

Capacitated Vehicle Routing Problem with Time Windows:  
A Case Study on Pickup of Dietary Products in Nonprofit Organization

by

Xiaoyan Li

A Thesis Presented in Partial Fulfillment  
of the Requirements for the Degree  
Master of Science

Approved July 2015 by the  
Graduate Supervisory Committee:

Ronald Askin, Chair  
Teresa Wu  
Rong Pan

ARIZONA STATE UNIVERSITY

December 2015

## ABSTRACT

This thesis presents a successful application of operations research techniques in nonprofit distribution system to improve the distribution efficiency and increase customer service quality. It focuses on truck routing problems faced by St. Mary's Food Bank Distribution Center. This problem is modeled as a capacitated vehicle routing problem to improve the distribution efficiency and is extended to capacitated vehicle routing problem with time windows to increase customer service quality. Several heuristics are applied to solve these vehicle routing problems and tested in well-known benchmark problems. Algorithms are tested by comparing the results with the plan currently used by St. Mary's Food Bank Distribution Center. The results suggest heuristics are quite complete: average 17% less trucks and 28.52% less travel time are used in heuristics' solution.

## ACKNOWLEDGMENTS

This thesis would not have been possible without the guidance and help of several individuals who contributed and extended their valuable assistance in the preparation and completion of this thesis.

Firstly, I would like to give my deepest appreciation to my advisor, Dr. Ronald Askin, for his continuous encouragement and guidance throughout my studies. He helped me grow from nothing to mature in my research; he appeared at any time when I encounter difficulties; he reminded me to pay attention to every detail and be rigorous. My thesis could not be completed without his fruitful discussions and inspiring suggestions. Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Teresa Wu and Prof. Rong Pan, for their patience, time, insightful comments and encouragement.

I am also grateful to my husband, Ruijin Cang. I feel very lucky to receive his unswerving support at any time for the past seven years. Most of all, I would like to thank my family and my friends for their love and ASU for every wonderful moment and experience I enjoyed.

# TABLE OF CONTENTS

|  | Page |
|--|------|
| LIST OF TABLES .....   | vi   |
| LIST OF FIGURES .....  | vii  |
| CHAPTER  |      |
| 1 INTRODUCTION.....  | 1    |
| 1.1 Background .....   | 1    |
| 1.2 Problem Overview .....   | 2    |
| 1.3 Problem Definition .....   | 3    |
| 1.4 Resource.....  | 4    |
| 1.5 Thesis Structure .....   | 4    |
| 2 LITUREATURE REVIEW .....   | 6    |
| 2.1 Introduction .....   | 6    |
| 2.2 Capacitated Vehicle Routing Problem .....                                | 9    |
| 2.3 Capacitated Vehicle Routing Problem with Time Windows (CVRPTW).....      | 9    |
| 2.4 Algorithms .....   | 10   |
| 2.4.1 Exact Algorithms.....  | 10   |
| 2.4.2. Classical Heuristic:.....   | 11   |
| 2.4.3 Metaheuristics .....   | 12   |
| 2.5 Researches on Distribution and Collection of Nonprofit Organization..... | 14   |
| 3 CAPACITATED VEHICLE ROUTING PROBLEM .....                                  | 15   |
| 3.1 Problem Description and Characteristics .....                            | 16   |
| 3.1.1 Transportation Requests .....  | 16   |
| 3.1.2 Objective Functions.....   | 16   |

| CHAPTER  | Page |
|--|------|
| 3.2 Formulation.....   | 18   |
| 3.3.1 Clarke and Wright Algorithm (Saving Algorithm):.....   | 21   |
| 3.3.2 Local Search .....   | 26   |
| 3.3.3 Record-to-Record Algorithm .....   | 28   |
| 3.4 Numerical Result .....   | 32   |
| 3.5 Case Study - Problem of St. Mary's Food Bank's Distribution Center (Without Time Windows)..... | 35   |
| 3.5.1 Input Data .....   | 35   |
| 3.5.2 Results .....  | 36   |
| 4 CAPACITATED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS (CVRPTW).....                              | 39   |
| 4.1. Scenario.....   | 39   |
| 4.2. Time Window Constraints.....  | 39   |
| 4.3 Problem Formulation .....  | 40   |
| 4.4 Multiple Ant Colony System (MACS).....   | 44   |
| 4.4.1 MACS for Capacitated Vehicle Routing Problem with Time Windows .....                         | 44   |
| 4.4.2 ACS-VEI.....   | 47   |
| 4.4.3 ACS-TIME.....  | 49   |
| 4.4.4 Solution Constructive Procedure.....   | 50   |
| 4.5 Numerical Results .....  | 55   |
| 4.6 Case Study-Problem of St. Mary's Food Bank Distribution Center (With Time Windows).....        | 57   |
| 4.6.1 Data .....   | 58   |

| CHAPTER                                | Page |
|--|------|
| 4.6.2 Results .....                    | 58   |
| 5 CONCLUSIONS AND FUTURE RESEARCH..... | 61   |
| REFERENCES.....                        | 63   |
| APPENDIX                               |      |
| A RTR ALGORITHM.....                   | 68   |
| B ROUTING SOLUTION.....                | 71   |

## LIST OF TABLES

| Table   | Page |
|---|------|
| 2- 1 Brief History of Vehicle Routing Problem .....                               | 7    |
| 2- 2 VRP Variants and Its Characteristics.....                                    | 8    |
| 3- 1 Saving Algorithm .....   | 23   |
| 3- 2 Procedure of Merge for the Saving Algorithm.....                             | 25   |
| 3- 3 Local Search Operators.....  | 27   |
| 3- 4 RTR Algorithm.....   | 30   |
| 3- 5 Results for Christofides' Problem (Christofides et al., 1979).....           | 33   |
| 3- 6 Results for Taillard's Problem (Taillard, 1993).....                         | 34   |
| 3- 7 Routes on Thursday Generated by SMFB.....                                    | 36   |
| 3- 8 Routes on Thursday Generated by Local Search.....                            | 37   |
| 3- 9 Comparison between SMFB's Original Solution and Local Search's Solution..... | 37   |
| 4- 1 MACS Algorithm .....   | 46   |
| 4- 2 The Algorithm of ACS-VEI.....  | 48   |
| 4- 3 The Algorithm of ACS-TIME .....  | 50   |
| 4- 4 Solution Constructive Procedure.....   | 52   |
| 4- 5 Features of Benchmark Problems .....   | 56   |
| 4- 6 Results for Benchmark Problems (Solomon, 1987) .....                         | 57   |
| 4- 7 Routes on Thursday Generated by SMFB.....                                    | 59   |
| 4- 8 Routes on Thursday Generated by MACS .....                                   | 59   |
| 4- 9 Comparison of SMFB's Original Solution and MACS's Solution.....              | 60   |

## LIST OF FIGURES

| Figure   | Page |
|--|------|
| 1- 1 Donors That Need Be Visited on Thursday (a) and Monday (b)..... | 3    |
| 3- 1 Capacitated Vehicle Routing Problem.....                        | 15   |
| 3- 2 Flowchart of Saving Algorithm .....                             | 22   |
| 3- 3 Saving Algorithm .....  | 24   |
| 3- 4 Procedure of Mergence.....                                      | 24   |
| 3- 5 Flowchart of RTR Algorithm.....                                 | 29   |
| 3- 6 Effect of Parameters on Computation Time.....                   | 31   |
| 3- 7 Effect of Parameters on Results .....                           | 31   |
| 3- 8 Routing Map of SMFB (a) and Local Search (b).....               | 38   |
| 4- 1 Architecture of MACS .....                                      | 45   |
| 4- 2 Flowchart of MACS .....   | 47   |
| 4- 3 Flowchart of ACS-VEI and ACS-TIME.....                          | 49   |
| 4- 4 Flowchart of Solution Constructive Procedure .....              | 51   |
| 4- 5 Routing of SMFB (a) and Routing of MACS (b).....                | 60   |



## CHAPTER 1

### INTRODUCTION

#### 1.1 Background

Since the deep collapse in 2008 and 2009, the economic recovery is uncommonly slow. The rate of poverty in the U.S. has reached the highest rate since 1993. There were 45.3 million people who lived in poverty in 2013. It has remained as the largest number since the first statistics report that published more than 50 years ago. (Hunger in America 2014)

The hunger problem in Arizona is even more serious: in 2014, 17.8% of Arizonans were food insecure, which exceeded the national average of 15.9%. At the same time, 28.2% of Arizona children faced hunger. With 456,760 children facing hunger on a daily basis, Arizona ranked the third in the country for high child food insecurity rate, only behind New Mexico and Mississippi.

To solve this hunger problem, nonprofit organizations play an important role in providing essential food and service to the underserved and vulnerable members of society. Feeding America is such a nonprofit organization that feeds America's hungry through a nationwide network. Based on the hunger study of Feeding America, 3.3 billion meals have been brought to more than 46 million people by Feeding America and its network. A large amount of this food is donated from surplus food such as grocery chains and supermarkets. As an active member and one of the largest distribution centers in Feeding America, St. Mary's Food Bank collects dietary products from grocery chains or supermarkets, stores goods in a warehouse, and distributes meals to hungry people.

Founded in 1967, St. Mary's Food Bank (SMFB) is the world's first Food Bank. It has already distributed more than 700 million pounds of food to Arizona since it was set up. In addition, as the world has recognized its efficiency in food distribution, SMFB has been regarded as the

blueprint for Food Banks over the United States. With the rising demand yet the resources that are becoming more limited, the challenge that SMFB faces is even more complicated. Operations research methods have been applied to logistics problems and yielded many successful stories. Therefore, in this research we adopt the corresponding operations research technique to improve the efficiency of goods collection in SMFB.

## 1.2 Problem Overview

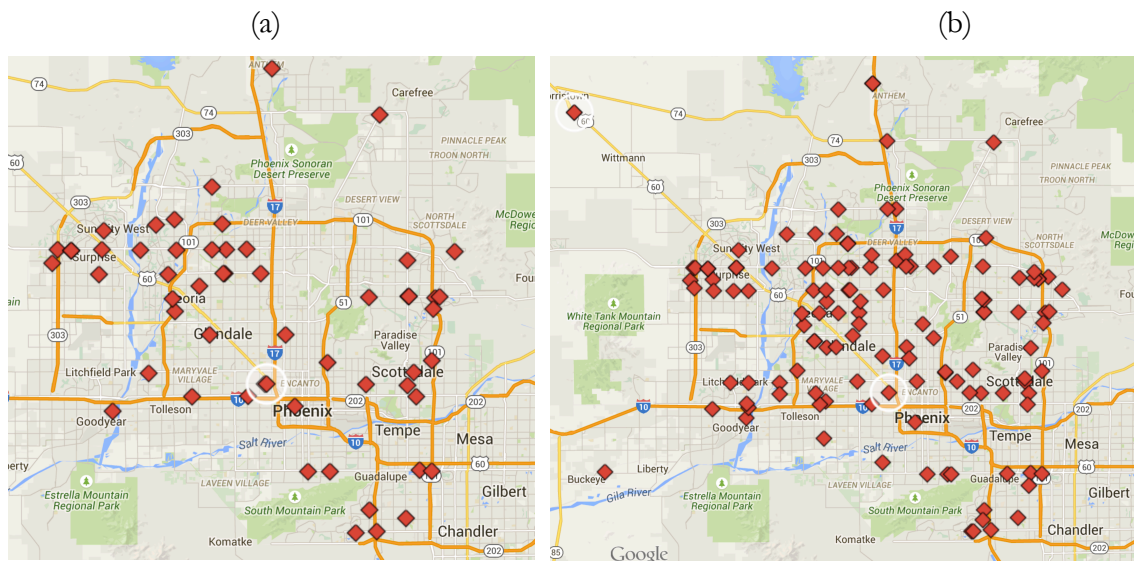
SMFB provides services all over the Arizona. 330 partner agencies are located in 430 different sites. Most of these partner agencies are dining halls, domestic violence shelters, churches, food pantries, schools, children's shelter, senior centers and halfway houses. As a nonprofit organization, SMFB faces great challenges in its operation. Firstly, demands and donations are uncertain. In SMFB, dietary products are collected from hundreds of resources, such as donations from different supermarkets and buying products from government with low price. With uncertain demands and limited distribution resources, sometimes SMFB cannot receive donated products in time. As a result, lots of fresh foods are wasted and a number of people remain hungry. Secondly, the inefficiency in distribution channel also needs to be settled. The SMFB's services cover over 81,000 square miles in center and Northern Arizona, including Coconino, Navajo, Apache, Gila, Yavapai Mohave Counties and two-thirds of Maricopa County. Only in the Northern Arizona communities, SMFB has distributed almost 12 million pounds of food in fiscal year 2011-2012, nearly 10 % increase over the previous year. Such a complex and large system is not easy to be managed efficiently. After collecting donated products, SMFB classifies food into different pallets, stores pallets in warehouses and distributes meals to different agencies. All of these challenging tasks requires the assistance from volunteers.

Among SMFB’s problems, distribution efficiency has a huge impact on the operational activities. In SMFB, every dollar could serve seven meals to hungry people, at the same time, transportation cost of distribution takes the majority of the budget. If the distribution center could improve the transportation efficiency, more hungry people could be served. In this thesis, our research will focus on improving the efficiency of collection between donation suppliers and SMFB distribution center.

### 1.3 Problem Definition

Figure 1- 1

Donors That Need Be Visited on Thursday (a) and Monday (b)



On each weekday, 12 trucks in SMFB are dispatched from distribution center (Depot) and each truck visits eight to seventeen donors (Customer) to pick up donated goods within capacity of 40,000lb. Every donor is visited only by one truck and takes an average of fifteen to thirty minutes to load goods. To reduce the transportation cost, the objective of this research is to minimize the number of trucks and total travel time. Figure 1 shows donors that need to be picked up on Thursday (a) and Monday (b). Take Thursday as an example.

On Thursday, 54 donors need to be visited and total 151,200 lb. goods need to be picked up. The donors are located in Maricopa County. At SMFB's distribution center, 12 trucks with a capacity of 40,000 lb. are available to pick up foods from 5 a.m. to 16 p.m. on Thursday. Our objective is minimizing the number of trucks and total travel time while picking up all products from 54 donors. Currently, in SMFB distribution center, all routes are designed by commercial software, which is widely used in routing problem. In the commercial route design software, it only considers the travel distance, several other important factors, such as truck capacity, traffic situation, weights of pickup and time windows are not considered in the solution, which makes solution relatively limited.

#### 1.4 Resource

We obtain the following information from SMFB: a depot (SMFB distribution center), 12 trucks that pick up food from donors, the capacity of each truck, addresses of donors and depot and weight of products that need to be picked up. To be more practical, all the travel time between donors and distribution center are calculated through Google map. Moreover, to compare performance of algorithms used in this thesis with other well-known algorithms, Christofides' benchmark (Christofides, 1969) and Taillard's benchmark (Taillard, 1993) are used to test algorithms in Capacitated Vehicle Routing Problem (CVRP). Solomon's instances are applied to test algorithms in the Vehicle Routing Problem with Time Windows (VRPTW).

#### 1.5 Thesis Structure

This thesis is organized as follows: After investigating SMFB distribution center's problem, this problem is focused to belong to the class of Vehicle Routing Problem (VRP) and its extensions. In chapter 2, the development of the vehicle routing problem and former research results are introduced as the literature review. With deep understanding of VRP, we discuss SMFB's distribution problem in detail in chapter 3. Based on SMFB's present situation, a

model of Capacitated Vehicle Routing Problem (CVRP) is built for this problem. Meanwhile, Clarke and Wright algorithm and local search algorithm are adopted to generate solution for SMFB. Since some of donors in SMFB have time windows for picking up donated products, in Chapter 4, SMFB's problem is extended to Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) by adding time window constraints. Multiple Ant Colony System (MACS) algorithm is applied to solve this complicated problem and improve the efficiency of distribution in SMFB. Last, conclusions are drawn and recommendation of future research on this problem is given.

## CHAPTER 2

### LITUREATURE REVIEW

#### 2.1 Introduction

Improving the efficiency of transportation for distribution centers is highly related to the Vehicle Routing Problem (VRP). The VRP is one of the most famous combinatorial optimization problems. Dantzig and Ramser (1959) introduced the VRP based on a real gasoline delivery problem. A mathematical programming formulation and an algorithm were proposed for VRP. Ever since then, the VRP has given rise to a well-known and important field in Operation Research. Generally, the VRP deals with the transportation of goods between customers and depots. All vehicles need to start at and return back to depot. Every customer can only be visited once. The goal of the VRP is to generate a minimum set of routes that satisfy all customers' demand. The VRP is easy to state and understand, while it is hard to solve in real world, especially combined with other practical constraints. In the domain of the VRP, there are numerous problems with specific constraints. For example, in the Heterogeneous VRP (Roberto Baldacci, 2007), more than one type of vehicles were assigned delivery tasks; Solomon (1995) presented the VRP with Time Windows (VRPTW), customers must be served in a given period; in the Periodic vehicle routing problem (PVRP) (Russell and Igo, 1979), the delivery plan involves several days at a time. In this research, we focus on VRPTW mainly.

In 1981, Lenstra and Rinnooy Kan (1981) studied the complexity of the VRP and concluded all VRP are NP-hard. In 1988, Solomon and Desrosiers (1988) also proved the VRPTW to be an NP-hard problem. As a result, only a small number of instances of VRPTW can be solved and proved optimality. The following is the brief history of VRP.

Table 2- 1

Brief History of Vehicle Routing Problem

| Decade | Events   |
|--------|--|
| 1950s  | Dantzig and Ramser (1959) introduced Vehicle Routing Problem;<br>Some small instances (10 to 20 costumers) were solved.  |
| 1960s  | Clarke and Wright algorithm (1964) was proposed to build routes;<br>2-opt and 3-opt were applied to improve the solution (Christofides, Eilon, 1969) ;<br>Some instances with 30 to 100 customers were solved. |
| 1970s  | 2-phase heuristic were developed (Gillett and Miller, 1974);<br>Some small instances (25 to 20 customers) have been solved with optimal solution.  |
| 1980s  | Interactive heuristic were developed (Cullen and Jarvis, 1981);<br>Some instances with 50 customers have been solved with optimal solution.  |
| 1990s  | Metaheuristics were proposed for VRP;<br>Some instances with 50 to 100 customers have been solved with optimal solutions.  |

There are many realistic assumptions in VRP, such as one route per vehicle, a homogeneous fleet of vehicles and single depot. If some additional constraints are introduced, these assumptions could be eliminated and the basic VRP could be extended to new VRP variants. Table 2-2 shows major VRP variants and their characteristics. Several researchers have studied this problem extensively.

Table 2- 2

VRP Variants and Its Characteristics

| Problem                                      | Objective  | Focus    | Comments   |
|--|--|----------|--|
| Traditional VRP                              | Minimize distance                                | Fleet    | Toth (1981) proposed an overview of the traditional VRP, Cordeau (1997) provided an overview of the heuristic to solve this problem.           |
| VRP with balance                             | Balance daily work                               | Driver   | Levy and Bodin [1988] developed constraints to obtain good solutions with balanced daily work and Sniezek et al. (2006) extended this problem. |
| Period Vehicle Routing Problem (PVRP)        | Account for time period constraints              | Demand   | Francis (2007) defined operational complexity as the difficulty in the PVRP  |
| Inventory Routing Problem (IRP)              | Ensure customer will not out of product          | Demand   | Demands in the IRP are stochastic based on monitoring and forecasts.   |
| Consistent Vehicle Routing Problem (Con VRP) | Ensure customers are served with the same driver | Customer | Each driver has the same routes to visit every day. Wong (2008) proposed some practical issues.  |
| Vehicle Routing Problem with Time Windows    | Minimize travel cost within time windows         | Customer | Specific time window constraints are considered in this problem.   |



## 2.2 Capacitated Vehicle Routing Problem

Capacitated Vehicle Routing Problem (CVRP) is an extension of VRP. In CVRP, the vehicle capacity constraint is now included. With capacity constraint, each vehicle has a given capacity and cannot load more goods than its capacity. The objective is to find the optimum road with minimum transportation cost and maximum satisfaction of customer with given constraints.

In the past three decades, numerous studies have been focused on solving this problem. Several approaches have been extended from direct tree search with Branch-and-Bound (Christofides and Eilon, 1969) to column generation, Branch-and-Cut algorithms and metaheuristics. All of these methods have provided a good quality solution under the scope of this type of problems.

## 2.3 Capacitated Vehicle Routing Problem with Time Windows (CVRPTW)

The Capacitated Vehicle Routing Problem with Time Windows is the CVRP with additional time window constraints. In real world, some customers can only be served in a given time period, we refer this time period as the time windows. Consequently, time window constraints should be taken into consideration. In CVRPTW, customers could begin to be served within a time window  $[E_i, L_i]$ . If vehicles arrive at customer  $i$  before  $E_i$ , vehicles could wait until  $E_i$ , since there is no additional cost to wait. If the vehicles arrive at customer  $i$  after  $L_i$ , as the customer is not available, the vehicle cannot pick up anything and additional penalty costs should be taken into consideration. Solomon (1987) proposed the sequential insertion heuristic to solve CVRPTW. Meanwhile, a problem set with different percentages of time windows, positioning and tightness were generated to test performance of algorithms used in CVRPTW.

## 2.4 Algorithms

Since the VRP is an NP-Hard problem, it is typically difficult to solve, especially in real world. Various studies have been investigated regarding to this problem for a long time. Generally, the algorithms can be broadly classified into three methods: exact algorithms, classic heuristic algorithms and metaheuristics algorithms. The common exact algorithms for solving VRPs focus on branch-and-bound algorithm and dynamic programming. However, even with the most modern computing technique, the best exact algorithm can only solve some instances with less than 100 vehicles (Fukasawa, 2006, Baldacci et. al, 2008). In practice, over thousands of customer locations and additional constraints could be involved in the distribution problems. As many assumptions are considered, the problem itself is only more challenging. Adopting heuristic methods to obtain a nearly good solution within a short time limit is necessary.

### 2.4.1 Exact Algorithms

Over the past 40 years, branch-and-bound algorithm and mathematical programming approaches are quite popular in the category of exact algorithm. Christofides (1969) proposed and compared branch-and-bound algorithm with ‘saving’ approach, 3-optimal tour method in solving VRPTW. To improve the performance of branch-and-bound algorithms, considerable effort has been spent on this problem. Christofides et. al (1981) generated two sharp lower bounds based on K-degree center tree(k-DCT) and q-routes. An instance with  $10 < n < 25$  has been solved with this two sharp lower bounds. Also with the same lower bound, Hadjiconstantinou (1995) solved the instance with  $n < 50$  with improved branch-and-bound algorithm. Fisher (1994) has solved several instances with even up to 134 nodes.

Eilon and Watson-Gandy (1971) first developed a dynamic programming to solve the VRP. Christofides et. al (1981) applied a state-space relaxation to solve the instance with up to 25 vehicles.

For exact algorithms, attention has also been focused on vehicle flow formulations and algorithms (Laporte and Nobert 1983), commodity flow formulations and algorithms (Gavish and Grave 1979, Hadjiconstantinou 2004), set partitioning formulations and algorithms (Balinski 1964, Baldacci 2008). Most successful instances are given by Hadjiconstantinou (2004) and Baldacci (2008). Hadjiconstantinou solved some instances with up to 135 by applying branch-and-cut. Baldacci used Set Partitioning formulation to build the model and several instances with up to 121 were solved by an integer linear programming method. With exact algorithms, only the instance with a small number of customers can be solved with an optimal solution. For problems with large number of customers or tight constraints, it's quite hard to obtain a feasible solution in a short time.

#### 2.4.2. Classical Heuristic:

From the 1960s to 1990s, various researches were focused on the classical heuristic, which includes saving algorithm and cluster-first, route-second heuristic.

##### 2.4.2.1 Saving Algorithm

Clarke and Wright (1964) developed a saving algorithm that could generate an optimal or near-optimal route rapidly for VRP. Saving Algorithm works equally well for the directed and undirected problems that the number of vehicles is not fixed and becomes a popular heuristic method for VRP. In Saving Algorithm, distance and travel timesaving would be calculated when two routes merge together. Based on the saving order, routes are merged together sequentially and a better feasible solution will be generated. Several improvements also have been proposed for Saving Algorithm. Golden (1977) added a positive weight to  $C_{ij}$  to avoid circumferential routes. Altinkemer and Gavish (1991) and Wark and Holt (1994) merged route with a matching algorithm to get a global optimal or near optimal solution. Otherwise,

Paessens (1988) accelerated the saving computation and Nelson (1985) chose efficient data structures to reduce the time in the consuming matching problem.

#### 2.4.2.2 Cluster-First, Route-Second Heuristic

Fisher and Jaikumar algorithm (1981) is also a well-known cluster-first, route-second algorithm. In this algorithm, seeds are generated first and a cluster is built to minimize the distance within the cluster by generalized assignment problem (GAP). In each cluster, a new route will be generated by solving the Traveling Salesman Problem (TSP). Some instances solved by this method do have road length constraints. However, the detail information on how to handle the length constraint was not quite clear.

Sweep algorithm (Wren, 1971) is another popular cluster-first, route-second method. Two parts are consisted in this algorithm: Split and TSP. A feasible initial cluster would be generated in Split phase and then a route would be obtained in this each cluster by solving a Travel Salesman Problem (TSP). It is first mentioned by Wren (1971) but was popularized by Gillett and Miller (1974).

#### 2.4.3 Metaheuristics

Most metaheuristics are adopted for improvement phase. A good metaheuristics algorithm could generate an equally good solution even with a low-quality initial solution. In recent years, metaheuristics are broadly applied in solving VRP problems with large size in a short time. When it comes to VRP, metaheuristics are widely classified into three methods: population search, learning mechanisms and local search.

#### 2.4.3.1 Population Search

Genetic Algorithms (GA) are the most famous method of this paradigm (Holland 1975). Just like genetic mutation, new offspring will be generated by recombining some extracted parts with current parent solution. If there is any improvement in objective function, the worst element will be replaced by the new offspring. To apply these methods to VRP, local search method is a necessary part to improve the offspring (Prins 2004, 2009, Mester and Braeysy 2005, 2007, Nagata 2007).

#### 2.4.3.2 Local Search

Local search has been proven to be an effective method to generate a good solution to VRPTW. Generally, local search defines a search neighborhood that contains feasible solutions generated by inter-routes moves. Tabu search (Gendreau, 1994) is a well-known search method that explores the solution space by forwarding to the best solution in a subset of its neighborhood. In tabu search, solutions that process a given attribute of the current solution are not considered for a number of iterations (Cordeau and Gendreau 1997, Derigs and Kaiser 2007).

Mladenovic (1997) proposed variable neighborhood search to build a heuristic by a systematic change of the neighborhood. To avoid the local optimum, neighborhoods will exchange with each other once a new better solution is identified. Savelsbergh (1991) has applied the local search algorithms to solve the VRPTW based on k-interchange concept.

Chris Groer (2008) has developed an open source C++ package named VRPH to solve the VRP. In VRPH, a well-documented and modular library was developed. Many well-known local search heuristic (Clarke-Wright, Two-OPT, Three-OPT, OR-OPT, Cross exchange etc.) were applied to generate the solution to the VRP. In well-studied VRP benchmark instance, the solution given by VRPH is typically within a percent or two of the best-known solution.

#### 2.4.3.3 Learning Mechanisms

Ant colony optimization (ACO) is a well-known form of learning mechanism. In real world, ants deposit pheromone (chemical traces) on their path when they are searching for food. Shorter routes will attract more ants by stronger pheromone with time. In the ACO mechanism, it mimics the behavior of real ants. The edge generating more good solutions will be given more weight. This mechanism was firstly introduced by Dorigo (1992) in his PhD thesis. To improve the quality of the solution and computation time, Luca Maria Gambardella (1999) coordinated the action of different ant colonies, each colony operates a specific objective by using different pheromone trails. This mechanism is competitive with the existing methods in solving VRPTW.

#### 2.5 Researches on Distribution and Collection of Nonprofit Organization

Establishing efficient network to collect and distribute necessities from donators to needed people is one of the tightest bottlenecks for nonprofit organization. Lien et al. (2013) have researched distribution operations of Greater Chicago Food Depository. He proposed a multi-vehicle sequential allocation problem that considers providing equitable service and minimizing wasted donations for nonprofit operations. After studying the behavior of the model in clustering donors and agencies, the impacts of demand variability and supply availability on route composition and solution performance, they introduced an efficient decomposition-based heuristic method in this problem. However, it has not taken real traffic situation at different time and time window constraints of donors into consideration while all of these constraints are the key factors affecting the result in practice.

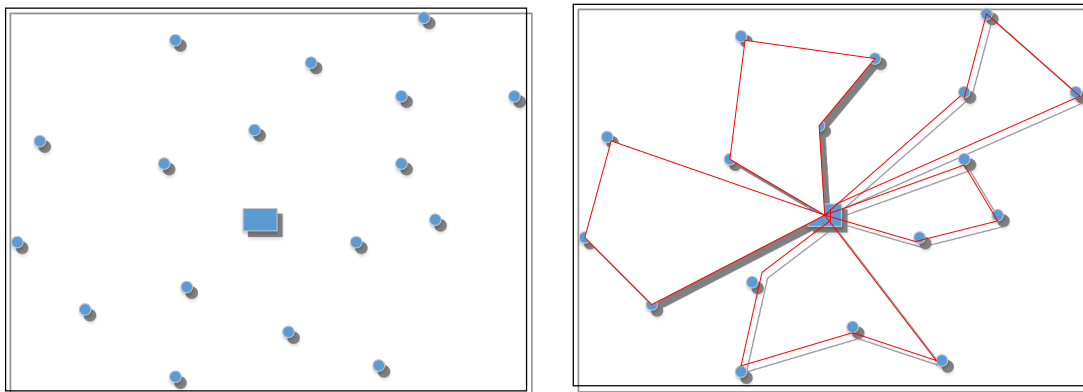
## CHAPTER 3

### CAPACITATED VEHICLE ROUTING PROBLEM

Currently, in SMFB distribution center, all routes are designed by commercial software, which only considers the travel distance. Several other important factors, such as truck capacity, traffic situations, weight of pickup and time windows are not considered in the solution. Also the traditional route design software cannot fulfil the expectations of dairy products pickup and may be quite inefficient for distribution. In this thesis, we begin to apply operations research methods in SMFB distribution center. In SMFB Distribution center, all trucks have a given capacity, which is 40,000 lb. On every weekday, trucks will depart from distribution center, visit donors one by one and return back to depot finally. This is a typical Capacitated Vehicle Routing Problem (CVRP). Currently SMFB distribution center uses the commercial logistic design software called Road Designer to design the road sequence. As the software only considers location of donors and total travel distance, it generates limited solutions for SMFB's distribution center. In this chapter, we build a model of CVRP for this problem and adopt saving algorithm and local search to improve the solution.

Figure 3- 1

Capacitated Vehicle Routing Problem



### 3.1 Problem Description and Characteristics

CVRP is a classical VRP with additional capacity constraint that each vehicle should have a uniform capacity. It is a fundamental problem in combinatorial optimization with wide application in practice and also within the core of logistics planning. As Figure 3-1 shows, in CVRP, all nodes will be visited exactly one time by a single truck. Each truck has a fixed capacity and starts from and returns back to the same depot. All donors have known weight of donated goods to be picked up. In addition, all donors should be serviced and the travel time should be minimized. All trucks are capable of picking up any donation supplier and have the same capacity. Regarded as a well-known VRP variant, CVRP has the following characteristics.

#### 3.1.1 Transportation Requests

The way transportation requests become available is an important characteristic of routing problems. There are two ways to know requests, dynamic way and statistic way. In a dynamic situation, some demands are known during the accomplishment of the route. In a static situation, all the requests are known before constructing the routes. In CVRP, we typically assume all the transportation requests are static and known before constructing the routes.

#### 3.1.2 Objective Functions

A number of objective functions have been applied in VRP. The most well-known ones are discussed as follows.

***Minimize the number of trucks:*** The number of trucks refers to total trucks that pick up donated products from donation nodes every business day. In distribution center, trucks and drivers cost the most in the transportation cost. Accordingly, minimizing the number of vehicles is usually the main objective of route optimization.



**Minimize route length:** The route length is the total distance that trucks have traveled from the depot to different donation nodes to pick up donated goods and return back to the depot. It seems the route length could reflect the travel cost immediately. However, in route length, the traffic situation, which is the key factor that affects the efficiency and cost of the transportation, has not been taken into consideration.

**Minimize duration:** The route duration takes all the time into consideration. It includes travel time, break time, waiting time, loading and unloading time. If the start time of the truck leaving the distribution center is set to zero, the completion time should be the arrival time when the truck returns back to the distribution center.

**Minimize travel time:** The route travel time refers to the total time spent on the transportation between different donation nodes. In Food Bank, all drivers' salaries are fixed and their working time is at most 11 hours every weekday. All donated goods have been packed in pallets before picking up and it is quite quick to load goods on the truck. We typically assume the loading time is fixed. With these specific situations, minimizing travel time is more reasonable than minimizing road duration, since there is no other cost for distribution center in the waiting times, break times, loading and unloading times. In this research, we obtain the real travel time matrix through Google Map. Traffic and road situations have already been considered in the travel time matrix through Google Map. This will make our results more practical.

Based on the discussion above, both of **the travel time** and **the number of trucks** should be involved in the objective function. As the trucks and drivers account for most of the costs,

the number of trucks should weight more than travel time. Thus a solution with fewer trucks is always better than others.

### 3.2 Formulation

The CVRP is formulated as a mixed integer program with the minimization of the number of trucks and total travel time. Since drivers and trucks contribute most of the costs in transportation, the number of tracks is more important than total travel time in the evaluation. In the objective function of this formulation, to enforce the number of trucks to be more important, a large constant  $B$  multiplies with the number of routes. Accordingly, in the objective function, the first term, which is also the most important term, is the number of trucks, and the second term, which minimizes the travel time, is the total time spent on all transportation.

In CVRP, we assume there are  $NV$  identical vehicles, each with capacity  $C$ ,  $NN$  number of donation nodes and a directed network that connects the distribution center and donation nodes. Donation nodes are denoted by  $1, 2 \dots NN$ . The distribution center (depot) is denoted with  $0$ . All trucks start at and return back to depot  $0$ . The connections between nodes are donated as the corresponding arc of the network. A travel time  $t_{ij}$  is assigned for each arc  $(i, j)$ . To be more practical, the travel time  $t_{ij}$  is calculated by Google Map and a real travel time matrix is obtained by this calculation. Each truck has a given capacity  $C$  and each donation node has a pick up demand  $Q_j$ .

**Parameters:**

$NN$ : Number of donation nodes

$NV$ : Number of vehicles

$C$ : Capacity of vehicles

$Q_j$ : Demand at node  $j$  ( $Q_i=0$ )

$s_i$ : Loading and unloading time at node  $i$  ( $t_i=0$ )

$t_{ij}$ : Travel time from node  $i$  to  $j$  ( $t_{ij}=\infty$ )

$D_{kj}$ : Departure time of truck  $k$  at node  $i$

$E_i$ : Earliest time available at node  $i$

$L_i$ : Latest time available at node  $i$

$M, B$ : Big constant number

**Set of nodes:**

$N(k)$ : Set of nodes visited by truck  $k$

**Variables:**

$$l_{ijk} = \begin{cases} 1 & \text{arc } i, j \text{ is traversed by vehicle } k \\ 0 & \text{Otherwise} \end{cases}$$

$$V_k = \begin{cases} 1 & \text{if vehicle } k \text{ used} \\ 0 & \text{Otherwise} \end{cases}$$

**Formulation:**

$$\text{Min } Z = B * \sum_{k=1}^{NV} V_k + \sum_{i=0}^{NN} \sum_{j=0}^{NN} \sum_{k=1}^{NV} l_{ijk} * t_{ij} \quad (3-0)$$

Subject to:

$$\sum_{i=0}^{NN} \sum_{k=1}^{NV} l_{ijk} = \sum_{i=0}^{NN} \sum_{k=1}^{NV} l_{jik} \quad \text{for all } j \quad (3-1)$$

$$\sum_{i=0}^{NN} \sum_{k=1}^{NV} l_{ijk} = \mathbf{1} \quad \text{for all } j \quad (3-2)$$

$$C - \sum_{j=1}^{NN} \{ Q_j \sum_{i=0}^{NN} l_{ijk} \} \geq \mathbf{0} \quad \text{for all } k \quad (3-3)$$

$$\sum_{j=0}^{NN} l_{0jk} \leq \mathbf{1} \quad \text{for all } k \quad (3-4)$$

$$\sum_{i=0}^{NN} l_{i0k} \leq \mathbf{1} \quad \text{for all } k \quad (3-5)$$

$$\sum_{i=0}^{NN} \sum_{j=0}^{NN} l_{ijk} \leq M * V_k \quad \text{for all } k \quad (3-6)$$

$$\sum_{j=1}^{NN} l_{0jk} - V_k = 0 \quad \text{for all } k \quad (3-7)$$

$$\sum_{i=1}^{NN} l_{i0k} - V_k = 0 \quad \text{for all } k \quad (3-8)$$

$$u_{ik} - u_{jk} + C * l_{ijk} \leq C - Q_j \quad \text{for all } i, j, k \quad (3-9)$$

$$Q_i \leq u_{ik} \leq C \quad \text{for all } i, k \quad (3-10)$$

$$\sum_{i=0}^{NN} \sum_{j=0}^{NN} l_{ijk} * t_{ij} \leq 11 \text{ h} \quad \text{for all } k \quad (3-11)$$

In the objective function (3-0), the first term is the number tracks used. To set the hierarchical nature of objective, it is multiplied by a big constant  $B$ . The second term in the objective function is the total travel time of all routes. Constraints (3-1) forces the number of trucks flow into and out of node  $j$  to be the same. Constraints (3-2) states that the demand at every donation node can be completely satisfied by exact one truck. Since each truck has a given capacity, constraints (3-3) enforces no truck can visit more donation nodes than its capacity enable it to. Constraints (3-4), (3-5) state that the truck  $k$  could only be used at most once every day and a constraint (3-6) makes sure  $V_k$  is 1 if vehicle  $k$  is used. Otherwise,  $V_k$  is zero. Constraints (3-7) (3-8) ensure the truck  $k$  should start from and return to the depot. Constraints (3-9) are sub-tour elimination constraint and capacity constraint. If vehicle  $k$  doesn't visit from  $i$  to  $j$ , constraints (3-9) is always true; if it does,  $l_{ijk}$  is 1. To make sure no other sub-tour starts from node  $j$ , the load of  $j$  cannot be more than the sum of load of  $i$  and the pickup weight at  $j$ . Constraints (3-10) enforces vehicle's load will not exceed vehicle's capacity after visit customer  $i$ . Constraints (3-11) makes sure the work time for every driver cannot exceed 11 hours.

However, as the feasible region for this formulation is nonconvex, the formulation itself could only provide us the understanding in mathematical perspective. However, the computational complexity is still very challenging. Even with most modern computing technique, the best exact algorithm can only solve some instances with less than 100 vehicles.

### 3.3 Solution Algorithm

In SMFB, trucks visit hundreds of donors to pick up donated products. Solving this problem with exact algorithm within reasonable time limit is very difficult. Meanwhile, several complicated conditions are added to this problem, such as truck capacity, time windows and route length constraints. These additional constraints make this problem more difficult. Numerous heuristics algorithms have been applied to this problem successfully. Heuristic methods based on tabu search have received significant attention in operations research (Chiang W.C. and Russell R.A., 1997). Lee et al. (2003) has compared tabu search and other competing heuristics and found that tabu search method has been shown to offer better feasible solutions. Accordingly, our research focuses on applying local search methods enhanced by tabu structure to solve CVRP.

The solution process is separated into two steps, initialization and improvement. In initialization step, Clarke and Wright (Saving) algorithm is adopted to generate an initial feasible solution in a short time. In improvement phase, several local search operators are applied in Record-to-Record algorithm to improve the solution.

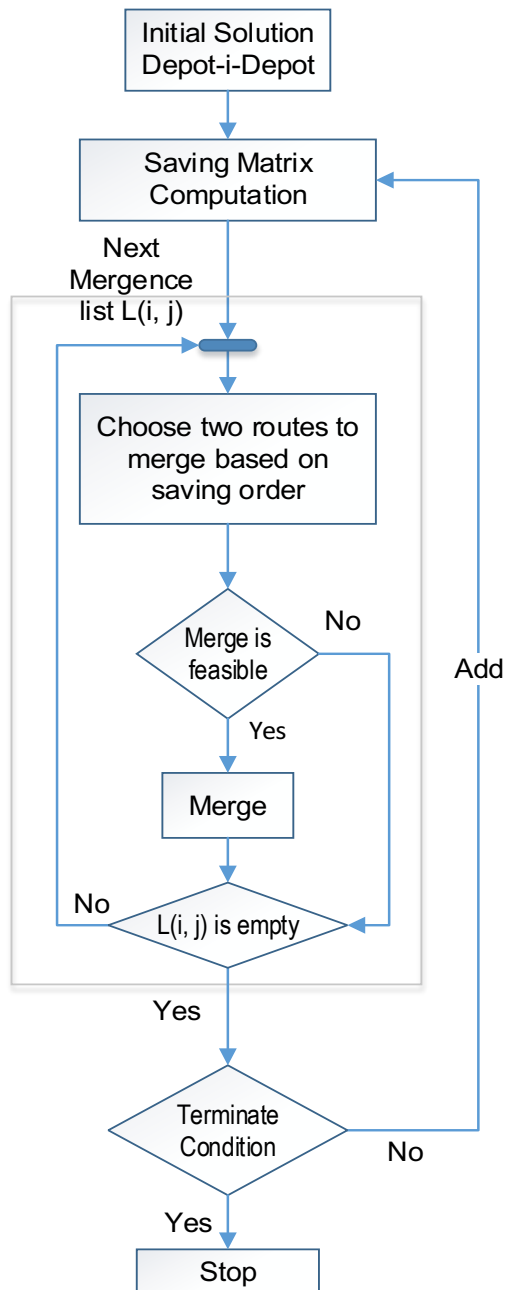
#### 3.3.1 Clarke and Wright Algorithm (Saving Algorithm):

In 1964, Clarke and Wright proposed a saving algorithm to solve the VRP when the number of vehicles is not fixed. This saving algorithm has become a well-known heuristic for solving VRP since that time. As a heuristic algorithm, Clarke and Wright algorithm cannot promise to

get the optimal solution but can guarantee a feasible solution. As Figure 3-2 shows, the basic idea of this algorithm is to combine routes together based on the saving matrix.

Figure 3- 2

Flowchart of Saving Algorithm



**Terminate Condition:**

- (1) No feasible mergence is found
- (2) or exceeds calculation time

Table 3- 1

Saving Algorithm

|   |   |
|---|---|
| Step 0: Initialization.   |   |
|   | Every customer node is connected with depot as Figure 3-3 (a) shows.<br>There are $NN$ routes (Depot-Supplier $i$ - Depot) in the initial solution. |
| Step 1: Saving Matrix computation   |   |
|   | The saving $S_{ij} = t_{io} + t_{oj} - t_{ij}$ for $i, j = 1, \dots, NN$  |
|   | Create list $L(i, j)$ and sorted $(i, j)$ in descending order by merging savings.   |
| Step 2: Routes mergence   |   |
|   | Based on list $L$ , begin to merge routes from the beginning of list $L$ :  |
|   | Test the feasibility of the mergence:   |
|   | If the mergence is feasible, then two routes are merged together as Figure 3-3 (b) shows and a new route is generated;                              |
|   | Test the next two routes $(i, j)$ in list $L$ until all routes combination in saving matrix are tested.   |
| Step 3 After all route combinations in list $L$ are considered, termination conditions are tested, if it satisfies termination conditions, stop the calculation; else, update saving matrix with new merged routes and repeat routes mergence |   |

In the initial step, all nodes are connected to the depot and  $NN$  (number of nodes) routes are generated as an initial solution. Secondly, the saving matrix is generated by calculating the saving travel time for merging any two possible routes  $(i, j)$  together. Based on the saving matrix, a list  $L(i, j)$  is created and sorted in descending order by merging savings. With list  $L(i, j)$ , road  $i$  and road  $j$  are selected to be tested with the feasibility of mergence. If the mergence is feasible, these two roads are merged together and a new merged route is created. After all the routes combination in list  $L(i, j)$  are considered, it need test the termination conditions. If no feasible mergence is found in last iteration or it exceeds given calculation time, the process

is terminated. Otherwise, the saving matrix is updated with new routes. The detail of the procedure is described in table 3-1.

Figure 3- 3

Saving Algorithm

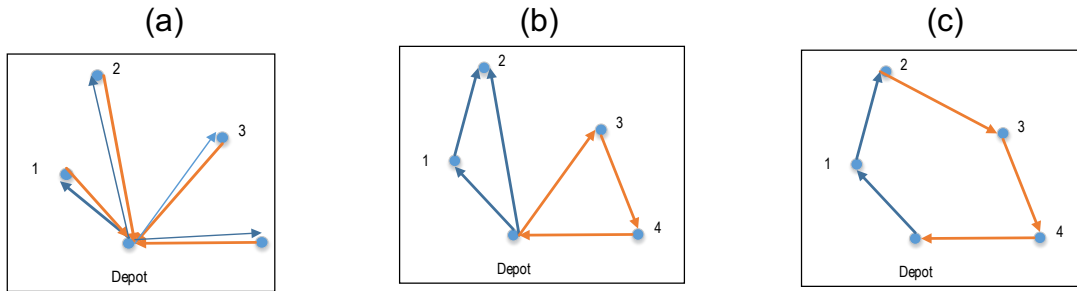
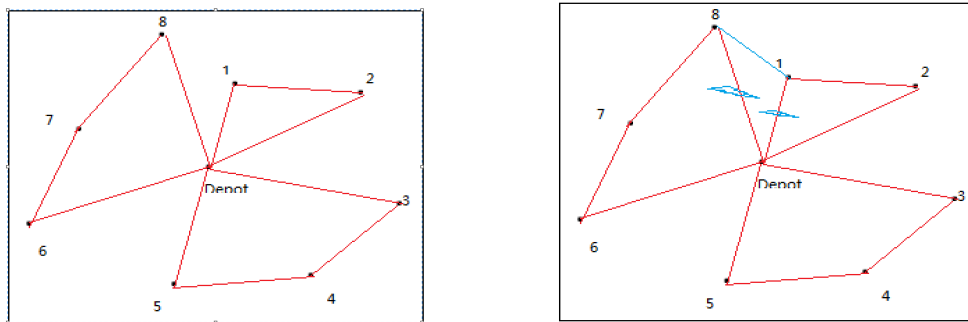


Figure 3- 4

Procedure of Mergence



In the saving algorithm, the key process is the routes mergence, where possible routes are combined together. In the mergence step, all the nodes are classified into four status: UNUSED, ADDED, INTERIOR and DEPOT. After the initialization, all the customer nodes are only connected with DEPOT and haven't been merged yet. They belong to UNUSED Status. After saving matrix computation, some routes are merged together. In merged routes, the first and the last customer node belong to ADDED status, since other routes can be added into these routes though connecting with these two nodes. For the other nodes in the merged route, they belong to INTERIOR and cannot be connected with other routes. If two routes are merged by connecting two nodes together, the status of these two



nodes is changed from ADDED to INTERIOR. For instance, in Figure 3-4, route (1) and (2) are merged together by connecting node 8 and node 1. The status of node 8 and node 1 is changed from ADDED to INTERIOR and these two nodes cannot be connected with other routes anymore. Since we only need to consider the possibility of connecting ADDED nodes together, the number of mergence is reduced fast when more and more ADDED nodes are changed to INTERIOR. The detail of mergence is given in Table 3-2.

Table 3- 2

Procedure of Merge for the Saving Algorithm

|                                      |   |
|--------------------------------------|---|
| For each combination $(i, j)$        |   |
| 1. if both $i$ and $j$ are UNUSED    |   |
|                                      | If the mergence is feasible,<br>Merge two routes;<br>status[ $i$ ]= ADDED; status[ $j$ ]= ADDED;      |
| 2. if $i$ is ADDED but $j$ is UNUSED |   |
|                                      | If the mergence is feasible,<br>Merge two routes<br>status[ $i$ ]= INTERIOR; status[ $j$ ]= ADDED;    |
| 3. if $j$ is ADDED but $i$ is UNUSED |   |
|                                      | If the mergence is feasible,<br>Merge two routes<br>status[ $j$ ]= INTERIOR; status[ $i$ ]= ADDED;    |
| 4. if $i$ and $j$ both are ADDED     |   |
|                                      | If the mergence is feasible,<br>Merge two routes<br>status[ $i$ ]= INTERIOR; status[ $j$ ]= INTERIOR; |
| 5. if $i$ and $j$ both are INTERIOR  |   |
|                                      | Skip  |

### 3.3.2 Local Search

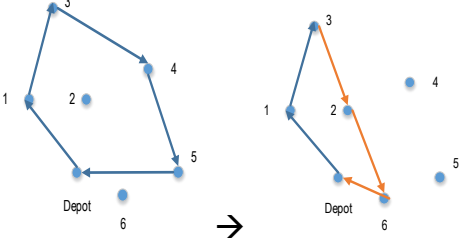
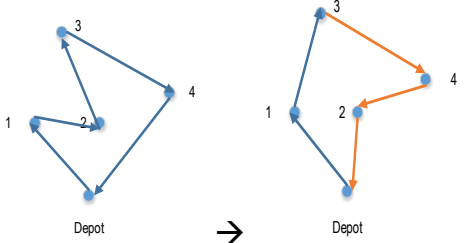
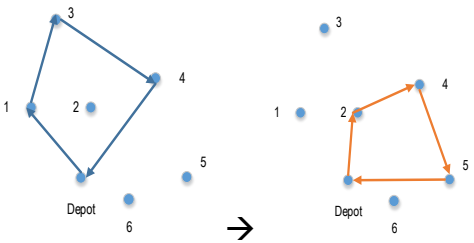
In the second phase, local search procedure is used to improve the initial solution. Local search is a widely-used technique to solve the combinatorial optimization. In a given minimization problem,  $\text{Min } f(x)$ , Subject to  $x \in S$ , where  $f(x)$  is the objective function and  $S$  is a solution space. Given a couple of constraints, we generate a feasible solution and its neighborhood  $N(x)$ , which are also feasible for all constraints. Then we will search for new solutions within the neighborhood. If we find a new  $x'$  and  $f(x') < f(x)$ , then it's an improving movement for the result. The key in this process is to generate a good solution within the constraints. Local search is an effective way to generate a good solution to classical VRP as well as VRPs with additional constraints such as multiple depots and vehicle with given capacity. From an existing solution, a new search neighborhood could be obtained though some well-defined movements such as swapping the visit sequence of some donation nodes or removing edges from routes and replacing them with new other edges.

A general local search operator was formally described in Tabu search algorithm (E. Hadjiconstantinou et. al. 1995). In this general local search, the position of  $n$  customer will be changed with  $\pi$  customers in another route. To control the computation complexity,  $n$  and  $\pi$  are usually less than three. The One-point move, Two-point move, Three-point move, Two-opt move, Three-opt move are special cases of the general local search operator. E. Taillard (1993) applied such operators to improve the solution and got a number of the best solutions for many benchmark instances. In this thesis, seven operators are used to generate new solutions in local search, they are One-point move, Two-point move, Two-opt move, Three-point move, Three-opt move, cross change, Or-opt. These local search operators are typically Tabu search operators. In  $n$ -point move,  $n$  nodes will be replaced with  $n$  nodes in another route. Similarly, in  $n$ -opt move,  $n$  edges will also be replaced with new other edges. The detail descriptions of these seven operators are summarized in table 3-3.

Table 3- 3

Local Search Operators

| Operator              | Description   | Example |
|-----------------------|---|---------|
| <p>One-point move</p> | <p>Replace an existing node to a new position;</p> <p><math>0-1-2-0 \rightarrow 0-1-3-0</math></p>  |         |
| <p>Two-point move</p> | <p>Swap the position of two nodes:</p> <p><math>0-1-2-3-4-0 \rightarrow 0-1-3-2-4-0</math></p>  |         |
| <p>Two-opt move</p>   | <p>Remove two edges from the original solution and replace them with two new edges:</p> <p><math>0-1-3-4-2-0 \rightarrow 0-1-3-4-5-0</math></p>       |         |
| <p>Or-opt move</p>    | <p>Remove a string of two, three, or four nodes and insert the string into a new position :</p> <p><math>0-1-3-5-2-0 \rightarrow 0-1-3-5-0</math></p> |         |

|                            |  |   |
|----------------------------|--|---|
| <p>Three-opt move</p>      | <p>Remove three edges from the original solution and replace them with three new edges:</p> <p>0-1-3-4-5-0 → 0-1-3-2-6-0</p> |   |
| <p>Three-point move:</p>   | <p>Swap the position of a pair of adjacent nodes with the position of a third node :</p> <p>0-1-2-3-4-0 → 0-1-3-4-2-0</p>    |   |
| <p>Cross-exchange move</p> | <p>Remove four edges from two different routes and replace them with four new edges :</p> <p>0-1-3-4-0 → 0-2-4-5-0</p>       |  |

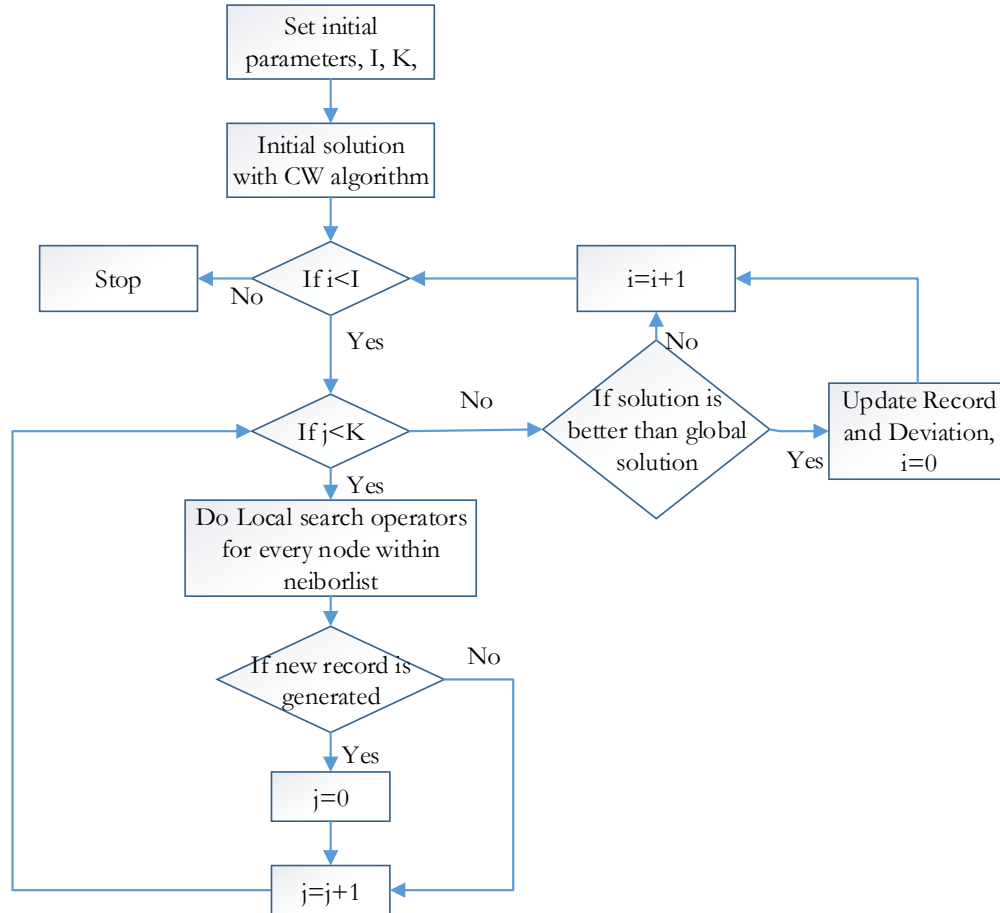
### 3.3.3 Record-to-Record Algorithm

When using local search to find a new solution, we use Record-to-Record (RTR) algorithm to record the best solution. RTR was introduced by Golden et al. (1998) and applied by Li et al. (2005) to solve large-scale VRPs. In RTR algorithm, Clarke and Wright algorithm would be used to generate the initial solution. Second, a local search operator, for example, One-point move, Two-point move and so on would be applied by using RTR algorithm. With satisfying all the constraints, nodes are exchanged between different routes. In this step, after a series of local operation exchanges, routes are allowed to improve and search in a local search process. Every time when a local optimum solution is obtained, a global re-initialization will perturb

the local optimum solution, exchange nodes with local search operators, improve and re-sequence routes. This process is repeated until the termination constraint is satisfied.

Figure 3- 5

Flowchart of RTR Algorithm



To reduce the computing complexity, we also consider only a given number of neighbors for each node when we use local search operators to improve the solution. The fixed number of neighbors' concept is not new, it was used in the traveling salesman problem before. Usually, we begin with a traditional fixed number of neighbor list within  $k=40$ . For every node  $i$ , all the edges whose length are greater than  $a * L$  will be removed, where  $L$  is the maximum length in all the neighbor list edges. We set  $a$  to be 1, if the number of neighbor list is equal to 40, otherwise,  $a$  is less than 1, if fewer edges are taken into consideration. Generally, as an increase,

the running time and accuracy will be expected to be increased. The detail of the algorithm is described in Figure 3-5 and Table 3-4.

Table 3- 4

RTR Algorithm

|   |   |
|---|---|
| Step 0: Initialization.                             |   |
|   | Parameters are $I$ , $K$ , and $\lambda$ . Set $I=30$ , $K=5$ , and $\lambda \in \{0.6, 1.4, 1.6\}$ .                 |
| Step 1: Starting solution.                          |   |
|   | Generate an initial feasible solution using the modified Clarke and Wright algorithm with parameter $\lambda$ .       |
|   | Set Record = objective function value of the current solution. Set Deviation= $0.01 \times$ Record.                   |
| Step 2: Improve the current solution.               |   |
|   | For $i=1$ to $I$ (I loop)   |
|   | Do local search operator and check feasibility  |
|   | If no feasible move is made, go to Step 3.  |
|   | If a new record is produced, update Record and Deviation.   |
|   | End $I$ loop  |
| Step 3: Repeat for $K$ consecutive iterations.      |   |
|   | If no new record is produced, go to Step 4.   |
|   | Otherwise, go to Step 2.  |
| Step 4: Perturb the solution.                       |   |
|   | Compare the solution generated after perturbation to the best solution generated so far and keep the better solution. |
| Step 5: Keep the best solution generated so far.    |   |
| Go to Step 1 and select a new value for $\lambda$ . |   |

Figure 3- 6

Effect of Parameters on Computation Time

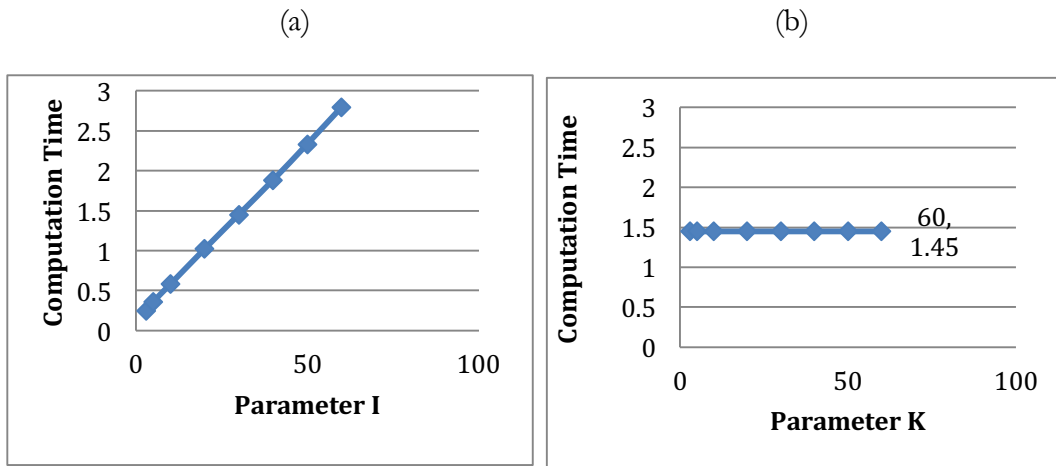
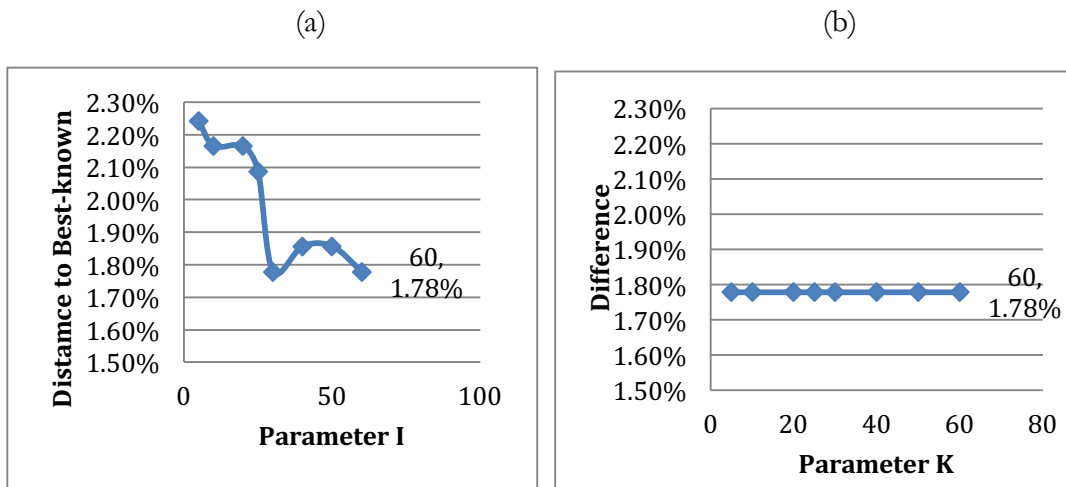


Figure 3- 7

Effect of Parameters on Results



In this algorithm, parameter  $I$  controls the maximum number of consecutive iterations allowed when the global record has not been updated and parameter  $K$  controls the maximum number of consecutive iterations allowed when the local record has not been updated. Increasing the values of  $I$  and  $K$  will lead to more iterations and more different solutions. Here, two Christofides' problems (Christofides, 1979) are chosen to test the effect of these two parameters on the computation time and the quality of the results. In Figure 3-6, we test the

effect on computation time. In Figure 3-6 (a) and (b), Parameter  $I$  and  $K$  are set to different values to obtain the computation time for the best result separately. From these two figures, we find that the computation time will increase as the parameter  $I$  increase while parameter  $K$  has little effect on the computation time. In Figure 3-7 (a) and (b), we test the effect of parameters on the quality of results. With given computation time, the distance to the best-known result is calculated as the quality of result based on different parameter  $I$  and  $K$ . From Figure 3-7, we find that, as parameter  $I$  increases, the result will be closer to best result, while parameter  $K$  has little impact on results. Based on the above observation, we choose  $I=30$  as there is no significant decrease on distance to the best result after  $I=30$  and  $K=5$  while  $K$  has little effect on the results and computation time.

### 3.4 Numerical Result

Several benchmark problems are widely used to test performance of algorithms in the CVRP. Among those benchmarks, Christofides' problem (Christofides, 1979) and Taillard's problem (Taillard, 1993) are most famous. In this chapter, Clarke and Wright algorithm and local search operators are coded in C++ in Linux system to solve this problem. These two problem sets were executed on Intel(R) Core™ i7-4770S CPU@3.10GHz. Our results are solving through the heuristics described in section 3.4.

The numerical results of Christofides' problem and Taillard's problem are presented in Table 3-5 and Table 3-6 separately. In Table 3-5 and Table 3-5, the first and the second column describe the name and the number of customer nodes in the test problems. The third and the fourth column are the number of vehicles and total travel distance in the solution.



Table 3- 5

Results for Christofides' Problem (Christofides et al., 1979)

| Problem | Number of Nodes | Number of Vehicles | Total Distance by Local Search | Best-known solution  | Difference above best known |
|---------|-----------------|--------------------|--------------------------------|----------------------|-----------------------------|
| 1       | 50              | 5                  | 524.61                         | 524.61 <sup>a</sup>  | 0.00%                       |
| 2       | 75              | 10                 | 845.18                         | 835.26 <sup>a</sup>  | 1.19%                       |
| 3       | 100             | 8                  | 830.17                         | 826.41 <sup>a</sup>  | 0.45%                       |
| 4       | 150             | 12                 | 1038.94                        | 1028.42 <sup>a</sup> | 1.02%                       |
| 5       | 199             | 16                 | 1316.74                        | 1293.24 <sup>b</sup> | 1.78%                       |
| 6       | 50              | 6                  | 555.43                         | 555.43 <sup>a</sup>  | 0.00%                       |
| 7       | 75              | 11                 | 923.52                         | 909.68 <sup>a</sup>  | 1.52%                       |
| 8       | 100             | 9                  | 865.94                         | 865.94 <sup>a</sup>  | 0.00%                       |
| 9       | 150             | 14                 | 1169.83                        | 1162.55 <sup>a</sup> | 0.75%                       |
| 10      | 199             | 18                 | 1412.92                        | 1395.85 <sup>a</sup> | 1.22%                       |
| Average |                 |                    |                                |                      | 0.80%                       |

<sup>a</sup> Rochat and Taillard, 1995'<sup>b</sup> Mester and Braysy, 2007

generated by our local search operators. The fifth column gives the total distance of best-known solution of the test problem. The last column compares our solution with the best known results and gives the difference between our solution and the best known solution. The difference is calculated by the equation (3-12), where DL represents the total travel distance in our solution, while DB is the total travel distance in best-known result.

$$\text{Difference} = (\text{DL} - \text{DB}) / \text{DB} * 100\% \quad (3-12)$$

Table 3- 6

Results for Taillard's Problem (Taillard, 1993)

| Problem | Number of Nodes | Number of Vehicles | Total Distance by Local Search | Best-known solution  | Difference above best known |
|---------|-----------------|--------------------|--------------------------------|----------------------|-----------------------------|
| 100A    | 100             | 11                 | 2077.58                        | 2041.34 <sup>a</sup> | 1.78%                       |
| 100B    | 100             | 11                 | 1948.26                        | 1939.9 <sup>a</sup>  | 0.43%                       |
| 100C    | 100             | 11                 | 1408.73                        | 1406.2 <sup>a</sup>  | 0.18%                       |
| 100D    | 100             | 11                 | 1598.25                        | 1580.46 <sup>b</sup> | 1.13%                       |
| 150A    | 150             | 15                 | 3148.72                        | 3055.23 <sup>a</sup> | 3.06%                       |
| 150B    | 150             | 14                 | 2810.34                        | 2727.2 <sup>b</sup>  | 3.05%                       |
| 150C    | 150             | 15                 | 2403.58                        | 2341.84 <sup>c</sup> | 2.64%                       |
| 150D    | 150             | 14                 | 2662.51                        | 2645.4 <sup>c</sup>  | 0.65%                       |
| Average |                 |                    |                                |                      | 1.61%                       |

<sup>a</sup> Mester and Braysy, 2007.<sup>b</sup> Nagata-Braysy, 2008l.<sup>c</sup> Taillard, 1993.

Based on the results given in Table 3-5 and 3-6, local search algorithms are able to generate high-quality solutions for CVRP. For Christofides' Problem, local search operators can generate solutions that are within 1.9% of best-known results and with an average of 1.21 CPU seconds. For Taillard's Problem, local search operators can produce solutions that are average 1.61% above best-known results with average of 1.48 CPU seconds. With success in CVRP, local search operators are believed to generate a good solution for problem of St. Mary's Food Bank Distribution Center.

### 3.5 Case Study - Problem of St. Mary's Food Bank's Distribution Center (Without Time Windows)

In this section, local search operators are applied to optimize the routing of SMFB's Distribution Center. Thursday is taken as an example to compare our solution with SMFB's current solution. Solutions of other weekdays are displayed in the appendix. On each weekday, 12 trucks in SMFB are available to pick up donated goods within capacity of 40,000lb. Every donor is visited only by one truck and takes an average of fifteen to thirty minutes to load goods. To reduce the transportation cost, the objective of my research is to minimize the number of trucks and total travel time. Figure 1 shows donors that need to be picked up on Thursday (a) and Monday (b). Take Thursday as an example. On Thursday, there are 54 donors need to be visited and total 151,200lb goods need to be picked up. Most of these donors are located in Maricopa County. At SMFB's distribution center, 12 trucks with a capacity of 40,000 lb are available to pick up goods from 5 a.m. to 4 p.m. on Thursday. Our objective is to minimize the number of trucks and total travel time while picking up all goods from 54 donors.

#### 3.5.1 Input Data

In VRP, several practical considerations are usually not considered in theoretical studies. For example, most researches on VRP are based on Euclidean distance yet Euclidean distance cannot reflect real travel cost in practice. In practice, traffic situation, weather, road quality and other factors will have a large impact on travel cost. In this thesis, as more practical features in real world can be reflected on travel time, travel time will be adopted to generate the routing sequence for SMFB distribution center. To calculate the real travel time between donors and distribution center, Google Map is applied to generate the travel time matrix. In the SMFB distribution center, there are 436 donors to be visited in every week, which means thousands distances need to be calculated. With the large amount of address combinations, a VBA macro was coded to automatically connect Google Map and calculate the travel time matrix for each weekday. The detail travel time matrix and VBA code is included in appendix.

In addition, based on the present routing sequence of SMFB, every donor is represented by a sequence number. The SMFB distribution center (Depot) is represented by 0. The demands, donors' addresses and their relative sequence numbers are also included in appendix.

### 3.5.2 Results

This section generates a routing sequence for SMFB distribution center. Clarke and Wright algorithm and local search operators are coded in C++ in Linux system to solve this problem. It was executed on Intel(R) Core™ i7-4770S CPU@3.10GHz. The average computation time is 0.92 CPU seconds for generating a route sequence of each weekday.

Table 3- 7

Routes on Thursday Generated by SMFB

| Route | Travel time(min) | Load(b) | Donors | Sequence                                      |
|-------|------------------|---------|--------|---|
| 1     | 153              | 39200   | 14     | 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-0          |
| 2     | 156              | 39200   | 14     | 0-15-16-17-18-19-20-21-22-23-24-25-26-27-28-0 |
| 3     | 157              | 36400   | 13     | 0-29-30-31-32-33-34-35-36-37-38-39-40-41-0    |
| 4     | 148              | 36400   | 13     | 0-42-43-44-45-46-47-48-49-50-51-52-53-54-0    |

Currently, in SMFB distribution center, all routes are designed by commercial software, which is widely used in routing problem. To test the efficiency of our algorithm, we simulate one week of truck routing for SMFB distribution center and compare the results with SMFB's current solution, which is generated by route design software. Both the total travel time of our solution and SMFB's solution are calculated though Google map. Table 3-7 and Table 3-8 show the solutions of Thursday generated by SMFB and our algorithm separately. In these two tables, the first column gives the route number. The second column and the third column present total travel time and total weights in each route. The fourth column is the total number of donors visited by each route and the final column shows the routing sequence of route. In SMFB's solution, 4 trucks are applied to visit 54 donors and every truck visits 13 to 14 donors

with picking up average 37,800 b donated products. Compared with SMFB’s solution, our solution has reduced 16.29% total travel time while using the same number of trucks.

Table 3- 8

Routes on Thursday Generated by Local Search

| Route | Travel time(min) | Load(b) | Donors | Sequence                                     |
|-------|------------------|---------|--------|--|
| 1     | 109              | 39200   | 14     | 0-1-5-4-8-3-10-13-12-11-9-6-7-20-28-0        |
| 2     | 161              | 39200   | 14     | 0-2-32-33-31-34-30-35-37-36-38-39-40-44-29-0 |
| 3     | 130              | 36400   | 13     | 0-15-17-22-23-21-25-24-14-26-27-19-18-16-0   |
| 4     | 114              | 36400   | 13     | 0-42-43-45-41-46-47-53-51-52-48-54-49-50-0   |

Table 3- 9

Comparison between SMFB’s Original Solution and Local Search’s Solution

|           | SMFB |      | Local Search |        |      |        |
|-----------|------|------|--------------|--------|------|--------|
|           | VEI  | TIME | VEI          |        | TIME |        |
| Monday    | 10   | 1377 | 8            | 20.00% | 921  | 33.12% |
| Tuesday   | 5    | 644  | 4            | 20.00% | 481  | 25.31% |
| Wednesday | 8    | 1000 | 6            | 25.00% | 692  | 30.80% |
| Thursday  | 4    | 614  | 4            | 0.00%  | 514  | 16.29% |
| Friday    | 10   | 1399 | 8            | 20.00% | 880  | 37.10% |
| Average   |      |      |              | 17.00% |      | 28.52% |

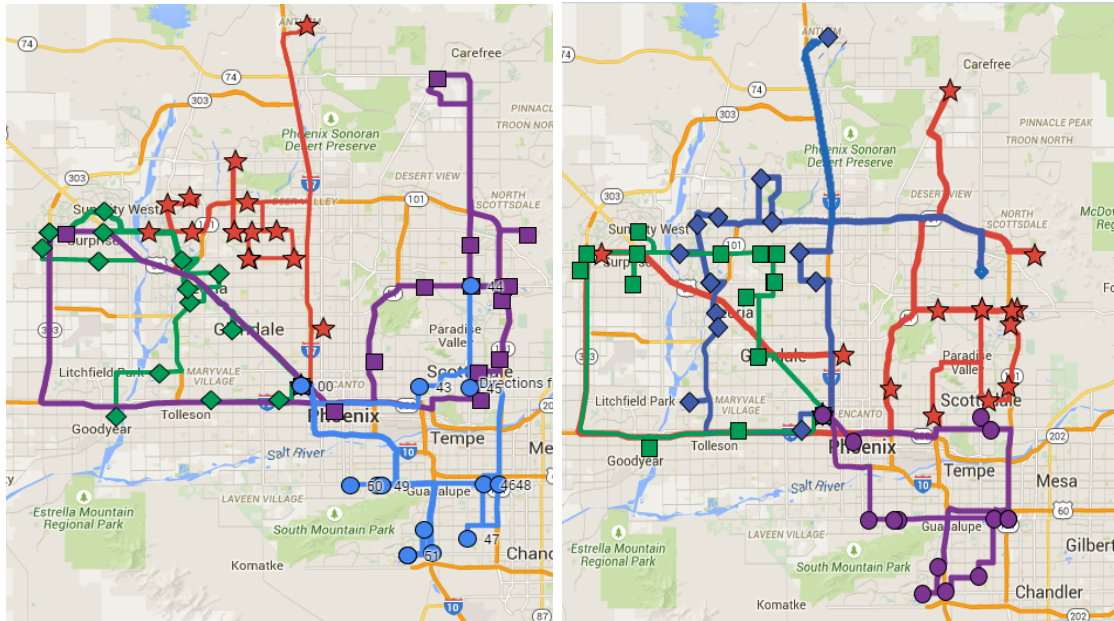
In addition, between SMFB’s solution and our solution, we also compare the total number of vehicles and total travel time for one week in Table 3-9. Obviously, local search is very competitive: it has decreased average 17% in the number of trucks and 28.52% of total travel time. From this result, it’s clear that local search operators are able to produce relatively good solutions in a short amount of time. Figure 3-8 shows the solutions generated by SMFB and local search on the google map.

Figure 3- 8

Routing Map of SMFB (a) and Local Search (b)

(a)

(b)



## CHAPTER 4

### CAPACITATED VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

#### (CVRPTW)

In chapter 3, we have improved current routes of SMFB distribution center by using different local search operators. To further improve the service of distribution center, additional constraints need to be considered, such as time windows for customers to be served, limits on work hours for drivers and lengths of routes. In this chapter, SMFB's problem is extended to Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) and Multiple Ant Colony System (MACS) is applied to solve this problem. Moreover, results will be also compared with SMFB's current solution in this chapter.

#### 4.1. Scenario

Over the past years, the research focus has shifted from fleet-oriented to customer-oriented. As our research is customer-oriented, our goal is not only improving efficiency of results, but also satisfying customer's requirements. In SMFB, several partner agencies are not available at regular business hours, donated products can only be picked up during a specific time period, i.e. within specified time windows. According to this situation, trucks should visit donors within their available time period. In SMFB distribution center, there are 12 trucks to pick up donated products from donors. The weight of donated products ("Demand") and available time windows in each donor varies with donation suppliers. Time windows are set based on neighborhood noise considerations as well as supplier workload constraints.

#### 4.2. Time Window Constraints

In addition to capacity constraints, time window constraints arise in many VRP. If time window constraints are very tight, it complicates the problem and is quite hard to find a

feasible solution. There are two kinds of time window constraints, one is related to donors (transportation requests) and the other one is related to distribution center.

**Time window constraints related to donors (transportation requests):** Since most of donation nodes are business markets, they are only available at a given period. They usually impose a time period  $[E_i, L_i]$  during which they could be served (with  $E_0$  the earliest start time and  $L_0$  the latest return time of each truck to the distribution center). For each donation node  $i$ , donated products can only be picked up between  $E_i$  (the earliest time of time windows) and no later than  $L_i$  (the latest time of time windows). If a truck arrives donation node  $i$  before  $E_i$ , it is permitted to wait until the donation node is available since there is no additional cost for waiting. Fortunately, in SMFB, not all the donation nodes have specific time window requests, only half of them need to be served within a given time interval.

**Time window constraints related to distribution center:** In SMFB distribution center, drivers and trucks are the most important factors that affect the distribution service. Both of drivers and trucks cannot be available for all the day as drivers need time to take a rest and trucks need time to be maintained. In SMFB distribution center, drivers and trucks are available from 5 am to 4 pm, on Monday to Friday, which is not as tight as time windows in donors. Primarily collections occur in the morning with afternoon seeing primarily deliveries by a different fleet.

#### 4.3 Problem Formulation

Before giving formal mathematical formulation of CVRPTW, we should firstly define the following constants and variables.

- $A_{kj}$ : The arrival time for truck  $k$  at supplier  $j$ ;
- $S_j$ : The service time for supplier  $j$ ;



- $w_{kj}$  The waiting time for truck  $k$  at supplier  $j$ ;
- $N(k)$ : Set of nodes visited by truck  $k$
- $T(k)$ : Set of trucks used in solution

In CVRPTW, two objectives are involved, minimizing the number of trucks and minimizing the total travel time (which means minimizing the fuel cost). The number of trucks takes precedence over total travel time. A solution adopts fewer trucks is always preferred even with more travel time. For solutions with the same number of trucks, they will be ranked by the total travel time. We use travel time as a surrogate for fuel consumption.

In addition to CVRP, there are some more features in CVRPTW. In SMFB, as driver's salary is fixed, there is no addition cost for waiting, if a truck arrives at donor before the beginning time of time windows  $E_j$ , it could wait until the donor is available. At the same time, the arrival time is not allowed to be later than the end time of time windows  $L_j$ . Otherwise, as drivers and trucks are available from 5 a.m. to 4 p.m., for each truck  $k$ , the earliest start time  $E_0$  should be later than 5 a.m. and the latest return time  $L_0$  should be earlier than 4 p.m..

Consider truck set  $T(k)$ , there are  $NN$  suppliers that needed to be visited. The arriving time at supplier  $j$  could be defined as the arrival time at previous supplier  $j-1$  plus the service time at this supplier and the travel time from supplier  $j-1$  to  $j$ .

$$A_{kj} = A_{k_{j-1}} + S_{k_{j-1}} + t_{j-1,j} \quad \text{for all } j \in N(k), k \in T(k)$$

If a truck  $k$  arrives supplier  $j$  before the beginning time of time windows  $E_j$ , it is allowed to wait until the donation node is available.

$$A_{kj} = \text{Max} \{E_j, A_{k_{j-1}} + S_{j-1} + t_{j-1,j}\} \quad \text{for all } j \in N(k), k \in T(k)$$

Accordingly, the waiting time

$$w_{kj} = \text{Max} \{0, E_j - A_{k_{j-1}} - S_{j-1} - t_{j-1,j}\} \quad \text{for all } j \in N(k), k \in T(k)$$

As required by supplier, the arrival time at supplier  $j$  must not be greater than  $L_j$

$$A_{kj} \leq L_j \quad \text{for all } j \in N(k), k \in T(k)$$

As required by distribution center, the start time of truck  $k$  should later than 5 a.m. and the return time should be earlier than 4 p.m. (which means the total working time must be less than 660 minutes).

$$A_{k_0} \geq 0, \text{ for all } k \in T(k)$$

As a result, combined with the formulation of CVRP described in Chapter 3, the problem is formulated as follows. Constraint sets (4-1) to (4-10) are formulated for CVRP as chapter 3 describes. Constraint sets (4-11) to (4-16) are added to extend CVRP to CVRPTW.

**Parameters:**

$NN$ : number of donation nodes

$NV$ : number of vehicles

$C$ : capacity of vehicles

$Q_j$ : demand at node  $j$  ( $Q_r=0$ )

$s_i$ : Loading and unloading time at node  $i$  ( $t_0=0$ )

$t_{ij}$ : travel time from node  $i$  to  $j$  ( $t_{ij}=\infty$ )

$D_{ki}$ : departure time of truck  $k$  at node  $i$

$E_i$ : earliest time available at node  $i$

$L_i$ : latest time available at node  $i$

$M, B$ : Big constant number

$u_{ik}$ : Load of vehicle  $k$  after visiting customer  $i$ ;

$A_{kj}$ : The arrival time for truck  $k$  at supplier  $j$ ;

$S_j$ : The service time for supplier  $j$ ;

$w_{kj}$ : The waiting time for truck  $k$  at supplier  $j$ ;

**Set of nodes:** $N(k)$ : Set of nodes visited by truck  $k$  $T(k)$ : Set of trucks used in solution**Variables:**

$$l_{ijk} = \begin{cases} 1 & \text{arc } i, j \text{ is traversed by vehicle } k \\ 0 & \text{Otherwise} \end{cases}$$

$$V_k = \begin{cases} 1 & \text{if vehicle } k \text{ used} \\ 0 & \text{Otherwise} \end{cases}$$

**Formulation:**

$$\text{Min } Z = B \sum_1^{NV} V_k + \sum_{i=0}^{NN} \sum_{j=0}^{NN} \sum_{k=1}^{NV} l_{ijk} * t_{ij}$$

**Subject to:**

$$\sum_{i=0}^{NN} \sum_{k=1}^{NV} l_{ijk} = \sum_{i=0}^{NN} \sum_{k=1}^{NV} l_{jik} \quad \text{for all } j \quad (4-1)$$

$$\sum_{i=0}^{NN} \sum_{k=1}^{NV} l_{ijk} = 1 \quad \text{for all } j \quad (4-2)$$

$$C - \sum_{j=1}^{NN} \{Q_j \sum_{i=0}^{NN} l_{ijk}\} \geq 0 \quad \text{for all } k \quad (4-3)$$

$$\sum_{j=0}^{NN} l_{0jk} \leq 1 \quad \text{for all } k \quad (4-4)$$

$$\sum_{i=0}^{NN} l_{i0k} \leq 1 \quad \text{for all } k \quad (4-5)$$

$$\sum_{i=0}^{NN} \sum_{j=0}^{NN} l_{ijk} \leq M * V_k \quad \text{for all } k \quad (4-6)$$

$$\sum_{j=1}^{NN} l_{0jk} - V_k = 0 \quad \text{for all } k \quad (4-7)$$

$$\sum_{i=1}^{NN} l_{i0k} - V_k = 0 \quad \text{for all } k \quad (4-8)$$

$$u_{ik} - u_{jk} + C * l_{ijk} \leq C - Q_i \quad \text{for all } i, j \in N(k), k \quad (4-9)$$

$$Q_i \leq u_{ik} \leq C \quad \text{for all } i, k \quad (4-10)$$

$$A_{k_j} = A_{k_{j-1}} + S_{j-1} + t_{j-1,j} \quad \text{for all } j \in N(k), k \quad (4-11)$$

$$A_{k_j} = \text{Max} \{E_j, A_{k_{j-1}} + S_{j-1} + t_{j-1,j}\} \quad \text{for all } j \in N(k), k \quad (4-12)$$

$$w_{k_j} = \text{Max} \{0, E_j - A_{k_{j-1}} - S_{j-1} - t_{j-1,j}\} \quad \text{for all } j \in N(k), k \quad (4-13)$$

$$A_{k_j} \leq L_j \quad \text{for all } j \in N(k), k \quad (4-14)$$

$$A_{k_0} \geq 0 \quad \text{for all } k \quad (4-15)$$

$$\sum t_{j-1,j} + \sum_j w_{k_j} + \sum_j S_{k_j} \leq 660 \quad \text{for all } j \in N(k), k \quad (4-16)$$

#### 4.4 Multiple Ant Colony System (MACS)

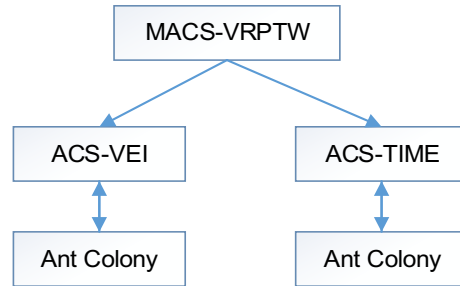
Numerous exact and heuristic methods have been used to solve VRPTW. Kohl et al. (1997) solved an instance with 100 customers with exact algorithm, which is one of the most efficient and succeeded in solving this problem. However, for some instances with tight constraints, it's hard to obtain an optimal feasible solution. Accordingly, heuristic algorithms were proposed to solve this hard problem. Recently, a number of ant colony optimization (ACO) algorithms have been developed to solve combinatorial optimization problems. Particularly, when combined with some local search operators, ACO algorithms are quite efficient to solve traveling salesman problem instances. (Dorigo and Gambardella, 1997a, 1997b). Gambardella et al. (1999) extended ACO to Multiple Ant Colony System (MACS) to solve VRPTW and MACS has been shown to generate competitive results as compared to best existing methods both in terms of solution quality and computation time. Accordingly, this thesis focuses on applying MACS to solve CVRPTW and SMFB's problem.

##### 4.4.1 MACS for Capacitated Vehicle Routing Problem with Time Windows

In real world, ants deposit pheromone (chemical traces) on their paths when they are searching for food. The shorter route will attract more ants by stronger pheromone with time. Similar to the real world, a set of artificial ants coordinate to solve complex problems through low-level communications. In the MACS, we mimic the behavior of real ants. The edge generating more good solutions will be given more weight. As many ants build solutions in MACS, it's more efficient than classic Ant Colony algorithms.

Figure 4- 1

Architecture of MACS



Two objectives are involved in CVRPTW: minimizing the number of vehicles and total travel time. The number of trucks counts more than the total travel time. In MACS, two objectives are optimized by two separate ant colonies, ACS-VEI and ACS-TIME. The architecture of MACS is described as Figure 4-1. In MACS, the objective of ACS-VEI is finding a solution with fewer vehicles while the goal of ACS-TIME is to optimize the total travel time. Two independent artificial ant colonies are used to optimize these two objectives separately. Even though they are independent, they share the same shortest tour  $\varphi^{gb}$ .

As Figure 4-2 shows, in MACS,  $\varphi^{gb}$  is initially generated by nearest neighbor heuristic. Then,  $\varphi^{gb}$  is improved by two artificial ant colonies, ACS-VEI and ACS-TIME. When ACS-VEI works, artificial ants try to find a solution with one vehicle less than the global shortest tour  $\varphi^{gb}$ . In this process, colony ACS-VEI maintains an infeasible solution with  $\nu-1$  vehicles. If a solution with no unvisited node is obtained, colony ACS-VEI has succeed in finding a feasible solution with  $\nu-1$  vehicles and the current global solution  $\varphi^{gb}$  is updated to the feasible solution with  $\nu-1$  vehicles. Then ACS-TIME begin to find a solution with less travel time than global solution  $\varphi^{gb}$ . Every time an improved feasible solution is found by one of the colonies,  $\varphi^{gb}$  will be updated. Two new artificial ant colonies will be generated to work with new  $\varphi^{gb}$ . The process is iterated until satisfying the termination conditions. There are three termination

conditions, the process will terminate if one of these three termination conditions is met. There are three conditions described as follows: (1) A fixed CPU time has elapsed; (2) A fixed number of solutions have been generated; (3) No improvement has been achieved during a given number of iteration. Figure 4-3 shows the flowchart of ACS-VEI and ACS-TIME and table 4-1 shows the algorithm of MACS.

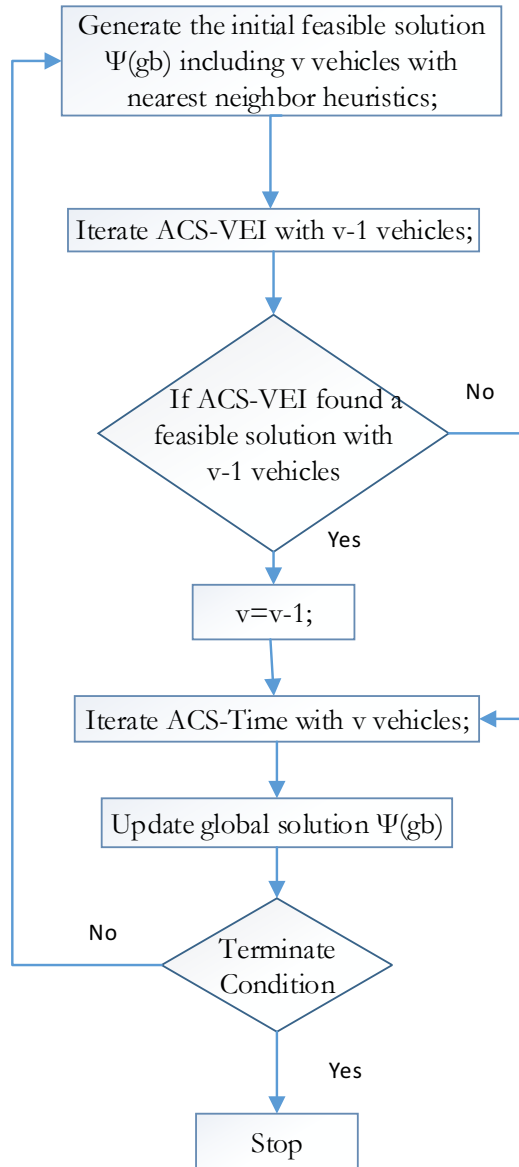
Table 4- 1

MACS Algorithm

|                         |  |
|-------------------------|--|
| Step 0: Initialization. |  |
|                         | Generate the initial feasible tour $\varphi^{gb}$ including $v$ vehicles with nearest neighbor heuristics; |
| Step 1: Improve         |  |
|                         | Repeat   |
|                         | Iterate ACS-VEI with $v-1$ vehicles: ACS-VEI ( $v-1$ );  |
|                         | If ACS-VEI( $v-1$ ) found a feasible solution with $v-1$ vehicles  |
|                         | Decrease the number of vehicles: $v = v-1$   |
|                         | Iterate ACS-Time with $v$ vehicles: ACS-Time ( $v$ );  |
|                         | Update the global solution: $\varphi^{gb}$   |
|                         | Until termination criterion is met   |

Figure 4- 2

Flowchart of MACS



Terminate Condition:

- (1) A fixed CPU time has elapsed;
- (2) A fixed number of solutions has been generated;
- (3) No improvement has been achieved during a given number of iteration.

#### 4.4.2 ACS-VEI

In MACS, two objectives are optimized by ACS-VEI and ACS-TIME. The goal of ACS-VEI is minimizing the number of trucks used in collection. Assume there are  $v$  trucks in the current  $\varphi^{gb}$ , we start ACS-VEI with  $v-1$  trucks to find a feasible solution. Firstly, nearest neighbor

heuristics is applied to generate an initial tour  $\varphi^{\text{ACS-VEI}}$ , which is probably infeasible as there are  $m^{\text{ACS-VEI}}$  supplier nodes are not visited by this tour. Then colony ACS-VEI begins to search a better solution, in which fewer nodes are unvisited. If an ant generates a solution  $\varphi^k$  with less unvisited nodes, the solution  $\varphi^{\text{ACS-VEI}}$  is updated to solution  $\varphi^k$ . If a solution with no unvisited node is obtained, colony ACS-VEI has succeeded in finding a feasible solution  $\varphi^{\text{ACS-VEI}}$  with  $\nu-1$  vehicles. Then the global solution  $\varphi^{\text{gb}}$  is updated to solution  $\varphi^{\text{ACS-VEI}}$  and the count of vehicles in  $\varphi^{\text{gb}}$  is updated to  $\nu-1$ . Then the next iteration will start with  $\nu-2$ , until the termination condition is satisfied. The detail algorithm of ACS-VEI is described in Table 4-2.

Table 4- 2

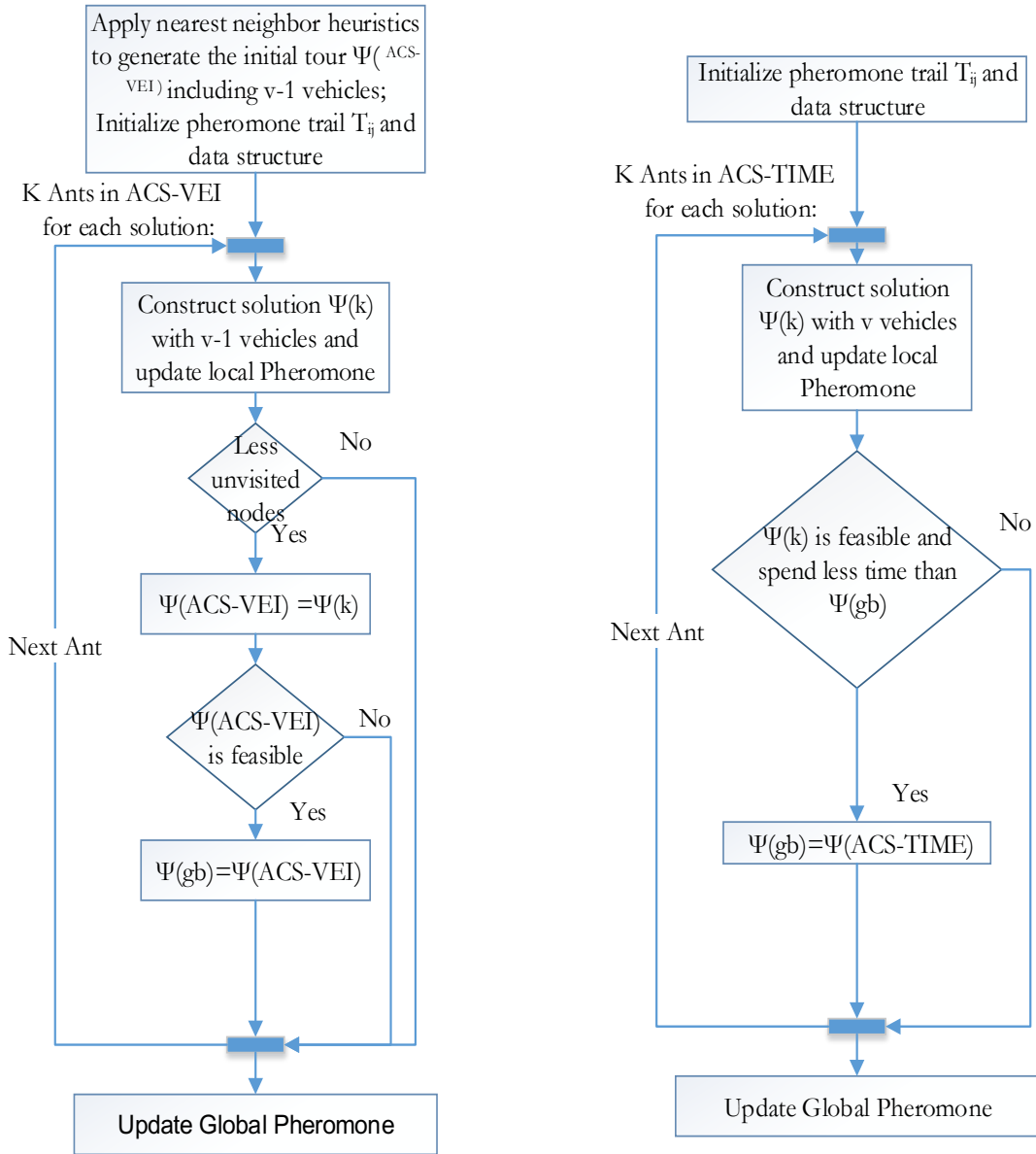
The Algorithm of ACS-VEI

|                         |   |
|-------------------------|---|
| Step 0: Initialization. |   |
|                         | Apply nearest neighbor heuristics to generate the initial tour $\varphi^{\text{ACS-VEI}}$ including $\nu-1$ vehicles and $m^{\text{gb}}$ unvisited nodes; ( $\nu$ is the number of vehicles used in current $\varphi^{\text{gb}}$ in MACS-VRPTW);<br>Initialize pheromone trail $\tau_{ij}$ and data structure;   |
| Step 1: Search          |   |
|                         | Repeat<br>For each artificial ant k:<br>Generate a new solution $\varphi^k$ with $\nu-1$ vehicles;<br>If $m^k$ (the number of unvisited nodes in solution $\varphi^k$ ) $<$ $m^{\text{ACS-VEI}}$ (the number of visited customers in solution $\varphi^{\text{ACS-VEI}}$ ),<br>Then:<br>1. $\varphi^{\text{ACS-VEI}} = \varphi^k$ ;<br>If $\varphi^{\text{ACS-VEI}}$ is feasible, then $\varphi^{\text{gb}} = \varphi^{\text{ACS-VEI}}$ ;<br>2. Update pheromone trail globally with $\tau_{ij} = (1-\rho) * \tau_{ij} + \rho / J_{\varphi}^{\text{gb}}$ ;<br>Stop until a termination constraint is satisfied. |



Figure 4- 3

Flowchart of ACS-VEI and ACS-TIME



#### 4.4.3 ACS-TIME

Different from ACS-VEI, the goal of ACS-TIME is minimizing the total travel time. In ACS-TIME,  $m$  artificial ants are constructing solutions separately. Each ant builds a solution  $\varphi^k$  and the solution is improved by local search operators we introduced in chapter 3. Once a feasible solution with less total travel time is found, the global optimal will be replaced by this

update solution  $\varphi^{gb}$ . The detail procedure of ACS-TIME is described as follows:

Table 4- 3

The Algorithm of ACS-TIME

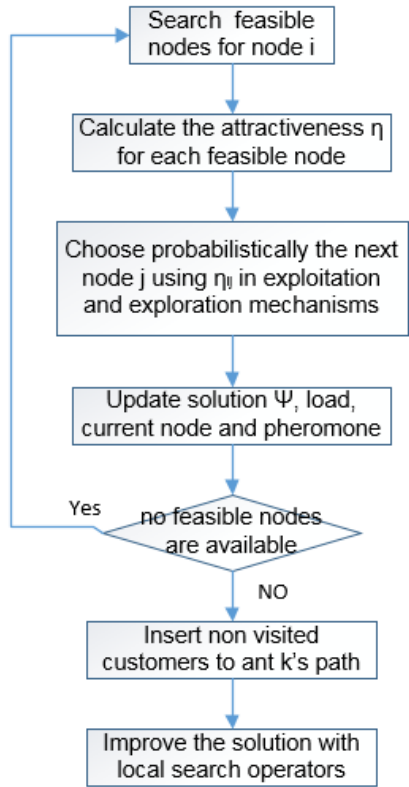
|                         |  |
|-------------------------|--|
| Step 0: Initialization. |  |
|                         | Initialize pheromone trail $\tau_{ij}$ and data structure using $\nu$ vehicles (same number of vehicles as current global solution $\varphi^{gb}$ ). |
| Step 1: Improve         |  |
|                         | Repeat   |
|                         | For each artificial ant $k$ :  |
|                         | Generate a new solution $\varphi^k$ ;  |
|                         | If $\varphi^k$ is feasible and total travel time of $\varphi^k$ is less than total travel time of $\varphi^{gb}$ ,                                   |
|                         | 1. $\varphi^{gb} = \varphi^k$ ;  |
|                         | 2. Update pheromone trail globally with $\tau_{ij} = (1-\rho) * \tau_{ij} + \rho / J_{\varphi}^{gb}$   |
|                         | Until a termination constraint is satisfied.   |

#### 4.4.4 Solution Constructive Procedure

Both ACS-TIME and ACS-VEI use the same solution constructive procedure to build solution for each artificial ant. As Figure 4-4 shows, each ant will start from the depot and build a solution by adding nodes iteratively. When selecting the next node to visit, a feasible set  $N_i^k$  is firstly searched for current node  $i$  with following constraints: (1) Not being visited; (2) Satisfy the time window constraints (3) Satisfy the vehicle capacity.

Figure 4- 4

Flowchart of Solution Constructive Procedure



Within the feasible set  $N_i^k$ , the attractiveness  $\eta_{ij}$  is calculated for each node. Based on the attractiveness  $\eta_{ij}$ , the next node  $j$  is chosen probabilistically by using exploitation and exploration, where the component that maximizes the pheromone trail is chosen in exploitation and each component will be associated with a probability ( $p_{ij}$ ) of being chosen in exploration. Every time ant  $k$  moves from one node to another, the current parameters, solution  $\varphi^k$  and pheromone  $\tau_{ij}$  are updated to start the next iteration. This iteration continues until no feasible nodes are available. At the end of solution constructive phase, as some nodes might haven't been visited, the solution is extended with further insertion, in which all non-visited nodes are considered by decreasing delivery quantities. In addition, In ACS-TIME, local search operators, which include Two-opt, Or-opt and cross exchange, are applied to improve the solution. The detailed procedure is described in Table 4-4.

Table 4- 4

## Solution Constructive Procedure

|  |   |
|--|---|
| Step 0: Initialization.  |   |
|  | <ol style="list-style-type: none"> <li>1. Artificial Ant <math>k</math> start tour from depot <math>0</math>;</li> <li>2. Current time <math>t_k=0</math> and current Load <math>l_k=0</math>;</li> </ol>   |
| Step 1: Build turns for artificial ants:                                   |   |
|  | <p>Repeat: For each artificial ant <math>k</math>:</p> <ol style="list-style-type: none"> <li>1. Search the feasible set <math>N_i^k</math> for node <math>i</math>, all nodes in feasible set <math>N_i^k</math> should satisfy the time window constraints and capacity constraints.</li> <li>2. Compute the attractiveness <math>\eta_{ij}</math> for each node in set <math>N_i^k</math>: <ol style="list-style-type: none"> <li>(1) Delivery time for node <math>j</math>: <math>D_j = \text{Max}(\text{Current time } t_k + t_{ij}, E_j)</math> ;</li> <li>(2) Delta time between <math>i</math> and <math>j</math>: <math>\Delta_{ij} = D_j - t_i</math>;</li> <li>(3) Distance between <math>i</math> and <math>j</math>: <math>\text{Dis}_{ij} = \Delta_{ij} * (L_j - t_i)</math> ;</li> <li>(4) Distance between <math>i</math> and <math>j</math>: <math>\text{Dis}_{ij} = \text{Max}(1, \text{Dis}_{ij} - in_j)</math></li> <li>(5) Attractiveness <math>\eta_{ij} = 1 / \text{Dis}_{ij}</math>;</li> </ol> </li> <li>3. Choose the next node <math>j</math> for node <math>i</math> probabilistically considering both exploitation and exploration. <p>In Exploitation, there are probability <math>q_0</math> to choose a node with the highest <math>\tau_{ij} * c_{ij}</math>;</p> <p>In Exploration, there are probability <math>(1-q_0)</math> to choose node with probability <math>p_{ij}</math>;</p> </li> <li>4. Update parameters: <p>Tour <math>k</math>: <math>\varphi^k = \varphi^k + \text{node } j</math>;</p> <p>If <math>j</math> is depot, then Current time <math>t_j = 0</math> and Current load: <math>l_k = 0</math>;</p> <p>Else Current time <math>t_j = D_j</math> and Current load <math>k</math>: <math>l_k = l_k + Q_j</math>;</p> <p>Update local pheromone <math>\tau_{ij} = (1-\rho) * \tau_{ij} + \rho * \tau_0</math>; Current Node <math>i=j</math>;</p> <p>Until no feasible nodes are left: <math>N_i^k = \{ \}</math>;</p> </li> </ol> |
| Step 2: Extend tour $\varphi^k$ by inserting unvisited nodes tentatively   |   |
| Step 3: For ACS-TIME, improve the feasible tour with local search operator |   |

In solution building process, selecting the next node  $j$  is the key procedure. Choosing the next node is based on the exploration and exploitation mechanisms. The following part gives the detail discussion on exploration and exploitation mechanisms in MACS and its key parameters. Exploration and exploitation are two ways to find next node. In exploitation, the component that maximizes the pheromone trail is chosen. Meanwhile, in exploration, each component will be associated with a probability ( $p_{ij}$ ) of being chosen to construct a solution. MACS adopts a pseudorandom proportional mechanism in choosing next node. In the exploitation, there are probability  $q_0$  to choose a node with highest  $\tau_{ij} * c_{ij}$ . At the same time, in the exploration, there is also a probability of  $(1 - q_0)$  to choose a node  $j$  with probability of  $p_{ij}$  of choosing node  $j$  as the next node to be visited, where  $q_0$  is the relative importance of exploitation versus exploration. As the parameter  $q_0$  closes to zero, more probability to adopt the probability rule  $p_{ij}$  the search is promoted to exploration of new solutions, whereas larger value narrows the search towards the best solution.

#### 4.4.4.1 Exploitation

In the exploitation, there are probability  $q_0$  to choose a node with the highest  $\tau_{ij} * c_{ij}$ . Closeness  $c_{ij}$  and the pheromone trail  $\tau_{ij}$  are two most important measures in choosing next node. For a given problem, the closeness  $c_{ij}$ , defined as the inverse of distance, is fixed while the pheromone trail  $\tau_{ij}$  is changing with time and used for both exploitation and exploration. Accordingly, the pheromone trail  $\tau_{ij}$  is the most important component of MACS. In real world, ants will follow the pheromone to find the food, the stronger the pheromone is, the more chance ants will follow this route. Same as in the real world, pheromone trails  $\tau_{ij}$  is a measure of how desirable it is to choose this arc in a solution. In MACS, pheromone trail  $\tau_{ij}$  is updated through two ways, locally in each solution construction and globally in global solution updating. During the tour construction, every time an ant uses an edge, the

pheromone trail  $\tau_{ij}$  of this edge will decrease and it's less chance to choose this edge next time. The following is how the pheromone trail of edge  $(i, j)$   $\tau_{ij}$  decreased after an ant using the edge  $(i, j)$ .

$$\tau_{ij} = (1 - \rho) * \tau_{ij} + \rho * \tau_0 \text{ for all } j \quad (4-18)$$

Where:

$\tau_0$ : The initial pheromone trail value and  $\tau_0 = 1 / (NN * J_{\varphi}^h)$

$J_{\varphi}^h$ : The length of initial solution generated by the nearest neighbor heuristic (Flood, 1956)

$NN$ : The number of nodes

$\rho$ : A parameter that between 1 and 0;

$\varphi$ : The shortest tour generated by ants as so far;

In addition, at the end of tour construction, the best solution will be used to update the pheromone globally. With global updating, the neighborhood that is used to compute the best solution will be intensified. In the global updating, only the best solution is applied to modify the pheromone trail. As the “best solution” generated in the last iteration will be memorized in the pheromone trail matrix, and ants will use this pheromone trail matrix to generate new tour in the neighborhood of this ‘best solution’, this updating strategy is more efficiency than updating with all solutions. The global updating is described as follows:

$$\tau_{ij} = (1 - \rho) * \tau_{ij} + \rho / J_{\varphi}^{gb} \quad (4-19)$$

Where:

$\varphi^{gb}$ : The shortest tour generated by ants as so far;

$J_{\varphi}^{gb}$ : The length of  $\varphi$ ;

#### 4.4.4.2 Exploration

In exploration, the probability  $p_{ij}$  is calculated based on two values, the pheromone level  $\tau_{ij}$

which has been discussed in exploitation and a heuristic value  $\eta_{ij}$  that led closer nodes more attractive. The probability  $p_{ij}$  is formulated as follows:

$$p_{ij} = \begin{cases} \frac{\tau_{ij} \cdot [\eta_{ij}]^\beta}{\sum_{l \in N_i^k} \tau_{il} \cdot [\eta_{il}]^\beta} & \text{if } j \in N_i^k \\ 0 & \text{otherwise} \end{cases} \quad (4-20)$$

where  $\beta$  is a parameter that weights the relative importance of the heuristic value  $\eta_{ij}$  versus  $\tau_{ij}$ .  $N_i^k$  is the set of feasible neighbors of node  $i$  for route  $k$ . As the measure of attractiveness, the heuristic value  $\eta_{ij}$  considers distance to customer, urgency to pick up and frequency to be selected. It is calculated as follows:

$$\eta_{ij} = \text{Max}\{1, (\text{Max}\{A_{ij}^k, E_j\} - D_i)(L_i - D_i) - \text{in}_j\}^{-1} \quad (4-21)$$

where  $D_i$  is the departure time from  $i$  and  $\text{in}_j$  is the number of times that has not been chosen as next node. In the equation of  $\eta_{ij}$ , the first factor takes the distance to the customer into consideration, the second considers the urgency of pickup and the last one yields nodes that are seldom visited more attractive. Then, pheromone trail  $\tau_{ij}$  is updated with the best solution found in the beginning trail.

#### 4.5 Numerical Results

This section analyzes computational results and efficiency of MACS. MACS has been tested on a group of 18 benchmark problems (Solomon, 1987) and been compared with best known results. In these 18 problems, it considers the number of customers, percentage of customers with time window, tightness of time windows and geographical data. The names of these problems have the following meaning: Problem sets R generate nodes with random location, problem set C generates clustered nodes whose time windows were generated with a known solution. Problem set RC is a combination of clustered customers and random location. In each kind of geographical type, we set different number of customers, 25, 50, 100. Table 4-5

describes the detail features of benchmark problems we use. The first and the second column describe the name and the geographical data of test problem set. The third and the fourth column are the percentage of customers with time windows and capacity of vehicles.

Table 4- 5

Features of Benchmark Problems

| Problem | Geographical data           | # of customers | Vehicle Capacity |
|---------|-----------------------------|----------------|------------------|
| R1      | Randomly generated          | 25,50,100      | 200              |
| R2      | Randomly generated          | 25,50,100      | 200              |
| C1      | Clustered                   | 25,50,100      | 200              |
| C2      | Clustered                   | 25,50,100      | 200              |
| RC1     | Mix of Random and clustered | 25,50,100      | 200              |
| RC2     | Mix of Random and clustered | 25,50,100      | 200              |

We compare performance of MAR-VRPTW with best known result in the number of vehicles and total travel distance in Table 4-6. MACS was executed on Intel(R) Core™ i7-4770S CPU@3.10GHz. The average computation time is 30 CPU seconds. For most problems, MACS algorithm performs very well and quite close to the best known result. Especially for R problem sets in which nodes are randomly generated, MACS generates better solutions than best known result in the number of vehicles. As most of donation suppliers of SMFB are located all over the Phoenix randomly, problem R should be the most similar with real problem of SMFB distribution center. Accordingly, the following section will solve this sequence problem for SMFB distribution center with MACS algorithm.



Table 4- 6

Results for Benchmark Problems (Solomon, 1987)

| Problem   | Best Known<br>(Solomon, 2005) |        | MARC-VRPTW |        |         |        |
|-----------|-------------------------------|--------|------------|--------|---------|--------|
|           | VEI                           | DIST   | VEI        |        | DIST    |        |
| R101-25   | 8                             | 617.1  | 8          | 0.00%  | 618.33  | 0.20%  |
| R101-50   | 12                            | 1044   | 11         | -8.33% | 1100.72 | 5.43%  |
| R101-100  | 20                            | 1637.7 | 19         | -5.00% | 1951.97 | 19.19% |
| C101-25   | 3                             | 191.3  | 3          | 0.00%  | 191.81  | 0.27%  |
| C101-50   | 5                             | 362.4  | 5          | 0.00%  | 363.25  | 0.23%  |
| C101-100  | 10                            | 827.3  | 10         | 0.00%  | 828.94  | 0.20%  |
| RC101-25  | 4                             | 461.1  | 4          | 0.00%  | 462.16  | 0.23%  |
| RC101-50  | 8                             | 944    | 8          | 0.00%  | 946.25  | 0.24%  |
| RC101-100 | 15                            | 1619.8 | 15         | 0.00%  | 1647.57 | 1.71%  |

#### 4.6 Case Study-Problem of St. Mary's Food Bank Distribution Center (With Time Windows)

In SMFB distribution center, collection activities begin at 5 a.m. and take about 9 to 11 hours to finish picking up donated products. Moreover, loading and unloading goods will need cooperators from suppliers, some of which can only be available within a specific time period (Time Windows). If trucks arrive before the earliest time of time windows, trucks must wait until the supplier is available. Otherwise, if trucks arrive at the supplier store after the upper bound of time windows, the supplier cannot be served and nothing could be picked up. Usually, these time windows are two hours' time periods between 5 a.m. and 3 p.m. In SMFB, half of donors have time windows and are available between 6 a.m. to 8 a.m. and 11 a.m. to 1 p.m. With above descriptions, this pickup problem can be defined as a Capacitated Vehicle Routing Problem with Time Windows (CVRPTW).

#### 4.6.1 Data

Compared with CVRP, time window constraints are added to the problem in CVRPTW. Except data of time windows, all data is the same with chapter 3. In SMFB, half the donation suppliers have time windows. Most of these time windows are time periods between 6 a.m. to 8 a.m. and 11 a.m. to 1 p.m.. Otherwise, drivers and trucks are available from 5 a.m. to 4 p.m. To normalize the data of time windows, we set 5 a.m. as start time 0 and other time as the length of time period between start time and that time in minute unit. For example, 6 a.m. can be represented as 60, which is  $(6-5)*60$ , and 8 a.m. is represented as 180.

#### 4.6.2 Results

MACS are coded in C++ in Linux system to generate routing sequence for SMFB distribution center. It was executed on Intel(R) Core™ i7-4770S CPU@3.10GHz. The average computation time is 10 CPU seconds for generating a route sequence of each weekday. To test the efficiency of our algorithm, we compare our solution with SMFB's current solution. Both the total travel time of our solution and SMFB original solution are calculated through Google Map. Currently, SMFB use commercial routing design software to design the route sequence for every weekday. Same as Chapter 3, we also use Thursday as an example to show our solution, results of other work days are included in appendix. Table 4-7 and Table 4-8 show the solution of Thursday generated by SMFB and by our algorithm separately. In this two table, the first column gives the route number. The second column and the third column present total travel time and total weights in each route. The fourth column is the total number of donors visited by each route and the final column shows the routing sequence of current route. In SMFB's solution, 4 trucks are applied to visit 54 donors and every truck visits 13 to 14 donors with picking up average 37,800 b donated products. Compared with SMFB's solution, MACS has reduced 7.17% total travel time. Figure 4-5 separately shows routes that are generated by SMFB and by MACS on Google Map.

In addition, we also compare the total number of vehicles and total travel time for a week in Table 4-9. Obviously, MACS generates relatively competitive results: it has decreased average 10% in number of trucks and 9.87% in total travel time. More importantly, time windows

requirements are satisfied with the solution. From this result, it's clear that MACS is able to produce relatively good solutions in a short amount of time.

Table 4- 7

Routes on Thursday Generated by SMFB

| Route | Travel time(min) | Load(lb.) | Donors | Sequence                                      |
|-------|------------------|-----------|--------|---|
| 1     | 153              | 39200     | 14     | 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-0          |
| 2     | 156              | 39200     | 14     | 0-15-16-17-18-19-20-21-22-23-24-25-26-27-28-0 |
| 3     | 157              | 36400     | 13     | 0-29-30-31-32-33-34-35-36-37-38-39-40-41-0    |
| 4     | 148              | 36400     | 13     | 0-42-43-44-45-46-47-48-49-50-51-52-53-54-0    |

Table 4- 8

Routes on Thursday Generated by MACS

| Route | Travel time(min) | Load(b) | Donors | Sequence                                       |
|-------|------------------|---------|--------|--|
| 1     | 163              | 39200   | 14     | 0 - 24- 23-22-21-25-26-9-3-2-10-13-12-11-14 -0 |
| 2     | 165              | 39200   | 14     | 0-32-33-31-34-30-35-37-36-46-48-52-51-53-0     |
| 3     | 148              | 36400   | 13     | 0-38-39-40-44-41-45-43-29-1-42-50-49-54-0      |
| 4     | 94               | 36400   | 13     | 0-15-16-17-18-19-27-20-6-7-8-4-5-28-0          |

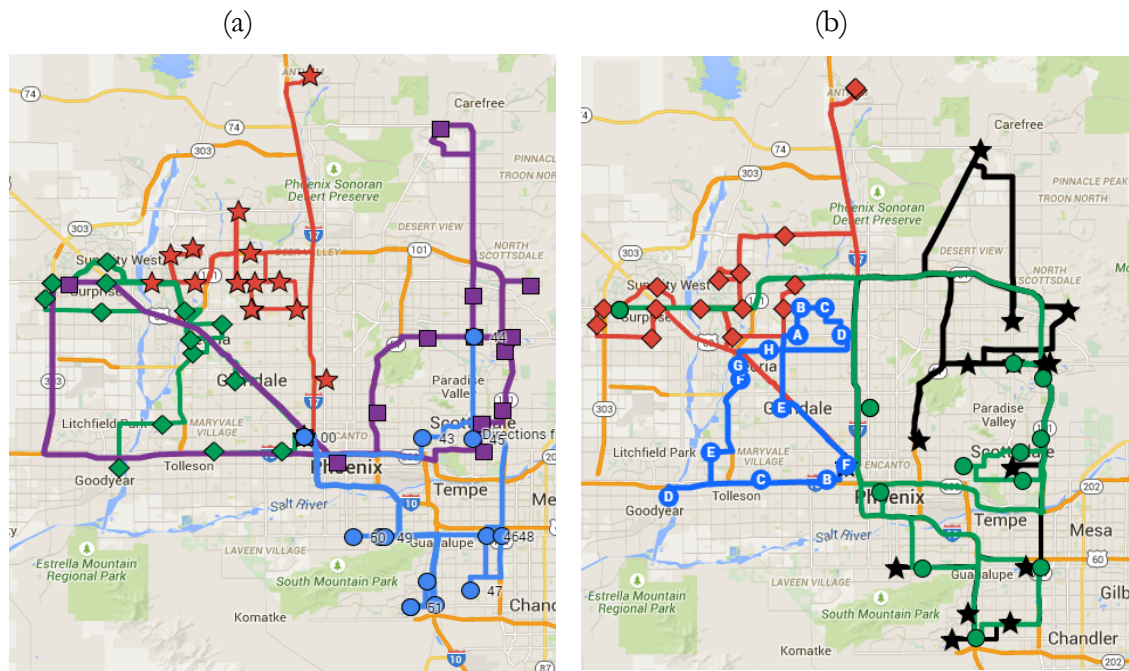
Table 4- 9

Comparison of SMFB's Original Solution and MACS's Solution

|           | SMFB |      | MACS |        |      |        |
|-----------|------|------|------|--------|------|--------|
|           | VEI  | TIME | VEI  | DIF    | VEI  | DIF    |
| Monday    | 10   | 1377 | 9    | 10.00% | 1237 | 10.17% |
| Tuesday   | 5    | 644  | 4    | 20.00% | 555  | 13.82% |
| Wednesday | 8    | 1000 | 6    | 25.00% | 931  | 6.90%  |
| Thursday  | 4    | 614  | 4    | 0.00%  | 570  | 7.17%  |
| Friday    | 10   | 1399 | 9    | 10.00% | 1241 | 11.29% |
| Average   |      |      |      | 10.00% |      | 9.87%  |

Figure 4- 5

Routing of SMFB (a) and Routing of MACS (b)



## CHAPTER 5

### CONCLUSIONS AND FUTURE RESEARCH

In this thesis, we address the capacitated vehicle routing problem and extend it to a capacitated vehicle routing problem with time windows. We present a practical vehicle routing problem faced by SMFB. The problem is modeled as a capacitated vehicle routing problem to improve the distribution efficiency and is extended to capacitated vehicle routing problem with time windows to increase customer service quality separately. As VRP is known to be NP-hard, metaheuristics are adopted to solve these problems. Moreover, the travel time matrix generated from Google map is used as an input to solve the problem. The corresponding solution is highly pragmatic and realistic.

Research on modeling and algorithms of the VRP and its variants are reviewed comprehensively in chapter 2. Motivated by the truck routing problem of SMFB distribution center, the CVRP is studied in detail in chapter 3. The routing problem of SMFB is modeled as CVRP that considers two critical objectives: minimizing the number of trucks and total travel time. To solve this problem, Clarke and Wright algorithm is applied to generate the initial solution and local search operators are used to improve the results. Using different local search operators, we are able to improve the current routes of SMFB distribution center, with using average 17% less trucks and 28.52% less travel time. In addition, as SMFB requests, time window constraints are added to the problem and we model the problem as CVRPTW. We deal with this problem though MACS and generate high quality solutions, in which 10% less trucks and 9.87% travel time while satisfying all time windows. These results assume randomly generated time windows as actual values were not available for the data provided. Moreover, as a key input, travel time matrix is generated from Google Map, which considers most of unexpected traffic situations and makes solutions highly realistic and applicable in life. We also develop a macro to collect travel time automatically from Google Map.

This thesis addressed truck routing problem based on assumption of known customer demands. The stochastic demand situation was not considered. In practice, demands are

usually unknown until the trucks arrive customers. Accordingly, future research can consider a problem with stochastic pickup demand, a forecasting model can be introduced to forecast the pickup demand with historical data. Based on the forecasting data, routes could be designed in advance to improve distribution efficiency. To be more convenient for distribution center and customers, a real distribution management system can be developed based on our algorithms. This system can connect distribution center and donors together, route sequences can be optimized based on the dynamic data from donors and be sent to donors through email in time.

## REFERENCES

- [1] Altinkemer K., Gavish B. (1991). Parallel savings based heuristics for the delivery problem. *Operation Research* 39(3), 456-469.
- [2] Baldacci R., Christofide N. and Mingozzi A. (2008). An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Math. Programming Ser. A* 115, 351-385
- [3] Baldacci R., Hadjiconstantinou E., Mingozzi A. (2004). An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation. *Operation Research* 52(5), 723-738.
- [4] Baldacci R., Dell'Amico M. (2007). The capacitated m-ring-star problem. *Operations Research*, 55(6), 1147-1162
- [5] Baldacci R., Christofides N, Mingozzi A. (2008). An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming* 115(2), 351-385.
- [6] Balinski M.L., Quandt R.E. (1964). On an integer program for a delivery problem. *Operations Research* 12(2), 300-304.
- [7] Bruno-Laurent G., Potvin J. Y. and Rousseau J. M., (1994). A parallel implementation of the Tabu search heuristic for vehicle routing problems with time window constraints, *Comput. Oper. Res.*, vol. 21, no.9, pp.1025-1033,
- [8] Chiang W.C. and Russell R.A., (1997). A reactive Tabu search metaheuristic for the vehicle routing problem with time windows." *INFORMS J. Comput.*, vol. 9, pp. 417-430.

- [9] Christofides N., Eilon S. (1969). An Algorithm for the Vehicle-Dispatching Problem. *OR*, 20(3), 309-318
- [10] Christofides N., Mingozzi A. and Toth P. (1981). Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. *Mathematic Programming* 20(1), 255-282.
- [11] Clarke G. and Wright J.W. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*. 12(4), 568-581.
- [12] Cullen FH, Jarvis J. and Ratliff H. D. (1981). Set partitioning based heuristics for interactive routing. *Networks*, 11(2), 125-143.
- [13] Cordeau J., Gendreau M., Laporte G. (1997). A Tabu Search Heuristic for Periodic and Multi-Depot Vehicle Routing Problems. *Networks*, 30:105-119.
- [14] Dantzig GB, Ramser JH. (1959). The truck dispatching problem. *Management Science*, 6:80-91.
- [15] Derigs U., R. Kaiser. (2007). Applying the attribute based hill climber heuristic to the vehicle routing problem. *Operational Research* 177(2), 719-732.
- [16] Eilon S., Watson-Gandy C.D.T., Heilbron A. (1971). A vehicle fleet costs more. *Physical Distribution* 1(3), 126-132.
- [17] Fox K.R., Gavish B., Graves S.C. (1980). Technical Note-An n-Constraint Formulation of the (Time-Dependent) Traveling Salesman Problem. *Operations Research* 28(3), 1018-1021.
- [18] Fukasawa R., Longo H. (2006). Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming* 106(3), 497-511.



- [19] Francis P., Smilowitz K., and Tzur M. (2007). Flexibility and complexity in periodic distribution problems. *Naval Research Logistics*, 54:136–150.
- [20] Gillett B.E., Miller. L.R. (1974). A Heuristic Algorithm for the Vehicle-Dispatch Problem. *Operations Research* 22(2), 340-349.
- [21] Groer C., Golden B., Wasil E. (2008). A parallel algorithm for the vehicle routing problem. *Computing* 23(2) 315-330.
- [22] Gambardella LM, Taillard T. and Agazzi G. (1999). MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows.
- [23] Gendreau M., Hertz A., Laporte G. (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science* 40(10), 1276-1290.
- [24] Hunger in America (2014). <http://www.feedingamerica.org/hunger-in-america/our-research/hunger-in-america/>
- [25] Hadjiconstantinou E., Christofides N., Mingozzi A. (1995). A new exact algorithm for the vehicle routing problem based on q-paths and k-shortest paths relaxation. *Annals of Operations Research* 61(1), 21-43.
- [26] Laporte G., Nobert Y., Pelletier P. (1983). Hamiltonian location problem. *European Journal of Operational Research* 12(1), 82-89.
- [27] Lenstra J.K., Kan R. (1981). Complexity of Vehicle Routing and Scheduling Problem. *Networks*, 11(2), 221-227

- [28] Levy L. and Bodin L. (1988). Scheduling the postal carriers for the United States postal service: An application of arc partitioning and routing. *Vehicle Routing: Methods and Studies*. 359–394.
- [29] Mester D. and O. Braysy. (2007). Active-guided evolution strategies for large-scale vehicle routing problems. *Computer and Operations Research*, 34, 2964-2975.
- [30] Marshall L. Fisher and Jaikumar R. (1981). A generalized assignment heuristic for vehicle routing. *Networks*, 11(2), 109-124.
- [31] Marshall L. Fisher. (1994). Optimal solution of vehicle routing problems using minimum k-trees. *Operation Research* 42(4), 626-642.
- [32] Mladenovic N, Hansen P. (1997). Variable neighborhood search. *Computer & Operations Research* 24(11), 1097-1100.
- [33] Nagata Y. and O. (2008). Braysy. Efficient local search limitation strategies for vehicle routing problems. *EvoCOP*, 4972, 48-60.
- [34] Paessens H. (1988). The savings algorithm for the vehicle routing problem. *Operational Research* 34(3), 336-344.
- [35] Russell R., Igo W. (1979). An assignment routing problem. *Networks*, 9, 1-17.
- [36] Rochat Y. and Taillard E. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1, 147-167.
- [37] Rober W. Lien, Sayed M. R Iravani and Karen R. Smilowitz. (2013). Sequential Resource Allocation for Nonprofit Operations. *Operation Research* Vol. 62, No. 2, pp. 301-317

- [38] Sniezek J. and Bodin L. (2006). Using mixed integer programming for solving the capacitated arc routing problem with vehicle/site dependencies with an application to the routing of residential sanitation vehicles. *Annals of Operations Research*, 144:33–58.
- [39] Solomon M.M. (1995). Algorithms for the vehicle routing problem with time windows. *Transportation Science*, 29(2), pp. 156-166
- [40] Solomon M.M., (2005). <http://web.cba.neu.edu/~msolomon/problems.htm>
- [41] Solomon M.M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operation Research* 15(2), 254-265.
- [42] Solomon M.M, Desrosiers J. (1988). Time window constrained routing and scheduling problem. *Trans. Sci.* 22, 1-13.
- [43] Taillard E. (1993). Parallel iterative search methods for vehicle routing problems. *Networks* 23 661-673
- [44] Taillard. E. (1993). A computational modification to the savings method of vehicle scheduling. *Operational Research Quarterly*, 21, 281-293.
- [45] Wren A. (1971). Bus scheduling: an interactive computer method. *Transport Planning and Operation*. 1(2), 115-122
- [46] Wark P., Holt J. (1994). A Repeated Matching Heuristic for the vehicle routing problem. *Operation Research* 45(10), 1156-1167.

APPENDIX A  
RTR ALGORITHM

1. One-Point Move with Record-to-Record Travel

|  |  |
|--|--|
| One-point Move with record-to-record travel        |  |
| For each node $i$ in the current solution (I loop) |  |
|  | For each edge $j$ in the current solution whose one end is in node $i$ 's neighbor list (J loop)                                   |
|  | Calculate the savings of inserting node $i$ between edge $j$   |
|  | If such a move is feasible & If the savings is greater than or equal to zero, then make the move and continue with the I loop.     |
|  | Store the value of the largest savings so far.   |
|  | If the tour length $-$ largest savings from the J loop $\leq$ Record + Deviation, then make the move and continue with the I loop. |
|  | End I loop   |
| End J loop   |  |

## 2. Two-opt Move with Record-to-Record Travel

|   |   |
|---|---|
| Two-opt Move with record-to-record travel                     |   |
| For each edge $i$ in the current solution (I loop)            |   |
| For each node $j$ from the neighbor list of node $i$ (J loop) |   |
|   | Calculate the savings of the two-opt move with $i$ and $j$ ,                    |
|   | If the move is feasible, that is, both capacity and distance constraints        |
|   | are satisfied & If the savings is greater than or equal to zero                 |
|   | Then make the move and go to the I loop.  |
|   | Store the value of the largest savings so far.                                  |
|   |   |
|   | If the tour length – largest savings from the J loop $\leq$ Record + Deviation, |
|   | Then make the move and continue with the I loop.                                |

## 3. Perturb a Feasible Solution

|  |  |
|--|--|
| Perturb a feasible solution  |  |
| <p>For each node <math>i</math>, define <math>r(i) = d(i)/s(i)</math>, where <math>d(i)</math> is the demand of customer <math>i</math> and <math>s(i) = \text{dist}(\text{prior}(i), i) + \text{dist}(i, \text{next}(i)) - \text{dist}(\text{prior}(i), \text{next}(i))</math>, where <math>\text{dist}(i, j)</math> is the distance from node <math>i</math> to node <math>j</math>, <math>\text{prior}(i)</math> is the node serviced before node <math>i</math>, and <math>\text{next}(i)</math> is the node serviced after node <math>i</math>.</p> |  |
| <p>Sort the <math>r(i)</math> values in ascending order and select the first <math>M</math> nodes where <math>M = \min(20, n/10)</math>, where <math>n</math> is the number of nodes. Try to insert these nodes, one by one, into a new location on a tour using least-cost insertion while maintaining feasibility.</p>   |  |

APPENDIX B  
ROUTING SOLUTION

1.1 Routes on Monday generated by SMFB

| Route | Travel time(min) | Load(lb.) | Donors | Sequence  |
|-------|------------------|-----------|--------|---|
| 1     | 166              | 28235     | 12     | 0-1-2-3-4-5-6-7-8-9-10-11-12-0  |
| 2     | 129              | 30588     | 13     | 0-13-14-15-16-17-18-19-20-21-22-23-24-25-0                              |
| 3     | 103              | 28235     | 12     | 0-26-27-28-29-30-31-32-33-34-35-36-37-0                                 |
| 4     | 119              | 23520     | 10     | 0-38-39-40-41-42-43-44-45-46-47-0                                       |
| 5     | 105              | 32941     | 14     | 0-48-49-50-51-52-53-54-55-56-57-58-59-60-61-0                           |
| 6     | 140              | 28235     | 12     | 0-62-63-64-65-66-67-68-69-70-71-72-73-0                                 |
| 7     | 174              | 28235     | 12     | 0-74-75-76-77-78-79-80-81-82-83-84-85-0                                 |
| 8     | 132              | 28235     | 12     | 0-86-87-88-89-90-91-92-93-94-95-96-97-0                                 |
| 9     | 122              | 28235     | 12     | 0-98-99-100-101-102-103-104-105-106-107-108-109-0                       |
| 10    | 187              | 40000     | 17     | 0-110-111-112-113-114-115-116-117-118-119-120-121-122-123-124-125-126-0 |



### 1.2 Routes on Monday generated by Local Search

| Route | Travel time(min) | Load(b) | Donors | Sequence  |
|-------|------------------|---------|--------|---|
| 1     | 103              | 32928   | 14     | 0-1-48-54-55-56-57-58-59-60-126-8-6-7-122-103-102-0           |
| 2     | 140              | 37632   | 16     | 0-3-116-125-124-119-117-118-5-123-113-114-115-112-111-4-110-0 |
| 3     | 131              | 37632   | 16     | 0-13-86-14-15-21-25-16-17-24-19-20-18-23-22-98-0              |
| 4     | 131              | 35280   | 15     | 0-36-37-47-46-45-35-34-44-43-41-42-40-39-38-82-0              |
| 5     | 101              | 35280   | 15     | 0-73-91-92-93-94-95-90-96-76-77-78-97-89-88-87-0              |
| 6     | 115              | 37632   | 16     | 0-84-108-83-26-27-80-81-74-79-75-67-69-70-107-106-105-0       |
| 7     | 79               | 37632   | 16     | 0-85-12-120-121-61-10-53-52-51-9-50-49-11-101-100-99-0        |
| 8     | 121              | 35280   | 15     | 0-28-29-30-31-32-33-62-63-64-65-66-68-71-72-104-109-0         |

### 1.3 Routes on Monday Generated by MACS

| Route | Travel time(min) | Load(lb.) | Donors | Sequence  |
|-------|------------------|-----------|--------|---|
| 1     | 137              | 32928     | 14     | 0-4-9-51-1-48-55-54-6-7-53-52-10-8-126-0                |
| 2     | 116              | 32928     | 14     | 0-50-49-120-61-110-103-102-100-101-11-121-122-19-23-0   |
| 3     | 139              | 32928     | 14     | 0-76-96-90-77-89-88-87-78-91-92-94-93-95-97-0           |
| 4     | 157              | 32928     | 14     | 0-108-83-26-27-74-79-107-106-105-82-84-73-24-25-0       |
| 5     | 140              | 32928     | 14     | 0-117-124-125-116-3-111-112-113-115-114-123-5-118-119-0 |
| 6     | 127              | 32928     | 14     | 0-80-81-63-65-64-62-32-33-31-30-47-46-36-37-0           |
| 7     | 145              | 32928     | 14     | 0-109-104-29-28-66-70-75-69-67-68-71-72-34-44-0         |
| 8     | 136              | 32928     | 14     | 0-99-38-39-42-41-43-40-56-57-58-59-60-35-45-0           |
| 9     | 140              | 32928     | 14     | 0-13-14-15-20-18-2-86-98-85-12-22-16-17-21-0            |

### 2.1 Routes on Tuesday Generated by SMFB

| Route | Travel time(min) | Load(b) | Donors | Sequence                             |
|-------|------------------|---------|--------|--------------------------------------|
| 1     | 137              | 30800   | 11     | 0-1-2-3-4-5-6-7-8-9-10-11-0          |
| 2     | 115              | 22400   | 8      | 0-12-13-14-15-16-17-18-19-0          |
| 3     | 111              | 30800   | 11     | 0-20-21-22-23-24-25-26-27-28-29-0    |
| 4     | 149              | 30800   | 11     | 0-30-31-32-33-34-35-36-37-38-39-40-0 |
| 5     | 132              | 28000   | 10     | 0-41-42-43-44-45-46-47-48-49-50-0    |

### 2.2 Routes on Tuesday Generated by Local Search

| Route | Travel time(min) | Load(b) | Donors | Sequence                                      |
|-------|------------------|---------|--------|---|
| 1     | 142              | 39200   | 15     | 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-0          |
| 2     | 83               | 39200   | 8      | 0-15-16-17-18-19-20-21-22-23-24-25-26-27-28-0 |
| 3     | 145              | 36400   | 14     | 0-29-30-31-32-33-34-35-36-37-38-39-40-41-0    |
| 4     | 113              | 36400   | 13     | 0-42-43-44-45-46-47-48-49-50-51-52-53-54-0    |

### 2.3 Routes on Tuesday Generated by MACS

| Route | Travel time(min) | Load(b) | Donors | Sequence                                      |
|-------|------------------|---------|--------|---|
| 1     | 114              | 28000   | 10     | 0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-0          |
| 2     | 115              | 33600   | 12     | 0-15-16-17-18-19-20-21-22-23-24-25-26-27-28-0 |
| 3     | 144              | 39200   | 14     | 0-29-30-31-32-33-34-35-36-37-38-39-40-41-0    |
| 4     | 180              | 40000   | 15     | 0-42-43-44-45-46-47-48-49-50-51-52-53-54-0    |

### 3.1 Routes on Wednesday Generated by SMFB

| Route | Travel time(min) | Load(lb.) | Donors | Sequence                                   |
|-------|------------------|-----------|--------|--|
| 1     | 109              | 20000     | 8      | 0-1-2-3-4-5-6-7-8-0                        |
| 2     | 114              | 22500     | 9      | 0-9-10-11-12-13-14-15-16-17-0              |
| 3     | 106              | 17500     | 7      | 0-18-19-20-21-22-23-24-0                   |
| 4     | 105              | 27500     | 11     | 0-25-26-27-28-29-30-31-32-33-34-35-0       |
| 5     | 109              | 30000     | 12     | 0-36-37-38-39-40-41-42-43-44-45-46-47-0    |
| 6     | 167              | 32500     | 13     | 0-48-49-50-51-52-53-54-55-56-57-58-59-60-0 |
| 7     | 135              | 22500     | 9      | 0-61-62-63-64-65-66-67-68-69-0             |
| 8     | 161              | 27500     | 11     | 0-70-71-72-73-74-75-76-77-78-79-80-0       |

### 3.2 Routes on Wednesday Generated by Local Search

| Route | Travel time(min) | Load(b) | Donors | Sequence  |
|-------|------------------|---------|--------|---|
| 1     | 96               | 40000   | 16     | 0-1-2-3-47-4-5-74-46-37-38-45-43-44-36-8-24-0       |
| 2     | 142              | 40000   | 16     | 0-6-7-75-76-78-71-70-72-73-79-80-77-40-39-42-41-0   |
| 3     | 157              | 40000   | 16     | 0-17-13-12-16-11-10-15-14-9-61-62-63-48-58-69-59-0  |
| 4     | 154              | 40000   | 16     | 0-18-19-20-28-50-51-49-57-27-21-22-23-31-32-33-34-0 |
| 5     | 130              | 37500   | 15     | 0-35-25-30-29-26-52-56-53-68-54-55-66-67-65-64-60-0 |

#### 4.1 Routes on Friday Generated by SMFB

| Route | Travel time(min) | Load(lb.) | Donors | Sequence  |
|-------|------------------|-----------|--------|---|
| 1     | 141              | 27500     | 11     | 0-1-2-3-4-5-6-7-8-9-10-11-0   |
| 2     | 129              | 32500     | 13     | 0-12-13-14-15-16-17-18-19-20-21-22-23-24-<br>0                          |
| 3     | 122              | 32500     | 13     | 0-25-26-27-28-29-30-31-32-33-34-35-36-37-<br>0                          |
| 4     | 117              | 36400     | 10     | 0-38-39-40-41-42-43-44-45-46-47-0                                       |
| 5     | 105              | 35000     | 14     | 0-48-49-50-51-52-53-54-55-56-57-58-59-60-<br>61-0                       |
| 6     | 142              | 32500     | 13     | 0-62-63-64-65-66-67-68-69-70-71-72-73-74-<br>0                          |
| 7     | 178              | 32500     | 13     | 0-75-76-77-78-79-80-81-82-83-84-85-86-87-<br>0                          |
| 8     | 111              | 30000     | 12     | 0-88-89-90-91-92-93-94-95-96-97-98-99-0                                 |
| 9     | 147              | 27500     | 11     | 0-100-101-102-103-104-105-106-107-108-<br>109-110-0                     |
| 10    | 157              | 40000     | 16     | 0-111-112-113-114-115-116-117-118-119-<br>120-121-122-123-124-125-126-0 |

#### 4.2 Routes on Friday Generated by Local Search without Time Windows

| Route | Travel time(min) | Load(lb.) | Donors | Sequence   |
|-------|------------------|-----------|--------|--|
| 1     | 98               | 35000     | 14     | 0-12-13-14-21-1-22-17-18-19-20-16-23-15-88-0                 |
| 2     | 139              | 40000     | 16     | 0-24-123-114-116-115-3-113-112-2-124-4-117-119-118-125-105-0 |
| 3     | 106              | 40000     | 16     | 0-27-66-67-68-71-62-63-65-64-72-73-29-30-28-36-38-0          |
| 4     | 101              | 40000     | 16     | 0-37-74-93-94-95-96-97-92-98-77-78-79-99-91-90-89-0          |
| 5     | 140              | 40000     | 16     | 0-39-41-40-42-43-32-31-33-44-45-46-110-47-34-35-84-0         |
| 6     | 100              | 40000     | 16     | 0-48-54-55-58-56-57-59-60-126-7-61-121-120-104-102-101-0     |
| 7     | 111              | 40000     | 16     | 0-86-109-85-25-26-80-82-83-75-81-76-69-70-108-107-106-0      |
| 8     | 85               | 40000     | 16     | 0-87-11-111-122-6-5-9-53-51-52-8-50-49-10-103-100-0          |

### 4.3 Routes on Friday Generated by MACS with Time Windows

| Route | Travel time(min) | Load(lb.) | Donors | Sequence   |
|-------|------------------|-----------|--------|--|
| 1     | 142              | 37500     | 15     | 0-83-82-81-75-80-26-25-76-92-98-93-97-79-99-0              |
| 2     | 151              | 37500     | 15     | 0-20-15-23-16-14-13-88-55-54-44-45-84-35-36-0              |
| 3     | 132              | 37500     | 15     | 0-46-47-38-101-103-102-111-2-125-7-5-6-61-105-0            |
| 4     | 165              | 37500     | 15     | 0-70-66-65-64-63-62-71-68-67-69-32-43-72-73-0              |
| 5     | 93               | 37500     | 15     | 0-34-10-49-50-52-51-53-9-8-121-120-104-87-11-0             |
| 6     | 145              | 35000     | 14     | 0-31-33-42-40-41-39-48-110-56-57-58-59-60-0                |
| 7     | 148              | 37500     | 15     | 0-30-29-27-28-12-100-21-1-17-18-19-24-74-37-0              |
| 8     | 130              | 40000     | 16     | 0-22-122-112-113-123-114-116-115-3-4-124-119-117-118-126-0 |
| 9     | 135              | 37500     | 15     | 0-95-96-94-77-78-91-90-89-106-107-86-109-108-85-0          |