.

The vehicle capacity constraint is another aspect to consider. The vehicle capacity is virtually infinite for pharmaceutical products. Expansion of the model could accommodate convalescent aides (e.g. beds, wheelchairs, etc…) and oxygen products. These larger products introduce binding restrictions on vehicle capacity. An appropriate capacity model could be incorporated into the routing program.

The current model also assumes all demand locations are known prior to vehicle departure. The results of the model provide a static solution to the known delivery points. To possibly better satisfy the needs of some pharmacies, a dynamic model could be developed to handle sudden, unplanned delivery scenarios. These demand occurrences
could reflect customer orders entering the system the same day, or same hour, of needing delivered.

To better serve customers in the delivery process, time-window analysis may be employed. Adding a time-window feature to the model would allow the customer to be assigned an estimated time of delivery. From a research approach, time-window analysis may allow a pharmacy to generalize regions to a specific day of the week. At the observed pharmacy, there are 80 reoccurring weekly orders that utilize the DISPILL® (www.dispill-usa.com) packaging system. The orders are split between “will-call” and “delivery.” Sixty of these orders are considered “delivery” and are delivered throughout the week, Monday through Thursday. The customer receives their order on the same day each week. The day to deliver these reoccurring orders should be incorporated into the model. A scaled model could be capable of scheduling weekly order flow, based on expected daily order flow.

The exact costs of providing the delivery service remain unknown. The program optimizes on route delivery time, thus focusing on reducing the labor expenditure. Thorough financial evaluation of delivery costs would include several factors (Table 18). The objective of future analysis may be to increase the profitability of the delivery service. Another system unknown is the rate of error occurrence in product handling by delivery personnel. Errors can be costly to the company and potentially harmful to the customer.

Certain products offered for delivery by a pharmacy may require special handling. One set of products are temperature-sensitive pharmaceuticals. Improper handling may compromise the effectiveness of the medicine. More research may resolve special
handling aspects such as the best way to keep products cool. Another product is the compressed oxygen units. This particular product poses physical danger to those in proximity when mishandled. The process of handling and transporting should offer security to the product to prevent potential dangers.

<table>
<thead>
<tr>
<th>Cost Category</th>
<th>Imposing Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vehicle</td>
<td>1. Fuel consumption</td>
</tr>
<tr>
<td></td>
<td>2. Purchase and salvage costs</td>
</tr>
<tr>
<td></td>
<td>3. Operating &amp; maintenance costs</td>
</tr>
<tr>
<td>Labor</td>
<td>1. Pick, pack, and load time</td>
</tr>
<tr>
<td></td>
<td>2. Delivery time</td>
</tr>
<tr>
<td></td>
<td>3. Scheduled break time</td>
</tr>
<tr>
<td></td>
<td>4. Variation in workforce</td>
</tr>
<tr>
<td>Material</td>
<td>1. Vehicle organization</td>
</tr>
<tr>
<td></td>
<td>2. Packing material</td>
</tr>
<tr>
<td></td>
<td>3. Fragile handling equipment (i.e., temperature, packaging)</td>
</tr>
</tbody>
</table>

*Table 18: Recognized Cost Factors of Delivery Services*

In summary, there are many future research opportunities for attaining operational improvements at a retail pharmacy, specifically with respect to delivery services. A valuable addition to the proposed routing program would be a parameter limiting the number of routes created by relaxing the duration constraint. The capacity constraint could be expanded to accommodate the space utilization of larger products, such as convalescent aid and oxygen units. The existing routing program routine solves for a static solution, yet a system may face instances of dynamic decision-making. Time-windows offer another point for system improvements, where benefits may be seen by both the customer and pharmacy. This research did not acknowledge financial aspects of providing delivery services. It is important to understand the costs of providing a service.
to maintain competitive advantage. Finally, pharmacy products introduce special handling considerations for the delivery process.

These are some of many future research opportunities, as there is little relevant existing literature for product delivery services by a retail pharmacy.
Appendix A

Order Flow Chart
The following process map summarizes the processing of customer orders and products through the pharmacy system. The responsibilities of delivery personnel are boxed.
Appendix B

MICROSOFT EXCEL ® VBA CODE – Customer Database
The following subroutines are called into the Customer Database file in the routing program. The routines include matrix generators and database management tools.

```vba
Sub generateNewTravelMatrix()

    ' This subroutine generates an entirely new travel matrix with
    ' angle values for the petal algorithm. This option is ideal
    ' for starting out or to give a thorough update. Note, that
    ' editing of customers should be done via the 'Delete Customer'
    ' button located on the "New Customer" tab.

    Application.ScreenUpdating = False

    Dim addressArray() As Variant
    Dim data As Collection
    Dim a As String
    Dim b As String
    Dim aLong As String
    Dim aLat As String

    ' determine the size of matrix to generate (a line for the Depot
    ' is added by the "+1")
    nLocations = Range("customerCount").Value + 1

    ' a copy of the Depot ("Customer Database," Row 1) and customer
    ' addresses are written to the "Travel Matrix Database" tab to
    ' form the header column and row of the travel matrix
    Worksheets("Customer Database").Select
    Range("addressAnchor").Select
    Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 1, 0)).Copy
    Worksheets("Travel Matrix Database").Select

    ' header column
    Range("matrixAnchor").Offset(0, -1).PasteSpecial Paste:=xlPasteValues

    ' header row
    Range("matrixAnchor").Offset(-1, 0).Select
    Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(0, nLocations - 1)).Select
    With Selection
        False, Transpose:=True
    End With

    ' the following two-stage loop walks through the matrix one cell
    ' at a time to determine the travel information FROM (row) a
    ' certain location TO (column) another location. Note that the
    ' matrix generated is conducive to asymmetry.
```
For i = 1 To nLocations  'i controls column
    For j = 1 To nLocations 'j controls row

' selects the cell to contain the travel information
Range("matrixAnchor").Offset(j - 1, i - 1).Select

'a is the variable for the FROM location
a = Selection.Offset(0, -i).Value

' log latitude and longitude for each location
If i = 1 Then
    ' using the TomTom website, the latitude and
    ' longitude are retrieved for each location
    With CreateObject("MSXML2.XMLHTTP")
        & a & "/json/1e2099c7-eaa9-476b-aac9-b20dc7100af1;language=en ;map=basic" _
        .send
        Do: DoEvents: Loop Until .readyState = 4
        aLong = Split(Split(.responseText, "longitude":"))(1), ",")
        aLat = Split(Split(.responseText, "latitude":"))(1), ",")
        .abort
    End With

' j is a counter for a location number
'(Depot = 1, Customer n = n + 1)
Range("matrixAnchor").Offset(j - 1, -5).Value = j

'records location latitude
Range("matrixAnchor").Offset(j - 1, -4).Value = aLat

'records location longitude
Range("matrixAnchor").Offset(j - 1, -3).Value = aLong

'the bearing from Depot to customer is
' calculated with straight East assumed as
' 0 degrees, increasing in a counter-
' clockwise direction
If j > 1 Then
    Range("matrixAnchor").Offset(j - 1, -2).FormulaR1C1 = "= _
    DepHdg(R[" & 1 - j & "]C[-2],R[" & 1 - j & "]C[-1],RC[-2],RC[-1])"
End If

End If

'b is the variable for the TO location
b = Selection.Offset(-j, 0).Value

'GetTimeAndDistance is a formula available through another macro (Module 2). Again, the TomTom site is accessed to get travel time (could easily be changed to travel distance) from a, to b. The 'travel magnitude is recorded in the matrix.
If i = j Then
Else
    Set data = GetTimeAndDistance(a, b, True, True, vbThursday, 960)
    Range("matrixAnchor").Offset(j - 1, i - 1).Value = data("Time") * 0.016666666667
End If

Next j
Next i

'returns to the main tab before finishing
Worksheets("New Customer").Select
Application.ScreenUpdating = True

'the user is notified that the program has finished
MsgBox "A new travel matrix has been configured in the Travel Matrix Database tab.,", "Matrix Complete"
End Sub

Sub updateTravelMatrix()
'The purpose of this subroutine is to update an existing 'travel matrix for newly added customers to the "Customer 'Database." The program does not change or update any existing 'matrix information.
Application.ScreenUpdating = False
Dim nCustomers As Integer
Dim addressArray() As Variant
Dim data As Collection
Dim a As String
Dim b As String
Dim aLong As String
Dim aLat As String
'determine the size of matrix to generate (a line for the Depot
'is added by the "+1").
   nLocations = Range("customerCount").Value + 1

'to know how many lines exist in the matrix, the variable
'nInMatrix' is found for comparison to 'nLocations'
   With Worksheets("Travel Matrix Database").UsedRange
      nInMatrix = .Rows.Count - 1 'the first row has addresses
   End With

'a copy of the new customer addresses are written to the
"Travel Matrix Database" tab, adding to the bottom of the
'header column and row of the travel matrix
   Worksheets("Customer Database").Select
   Range("addressAnchor").Select
   Range(ActiveCell.Offset(nInMatrix, 0), ActiveCell.Offset(nLocations - 1, 0)).Copy
   Worksheets("Travel Matrix Database").Select

'header column
   Range("matrixAnchor").Offset(nInMatrix, -1).PasteSpecial Paste:=xlPasteValues

'header row
   Range("matrixAnchor").Offset(-1, nInMatrix).Select
   With Selection
         False, Transpose:=True
   End With

'the following two-stage loop walks through the matrix one cell
'at a time to determine the travel information FROM (row) a
certain location TO (column) another location. Note that the
'matrix generated is conducive to asymmetry.
   For i = 1 To nLocations   'i controls column
      For j = 1 To nLocations 'j controls row

'selects the cell to contain the travel information
   Range("matrixAnchor").Offset(j - 1, i - 1).Select

   If i = j Then
      Else
         'new travel values are only found for blank cells
         If Selection = BLANK Then

'a is the variable for the FROM location
   a = Selection.Offset(0, -i).Value
'log latitude and longitude for each location
If i = 1 Then

'using the TomTom website, the latitude and
'longitude are retrieved for each location
With CreateObject("MSXML2.XMLHTTP")

'GeoCode aEnd
.Open "GET," "http://routes.tomtom.com/lbs/services/geocode/1/query/" &_
a & "/json/1e2099c7-eeea9-476b-aac9-_
b20dc7100af1;language=en;map=basic"
.send
Do: DoEvents: Loop Until .readyState = 4
aLong = Split(Split(.responseText, "longitude":"")(1), ",") (0)
aLat = Split(Split(.responseText, "latitude":"")(1), ",") (0)
.abort
End With

'j is a counter for a location number
'(Depot = 1, Customer n = n + 1)
Range("matrixAnchor").Offset(j - 1, -5).Value = j

'records location latitude
Range("matrixAnchor").Offset(j - 1, -4).Value = aLat

'records location longitude
Range("matrixAnchor").Offset(j - 1, -3).Value = aLong

'the bearing from Depot to customer is
'calculated with straight East assumed as
'0 degrees, increasing in a counter-
'clockwise direction
If j > 1 Then

Range("matrixAnchor").Offset(j - 1, -2).FormulaR1C1 = 
"=DepHdg(R[" & _
1 - j & "]C[-2],R[" & 1 - j & "]C[-1],RC[-2],RC[-1])"
End If

End If

'b is the variable for the TO location
b = Selection.Offset(-j, 0).Value

'GetTimeAndDistance is a formula available through
'another macro (Module 2). Again, the TomTom site
'is accessed to get travel time (could easily be
'changed to travel distance) from a, to b. The
'travel magnitude is recorded in the matrix.
Set data = GetTimeAndDistance(a, b, True, True, vbThursday, 960)
Range("matrixAnchor").Offset(j - 1, i - 1).Value = data("Time") * 0.016666666667
End If
End If

Next j
Next i

returns to the main tab before finishing
Worksheets("New Customer").Select
Application.ScreenUpdating = True

the user is notified that the program has finished
MsgBox "An updated travel matrix has been configured in the Travel Matrix Database tab.," , "Matrix Complete"

End Sub

(www.MrExcel.com)

Public Function GetTimeAndDistance(aEnd As String, _
bEnd As String, _
Optional avoidTraffic As Boolean, _
Optional includeTraffic As Boolean, _
Optional DayofWeek As VbDayOfWeek, _
Optional time As Long) As Collection
Dim aLong   As String
Dim aLat    As String
Dim bLong   As String
Dim bLat    As String
Dim url     As String
Dim ret     As Collection
Dim Days    As Variant

Set ret = New Collection

"friday," "saturday")

With CreateObject("MSXML2.XMLHTTP")
'GeoCode aEnd
.Open "GET," "http://routes.tomtom.com/lbs/services/geocode/1/query/" & aEnd _
& "/json/1e2099c7-eea9-476b-aac9-b20dc7100af1;language=en;map=basic"
.send
Do: DoEvents: Loop Until .readyState = 4
aLong = Split(Split(.responseText, "longitude"";"')(1), ',')'(0)
aLat = Split(Split(.responseText, "latitude"";"')(1), ',')'(0)
.abort

'GeoCode bEnd
.Open "GET," "http://routes.tomtom.com/lbs/services/geocode/1/query/" & bEnd _
& "/json/1e2099c7-eea9-476b-aac9-b20dc7100af1;language=en;map=basic"
.send
Do: DoEvents: Loop Until .readyState = 4
bLong = Split(Split(.responseText, "longitude"";"')(1), ',')'(0)
bLat = Split(Split(.responseText, "latitude"";"')(1), ',')'(0)
.abort

'Get Route Info
.Open "GET," "http://routes.tomtom.com/lbs/services/route/1/" _
& aLat & ',' & aLong & ';' & bLat & ',' & bLong & _
'/Quickest/json/1e2099c7-eea9-476b-aac9-b20dc7100af1;language=en;' _
& "avoidTraffic=" & LCase(avoidTraffic) _
& ";includeTraffic=" & LCase(includeTraffic) _
& ";day=" & Days(DayofWeek) _
& ";time=" & IIf(time = 0, "now," time) _
& ";iqRoutes=2;trafficModelId=-1;map=basic"
.send
Do: DoEvents: Loop Until .readyState = 4
ret.Add Val(Split(.responseText, "totalDistanceMeters"";"')(1)), "Distance"
ret.Add Val(Split(.responseText, "totalTimeSeconds"";"')(1)), "Time"
.abort
End With

Set GetTimeAndDistance = ret

End Function

(www.MrExcel.com)

'Returns the bearing (borrowed code: http://www.mrexcel.com/forum/excel-
questions/626081-calculate-bearing-direction-between-2-coordinates.html)
Public Function DepHdg(ByVal lat1 As Double, ByVal lon1 As Double, _
ByVal lat2 As Double, ByVal lon2 As Double) As Double
Const pi As Double = 3.14159265358979
Const D2R As Double = pi / 180#
lat1 = D2R * lat1
lat2 = D2R * lat2
lon1 = D2R * lon1
lon2 = D2R * lon2

due to the curvature of the earth, bearing is used rather
than a general trigonometric calculation.
DepHdg = WorksheetFunction.Atan2(Cos(lat1) * Sin(lat2) - Sin(lat1) * Cos(lat2) * _
   Cos(lon1 - lon2), Sin(lon2 - lon1) * Cos(lat2)) / D2R
If DepHdg < 0 Then DepHdg = DepHdg + 360
End Function

Sub recordNewCustomer()
' the following macro takes new customer information and writes
' it to the end of the customer list in the "Customer Database"

    Application.ScreenUpdating = False

    Dim nCustomers As Integer

    nCustomers = Range("customerCount").Value + 1
    Range("newCustomerInformation").Select
    Range(ActiveCell.Offset(0, 1), ActiveCell.Offset(0, 4)).Copy
    Worksheets("Customer Database").Select
    Range("lastNameAnchor").Offset(nCustomers, 0).PasteSpecial xlPasteValues
    Range("phoneAnchor").Offset(nCustomers, 1).Value = ":=R[-1]C+1"

    Worksheets("New Customer").Select

    Application.CutCopyMode = False
    Application.ScreenUpdating = True
End Sub

Sub clearEntries()
' Cleanliness: previously entered customer information is
' cleared for new customer information.

    Application.ScreenUpdating = False

    Range("newCustomerInformation").Select
    Range(ActiveCell.Offset(0, 1), ActiveCell.Offset(0, 4)).ClearContents

    Application.ScreenUpdating = True
End Sub
Sub deleteCustomer()
' To keep a clean database, customers can be removed from the
' database list through the entry of last name or reference
' code information for that particular customer. The matrix is
' also edited in the "Travel Matrix Database"

Dim x As String

Application.ScreenUpdating = True

nCustomers = Range("customerCount").Value
x = Range("deleteCustomer").Value

Worksheets("Customer Database").Select
For i = 1 To 2
    For j = 1 To nCustomers
        y = Range("lastNameAnchor").Offset(j, i - 1).Value
        If x = y Then
            Rows(j + 1).Delete
            k = j
            i = 2
            j = nCustomers
        Else
            If i = 2 Then
                If j = nCustomers Then
                    MsgBox "Customer information not found."
                    Exit Sub
                End If
            End If
        End If
    Next j
Next i

Range("phoneAnchor").Offset(0, 1).Value = 1
For i = 1 To nCustomers - 1
    Range("phoneAnchor").Offset(i, 1).Value = Range("phoneAnchor")._
        Offset(i - 1, 1).Value + 1
Next i

Worksheets("Travel Matrix Database").Select
Rows(k + 2).Delete
Columns(k + 6).Delete
Range("A2").Value = 1
For i = 1 To nCustomers - 1
    Range("A2").Offset(i, 0).Value = Range("A2").Offset(i - 1, 0).Value + 1
Next i

Worksheets("New Customer").Select
Range("deleteCustomer").ClearContents

Application.ScreenUpdating = True

End Sub
Appendix C

MICROSOFT EXCEL ® VBA CODE – Routing Program
The following subroutines are the primary functions of the routing program. The first subroutine creates a travel matrix for the locations to be routed. The second subroutine is the petal routing procedure. Finally, the traveling salesman problem subroutine is stated.

**Sub generateMatrix()**

'Using the matrix database in the "Thesis_VRP_CustomerResource" file, this program develops a matrix pertaining only to the locations needing a delivery (from user entered information in the "Input" tab.

Application.ScreenUpdating = False

Dim addressArray() As Variant
Dim data As Collection
Dim a As Integer
Dim b As Integer

'The "Raw" worksheet will contain the newly generated travel matrix.
Sheets.Add.Name = "Raw"

'The "angleCache" worksheet manages delivery order through organization of angles for the petal algorithm.
Sheets.Add.Name = "angleCache"

'Determine the number of customers to be included in the matrix generation process. In addition to the customers listed, the Depot is added into the matrix for routing purposes.
Worksheets("Input").Select
Range("customerAnchor").Select
With Selection
    .End(xlDown).Select
End With
nLocations = ActiveCell.Offset(0, -1).Value + 1 'include the depot in the number of locations

'The order of the customers as entered is tracked
Range("nAnchor").Select
Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 2, 0)).Copy
Worksheets("angleCache").Select
Range("B1").PasteSpecial xlPasteValues

'The customer reference number corresponds to the row/column in the matrix database of the other file.
Worksheets("Input").Select
'build the list of FROM customers for the matrix
Worksheets("Raw").Select
Range("A3").PasteSpecial Paste:=xlPasteValues

'adding in the reference for the Depot
Range("A2").Select
Selection.Value = 1
Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 1, 0)).Copy

'list TO customers for the matrix
Range("B1").Select
With Selection
End With

'The number of units to be delivered to each customer plays
'into vehicle capacity constraints during routing.
Worksheets("Input").Select
Range("capAnchor").Select
Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 1, 0)).Copy
Worksheets("angleCache").Select
Range("C1").PasteSpecial xlPasteValues

'For loops permit the matrix generation down through each row
'and then across through each column of locations.
For i = 1 To nLocations    'i controls column
    For j = 1 To nLocations 'j controls row

        If i = j Then
            Else

            'select the cell at the cross-section between the FROM
            'and TO addresses
            Worksheets("Raw").Select
            Range("B2").Offset(j - 1, i - 1).Select

            'a is the FROM location
            a = Selection.Offset(0, -i).Value

            'b is the TO location
            b = Selection.Offset(-j, 0).Value

        End If
    Next j
Next i
'determine the desired row and column number to read
'from the matrix database of the other file
Worksheets("Raw").Select
  temp = Cells(a, b + 2).Address
  Range("B2").Offset(j - 1, i - 1).Select

'GetValue is a macro used to read values from another
'file. It is based on location "temp" relative to the
'matrix anchor on the "Travel Matrix Database" tab of
'the other file.
  Selection = GetValue(path, file, sheet, temp)

If i = 1 Then
  If j > 1 Then
    'while reading travel values, this set allows
    'for the angle values to be included on the
    "angleCache" tab
    Worksheets("angleCache").Select
    temp = Cells(a, 1).Address
    Range("A1").Offset(j - 2, 0).Select
    Selection = GetValue(path, file, sheet, temp)
  End If
End If

End If

Next j
Next i

'formulas are removed from the "angleCache" tab
  Worksheets("angleCache").Select
  Range("A1").Select
    With Selection
    Range(.Offset(0, 0), .Offset(nLocations - 2, 2)).Copy
    Range(.Offset(0, 0), .Offset(nLocations - 2, 2)).PasteSpecial xlPasteValues
    End With

'In preparation for the petal algorithm, the delivery
'locations are sorted in ascending order.
  Call sortDegrees

'Customer reference values are no longer needed. Henceforth,
'customers are referred to by their order on the "Input" tab
'(rather than their listed position in the other file).
  Worksheets("Raw").Select
  Rows(1).Delete
Columns(1).Delete

'Returns the screen to the "Input" tab
    Worksheets("Input").Select

'The user is informed of program completion.
    'MsgBox "Travel matrix and angles complete."

Application.ScreenUpdating = True

End Sub

(Versions 1: Petal)
Sub petalRouting()
 'The petalRouting() subroutine evaluates the delivery locations
 'by using the Petal Algorithm developed for Vehicle Routing
 'Problems. Two forms of route constraints are included in the
 'evaluation: duration and capacity. If any location requires
 'more duration or capacity than permitted, it is given a
 'unique route.

startTime = Timer

Call generateMatrix

Application.ScreenUpdating = False

Dim DNew As Single, DMax As Single, DTotal As Single, DTemp As Single
Dim QCurrent As Integer, QNew As Integer, QMax As Integer
Dim cycOrder(1 To 1000) As Integer, routeSet(1 To 1000) As Integer, Duration(1 To 1000) As Single, Deliveries(1 To 1000) As Integer
Dim nRoute As Integer, nDeliveries As Single
Dim tempDRoute As Single
Dim optDTotal As Single, optNRoutes As Integer, optCycOrder(1 To 1000) As Integer, optRouteSet(1 To 1000) As Integer, optDuration(1 To 1000) As Single, optDeliveries(1 To 1000) As Integer

""temp" is used in the process of developing the matrix while
'the Matrix worksheet will contain the final sub-matrix
    Sheets.Add.Name = "temp"
    Sheets.Add.Name = "Matrix"
    Sheets.Add.Name = "Report"
    Sheets.Add.Name = "C"
    Sheets.Add.Name = "D"
    Sheets.Add.Name = "cost"
'SECTION 1: Inputs

'Number of Locations (Depot = 0, Customer Locations = 1, . , n)
Worksheets("angleCache").Select
Range("A1").Select
Do Until ActiveCell = BLANK
    nLocations = nLocations + 1
    ActiveCell.Offset(1, 0).Select
Loop

'Route Duration (D) - taken from user input
DMax = Worksheets("Input").Range("maxDuration").Value

'Route Capacity (Q) - taken from user input
QMax = Worksheets("Input").Range("maxCapacity").Value

'Cyclic Order - taken from the radially ordered angles
Worksheets("angleCache").Select
Range("A1").Select

'SECTION 2: Body

'Duration Management
The following For loop takes the reference number, listed in Column B, for both the FROM and TO locations. Using the two location values, three travel information points are found.
' 1. Distance from Depot to current location
' 2. Distance from previous location to current location
' 3. Distance from current location to Depot
'All three values are logged into the row for given location 'on the "angleCache" tab.
For j = 1 To nLocations

'assign FROM and TO locations to variables aFrom & bTo
If j = 1 Then
    aFrom = Range("A1").Offset(nLocations - 1, 1).Value
    bTo = Range("A1").Offset(0, 1).Value
Else
    aFrom = Range("A1").Offset(j - 2, 1).Value
    bTo = Range("A1").Offset(j - 1, 1).Value
End If

'look up travel information and store to variables
Worksheets("Raw").Select
DdepotTOcustomer = Range("A1").Offset(0, bTo).Value
DcustomerTOcustomer = Range("A1").Offset(aFrom, bTo).Value
DcustomerTOdepot = Range("A1").Offset(bTo, 0).Value
'log travel information by respective location
Worksheets("angleCache").Select
    Range("A1").Offset(j - 1, 3).Value = DdepotTOcustomer
    Range("A1").Offset(j - 1, 4).Value = DcustomerTOcustomer
    Range("A1").Offset(j - 1, 5).Value = DcustomerTOdepot
Next j

'Capacity Management
'i is the starting location for the sweep
i = 1

'values on the "angleCache" tab are duplicated for the ease of
'progressing through iterations with different starting
'locations
    Range("A1").Select
    Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 1, 5)).Copy
    Selection.Offset(nLocations, 0).PasteSpecial xlPasteValues

'The Do loop allows for routes to formulated from each of the
'n possible starting locations (customer locations). The
'purpose is to gain a more optimal solution that using only
'one starting location.
Do Until i = nLocations + 1
    Worksheets("angleCache").Select
    Range("A1").Select

    'the "cycOrder" array contains ordered location values
    'for the given iteration
    'Deliveries array tracks the number of deliveries in
    'each route.
    'Duration array tracks the duration within each route.
    'routeSet array tracks the first customer location
    'for each route. This code resets the different arrays.
    For j = 1 To nLocations
        cycOrder(j) = ActiveCell.Offset(i + j - 2, 1).Value
        Deliveries(j) = 0
        Duration(j) = 0
        routeSet(j) = 0
    Next j

    'variable to count number of routes needed
    nRoutes = 1

    'variable for used capacity and duration on current
    'route
QCurrent = 0
DCurrent = 0

'The nDeliveries variable tracks the number of
deliveries within a certain route as the route is
built.
nDeliveries = 0

'The starting delivery location will be the first
location in the first route.
routeSet(1) = Range("A1").Offset(i - 1, 1).Value

'The For loop with counter 'k' incrementally assigns
the next location to either the existing route or a
'new route.
For k = 1 To nLocations

C = i + k - 2

'QNew is the would-be incoming capacity needed to
'add the new location to an existing route.
QNew = Range("A1").Offset(C, 2).Value

If (QNew + QCurrent) > QMax Then

'In the case of insufficient capacity:

'The following conditional statement assigns
'those customers exceeding the capacity
'contraint to a separate route.
If QCurrent = 0 Then

nRoutes = nRoutes + 1
routeSet(nRoutes) = Range("A1").Offset(C + 1, 1).Value

QCurrent = 0
DCurrent = 0

Duration(nRoutes - 1) = Range("A1").Offset(C, 3).Value + _
Range("A1").Offset(C, 5).Value
Deliveries(nRoutes - 1) = 1

nDeliveries = 0

Else
'If the Duration constraint is exceeded, the user is notified and the program terminates.
"If QCurrent = 0 Then
nRoutes = nRoutes + 1

'Record the incoming location as the first location for the next route.
routeSet(nRoutes) = Range("A1").Offset(C, 1).Value

'The new capacity that was needed becomes the current capacity of the new route.
QCurrent = QNew

'The duration and number of deliveries of the closed route is added to the respective arrays.
Duration(nRoutes - 1) = DCurrent
Deliveries(nRoutes - 1) = nDeliveries

'The duration of the new route includes the travel from the Depot to the new location, 'and the travel, then, back to the Depot.
DCurrent = (Range("A1").Offset(C, 3).Value) + (Range("A1").Offset(C, 5).Value)

'The number of deliveries is reset to 1 for the new route.
nDeliveries = 1

End If

Else
'The current vehicle has sufficient capacity. 'Now the duration constraint is tested.

If DCurrent = 0 Then
'No current duration of the route means an incoming duration of the trip to go 'from the Depot, to the new customer, and 'then back to the Depot.
DNew = (Range("A1").Offset(C, 3).Value) + (Range("A1").Offset(C, 5).Value)

Else
'If the current route already includes customers, then the new duration value to 'consider includes the travel time between
'the most recently added customer and this
'new customer, the travel time from the
'new customer back to the Depot, less the
'travel time from the previous customer to
'the Depot.

\[ D_{New} = (Range("A1").Offset(C, 4).Value) + (Range("A1").Offset(C, 5).Value) - (Range("A1").Offset(C - 1, 5).Value) \]

End If

If (DNew + DCurrent) > DMax Then

'The following conditional statement
'assigns those customers exceeding the
'duration constraint to a separate route.

If DCurrent = 0 Then

\[ n_{Routes} = n_{Routes} + 1 \]

routeSet(nRoutes) = Range("A1").Offset(C + 1, 1).Value

QCurrent = 0
DCurrent = 0

Duration(nRoutes - 1) = Range("A1").Offset(C, 3).Value + Range("A1").Offset(C, 5).Value
Deliveries(nRoutes - 1) = 1

nDeliveries = 0

Else

'In the case of insufficient duration:

'REPEATED FROM INSUFFICIENT CAPACITY (up)
'Add in a new route

nRoutes = nRoutes + 1

'Record the incoming location as the first
'location for the next route.

routeSet(nRoutes) = Range("A1").Offset(C, 1).Value

'The new capacity that was needed becomes the
'current capacity of the new route.

QCurrent = QNew
'The duration and number of deliveries of the
'closed route is added to the respective
'arrays.
Duration(nRoutes - 1) = DCurrent
Deliveries(nRoutes - 1) = nDeliveries

'The duration of the new route includes the
'travel from the Depot to the new location,
'and the travel, then, back to the Depot.
DCurrent = (Range("A1").Offset(C, 3).Value) + (Range("A1").Offset(C, _ 5).Value)

'The number of deliveries is reset to 1 for the
'new route.
nDeliveries = 1
End If

Else
'There is sufficient capacity and duration
'to add the new location to the existing
'route.

'Increase the number of deliveries on route
nDeliveries = nDeliveries + 1

'The current capacity and duration become
'the previous values with the incoming
'values associated with adding the new
'location to the route.
QCurrent = QCurrent + QNew
DCurrent = DCurrent + DNew

End If

End If

If k = nLocations Then
'The last location for this iteration has been
'implemented and the routes should be closed
'as there are no more points to consider.
Duration(nRoutes) = DCurrent
Deliveries(nRoutes) = nDeliveries
End If
Next k

'The iteration at the current starting location has
'finished processing through the constraints.
'Increments the iteration counter.
\[ i = i + 1 \]

'===============================================================
'=================================================================

Worksheets("Output").Select

'Route durations and number of deliveries are logged
For \( m = 1 \) To nRoutes
  Worksheets("Output").Select
  Range("Duration" & m).Value = Duration(m)
  Range("Deliveries" & m).Value = Deliveries(m)
Next m

Worksheets("Input").Select

k = 1

'The For loop writes each route to a specific form included on
'the "Output" tab.
For \( j = 1 \) To nLocations
  If cycOrder(j) = routeSet(k) Then
    'Customer j should be recorded to a new route.
    m = 0
    Range("solCustomerNumberAnchor" & k).Offset(m, 0).Value = cycOrder(j)
  Else
    'Customer j is added to current route.
    m = m + 1
    Range("solCustomerNumberAnchor" & k).Offset(m, 0).Value = cycOrder(j)
  End If
  If cycOrder(j + 1) = routeSet(k + 1) Then
    'The next route will be a new route, so increment
    'the variable 'k' to correctly write to the form.
    k = k + 1
  End If
Next j

Range("numberOfRoutes").Value = nRoutes

'The post-optimization of each route is completed
Call TSP_Heuristic

Worksheets("Output").Select

'Improvements from the post-optimization are overwritten to
'the Output route forms
For m = 1 To nRoutes
  Worksheets("Output").Select
  Duration(m) = Range("Duration" & m).Value
  Deliveries(m) = Range("Deliveries" & m).Value
Next m

k = 1

'The following loop saves changes back into the array terms
For j = 1 To nLocations
  If cycOrder(j) = routeSet(k) Then
    'Customer j should be recorded to a new route.
    m = 0
    cycOrder(j) = Range("solCustomerNumberAnchor" & k).Offset(m, 0).Value
  Else
    'Customer j is added to current route.
    m = m + 1
    cycOrder(j) = Range("solCustomerNumberAnchor" & k).Offset(m, 0).Value
  End If
  If cycOrder(j + 1) = routeSet(k + 1) Then
    'The next route will be a new route, so increment
    'the variable 'k' to correctly write to the form.
    k = k + 1
  End If
Next j

'DTotal is the value of the objective function (minimize)
DTotal = Range("totalDuration").Value

'The output form is cleared for the next iteration of entries
Call clearForm

===============================================================
'==============================================================
Worksheets("Raw").Select

'Reset of variables.
DNew = 0
k = 1

'As the iterations progress, the target is to find the
'best routing scheme as to reduce travel time (program
'objective). Thus, the code requires a benchmark
'for comparison, taken following the first iteration.
If i = 2 Then
    optDTotal = DTotal
    optNRoutes = nRoutes
    For m = 1 To nRoutes
        optRouteSet(m) = routeSet(m)
    Next m
    For m = 1 To nLocations
        optCycOrder(m) = cycOrder(m)
    Next m
    For m = 1 To nRoutes
        optDuration(m) = Duration(m)
    Next m
    For m = 1 To nRoutes
        optDeliveries(m) = Deliveries(m)
    Next m
End If

'Having the benchmark values stored, the results of
'each iteration can be compared on the basis of total
'duration time of all routes. If a better solution is
'found, the optimality variables and arrays are
'overwritten with the best-found solution.
If DTotal < optDTotal Then
    'optimized total duration
    optDTotal = DTotal
    'number of routes in optimized solution (O.S.)
    optNRoutes = nRoutes
    'starting location of each route in O.S.
    For m = 1 To nRoutes
        optRouteSet(m) = routeSet(m)
    Next m
'order of locations evaluated in O.S.
For m = 1 To nLocations
    optCycOrder(m) = cycOrder(m)
Next m

'duration of each route in O.S.
For m = 1 To nLocations
    optDuration(m) = Duration(m)
Next m

'number of deliveries in each route of O.S.
For m = 1 To nRoutes
    optDeliveries(m) = Deliveries(m)
Next m
End If

Loop

'SECTION 3: Delivery personnel Travel Report

The petal algorithm is complete. Now the best-found results
'are written to a travel report form on the "Output" tab. The
'intention is that these reports can be printed and followed
'by the delivery personnel.
    Worksheets("Output").Select

'O.S. route durations and number of deliveries are logged
    For m = 1 To optNRoutes
        Worksheets("Output").Select
        Range("Duration" & m).Value = optDuration(m)
        Range("Deliveries" & m).Value = optDeliveries(m)
    Next m

    k = 1

'The For loop writes each route to a specific form included on
'the "Output" tab.
    For j = 1 To optNRoutes
        For m = 0 To optDeliveries(j) - 1
            Range("solCustomerNumberAnchor" & j).Offset(m, 0).Value = optCycOrder(k)
            k = k + 1
        Next m
    Next j

'Clears out the duplicate information.
    Worksheets("angleCache").Select
    Range("A1").Offset(nLocations, 0).Select
With Selection
    Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations, 5)).ClearContents
End With

' Record number of routes and the total duration to a summary
' area on the "Input" tab.
Worksheets("Input").Select
    Range("numberOfRoutes").Value = optNRoutes
    Range("totalDuration").Value = optDTotal

Application.DisplayAlerts = False
"cost")).Delete
Application.DisplayAlerts = True

Application.ScreenUpdating = True

elapsedTime = Timer - startTime
MsgBox "Routing complete. The program took " & elapsedTime & " seconds to run."

End Sub

(Versions 2: Petal)
Sub petalRouting()
    ' The petalRouting() subroutine evaluates the delivery locations
    ' by using the Petal Algorithm developed for Vehicle Routing
    ' Problems. Two forms of route constraints are included in the
    ' evaluation: duration and capacity. If any location requires
    ' more duration or capacity than permitted, it is given a
    ' unique route.

    Application.ScreenUpdating = False
    Application.StatusBar = True

    Dim DNew As Single, DMax As Single, DTotal As Single, DTemp As Single
    Dim QCurrrent As Integer, QNew As Integer, QMax As Integer
    Dim cycOrder(1 To 1000) As Integer, routeSet(1 To 1000) As Integer, Duration(1 To 1000) As Single, Deliveries(1 To 1000) As Integer
    Dim nRoute As Integer, nDeliveries As Single
    Dim tempDRoute As Single
    Dim optDTotal As Single, optNRoutes As Integer, optCycOrder(1 To 1000) As Integer, optRouteSet(1 To 1000) As Integer, optDuration(1 To 1000) As Single, optDeliveries(1 To 1000) As Integer

    ' SECTION 1: Inputs *******************************************************
    ' Number of Locations (Depot = 0, Customer Locations = 1, . . . n)
Worksheets("angleCache").Select
Range("A1").Select
Do Until ActiveCell = BLANK
    nLocations = nLocations + 1
    ActiveCell.Offset(1, 0).Select
Loop

'Rout Duration (D) - taken from user input
DMax = Worksheets("Input").Range("maxDuration").Value

'Rout Capacity (Q) - taken from user input
QMax = Worksheets("Input").Range("maxCapacity").Value

'Cyclic Order - taken from the radially ordered angles
Worksheets("angleCache").Select
Range("A1").Select

'SECTION 2: Body ****************************************
'Duration Management
'The following For loop takes the reference number, listed in
'Column B, for both the FROM and TO locations. Using the two
'location values, three travel information points are found.
'1. Distance from Depot to current location
'2. Distance from previous location to current location
'3. Distance from current location to Depot
'All three values are logged into the row for given location
'on the "angleCache" tab.
For j = 1 To nLocations
    'assign FROM and TO locations to variables aFrom & bTo
    If j = 1 Then
        aFrom = Range("A1").Offset(nLocations - 1, 1).Value
        bTo = Range("A1").Offset(0, 1).Value
    Else
        aFrom = Range("A1").Offset(j - 2, 1).Value
        bTo = Range("A1").Offset(j - 1, 1).Value
    End If

    'look up travel information and store to variables
    Worksheets("Raw").Select
    DDepotTOcustomer = Range("A1").Offset(0, bTo).Value
    DcustomerTOcustomer = Range("A1").Offset(aFrom, bTo).Value
    DcustomerTOdepot = Range("A1").Offset(bTo, 0).Value

    'log travel information by respective location
Worksheets("angleCache").Select
    Range("A1").Offset(j - 1, 3).Value = DdepotTOcustomer
    Range("A1").Offset(j - 1, 4).Value = DcustomerTOcustomer
    Range("A1").Offset(j - 1, 5).Value = DcustomerTOdepot
Next j

'Capacity Management
'i is the starting location for the sweep
    i = 1

'values on the "angleCache" tab are duplicated for the ease of
'progressing through iterations with different starting
'locations
    Range("A1").Select
    Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 1, 5)).Copy
    Selection.Offset(nLocations, 0).PasteSpecial xlPasteValues

'The Do loop allows for routes to formulated from each of the
'n possible starting locations (customer locations). The
'purpose is to gain a more optimal solution that using only
'one starting location.
    Do Until i = nLocations + 1
        Worksheets("angleCache").Select
        Range("A1").Select

        'the "cycOrder" array contains ordered location values
        'for the given iteration
        Deliveries(j) = 0
        Duration(j) = 0
        routeSet(j) = 0
        Next j

        'variable to count number of routes needed
        nRoutes = 1

        'variable for used capacity and duration on current
        'route
        QCurrent = 0
        DCurrent = 0
The nDeliveries variable tracks the number of deliveries within a certain route as the route is built.

nDeliveries = 0

The starting delivery location will be the first location in the first route.

routeSet(1) = Range("A1").Offset(i - 1, 1).Value

The For loop with counter 'k' incrementally assigns the next location to either the existing route or a new route.

For k = 1 To nLocations

C = i + k - 2

QNew is the would-be incoming capacity needed to add the new location to an existing route.

QNew = Range("A1").Offset(C, 2).Value

If (QNew + QCurrent) > QMax Then

'In the case of insufficient capacity:

'The following conditional statement assigns those customers exceeding the capacity constraint to a separate route.

If QCurrent = 0 Then

nRoutes = nRoutes + 1
routeSet(nRoutes) = Range("A1").Offset(C + 1, 1).Value

QCurrent = 0
DCurrent = 0

Duration(nRoutes - 1) = Range("A1").Offset(C, 3).Value + _ Range("A1").Offset(C, 5).Value
Deliveries(nRoutes - 1) = 1

nDeliveries = 0

Else

'If the Duration constraint is exceeded, the user is notified and the program terminates.
"If QCurrent = 0 Then
nRoutes = nRoutes + 1

'Record the incoming location as the first
'location for the next route.
routeSet(nRoutes) = Range("A1").Offset(C, 1).Value

'The new capacity that was needed becomes the
'current capacity of the new route.
QCurrent = QNew

'The duration and number of deliveries of the
'closed route is added to the respective
'arrays.
Duration(nRoutes - 1) = DCurrent
Deliveries(nRoutes - 1) = nDeliveries

'The duration of the new route includes the
'travel from the Depot to the new location,
'and the travel, then, back to the Depot.
DCurrent = (Range("A1").Offset(C, 3).Value) + (Range("A1").Offset(C, _
5).Value)

'The number of deliveries is reset to 1 for the
'new route.
nDeliveries = 1

End If

Else
'The current vehicle has sufficient capacity.
'Now the duration constraint is tested.

If DCurrent = 0 Then
 'No current duration of the route means
 'an incoming duration of the trip to go
 'from the Depot, to the new customer, and
 'then back to the Depot.
 DNew = (Range("A1").Offset(C, 3).Value) + (Range("A1").Offset(C, _
5).Value)

Else
 'If the current route already includes
 'customers, then the new duration value to
 'consider includes the travel time between
 'the most recently added customer and this
 'new customer, the travel time from the
'new customer back to the Depot, less the travel time from the previous customer to the Depot. 
\[ D_{\text{New}} = (\text{Range("A1")}.\text{Offset(C, 4).Value}) + (\text{Range("A1")}.\text{Offset(C, 5).Value}) - (\text{Range("A1")}.\text{Offset(C - 1, 5).Value}) \]

End If

If (DNew + DCurrent) > DMax Then

'The following conditional statement assigns those customers exceeding the duration contraint to a separate route.
If DCurrent = 0 Then

nRoutes = nRoutes + 1
routeSet(nRoutes) = Range("A1").Offset(C + 1, 1).Value

QCurrent = 0
DCurrent = 0

Duration(nRoutes - 1) = Range("A1").Offset(C, 3).Value + 
                    Range("A1").Offset(C, 5).Value
Deliveries(nRoutes - 1) = 1

nDeliveries = 0

Else

'In the case of insufficient duration:
'REPEATED FROM INSUFFICIENT CAPACITY (up)
'Add in a new route
nRoutes = nRoutes + 1

'Record the incoming location as the first location for the next route.
routeSet(nRoutes) = Range("A1").Offset(C, 1).Value

'The new capacity that was needed becomes the current capacity of the new route.
QCurrent = QNew

'The duration and number of deliveries of the closed route is added to the respective arrays.
Duration(nRoutes - 1) = DCurrent
Deliveries(nRoutes - 1) = nDeliveries

"The duration of the new route includes the
'travel from the Depot to the new location,
'and the travel, then, back to the Depot.
DCurrent = (Range("A1").Offset(C, 3).Value) + (Range("A1").Offset(C, _
5).Value)

"The number of deliveries is reset to 1 for the
'new route.
nDeliveries = 1
End If

Else
"There is sufficient capacity and duration
'to add the new location to the existing
'route.

'Increase the number of deliveries on route
nDeliveries = nDeliveries + 1

"The current capacity and duration become
'the previous values with the incoming
'values associated with adding the new
'location to the route.
QCurrent = QCurrent + QNew
DCurrent = DCurrent + DNew

End If

End If

If k = nLocations Then
"The last location for this iteration has been
'implemented and the routes should be closed
'as there are no more points to consider.
Duration(nRoutes) = DCurrent
Deliveries(nRoutes) = nDeliveries
End If

Next k

"The iteration at the current starting location has
'finished processing through the constraints.
'Increments the iteration counter.
i = i + 1

Worksheets("Raw").Select

'Reset of variables. DTotal is the total duration of
'all routes.
DTotal = 0
DNew = 0
k = 1

'DTotal is calculated by summing up the duration of
'each route (stored in Duration array).
For j = 1 To nRoutes
    DTotal = DTotal + Duration(j)
Next j

'As the iterations progress, the target is to find the
'best routing scheme as to reduce travel time (program
'objective). Thus, the code requires a benchmark
'for comparison, taken following the first iteration.
If i = 2 Then
    optDTotal = DTotal
    optNRoutes = nRoutes
    For m = 1 To nRoutes
        optRouteSet(m) = routeSet(m)
    Next m
    For m = 1 To nLocations
        optCycOrder(m) = cycOrder(m)
    Next m
    For m = 1 To nRoutes
        optDuration(m) = Duration(m)
    Next m
    For m = 1 To nRoutes
        optDeliveries(m) = Deliveries(m)
    Next m
End If

'Having the benchmark values stored, the results of
'each iteration can be compared on the basis of total
'duration time of all routes. If a better solution is
'found, the optimality variables and arrays are
'overwritten with the best-found solution.
If DTotal < optDTotal Then
  'optimized total duration
  optDTotal = DTotal

'number of routes in optimized solution (O.S.)
optNRoutes = nRoutes

'starting location of each route in O.S.
For m = 1 To nRoutes
  optRouteSet(m) = routeSet(m)
Next m

'order of locations evaluated in O.S.
For m = 1 To nLocations
  optCycOrder(m) = cycOrder(m)
Next m

'duration of each route in O.S.
For m = 1 To nLocations
  optDuration(m) = Duration(m)
Next m

'number of deliveries in each route of O.S.
For m = 1 To nRoutes
  opt Deliveries(m) = Deliveries(m)
Next m
End If

Loop

'SECTION 3: Delivery personnel Travel Report ***************************
The petal algorithm is complete. Now the best-found results
'are written to a travel report form on the "Output" tab. The
'intention is that these reports can be printed and followed
'by the delivery personnel.
  Worksheets("Output").Select

'O.S. route durations and number of deliveries are logged
For m = 1 To optNRoutes
  Worksheets("Output").Select
  Range("Duration" & m).Value = optDuration(m)
  Range("Deliveries" & m).Value = optDeliveries(m)
Next m
Worksheets("Input").Select

k = 1

' The For loop writes each route to a specific form included on
' the "Output" tab.
For j = 1 To nLocations
    If optCycOrder(j) = optRouteSet(k) Then
        ' Customer j should be recorded to a new route.
        m = 0
        Range("solCustomerNumberAnchor" & k).Offset(m, 0).Value = optCycOrder(j)
    Else
        ' Customer j is added to current route.
        m = m + 1
        Range("solCustomerNumberAnchor" & k).Offset(m, 0).Value = optCycOrder(j)
    End If
    If optCycOrder(j + 1) = optRouteSet(k + 1) Then
        ' The next route will be a new route, so increment
        ' the variable 'k' to correctly write to the form.
        k = k + 1
    End If
Next j

' Clears out the duplicate information.
Worksheets("angleCache").Select
Range("A1").Offset(nLocations, 0).Select
With Selection
    Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations, 5)).ClearContents
End With

' Record number of routes and the total duration to a summary
' area on the "Input" tab.
Worksheets("Input").Select
Range("numberOfRoutes").Value = optNRoutes
Range("totalDuration").Value = optDTotal

' Complete post-optimization on each route of the best-found
' solution.
Call TSP_Heuristic

MsgBox "Routing complete."

Application.StatusBar = False
Application.ScreenUpdating = True
Sub TSP_Heuristic()
' Once running the petal algorithm, routes are built in order
' based on the angle measurements. With radius distances
' differing between customers sharing a route, sometimes a
' reduction in travel time can be achieved by optimizing each
' route individually. Thus, the TSP Furthest Insertion Algorithm
' is employed to do post-optimization on each route.

'The program is broken into 2 steps for each considered route:
' 1. Generate a sub-matrix of travel distances
' 2. Complete TSP optimization

' generate matrix
Dim addressArray() As Variant
Dim data As Collection
Dim g As String
Dim h As String

'TSP values
Dim i As Integer, j As Integer, temp As Integer, q As Integer
Dim maxDist As Single
Dim iteration As Integer ' find the loop and repeat
Dim master() As Double ' keeps track of what nodes are in cycle
Dim dist As Variant ' distance matrix
Dim cyc() As Double ' cycle matrix
Dim cycTemp As Variant
Dim totalTravel As Single
Dim a As Single, b As Single, C As Single
Dim nTrials As Integer
Dim travelRange As Variant
Dim optArray() As String

'The recorded number of routes from the Petal Routing
' Algorithm is maintained as the number of iterations for the
' TSP heuristic.
nRoutes = Range("numberOfRoutes").Value

' optimize order of each established route
For Z = 1 To nRoutes

'number of locations to be serviced (adding the depot)
  With Worksheets("Output")
    nLocations = Range("Deliveries" & Z).Value + 1
  End With

'no optimization is necessary when only one customer exists on
'a route
  If nLocations < 3 Then
    Else

'SECTION 1: Generate the sub matrix ================

"temp" is used in the process of developing the matrix while
'the Matrix worksheet will contain the final sub-matrix
  Worksheets("temp").Cells.ClearContents
  Worksheets("Matrix").Cells.ClearContents

  Worksheets("Output").Select

'customer identification information is copied for the route
'being analyzed
  Range("solCustomerNumberAnchor" & Z).Offset(0, -1).Select
  Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 2, 1)).Copy

  Worksheets("temp").Select
  Range("A4").PasteSpecial Paste:=xlPasteValues

'a line for Depot travel information is added
  Range("A3:B3").Value = 0

'the column of FROM customer information is duplicated to
'represent the TO customer information
  With Range("A3")
    Range(Offset(0, 0), .Offset(nLocations - 1, 1)).Copy
  End With

  Range("C1").Select
  Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(1, nLocations - 1)).Select

  With Selection
  End With
'the For loops build the travel sub-matrix from the main matrix
'in the "Raw" worksheet
For i = 1 To nLocations
    For j = 1 To nLocations

        Worksheets("temp").Select

        If i = j Then
            Else

                'select the cell to write the travel information
                Range("C3").Offset(j - 1, i - 1).Select

                'g is assigned the FROM customer information
                g = Selection.Offset(0, -i).Value

                'h is assigned the TO customer information
                h = Selection.Offset(-j, 0).Value

                'copy travel information
                Worksheets("Raw").Select
                Range("A1").Offset(g, h).Copy

                'record travel information
                Worksheets("temp").Select
                Range("C3").Offset(j - 1, i - 1).PasteSpecial xIPasteValues

        End If

    Next j
Next i

'only the travel matrix, without header rows/columns, is
'transferred to the "Matrix" worksheet
Worksheets("temp").Select
Range("C3").Select
Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 1, nLocations - 1)).Copy

Worksheets("Matrix").Select
Range("A1").PasteSpecial xIPasteValues

Worksheets("Input").Select

'SECTION 2: TSP Heuristic ============================================
'worksheets for different heuristic steps are added
Worksheets("Report").Cells.ClearContents
Worksheets("C").Cells.ClearContents
Worksheets("D").Cells.ClearContents
Worksheets("cost").Cells.ClearContents

'the number of locations is the number of attempts made to
'find optimality
For runCount = 1 To nLocations

'starting node for given iteration
i = runCount

dimension arrays for Furthest Insertion Algorithm
ReDim dist(1 To nLocations), cyc(1 To nLocations + 1), _
master(1 To nLocations + 1)

"Matrix" contains the sub-array from "Raw"
Worksheets("Matrix").Select
Range("A1").Select
dist = Worksheets("Matrix").Range(ActiveCell.Offset(i - 1, 0), _
    ActiveCell.Offset(i - 1, nLocations - 1))

'The 'master' array tracks which locations have been
'added to the cycle thus far. To start, only the
'starting location is included in the array.
master(1) = i

"D" contains the 'dist' array information
Worksheets("D").Select
Range("A1").Select
Range(Selection.Offset(0, 0), Selection.Offset(0, nLocations - 1)) = dist

'the 'iteration' For loop is the TSP process given the specific
'starting node
For iteration = 1 To nLocations - 1

'selection criteria: node furthest away
maxDist = WorksheetFunction.Max(dist)

'identify the node furthest away (incoming node)
For loopCount = 1 To nLocations
    If loopCount = i Then
    Else
        Worksheets("D").Select
        Range("A1").Offset(iteration - 1, loopCount - 1).Select
        If Selection = maxDist Then
q = loopCount
loopCount = nLocations
End If
End If
Next loopCount

'incoming node is stored to 'master' array
master(iteration + 1) = q

'If it is the first location to be added, then the
'travel distance will be from the starting node to the
'newly added node. There are no insertion location
'considerations to be made.
If iteration = 1 Then
Worksheets("Matrix").Select

"a' is the travel cost from the starting
'location to the first location entering the
'cycle
a = Range("A1").Offset(master(1) - 1, master(2) - 1)

"b' is the travel cost of returning from the
'entering location back to the starting
'location
b = Range("A1").Offset(master(2) - 1, master(1) - 1)

'no distance is saved by creating this first
'cycle
C = 0

"x' is the insertion cost
x = a + b - C

'record the insertion cost for tracking
'purposes
Worksheets("cost").Select
Range("A1").Offset(iteration - 1, 0) = x

'the current cycle, controlled by the 'cyc'
'array, is stored
With Worksheets("C").Select
    Range("A1").Value = master(1)
    cyc(1) = master(1)
    Range("B1").Value = master(2)
    cyc(2) = master(2)
    Range("C1").Value = master(1)
cyc(3) = master(1)
End With

'the 'totalTravel' variable compiles the length
'of the final route between all locations
totalTravel = x
End If

'if a cycle exists between the starting location and
'the first added location, then it is important to
determine where to place the incoming location
If iteration >= 2 Then

'In order to evaluate possible insertion locations,
'the following 'Count' loop finds and records the
different insertion costs
For Count = 1 To iteration

Worksheets("Matrix").Select

"a" is the cost FROM the first location
to the new location
a = Range("A1").Offset(cyc(Count) - 1, master(iteration + 1) - 1)

"b" is the cost TO the second location
from the new location
b = Range("A1").Offset(master(iteration + 1) - 1, cyc(Count + 1) - 1)

"C" is the cost saved, defined as the
cost from the first location to the
second location
C = Range("A1").Offset(cyc(Count) - 1, cyc(Count + 1) - 1)

'insertion cost calculation
x = a + b - C

'insertion cost is recorded
Worksheets("cost").Select
Range("A1").Offset(iteration - 1, Count - 1) = x

Next Count

given the different insertion costs, select the
minimum
Worksheets("cost").Select
Range("A1").Select
cycTemp = Worksheets("Cost").Range(ActiveCell.Offset(iteration - 1, 0), _ 
          ActiveCell.Offset(iteration - 1, nLocations - 1))
minCost = WorksheetFunction.Min(cycTemp)

' the column of the minimum insertion cost
'determines the two locations to insert the
' incoming location between
For loopCount = 1 To iteration
    If loopCount = temp Then
    Else
        Worksheets("cost").Select
        Range("A1").Offset(iteration - 1, loopCount - 1).Select
        If Selection = minCost Then
            iAfter = loopCount
            temp = loopCount
            loopCount = iteration 'exit loop sooner
        End If
    End If
Next loopCount

'The 'cyc' array is overwritten throughout each
' iteration, allowing recall of the cycle order for
' future cost calculations. The following code
' changes the array to reflect the insertion of the
' new location.
cycNew = cyc
    cycNew(iAfter + 1) = master(iteration + 1)
For loopCount = iAfter + 2 To nLocations + 1
    cycNew(loopCount) = cyc(loopCount - 1)
Next loopCount
    cyc = cycNew

' Adding for visual aid, the 'cyc' array is printed
' to the "C" tab following each insertion.
Worksheets("C").Select
    Range("A1").Select
    Range(ActiveCell.Offset(iteration - 1, 0), ActiveCell.Offset(iteration - 1, _ 
          nLocations)).Value = cyc

'Having added in a new location, the total route
'travel time is adjusted for the new cycle.
    totalTravel = totalTravel + minCost

End If
The travel information for the new location is copied and pasted to the "D" worksheet for the next iteration of determining the next insertion location.

Worksheets("Matrix").Select
Range("A1").Select
dist = Worksheets("Matrix").Range(ActiveCell.Offset(master(iteration + 1) - 1, _ 0), ActiveCell.Offset(master(iteration + 1) - 1, nLocations - 1))
Worksheets("D").Select
Range("A1").Select
Range(Selection.Offset(iteration, 0), Selection.Offset(iteration, nLocations - 1)) _
   = dist

Of the remaining locations to enter the cycle, the furthest insertion distance is taken of the minimum values for each location.

Range("A1").Select
Selection.Offset(iteration, 0).Select

minimum of the column is kept to represent the smallest travel value for that specific location
For j = 1 To nLocations
   x = WorksheetFunction.Min(Range(Selection.Offset(0, j - 1), Selection._
      Offset(-1, j - 1)))
   Selection.Offset(0, j - 1) = x
Next j

currently included locations are removed from available selection
For j = 1 To iteration + 1
   x = master(j)
   Selection.Offset(0, x - 1).Value = ""
Next j

the remaining row of inseriton values is saved to the 'dist' array
Range("A1").Select
dist = Range(ActiveCell.Offset(iteration, 0), ActiveCell.Offset(iteration, _
   nLocations - 1))

Next iteration

The solved cycle, along with the total travel time, are recorded into the "Report" worksheet before starting a new iteration.
Worksheets("Report").Select
Range("A1").Select
Range(ActiveCell.Offset(runCount, 0), ActiveCell.Offset(runCount, nLocations)).Value = cyc
Range("A1").Offset(runCount, nLocations + 1).Value = totalTravel

Next runCount

'Organize report
Worksheets("Report").Select
Range("A1").Name = "reportAnchor"

Range("reportAnchor").Select

'The best-found solution with the TSP heuristic is sorted
to the top to allow for the remaining solutions to be
deleted.
  Key:=Range(ActiveCell.Offset(0, nLocations + 1), _
  ActiveCell.Offset(nTrials, nLocations + 1)), _
  SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
With ActiveWorkbook.Worksheets("Report").Sort
  .SetRange Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nTrials, nLocations + 1))
  .Header = xlGuess
  .MatchCase = False
  .Orientation = xlTopToBottom
  .SortMethod = xlPinYin
  .Apply
End With

Rows("1").Delete
Rows("2:10000").Delete

'Since an assumption of the assigned routes will be that they
all start and end at the Depot, it is important to have the
first (1) location lead the recording of the following
delivery locations.
With Range("A1")
  temp = .Offset(0, nLocations + 1).Value
  Range(.Offset(0, 0), .Offset(0, nLocations - 1)).Copy
  .Offset(0, nLocations).PasteSpecial xlPasteValues
End With

'The variable 'x' is a simple measure to determine the offset
needed to reach the first customer delivery (not = 1)
x = 0
Do Until i = 1
    Range("A1").Offset(0, x).Select
    i = Selection.Value
    x = x + 1
Loop

'The customers corresponding to the initial order are copied 'into the "Report" worksheet for organization of the order 'deliveries are made.
Worksheets("Output").Select
With Range("solCustomerNumberAnchor" & Z)
    Range(.Offset(0, 0), .Offset(nLocations - 2, 0)).Copy
End With

Worksheets("Report").Select
Range("A1").Offset(1, x).Select
With Selection
        False, Transpose:=True
End With

'The customer reference numbers are sorted according to TSP 'output.
With Range("A1")
    Range(.Offset(0, x), .Offset(1, x + nLocations - 2)).Copy
    .Offset(2, 0).Select
With Selection
        False, Transpose:=True
End With

    Key:=Range(.Offset(2, 0), .Offset(nLocations, 0)), _
    SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
With ActiveWorkbook.Worksheets("Report").Sort
    .SetRange Range(ActiveCell.Offset(0, 0), ActiveCell.Offset(nLocations - 2, 1))
    .Header = xlGuess
    .MatchCase = False
    .Orientation = xlTopToBottom
    .SortMethod = xlPinYin
    .Apply
End With
Range(.Offset(2, 1), .Offset(nLocations, 1)).Copy
End With

The ordered customer deliveries are recorded over the Petal
Routing output in the "Output" worksheet.
Worksheets("Output").Select
Range("solCustomerNumberAnchor" & Z).Select
Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
:=False, Transpose:=False
Application.CutCopyMode = False

The new total route duration is written to the output.
Range("Duration" & Z).Value = temp
End If

Continue through each route for TSP optimization.
Next Z

Sum durations.
x = 0
For i = 1 To nRoutes
  x = x + Range("Duration" & i)
Next i

Calculate time savings through post-optimization.
i = Range("totalDuration").Value - x

Print new duration.
Range("totalDuration").Value = x

Worksheets("Input").Select

End Sub

The sortDegrees() subroutine orders the locations on the
"angleCache" tab based on the degree values pulled from the
database file.
Sub sortDegrees()
  Columns("A:B").Select
  ActiveWorkbook.Worksheets("angleCache").Sort.SortFields.Add
Key:=Range("A1:A300"), _    
SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal  
With ActiveWorkbook.Worksheets("angleCache").Sort  
.SetRange Range("A1:C300")  
.Header = xlGuess  
.MatchCase = False  
.Orientation = xlTopToBottom  
.SortMethod = xlPinYin  
.Apply  
End With  
End Sub  

'To clear clutter and possible overlapped information, the user  
'starts by clearing out previous routing information.  
Sub clearForm()  
  
  Worksheets("Output").Select  
  For i = 1 To 25  
   With Range("solCustomerNumberAnchor" & i)  
    Range(.Offset(0, 0), .Offset(5000, 0)).ClearContents  
   End With  
   Range("Duration" & i).ClearContents  
   Range("Deliveries" & i).ClearContents  
  Next i  
  
  Worksheets("Input").Select  
  Range("numberOfRoutes").ClearContents  
  Range("totalDuration").ClearContents  
  
End Sub  

'GetValue is a function used in the matrix generation and angle  
'capture from the database file.  
Public Function GetValue(path, file, sheet, ref)  
  path = "C:\Users\Starr\Desktop"  
  file = "Thesis_VRP_CustomerResource.xlsm"  
  sheet = "Travel Matrix Database"  
  ref = "F2:XFD1048576" 'F2 is the Range("matrixAnchor") cell  
  
    
  'Retrieves a value from a closed workbook
Dim arg As String

' Make sure the file exists
If Right(path, 1) <> "\" Then path = path & "\"
If Dir(path & file) = "" Then
    GetValue = "File Not Found"
    Exit Function
End If

' Create the argument
arg = "" & path & "[" & file & "]" & sheet & "!" & _
Range(ref).Range("D2").Address(, , xlR1C1)

' Execute an XLM macro
GetValue = ExecuteExcel4Macro(arg)
End Function
Appendix D

Delivery Personnel Standard Operating Procedure
The following process map is the proposed operating procedure for delivery personnel.
Appendix E

Sample Address Data
The following data was used for program analysis. The data is borrowed from a posted sample data set at [www.briandunning.com/sample-data](http://www.briandunning.com/sample-data).

<table>
<thead>
<tr>
<th>Sample Addresses</th>
<th>Location</th>
<th>Maryland</th>
<th>Michigan</th>
<th>New York</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depot</td>
<td>Maryland</td>
<td>Michigan</td>
<td>New York</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>115 Clemente St, Holyoke, MA, 1040</td>
<td>14842 Beech Daly Rd, Taylor, MI, 48180</td>
<td>548 W 28th St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>10 Kelley St, Palmer, MA, 1069</td>
<td>8575 Ronda Dr, Canton, MI, 48187</td>
<td>320 5th Ave, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1350 Main St, Springfield, MA, 1103</td>
<td>5333 Mcauley Dr, Ypsilanti, MI, 48197</td>
<td>460 W 34th St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>83 State St, Springfield, MA, 1103</td>
<td>345 S Prospect St, Ypsilanti, MI, 48198</td>
<td>1170 Broadway #503, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>33 State St, Springfield, MA, 1103</td>
<td>2955 Bellevue St, Detroit, MI, 48207</td>
<td>230 5th Ave, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1695 Main St, Springfield, MA, 1103</td>
<td>7601 Central St, Detroit, MI, 48210</td>
<td>243 W 30th St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>62 Avocado St, Springfield, MA, 1104</td>
<td>1901 Marston St, Detroit, MI, 48211</td>
<td>151 W 26th St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>399 Liberty St, Springfield, MA, 1104</td>
<td>899 Chalmers St, Detroit, MI, 48215</td>
<td>229 W 28th St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>179 Page Blvd, Springfield, MA, 1104</td>
<td>1701 W Lafayette Blvd, Detroit, MI, 48216</td>
<td>601 W 26th St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>53 Batavia St, Springfield, MA, 1109</td>
<td>2436 Bagley St, Detroit, MI, 48216</td>
<td>116 W 32nd St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1 Monarch Pl, Springfield, MA, 1144</td>
<td>17321 Telegraph Rd, Detroit, MI, 48219</td>
<td>330 7th Ave, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>476 Oak St, Indian Orchard, MA, 1151</td>
<td>1705 E 9mile Rd, Ferndale, MI, 48220</td>
<td>330 5th Ave, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>55 Spring St, Pittsfield, MA, 1201</td>
<td>350 Fair St, Ferndale, MI, 48220</td>
<td>10 W 33rd St, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>41 Park St, Adams, MA, 1220</td>
<td>553 E Jefferson Ave, Detroit, MI, 48226</td>
<td>320 5th Ave #905, New York, NY, 10001</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>180 Pond St, Leominster, MA, 1453</td>
<td>1 Woodward Ave, Detroit, MI, 48226</td>
<td>11 Rivington, New York, NY, 10002</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>20 Mohawk Dr, Leominster, MA, 1453</td>
<td>8225 Lyndon St, Detroit, MI, 48238</td>
<td>466 Grand St, New York, NY, 10002</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>8 S Main St, Millbury, MA, 1527</td>
<td>500 Renaissance Ctr, Detroit, MI, 48243</td>
<td>55 1st Ave, New York, NY, 10003</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>31 River St, Millbury, MA, 1527</td>
<td>1393 S Woodward Ave, Bloomfield Hills, MI, 48302</td>
<td>41 Union Sq W, New York, NY, 10003</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Address Details</td>
<td>Address Details</td>
<td>Address Details</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>----------------</td>
<td>-----------------</td>
<td>-----------------</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>172 Otis St, Northborough, MA, 1532</td>
<td>200 E Long Lake Rd #-165, Bloomfield Hills, MI, 48304</td>
<td>71 5th Ave, New York, NY, 10003</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Cudworth Rd, Oxford, MA, 1540</td>
<td>411 S Main St, Rochester, MI, 48307</td>
<td>66 E 1st St, New York, NY, 10003</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>779 Hartford Tpke, Shrewsbury, MA, 1545</td>
<td>1900 Northfield Dr, Rochester, MI, 48309</td>
<td>56 E 13th St, New York, NY, 10003</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>64 Mill St, Southbridge, MA, 1550</td>
<td>35454 Mound Rd, Sterling Heights, MI, 48310</td>
<td>30 Irving Pl, New York, NY, 10003</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>529 Ashland Ave, Southbridge, MA, 1550</td>
<td>14445 Mile Rd, Sterling Heights, MI, 48312</td>
<td>17 Battery Pl, New York, NY, 10004</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>109 Worcester Rd, Webster, MA, 1570</td>
<td>43805 Van Dyke Ave, Sterling Heights, MI, 48314</td>
<td>5 Hanover Sq, New York, NY, 10004</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>21 E Main St, Westborough, MA, 1581</td>
<td>13961 Hall Rd, Utica, MI, 48315</td>
<td>120 Wall St, New York, NY, 10005</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>25 Bridle Ln, Westborough, MA, 1581</td>
<td>46178 Winston Dr, Utica, MI, 48315</td>
<td>225 Broadway, New York, NY, 10007</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>1 Main St, Whitinsville, MA, 1588</td>
<td>45700 Mound Rd, Utica, MI, 48317</td>
<td>305 Broadway, New York, NY, 10007</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>47 Brownell St, Worcester, MA, 1602</td>
<td>6346 Orchard Lake Rd, West Bloomfield, MI, 48322</td>
<td>15 A Aven, New York, NY, 10009</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>107 Richmond Ave, Worcester, MA, 1602</td>
<td>5830 Andersonville Rd, Waterford, MI, 48329</td>
<td>307 3rd Ave, New York, NY, 10010</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>41 Sutton Ln, Worcester, MA, 1603</td>
<td>30790 W 8mile Rd, Farmington, MI, 48336</td>
<td>54 W 21st St, New York, NY, 10010</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>100 Grove St, Worcester, MA, 1605</td>
<td>3001 E Highland Rd, Highland, MI, 48356</td>
<td>824 Americas Aven, New York, NY, 1001</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>340 Main St #-910, Worcester, MA, 1608</td>
<td>3910 Center Rd, Highland, MI, 48357</td>
<td>104 E 25th St, New York, NY, 10010</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>28 Mechanic St, Worcester, MA, 1608</td>
<td>22313 Worcester Dr, Novi, MI, 48374</td>
<td>225 E 26th St, New York, NY, 10010</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>7 Dewey St, Worcester, MA, 1609</td>
<td>8415 Davison Rd #-a, Davison, MI, 48423</td>
<td>175 5th Ave, New York, NY, 10010</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>875 Main St, Worcester, MA, 1610</td>
<td>1358 Ortonville Rd, Ortonville, MI, 48462</td>
<td>20 W 22nd St, New York, NY, 10010</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>2 Vernon St, Framingham, MA, 1701</td>
<td>2284 S Ballenger Hwy, Flint, MI, 48503</td>
<td>17 W 17th Ave, New York, NY, 10010</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>8 Craig Rd, Acton, MA, 1720</td>
<td>1622 Berrywood Ln, Flint, MI, 48507</td>
<td>17 W 18th St, New York, NY, 10011</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>40 Main St, Ashland, MA, 1721</td>
<td>G3437 Miller Rd, Flint, MI, 48507</td>
<td>44 W 18th St, New York, NY, 10011</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>29 Domino Dr, Concord,</td>
<td>3623 Blackington Ave,</td>
<td>44 W 18th St, New York, NY, 10011</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3623 Blackington Ave,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

126
<table>
<thead>
<tr>
<th></th>
<th>MA, 1742</th>
<th>Flint, MI, 48532</th>
<th>York, NY, 10011</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>159 Ash St, Hopkinton, MA, 1748</td>
<td>1708 N Michigan Ave, Saginaw, MI, 48602</td>
<td>459 W 15th St, New York, NY, 10011</td>
</tr>
<tr>
<td>41</td>
<td>195 Merrimac St, Woburn, MA, 1801</td>
<td>1900 N Michigan Ave, Saginaw, MI, 48602</td>
<td>453 W 17th St, New York, NY, 10011</td>
</tr>
<tr>
<td>42</td>
<td>87 Olympia Ave, Woburn, MA, 1801</td>
<td>106 Hoyt Ave, Saginaw, MI, 48607</td>
<td>161 Hudson St, New York, NY, 10013</td>
</tr>
<tr>
<td>43</td>
<td>265 Winn St, Burlington, MA, 1803</td>
<td>320 W Cedar Ave, Gladwin, MI, 48624</td>
<td>17 Baxter St, New York, NY, 10013</td>
</tr>
<tr>
<td>44</td>
<td>359 Main St, Haverhill, MA, 1830</td>
<td>429 N State St, Caro, MI, 48723</td>
<td>304 Hudson St, New York, NY, 10013</td>
</tr>
<tr>
<td>45</td>
<td>60 Bailey Blvd, Haverhill, MA, 1830</td>
<td>1407 S Harrison Rd, East Lansing, MI, 48823</td>
<td>150 Varick St, New York, NY, 10013</td>
</tr>
<tr>
<td>46</td>
<td>707 Turnpike St, North Andover, MA, 1845</td>
<td>2205 Abbott Rd, East Lansing, MI, 48823</td>
<td>339 Greenwich St, New York, NY, 10013</td>
</tr>
<tr>
<td>47</td>
<td>30 Massachusetts Ave, North Andover, MA, 1845</td>
<td>1133 May St, Lansing, MI, 48906</td>
<td>476 Broadway, New York, NY, 10013</td>
</tr>
<tr>
<td>48</td>
<td>1271 Main St, Tewksbury, MA, 1876</td>
<td>1121 N Creyts Rd, East Lansing, MI, 48917</td>
<td>458 Greenwich St, New York, NY, 10013</td>
</tr>
<tr>
<td>49</td>
<td>301 Littleton Rd, Westford, MA, 1886</td>
<td>435 Washington Sq S, Lansing, MI, 48933</td>
<td>433 Canal St, New York, NY, 10013</td>
</tr>
<tr>
<td>50</td>
<td>210 Andover St #-21, Wilmington, MA, 1887</td>
<td>603 S Washington Ave, Lansing, MI, 48933</td>
<td>183 Canal St, New York, NY, 10013</td>
</tr>
<tr>
<td>51</td>
<td>17 Joyce St, Lynn, MA, 1902</td>
<td>412 N Walnut St, Lansing, MI, 48933</td>
<td>424 West St, New York, NY, 10014</td>
</tr>
<tr>
<td>52</td>
<td>12 Cleveland Pl, Gloucester, MA, 1930</td>
<td>2129 Portage St, Kalamazoo, MI, 49001</td>
<td>11 E 36th St, New York, NY, 10016</td>
</tr>
<tr>
<td>53</td>
<td>40 Beach St #-105, Manchester, MA, 1944</td>
<td>3715 W Main St, Kalamazoo, MI, 49006</td>
<td>245 5th Ave, New York, NY, 10016</td>
</tr>
<tr>
<td>54</td>
<td>905 Turnpike, Canton, MA, 2021</td>
<td>1717 Douglas Ave, Kalamazoo, MI, 49007</td>
<td>470 Park Ave S, New York, NY, 10016</td>
</tr>
<tr>
<td>55</td>
<td>1017 Turnpike St, Canton, MA, 2021</td>
<td>619 Cliff St, Battle Creek, MI, 49017</td>
<td>117 E 29th St, New York, NY, 10016</td>
</tr>
<tr>
<td>56</td>
<td>900 Providence Hwy, Dedham, MA, 2026</td>
<td>95 W Main St, Benton Harbor, MI, 49022</td>
<td>10 E 33rd St, New York, NY, 10016</td>
</tr>
<tr>
<td>57</td>
<td>275 Lenox St, Norwood, MA, 2062</td>
<td>124 S Saint Joseph St, Colon, MI, 49040</td>
<td>244 Madison Ave, New York, NY, 10016</td>
</tr>
<tr>
<td>58</td>
<td>378 Page St, Stoughton, MA, 2072</td>
<td>4777 Beechnut Dr, Saint Joseph, MI, 49085</td>
<td>140 E 38th St, New York, NY, 10016</td>
</tr>
<tr>
<td>59</td>
<td>405 West St, Walpole, MA, 2081</td>
<td>106 W Prutzman St, Three Rivers, MI, 49093</td>
<td>455 1st Ave, New York, NY, 10016</td>
</tr>
<tr>
<td></td>
<td>Address 1</td>
<td>Address 2</td>
<td>Address 3</td>
</tr>
<tr>
<td>---</td>
<td>------------------------------------</td>
<td>---------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>60</td>
<td>50 Congress St, Boston, MA, 2109</td>
<td>3030 S 11th St, Niles, MI, 49120</td>
<td>44 E 29th St, New York, NY, 10016</td>
</tr>
<tr>
<td>61</td>
<td>20 Winthrop Sq, Boston, MA, 2110</td>
<td>2732 S 3rd St, Niles, MI, 49120</td>
<td>10 E 40th St, New York, NY, 10016</td>
</tr>
<tr>
<td>62</td>
<td>77 N Washington St #-9, Boston, MA, 2114</td>
<td>1701 Terminal Rd, Niles, MI, 49120</td>
<td>271 Madison Ave, New York, NY, 10016</td>
</tr>
<tr>
<td>63</td>
<td>419 Boylston St, Boston, MA, 2116</td>
<td>145 S Jackson St, Jackson, MI, 49201</td>
<td>274 Madison Ave #-1501, New York, NY, 10016</td>
</tr>
<tr>
<td>64</td>
<td>128 Newbury St, Boston, MA, 2116</td>
<td>140 Price St, Jackson, MI, 49202</td>
<td>411 5th Ave, New York, NY, 10016</td>
</tr>
<tr>
<td>65</td>
<td>932 Dorchester Ave, Boston, MA, 2125</td>
<td>2545 Spring Arbor Rd, Jackson, MI, 49203</td>
<td>222 E 49th St, New York, NY, 10017</td>
</tr>
<tr>
<td>66</td>
<td>156 Lincoln St, Brighton, MA, 2135</td>
<td>155 N Winter St, Adrian, MI, 49221</td>
<td>41 E 42nd St, New York, NY, 10017</td>
</tr>
<tr>
<td>67</td>
<td>1505 Commonwealth Ave, Brighton, MA, 2135</td>
<td>28290m 60 E, Homer, MI, 49245</td>
<td>305 E 46th St, New York, NY, 10017</td>
</tr>
<tr>
<td>68</td>
<td>1160 Massachusetts Ave, Cambridge, MA, 2138</td>
<td>96 84th St Sw, Byron Center, MI, 49315</td>
<td>511 Lexington Ave, New York, NY, 10017</td>
</tr>
<tr>
<td>69</td>
<td>186 Alewife Brook Pky, Cambridge, MA, 2138</td>
<td>202 N Monroe St, Rockford, MI, 49341</td>
<td>820 2nd Ave, New York, NY, 10017</td>
</tr>
<tr>
<td>70</td>
<td>55 Wheeler St, Cambridge, MA, 2138</td>
<td>4634 12th St, Wayland, MI, 49348</td>
<td>630 3rd Ave, New York, NY, 10017</td>
</tr>
<tr>
<td>71</td>
<td>2267 Massachusetts Ave, Cambridge, MA, 2140</td>
<td>4395 S Parsons Ave, Fremont, MI, 49412</td>
<td>335 Madison Ave, New York, NY, 10017</td>
</tr>
<tr>
<td>72</td>
<td>36 Charles St, Cambridge, MA, 2141</td>
<td>128 Columbus St, Grand Haven, MI, 49417</td>
<td>501 5th Ave, New York, NY, 10017</td>
</tr>
<tr>
<td>73</td>
<td>145 Cedar St, Somerville, MA, 2144</td>
<td>12952 152nd Ave, Grand Haven, MI, 49417</td>
<td>535 5th Ave, New York, NY, 10017</td>
</tr>
<tr>
<td>74</td>
<td>339 Pleasant St, Malden, MA, 2148</td>
<td>321 Settlers Rd, Holland, MI, 49423</td>
<td>565 5th Ave, New York, NY, 10017</td>
</tr>
<tr>
<td>75</td>
<td>650 Broadway, Malden, MA, 2148</td>
<td>225 N 168th Ave, Holland, MI, 49424</td>
<td>1015 Americas Aven, New York, NY, 10018</td>
</tr>
<tr>
<td>76</td>
<td>104 Tremont St, Everett, MA, 2149</td>
<td>701 W Clay Ave, Muskegon, MI, 49440</td>
<td>231 W 39th St, New York, NY, 10018</td>
</tr>
<tr>
<td>77</td>
<td>210 Beacham St, Everett, MA, 2149</td>
<td>249 Irwin Ave, Muskegon, MI, 49442</td>
<td>323 W 39th St, New York, NY, 10018</td>
</tr>
<tr>
<td>78</td>
<td>1783 Revere Beach Pky, Everett, MA, 2149</td>
<td>614 174th Ave, Spring Lake, MI, 49456</td>
<td>315 W 36th St, New York, NY, 10018</td>
</tr>
<tr>
<td>79</td>
<td>777 N Shore Rd, Revere, MA, 2149</td>
<td>612 W Savidge St,</td>
<td>545 8th Ave, New York, NY, 10017</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>MA, 2151</td>
<td>Spring Lake, MI, 49456</td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>427 Riverside Ave, Medford, MA, 2155</td>
<td>60 Monroe Center St Nw, Grand Rapids, MI, 49503</td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>209 Middlesex Ave, Medford, MA, 2155</td>
<td>510 Grand Plaza Plac, Grand Rapids, MI, 49503</td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>430 Hancock St, Quincy, MA, 2171</td>
<td>904 Broadway Ave Nw, Grand Rapids, MI, 49504</td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>225 Fallon Rd, Stoneham, MA, 2180</td>
<td>2610 Glencairin Dr Nw, Grand Rapids, MI, 49504</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>290 Northern Ave, Boston, MA, 2210</td>
<td>2152 Plainfield Ave Ne, Grand Rapids, MI, 49505</td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>145 Northern Ave, Boston, MA, 2210</td>
<td>2660 S Division Ave, Grand Rapids, MI, 49507</td>
<td></td>
</tr>
<tr>
<td>87</td>
<td>56 Main, Easton, MA, 2334</td>
<td>5370 52nd St Se, Grand Rapids, MI, 49512</td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>133 Centre, Middleboro, MA, 2346</td>
<td>1001 Americas Ave #700, New York, NY, 10018</td>
<td></td>
</tr>
<tr>
<td>89</td>
<td>200 Wales St, Abington, MA, 2351</td>
<td>2455 29th St Se, Grand Rapids, MI, 49512</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>533 Washington St, Abington, MA, 2351</td>
<td>4900 Cascade Rd Se, Grand Rapids, MI, 49546</td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>627 Hancock St, Abington, MA, 2351</td>
<td>424 44th St Se, Grand Rapids, MI, 49548</td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>51 School St, Pembroke, MA, 2359</td>
<td>5312 S Division Ave, Grand Rapids, MI, 49548</td>
<td></td>
</tr>
<tr>
<td>93</td>
<td>278 Court St, Plymouth, MA, 2360</td>
<td>103 E Pine St, Cadillac, MI, 49601</td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>58 Teed Dr, Randolph, MA, 2368</td>
<td>201 Haynes St, Cadillac, MI, 49601</td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>66 Pond St, Whitman, MA, 2382</td>
<td>837 N Mitchell St, Cadillac, MI, 49601</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>59 Shore St, Falmouth, MA, 2540</td>
<td>123 Arthur St, Manistee, MI, 49660</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 N Water St, Nantucket, MA, 2554</td>
<td>757 E Silver Lake Rd S, Traverse City, MI, 49684</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>25 Sutton Pl S, New York, NY, 10022</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Address 1</td>
<td>Address 2</td>
<td>Address 3</td>
</tr>
<tr>
<td>---</td>
<td>-----------------------------------</td>
<td>-----------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>97</td>
<td>329 W Main St, Hyannis, MA, 2601</td>
<td>2550 Us Highway 23s S, Alpena, MI, 49707</td>
<td>153 E 53rd St, New York, NY, 10022</td>
</tr>
<tr>
<td>98</td>
<td>11 Thornton Dr, Hyannis, MA, 2601</td>
<td>494 S Ripley Blvd, Alpena, MI, 49707</td>
<td>488 Madison Ave, New York, NY, 10022</td>
</tr>
<tr>
<td>99</td>
<td>182 Lund Farm Way, Brewster, MA, 2631</td>
<td>5400 Tall Timber Trl, Gaylord, MI, 49735</td>
<td>500 Park Ave, New York, NY, 10022</td>
</tr>
<tr>
<td>100</td>
<td>67 Mechanic St, Attleboro, MA, 2703</td>
<td>327 S Sophie St, Bessemer, MI, 49911</td>
<td>575 Lexington Ave, New York, NY, 10022</td>
</tr>
</tbody>
</table>
Appendix F

Delivery Picking System
The three pictures below demonstrate the proposed order picking system.

System Application in Local Health Center

HangUP® Bag*

Bag Caddy*

*Image displayed on company website (www.hangupbags.com)


