INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

SERVICE LEVEL BASED VEHICLE ROUTING PROBLEM: MATHEMATICAL MODELS, NEURAL NETWORKS HEURISTIC AND OPERATIONAL IMPLICATIONS

by

RI XIA

A dissertation submitted to the Graduate Faculty in Business
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy, The City University of New York

2003

UMI Number: 3103185

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

This manuscript has been read and accepted for the Graduate Faculty in Business in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Aug 6, 2003
Date

Chair of Examining Committee

Aug 5 '03
Date

Executive Officer

Professor William Chien

Professor Georghios Sphicas

Professor Elsie Gottlieb

Supervisory Committee

The City University of New York

ABSTRACT

SERVICE LEVEL BASED VEHICLE ROUTING PROBLEM: MATHEMATICAL MODELS, NEURAL NETWORKS HEURISTIC AND OPERATIONAL IMPLICATIONS

by

Ri Xia

Advisors: Professor William Chien, Professor George Sphicas, and Professor Elsie Gottlieb

We extend the Vehicle Routing Problem (VRP) by taking the service level into consideration. The result is called the Service Level Based Vehicle Routing Problem (SLB-VRP). In this study, the service level perceived by a customer is measured by the waiting time of that customer. Three models are constructed for the SLB-VRP. Among them, two are mixed integer linear programming models that are based on the modeling of traffic flows. The third model uses decision variables that record the sequence of the customers who receive service from the vehicles and has a quadratic objective function. We have also studied how to deal with several varieties, such as the minimization of total traveled distance, the minimization of maximum waiting time, capacity constraint, and time windows.

We have developed a neural networks based heuristic to solve the SLB-VRP. The neural networks used in the heuristic are Hopfield networks. We have used a guided sampling process in the algorithm to save the computation time and estimated the

maximum and minimum number of customers in a route to reduce the complexity of the networks. Two local search procedures, an oscillating procedure and a 2-opt exchanging procedure, are also implemented to improve the performance of our heuristic.

We have tested the performance of our algorithm by three approaches. The first approach is to compare our solution to an estimated optimal solution. The second approach is to compare our solution to the solutions of other well-known heuristics. The third approach is to design a performance indicator based on our solution and the statistics of the traveling distance matrix. Our experiments have shown that our algorithm has good performance according to all three approaches.

Experiments are also conducted to study the operational implications, such as the effects of the shape of the service region and the location of the depot. We have found that Circle service region has the shortest Total Waiting Time and the average Total Waiting Time in the depot at corner case is estimated to be 1.5 times of the average Total Waiting Time in the depot at center case.

Acknowledgments

I would like to express my deep gratitude to Professor William Chien, Professor Georghios Sphicas, and Professor Elsie Gottlieb. Without their invaluable guidance, suggestions, and encouragement, this dissertation could not have been finished. Especial thanks should go to the chair of my dissertation committee, Professor William Chien, for his intensive supervision and critical discussion throughout this dissertation writing period. Thanks also goes to Professor Elsie Gottlieb for kindly providing me the integer programming software package.

I would also like to express my appreciation to Professor Gloria Thomas and Professor Georghios Sphicas for their financial support during all these years of my doctoral study. I also want to thank Ms. Molveine Karen, who helped me understand the dissertation preparation procedure and other administrative rules.

I also wish to thank my colleagues and friends David Chu and Hongdeng Chen for proofreading my dissertation and keeping me busy with helpful discussions and suggestions.

Finally, I want to thank my family, without whom I would never have been able to achieve so much. I especially wish to express my love for my wife, Mai Yang, for her constant support during all of these enduring years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter I

INTRODUCTION

1.1 Introduction

Vehicle Routing Problem (VRP) is a well-known problem in the field of Operations Management and Combinatorial Optimization. One brief definition for the basic type of VRP can be stated as:

One set of customers {1,2, ... , n} requires service from the depot {0}. There are m vehicles originally located at the depot. The vehicles will provide service when they visit the customers. There is a cost associated with each pair of points from the set {0,1,2, ... , n}. A feasible solution is defined as m routes that visit each customer exactly once. The optimal solution is the feasible solution with the least total cost.

Three entities are used in the definition of the VRP: customer, depot, and vehicle. The customer is the driving force of the whole vehicle routing system. Each customer requires a certain type of service. The whole system exists because we wish to serve the customers effectively and efficiently. The depot contains the resource that the customers ask for, and we will use vehicles to carry the required resource from the depot to the customers. These three entities should be treated as abstract terms. The customer may be a real customer, or a grocery store, or any entity requiring certain resource. The depot may be a real warehouse, or a data center, as long as it plays the role as the resource origin. The vehicle should be regarded as the general term for any facility that allows the resource to flow between the depot and the customers.

As one logistical problem, the VRP has significant economic impacts. According to Srivastava and Benton (1990), the combined transportation and warehousing costs account for at least 20 percent of the U.S. GDP each year. The potential huge gain from the improvement of the vehicle routing system is believed to be one major cause why the VRP is so popular in the literature. Besides, practical distribution cases have many specific requirements, which will prompt researchers to develop new models to accommodate these practical situations. Furthermore, the model and method used to study the VRP can be applied to problems in other fields. For example, Chen, Wan, and Xu (1998) formulate the rolling batch planning problem in the steel industry as one vehicle routing problem with time windows. Given the above-mentioned reasons, it is not a surprise to find many different VRP models in the literature.

In essence, the VRP can be regarded as a resource allocation problem. What the VRP considers is how to allocate the resource according to some time and space requirements. Space requirements come from the customers. The locations of the customers specify where the allocated resource should go. On the other hand, the time issue is affected by two factors. One is the spatial distribution of customers and the other one is the instrument used to deliver the requested resource. If we adopt this understanding of the VRP and treat the three entities of the VRP as abstract terms mentioned above, we can then extend the VRP to various related decision problems requiring resource allocation planning.

1.2 Motivation and research questions

The VRP originally arose from the distribution industry. Although there are some papers in the literature discussing the applications of the VRP in other fields, most models are constructed while physical distribution is kept in the researcher's mind. Therefore, current available models more or less take some characteristics of the distribution industry into granted, which limits the applicability of the VRP to other resource allocation problems.

As far as the resource is concerned, two parties are involved in the VRP. They are the resource owner and the resource user. The customers are the resource users and the depot can be regarded as the representative of the resource owner. The objective of the vehicle routing system should reflect the interests of these two parties. Since the resource owner is interested in decreasing the cost of delivering the resource while the resource users pay more attention to the service related to how the resource is received, the main objective of the VRP is either to minimize the delivery cost or to maximize the service level. In the distribution industry, the distribution cost is a major concern while we take its contribution to the GDP into consideration. On the other hand, few requirements on how service should be provided are specified, maybe because in most cases the resource is physical. Therefore, minimizing the system-wide cost is the main objective of the VRP. However, in some cases when the distribution cost is negligible, the service level or customer satisfaction level becomes much more important. Especially, while we consider the general resource allocation problem, the resource may not be physical goods and the usage of the resource may heavily depend on how the resource is received. Consequently, it is worthwhile to integrate the service level with the

ordinary vehicle routing requirements. The result is a relatively new class of VRP called the Service Level Based Vehicle Routing Problem (SLB-VRP), whose objective function is defined over the service level related cost items.

More specifically, we are interested in studying the SLB-VRP for the following three reasons. First, for the vehicle routing systems in the domain of public service, the service level is expected to be the first priority in the mind of decision-makers. Decision-makers cannot use cost as the only argument to justify their decisions. Hence, the service level will inevitably enter into the objective function. Second, in today's tough competition market, customer satisfaction is critical to gain competitive advantage. Finally, The recognition of the importance of the SLB-VRP benefits from the integration of the organization-wide operations. While treating the vehicle routing system as an independent system, we may need to sacrifice the cost savings to improve the service level; however, if we regard this system as one part of the whole supply chain, the achievement in the service level may lead to savings in the total cost for the entire supply chain.

In the literature, two groups of algorithms are developed to solve the VRP. The first group includes the exact algorithms. Most algorithms in this group use the branch and bound method. The second group includes the heuristics. Among these heuristics, a special class of algorithms, called metaheuristics, has been reported with good performance. Metaheuristics includes tabu search, simulated annealing, genetic algorithm, evolutionary strategies, and neural networks. Although pursuing optimality is the ultimate goal while studying one optimization problem, the intractability of the VRP limits the problem size we can handle using currently available exact algorithms.

Without a breakthrough in the theoretical base of integer programming, it is difficult to make progress in developing exact algorithms. Since one intention of this research is to promote the recognition of the SLB-VRP, we do need to demonstrate computational results. Therefore, in this research, we will use heuristics as the solution technique and at the same time, construct mathematical models to leave the door open for the future development of exact algorithms.

Our research questions come from three areas. The first one is the modeling. Can we construct mathematical models for the SLB-VRP? The second one is the solution technique. Are we able to develop an efficient and effective heuristic algorithm to solve the SLB-VRP? The last one is the operational implications. If we adopt the maximization of the service level as the objective, what is the effect of this adoption on operating the warehousing and distributing system?

At the end of this section, we discuss two possible applications of the SLB-VRP.

An Emergency Responding System A utility company has a fleet of vehicles that are responsible for emergencies occurring in a city. Instead of using the minimization of the total traveling cost as the objective, this company wants to minimize the total waiting time of the customers for the following two reasons. First, the highest priority is to minimize shutdown time, and total waiting time is a good measurement for shutdown time. Second, if the system is operated with this objective, no customer's service will be extremely delayed, which is helpful in reducing the number of complaints and maintaining a high level of the credibility of this system.

A waste collection system A company is responsible for collecting waste from

some dumping sites in a city. For certain economical and environmental reasons, the objective is to minimize the cumulative waste level in those dumping sites. The waste in each dumping site is accumulated at a certain constant rate. So the cumulative waste level in a dumping site can be expressed in terms of the waiting time of that site, where the waiting time of a site is defined as the time from the start of the planning period till the time that this site's waste gets collected. Therefore, the objective can be set as the minimization of the total waiting time of those dumping sites.

1.3 Research methodology and expected contributions

In this study, the service level is measured by the waiting time, where the waiting time of a customer is the time from the start of the planning period till the time that this customer receives service, assuming that all customer demands arrive at the start of the planning period. Hence, the objective of maximizing the system-wide service level can be achieved by the minimization of the total waiting time. Furthermore, if we assume that the vehicles have constant speed, the travelling time is proportional to the travelling distance. Since the waiting time of a customer is simply the traveling time from the depot to the location of this customer, the waiting time can be expressed in terms of the traveling distance. So in our SLB-VRP, we will minimize the summation of each customer's traveling distance. Here, the traveling distance of a customer is defined as the traveling distance from the depot to the location of this customer.

Three basic models are constructed for the SLB-VRP. Among them, two are mixed integer linear programming models that are based on the modeling of traffic flows. The first model uses a three-index decision variable that specifies the starting customer,

the ending customer, and the vehicle used for the trip. The second model is more compact because only the starting customer and the ending customer for the trip are specified in the decision variables. The third model that we propose is not constructed upon the traffic flows. Instead, its decision variables record the sequence of the customers who receive service from the vehicles. Since a trip is defined by a starting customer and an ending customer, we need two decision variables to decide whether a trip happens or not. Therefore, we will have a quadratic objective function in the third model. To illustrate the difference between the VRP and the SLB-VRP, we have done an experiment, that reveals that the value of the optimal solution changes significantly when we switch to the SLB-VRP model from the VRP model.

In our basic models, we do not consider the capacity constraints, but capacity constraints can be easily incorporated into our modeling framework. Besides, since our objective is to minimize the total waiting time, all available vehicles will be used in the optimal solution. So our basic models are constructed by treating the VRP as m-TSP. We will also present some variations of our basic models, taking more characteristics of VRP into consideration.

Although existing methods to the VRP can be modified to solve the SLB-VRP, we develop a new algorithm, partly because we want to add a new method to the domain of the VRP and partly due to the following two concerns for future research. First, although in this study we will only consider the basic measurement of the service level, which is the total waiting time, we can expect that in practical situations many other measurements will be used by the customers to evaluate the perceived service level. Therefore, our algorithm should have the potential to be extended to accommodate

various other measurements for the service level. The other concern is how fast the algorithm can provide a solution. If we construct the vehicle routing system around the intention to achieve a better service level, most likely we are facing a volatile environment. For example, the demand may occur randomly and it is possible to lose customers if we cannot provide timely service. Therefore, quick response will be critical to be competitive in handling the SLB-VRP, and our algorithm should be able to provide solutions instantaneously.

Based on the above-mentioned concerns, we choose neural networks as the solution technique since neurons can make instantaneous response and be modeled with different measurements for the service level. We construct Hopfield networks (Hopfield, 1982, 1984), which are stochastic neural networks, to solve the SLB-VRP because Hopfield networks have been used to solve the Traveling Salesman Problem (TSP), which can be regarded as a special case of the VRP. Our networks are constructed based on the third model because the Hopfield networks require the objective function be in a quadratic form. The Hopfield networks act as a global searching device and two local searching procedures are also developed to improve the performance. Furthermore, because the Hopfield networks have been criticized for their complexity and time-consuming computation in their application to the TSP, we provide two additional procedures to improve our networks by reducing the number of neurons and speeding up the computation.

We have taken two approaches to evaluate the performance of our Hopfield networks based algorithm. The first approach is to compare our solutions with estimated

optimal solutions or solutions obtained by other well known heuristics. We have to use estimated optimal solutions because SLB-VRP is NP-Complete (Sahni and Gonzalez, 1976); therefore, no optimal solution is available for large size SLB-VRP. The estimated optimal solution is estimated by a method based on order statistics of a sequence of solutions (Dannenbring, 1977). For solutions obtained by other heuristics, we have modified the Savings algorithm and the Sweep algorithm so that they can solve the SLB-VRP. The second approach is to directly measure the quality of a solution. We have designed a performance indicator to measure how short an average trip in a solution is. The smallness is measured based on the statistics of all elements in the travelling distance matrix.

Since the direct application of the SLB-VRP is still in the distribution industry, it is worthwhile to study the operational implications of the SLB-VRP to the warehousing/distributing system. The first issue is the shape of the service region. We want to know if the shape of service region has any effect on the system-wide service level. The second issue is the location of the depot. We want to know if there are significant differences in the service level for different locations of the depot. We are interested in these two issues because they are related to two tactical decisions the distribution manager needs to make. These two decisions are how to divide a large service region into smaller ones and where to locate the depot. Several experiments will be designed to address these concerns.

According to our research questions and preliminary research, we can expect the following contributions:

- Three mathematical models for the SLB-VRP are constructed.

- The difference between the VRP and the SLB-VRP is studied.
- A Hopfield-networks-based heuristic algorithm is built to solve the SLB-VRP.
- Two local searching procedures are developed to improve the performance.
- Two procedures are developed to improve the efficiency of the Hopfield networks by reducing the number of neurons and speeding up the computation.
- Savings algorithm and Sweep algorithm are modified to solve the SLB-VRP.
- A performance indicator is designed to measure the quality of a solution to SLB-VRP.
- The effects of the service regions with different shapes are examined.
- The effects of the location of the depot are studied.

Chapter II

LITERATURE REVIEW

Since the first paper about the VRP was discussed in Dantzig and Ramser (1959), many different models and solution techniques have appeared in the literature. In this chapter, we provide a literature review of the VRP. Since the purpose of this review is to gain insights to the current research directions, we will not provide a full scale and detailed review; instead, except for some classical papers, only recently published papers are included. For extensive reviews on this subject, readers can go to Christofides, Mingozzi, and Toth (1979), Bodin, Golden, Assad, and Ball (1983), Laporte and Nobert (1987), Laporte (1992), Assad and Golden (1995), Eiselt, Gendreau, and Laporte (1995a, 1995b), Laporte and Osman (1995) and Psaraftis (1995).

This chapter is organized as follows. To have a full understanding of the VRP, we provide a classification scheme in section 2.1. Some ideas about this scheme come from Golden, Magnanti, and Nguyen (1977), Bodin and Golden (1981), and Desrochers, Lenstra, and Savelsbergh (1990). Then, we will discuss the solution techniques. Exact algorithms are covered in section 2.2 and heuristic algorithms are covered in section 2.3.

2.1 Classification of Vehicle Routing Problem

To classify the VRP models, we need to first recognize those elements that specify a model. Essentially, there are two parts in a VRP model: an objective function and constraints. The objective function reflects the interest of the whole routing system, which is used to evaluate the performance of different ways to do the vehicle routing. On

the other hand, constraints describe the features of the problem. Since the depot, customer, and vehicle are the only three entities in the VRP, those features can be classified according to these entities. Some features only require the involvement of a single entity, but some features are defined as a certain type of relationship among entities. Therefore, we will use five elements to classify the VRP. They are objective function, depot, customer, vehicle, and relationship. After the discussion of each element, some research papers are introduced to illustrate the context of some special options that an element might take.

2.1.1 Objective function

The objective function is a function of costs. Many different types of cost items have been discovered in practical situations. While vehicles are traveling between the depot and the customers, traveling cost occurs. Usually this cost is calculated on the traveling distance or time. As mentioned above, we have constraints guiding us on what we have to do and what we cannot do. Sometimes, we can violate those constraints by paying a certain amount of penalty, which is called penalty cost. Another common type of cost is vehicle cost. It is fixed and depends on the vehicle type. The most frequently used objective function takes the form of minimizing the sum of the costs, which reflects the intention to minimize the system-wide total cost. Another form of the objective function is to minimize the maximum individual cost item. This objective function is used to do damage control or maintain a certain level of customer satisfaction.

Park and Koelling (1989) build an interactive computerized algorithm for multicriteria vehicle routing problems. The model consists of the following relevant

objectives: (1) the minimization of the travel distance of vehicles, (2) the minimization of the total deterioration of goods during transportation, and (3) the maximization of the total fulfillment of emergent services and conditional dependencies of customers. Anily and Federgrun (1990a) examine the general cost function defined over the length and the number of points on the route with the assumption that the cost function is nondecreasing and concave. Anily and Federgrun (1990b) study a distribution system with the objective to determine the feasible replenishment strategies that minimize the long run average transportation and inventory costs. Wijeratne, Turnquist and Mirchandani (1993) develop a method for determining a set of non-dominated routes in stochastic networks when multiple, uncertain measures form the basis of route evaluation. Karkazis and Boffey (1995) describe one case related to the transportation of hazardous materials where the objective is to minimize the expected damage effects on the population in case of an accident. Malmborg (1996) uses the service level as the cost item in a mail delivery system, where the service level is defined as the waiting time periods for one mail between the accumulation at the origin and the arrival at the destination. Golden, Laporte, and Taillard (1997) study the situation in which the objective function takes the minmax form. They cover a class of vehicle routing problems, including the Capacitated VRP, the Capacitated VRP with multiple use of vehicles, and the m-Traveling Salesman Problem with multiple use of vehicles. Herer and Levy (1997) integrate the VRP with inventory management. The objective in their model is to determine when to serve each customer and the route to be performed by each truck, in order to minimize the total discounted costs, including holding, transportation, fixed ordering, and stock out costs. Butt and Ryan (1999) assume that each customer provides a reward if visited, and study

how to maximize the collected reward if it is impossible to visit all customers. Hong and Park (1999) propose a bi-objective model. The objectives are to minimize both the total vehicle traveling time and the total customer waiting time. Solutions are generated by specifying both the target values of these objectives and the decision-maker's preference for the objectives expressed. Weintraub, Aboud, Fernandez, Laporte, and Ramirez (1999) study an emergency vehicle dispatching system, aiming to optimize service quality that was measured by response time while taking priority level into consideration.

2.1.2 Depot

Depot can be further classified according to three sub elements: number, type, and capacity. Most models assume there is only one depot, but a multiple depot case is not rare in the literature. In addition to the common type of depot, another facility, called satellite site, can be regarded as a special type of depot. Satellite site has some characteristics of the depot, such as providing resource that the customers are asking for, but it lacks other characteristics, such as the inability of vehicles to start from or return to it. With satellite facility, drivers are allowed to continue making deliveries until the end of their shifts without necessarily returning to the central depot for replenishment. Usually we will assume that the depot has enough capacity, but sometimes a limited capacity constraint is imposed over the depot. We may even have the situation where the depot is a workstation, in which case the production rate will be used to address the capacity requirement.

A multiple depot case is one recent trend in the literature. Usually, some location or allocation features are included in the model. Some examples can be found in Sumichrast and Markham (1995), Renaud, Laporte, and Boctor (1996), Salhi and Sari (1997), Hadjiconstantinou (1998), Salhi and Nagy (1999), Chan (2001). For satellite facility, Bard, Huang, Dror, and Jaillet (1998) use it as one way of safeguarding against unexpected demand. Sariklis and Powell (2000) study a special case of the VRP called the open vehicle routing problem, where the vehicles are either not required to return to the depot, or they have to return by revisiting the customers assigned to them in the reversed order.

2.1.3 Customer

This element has four sub-elements: location, service type, demand, and temporal constraint. If treating all paths that the vehicles can travel as one network, the customers, with demand, can be located on the edge or on the node. Of course, it is not necessary to require that all customers have the same location type. In terms of the service type, usually we consider only two alternatives: delivery and pickup. Sometimes, a group of customers asks for a delivery and another group wants a pickup; then, we have a mixed delivery and pickup case. The demand of customers may be known in advance, which is the deterministic case, or follow a certain type of probability distribution, which is the stochastic case. The temporal constraint of customers is called time windows, which is used to model the situation that some customers require service in some predetermined time intervals.

The VRP with time windows is an active topic in the literature. The main stream is the development of heuristic algorithms, especially metaheuristics, which will be covered in section 2.3. Balakrishnan (1993) describes a related temporal constraint, called the soft time windows, where the time windows constraint can be violated by paying a certain amount of penalty. The pickup and delivery problem is another interesting feature of the VRP. Dumas, Desrosiers, and Soumis (1991) study the construction of optimal routes to satisfy transportation requests, each requiring both pickup and delivery under capacity constraint. Min, Current, and Schilling (1992) tackle the case of customers who simultaneously request pickup and delivery. Shang and Cuff (1996) consider a special pickup and delivery problem where the customers are allowed to transfer between the vehicles and many-to-many transportation is integrated in the model. Mosheiov (1998) studies an important version of the VRP with pickup and delivery, called the delivery and backhaul problem, where the pickup transports goods from customers to the depot. Swihart and Papastavrou (1999) develop a stochastic and dynamic model for the pickup and delivery problem, where the pickup locations are independent and uniformly distributed over a service region and the demand for service arrives according to a Poisson process in time. The VRP with stochastic demand is one step forward in capturing the practical situation. The type of probability distribution and the related probabilistic analysis are two popular issues in the current literature. Some recent research papers are Benton and Rossetti (1992), Bertsimas (1992), Gendreau, Laporte, and Seguin (1996), Ong, Ang, Goh, and Deng (1997), Secomandi (2000, 2001), and Chan (2001).

2.1.4 Vehicle

This element has five sub-elements. The first sub-element is the fleet size. Fleet size can be fixed or be varied but restricted in one range specified by an upper bound and a lower bound. The second sub-element is the vehicle type. If all vehicles are the same, we have a homogeneous fleet; otherwise, we have a heterogeneous fleet. Capacity is the third sub-element. If the demand of customers has quantity characteristic, we usually put a capacity constraint on the vehicle. If the demand of customers is one kind of service, we may assume the vehicles have unlimited capacity. A special issue is compartment. The involvement of the compartment makes the concern of the capacity of vehicles more complicated. In the multi-product distribution case, vehicles may have compartments to store different products. A compartment itself can be divided into two types: interchangeable and dedicated. An interchangeable compartment can be used to store similar products, but dedicated compartment can be used only to store pre-specified products. The fourth sub-element is the temporal constraint. Vehicles can have time windows as well, which happens because sometimes vehicles are not able to work continuously. Another temporal constraint is the route duration. Some routing system may put a limit on the maximum time a vehicle can work. The last sun-element is the serving strategy. There are three common strategic decisions involved in the routing system. Serving quantity is one concern. The basic assumption is that a customer's demand will be satisfied in full amount. Some special cases will allow a demand split, which means some customers may be served by more than one vehicle or more than one visit. This decision can be made a priori in the deterministic demand case, or a posteriori in the stochastic demand case. Another decision is about backhauling. In case we have

mixed deliveries and pickups, backhauling is one common option, which means delivering first to empty the vehicle and then collecting loads on the way back to the depot. How many shifts one vehicle can take in one planning time period is the third decision problem. In most cases, a vehicle will take at most one shift per period, but a multiple shifts situation has been studied in the literature as well.

Desrochers and Verhoog (1991) address the problem of simultaneously selecting the composition and routing of a fleet of vehicles in order to efficiently serve customers with known demands for a central depot. This problem is called the fleet size and mix vehicle routing problem. Gendreau, Laporte, Musaraganyi, and Taillard (1999) study the heterogeneous fleet vehicle routing problem. In this problem, the assumption of identical vehicles is dropped from the classical VRP. Vehicles are different in capacity, variable cost, and fixed cost. Liu and Shen (1999) extend the fleet size and mix vehicle routing problem with time windows constraints and study the short term and long term implication of this problem. Vaidyanathan, Matson, Miller and Matson (1999) study the vehicle routing issue of a just-in-time production system. This problem is formulated as a Capacitated VRP. In addition, a non-linear capacity constraint is added to achieve the minimization of vehicle idle time and the inventories at the locations of the customers. Split delivery, backhauling, and multiple use of vehicles are three often-seen serving strategies in the literature. Examples can be found in Frizzle and Giffin (1995), Taillard, Laporte, and Gendreau (1996), Thangian, Potvin and Sun (1996), Brandao and Mercer (1997, 1998), Salhi and Nagy (1999), Toth and Vigo (1999), and Belenguer, Martinez, and Mota (2000).

2.1.5 Relationship

We consider five relationships among the entities of the VRP. They are customer-to-customer, customer-to-vehicle, customer-to-depot, vehicle-to-vehicle and vehicle-to-depot. Customer-to-customer refers to the relationship among the customers. There are four types of such relationship. The first type is precedence. If a customer must be visited before another customer, we say there is a precedence relationship between these two customers. The second type is the origin-destination pair, which models the situation when a customer sends out something to another customer. This is similar to precedence. The difference is that we usually view this origin-destination pair as one single customer. The other two types are inclusion set and exclusion set. Inclusion set refers to a set of customers that need to be visited in the same route. Exclusion set refers to a set of customers that cannot be visited in the same route. Customer-to-vehicle, customer-to-depot, and vehicle-to-depot can be regarded as types of inclusion set relationship since they just state that some entities need to be included in the same route. We list them separately simply for the sake of clarity. Vehicle-to-vehicle relationship considers the situation when vehicle synchronization is necessary. Usually vehicle synchronization is required because the customers have multiple services requested and we have a heterogeneous fleet with different vehicles providing different services.

Jansen (1992) develops an approximation algorithm for the general routing problem. The goal of the general routing problem is to find a minimum cost cycle that visits each vertex in a required subset exactly once and traverses each edge in a required subset of the network. Malmborg (1996) studies the origin-destination relationship in a

probabilistic sense, where a probability is assigned for each pair of customers to specify the strength of the relationship. Shang and Cuff (1996) discuss the many-to-many transportation relationships in a pickup and delivery case. Brandao and Mercer (1997) address a constraint that the access to some customers is restricted to some vehicles. Ong et al. (1997) suggest a selection criterion to serve a selected set of customers with the objective to minimize the total traveling cost and possible losses. Chao, Golden, and Wasil (1999) and Cordeau and Laporte (2001) study the site dependent vehicle routing problem. The characteristic of this problem is that each customer can only be served by a subset of a heterogeneous fleet. Ghiani and Improta (2000) tackle the generalized vehicle routing problem. This problem is an extension of the VRP featuring a customers set consisting of mutually exclusive and exhaustive clusters, each cluster having only one customer visited.

In table 2-1, we present the summary for this classification. This table contains three columns. The first column lists those five elements. The second column lists sub-elements of each element. And the last column lists possible options for each sub-element.

2.2 Exact algorithms

According to Laporte and Nobert (1987), exact algorithms can be classified into three categories: direct tree search methods, dynamic programming, and integer linear programming. In this section, we will provide a review for each category, including classical papers and available recent research results. Some algorithms may not aim at

optimality in their original research. We will regard them as exact algorithms as long as they can be extended to pursue optimality.

2.2.1 Integer linear programming

According to different types of decision variables, there are two major groups of integer linear programming models. The models in the first group use decision variables defined over cost items of pairs of vertices in the network. Fisher and Jaikumar (1981) develop a three-index vehicle flow formulation for the VRP. The decision variable is X_{ijk}, denoting whether the kth vehicle travels the trip from the ith customer/depot to jth customer/depot. Benders' decomposition is used to decompose the VRP into two sub-problems. One is the generalized assignment problem (GAP) and the other the traveling salesman problem with time windows (TSPTW). The procedure iterates between solving a GAP master problem that assigns vertices to vehicles, and solving a TSPTW to determine the best vehicle route for each vehicle. Laporte, Nobert, and Desrochers (1985) propose a two-index integer linear programming model by dropping the index k from the decision variable. The validity of the model is guaranteed by the flow conservation constraint. This model is solved by constraint relaxation and branching. The subtour elimination constraint is first taken out of the model. After getting one feasible solution, any violated subtour elimination constraint will be identified and the detected constraints will be added to the model. Then the model will be resolved until there is no violation of subtour elimination constraint. If the solution is integer, the solution will be reserved as the best known solution; otherwise, sub-problems will be created by branching on a fractional variable. The models in the second group use

decision variables defined over the cost of feasible routes. The model is formulated in finding the minimum cost of some routes that cover each customer once and only once. This kind of formulation is called the set partitioning formulation. Balinski and Quandt (1964) are among the first to propose such a formulation for the VRP. The large number of feasible routes is the major difficulty associated with this formulation. One popular method of getting around this difficulty is column generation. Here a column represents a feasible route. In column generation algorithm, a restricted subset of all feasible routes is used to solve the linear relaxation of the set partitioning formulation of the VRP. Then an algorithm is used to find the column with the least reduced cost. If this column has a non-negative reduced cost, the current solution is optimal. Otherwise, this column will enter the basis and the problem is re-optimized. This technique has been applied to the field of vehicle routing by Rao and Zionts (1968), Foster and Ryan (1976), Orloff (1976), Desrosiers, Soumis, and Desrochers (1984), Agarwal, Mathur, and Salkin (1989), Desrochers, Desrosiers, and Solomon (1992), and Butt and Ryan (1999).

2.2.2 Direct tree search method

In this category, the VRP is usually also formulated as an integer linear programming model. The difference between a direct tree search method and an integer linear programming method is that, in the direct tree search method, the network property of the VRP is used to provide lower bounds to facilitate the branch and bound process. Christofides, Mingozzi, and Toth (1981a) develop an algorithm for the symmetrical VRP based on the k-degree center tree relaxation. A k-degree tree is one spanning over the network, where the degree of the depot is k, where k equals to 2m-y. Here m is the

number of vehicles and y is an integer between 0 and m. We can construct a feasible solution for the VRP based on each k-degree center tree. After we get a tree, the edge set is divided into four subsets. Set 1 includes edges not belonging to the solution. Set 2 includes edges forming a k-degree center tree. Set 3 includes y edges incident to the depot. Set 4 includes m-y edges not incident to the depot. It is clear that sets 2, 3, and 4 can be used to form m routes. The VRP is formulated by finding the minimum-cost k-degree center trees, and Lagrangean relaxation is used to obtain a lower bound that can be used in the branch and bound scheme. Laporte, Mercure and Nobert (1986) provide an algorithm by capitalizing on the relationship between the VRP and one of its relaxations, the m-TSP. The m-TSP can be further transformed into the regular TSP according to Lenstra and Rinnooy Kan (1975). Then, a branch and bound process can be started with assignment problems as its sub-problems. Fischetti, Toth, and Vigo (1994) discuss a branch and bound algorithm with two new bounding procedures based on an additive approach. Fisher (1994) proposes another spanning tree based algorithm. For the VRP with n customers and k vehicles, a k-tree is defined as one set of n+k edges that span over the graph. The VRP is formulated as the problem of finding a minimum-cost k-tree with 2k edges incident to the depot. Then Lagrangean relaxation and branch and bound process are used to solve this problem as well. Achuthan, Caccetta, and Hill (1996) implement a branch and bound procedure with a new subtour elimination constraint. Fisher, Jornstern, and Madsen (1997) describe two optimization methods for the VRP with time windows. One is a k-tree relaxation and the other is a Lagrangean decomposition with two sub-problems: a semi-assignment problem and a series of shortest path problems. Kohl and Madsen (1997) present an optimization method based

on the Lagrangean relaxation of the constraint set requiring that each customer be served. The master problem consists of finding the optimal Lagrangean multipliers and the sub-problem is a shortest path problem with time windows and capacity constraints. Achuthan, Caccetta, and Hill (1998) provide eight new strong cutting planes for the Capacitated Vehicle Routing Problem (CVRP). These cutting planes come from three ideas. The first is the connectivity. By eliminating the subtour, some new constraints are constructed. The second idea is based on the fact that if several customers of a set S are required to be served by the same vehicle, the minimal number of the vehicles to serve S may increase. The lower bound from Bin Packing Problem (BPP) is used to develop several new strong cutting planes. The third idea is to study the relationship between the ordinary CVRP and the CVRP with unitary demand. Then some constraints called multistar constraints are developed. Augerat, Belenguer, Benavent, Corberan, and Naddef (1998) study how to efficiently identify valid inequalities in the branch and bound method. Emphasis is put on the capacity constraints, and three algorithms are developed to identify them. Those algorithms are a constructive heuristic capitalizing on the minimum capacity cut, a greedy randomized algorithm and a tabu search algorithm. Bard, Huang, Dror, and Jaillet (1998) show a branch and bound algorithm in which a subtour violation is recognized in the support graph, and some constraints out of a sequential lifting procedure are used as well.

2.2.3 Dynamic programming

Eilon, Watson-Gandy, and Christofides (1971) first propose a dynamic programming approach for the VRP. The basic idea is to treat the set of customers as

states and the number of vehicles as a stage. Let c(S) denote the cost of a vehicle route through depot and all vertices of set S. And let $f_k(U)$ be the minimum cost achievable, using k vehicles and delivering to a subset U of customer set V. We can have the following recursive relationship:

$$f_k(U) = \begin{cases} c(U) & k = 1 \\ \min_{U^* \subset U}[f_{k-1}(U \setminus U^*) + c(U^*)] & k > 1 \end{cases}$$

The solution cost is equal to $f_m(V)$ and the optimal solution corresponds to the U^*s in the recursive process. One drawback of this method is that a huge number of states need to be considered, even for a moderate size problem. Therefore, the state-space reduction technique is necessary to successfully implement the above-mentioned scheme. Usually, some side constraints, such as a capacity constraint, can provide some reduction in the number of states. Another efficient approach is state-space relaxation. This method is introduced in Christofides, Mingozzi, and Toth (1981b). A mapping function will be used to transfer the original state-space into a smaller cardinality state-space. At the same time, the original recursion will be transferred correspondingly. Lower bounds of the original problem can be obtained from the new recursive relationship and can be used in the branch and bound process to obtain optimal solution.

2.3 Heuristic algorithms

Due to the computational intractability, it is usually inefficient to implement exact algorithms. Heuristic algorithms can provide near-optimal or satisfactory solutions with significant reduction in computational resource. In the current literature, heuristic algorithms are the main solution techniques to handle the VRP. The basic idea is to do a

restricted search in the solution space. The rule of defining the search region and the mechanism to conduct the search are critical to the performance of a heuristic. In this section, some classical heuristic algorithms will be introduced. At the same time, a relatively newly formed group, called metaheuristics, will be reviewed as well.

2.3.1 Classical heuristics

Algorithms in this group usually take two strategies to achieve the satisfactory solution. One is the construct-then-improve. Initial routes will be constructed first, and then an improvement process is applied to get a better solution. The construction of routes often takes types of insertion, and the improvement process usually is done by exchanging customers' positions in routes. The other strategy is the cluster-then-route. The set of customers will first be divided into clusters, and then a route will be formed for each cluster. The core of this strategy is the clustering, since methods for TSP can be used to obtain optimal route for each cluster.

2.3.1.1 Construct-then-improve methods

Clarke and Wright 's (1964) savings algorithm is probably the most famous heuristic for the VRP. In this algorithm, initially, each customer is assumed to form one route separately. For each pair of customers, the saving in cost is calculated by assuming merging two single customer routes into one route. Then routes will be merged according to the savings between the first customer in one route and the last customer in the other route. Or (1976) introduces the Or-opt exchange method. In this method, a sequence of up-to-three adjacent nodes is removed and is inserted at another location

within the same route. This exchange method can be regarded as a special type of 3-opt exchange. Paessens (1988) uses one parametrical savings function to improve the result. Van Landeghem (1988) extends the savings algorithm for the VRP with time windows by incorporating an intuitive view of time influence on route building. Desrochers and Verhoog (1991) present a savings heuristic based on successive route fusion. A weighted matching problem is solved at each iteration to select the best fusion. Potvin and Rousseau (1993) build the routes in parallel and use a generalized regret measure over all unrouted customers to pick up the next candidate for insertion. Potvin and Rousseau (1995) compare different exchanging heuristics for the VRP with time windows. In their study, a new 2-opt exchange method is introduced and a hybrid approach, based on Or-opt and 2-opt, has been shown to be particularly successful. Thangian, Potvin, and Sun (1996) study the vehicle routing problem with backhauls and time windows. A route construction heuristic is described and several different local search heuristics are used to improve the initial solutions. Vigo (1996) proposes a heuristic algorithm that starts with an infeasible solution and determines the final set of routes through an insertion procedure as well as an intra-route and inter-route arc exchange procedure. Mosheiov (1998) constructs routes by breaking a TSP tour into disjoint segments. Basnet, Foulds and Wilson (1999) modify the savings algorithm by taking the tree structure of one special VRP into consideration. Salhi and Nagy (1999) study the case in which the backhauling is used. Their heuristic constructs the routes by first forming routes for linehaul customers only, then inserting backhaul customers in a cluster manner.

2.3.1.2 Cluster-then-route methods

Gillett and Miller 's (1974) sweep algorithm is a classical cluster first route second algorithm. Customers are first assigned to clusters according to their polar coordinates. Capacity constraint is used to decide if a cluster is full of customers or not. Then the final vehicle routes are obtained by solving the corresponding TSP exactly or approximately. Fisher and Jaikumar (1981) relax the original VRP to one generalized assignment problem. The solution to the generalized assignment problem is used to form clusters of customers. The final routes are constructed by sequencing customers in each cluster into one route through a TSP algorithm. Chung and Norback (1991) apply this cluster-then-route strategy for one large food distributor. A cluster and insertion heuristic is developed to obtain satisfactory solutions. Baker and Sheasby (1999) extend the generalized assignment heuristic, following Fisher and Jaikumar (1981), and do an analysis on the relationship between the solution to the VRP and the solution to the generalized assignment problem. Hong and Park (1999) develop a heuristic for a bi-objective VRP with time windows. The heuristic algorithm consists of a parallel insertion method for clustering, and a sequential linear goal programming procedure for routing. Liu and Shen (1999) consider a different type of clustering. Instead of constructing clusters according to the relationship among customers, based on a reasonable conjecture that a customer holds a higher probability of being served by nearby routes than by farther routes, clusters are formed by considering the relationship between route and customer. Toth and Vigo (1999) use Lagrangian relaxation to form clusters, and then apply one matching step and a routing step to solve the VRP with backhauls.

2.3.2 Metaheuristics

Like classical heuristic algorithms, metaheuristics still attempt to search in a restricted area of the solution space. The relationship among the customers, and hence the searching region, are defined on an abstract level. And some search mechanisms are borrowed from other disciplines, such as biology and physics. In the current literature, the popular metaheuristics include tabu search, neural networks, simulated annealing, and evolutionary metaheuristics.

2.3.2.1 Tabu search

Tabu search is a method that is very effective in avoiding being trapped in local optimal. Given the current solution, tabu search will find the next solution from a set of solutions called neighborhood. It is not necessary for the objective to be improved. Hence it is possible to leave the local optimal. To avoid cycling, solutions that are recently examined are forbidden to be selected for a certain number of iterations. These solutions are called tabu, and they will form a tabu list. After a solution is selected, this solution will be added to the end of the tabu list, and the first element in the tabu list will be removed and becomes non-tabu. In some studies, the whole neighborhood is explored and the best non-tabu move is selected; in some other studies, the first feasible non-tabu-improving move is selected. A solution in the tabu list is legally selected if it satisfies the aspiration criterion. The aspiration criterion is a measure solely designed to override the tabu status of one move if this move leads to a solution better than the best found solution so far. The strategies about search scope and direction can be classified into two groups.

Intensification strategies refer to those strategies that do intensive search in a promising region of the solution space; diversification strategies, on the other hand, are used to broaden the search into less explored regions. Some recent research papers related to this topic are Garcia, Potvin, and Rousseau (1994), Gendreau, Hertz, and Laporte (1994), Rego and Roucairol (1995), Gendreau, Laporte, and Seguin (1996), Renaud, Laporte, and Boctor (1996), Brandao and Mercer (1997), Golden, Laporte, and Taillard (1997), Rego (1998), Barbarosoglu and Ozgur (1999), Gendreau, Laporte, Musaraganyi, and Taillard (1999), Cordeau and Laporte (2001), Cordeau, Laporte, and Mercier (2001).

2.3.2.2 Neural networks

There are two types of neural networks used in the literature of VRP and TSP. One is the Hopfield networks (Hopfield, 1982, 1984) and the other is the self-organizing neural networks (Kohonen, 1982). Hopfield and Tank (1985) introduce the application of the neural networks into the field of TSP. In their network, each neuron indicates the combination of a customer and a position. Given a TSP with n customers, there will be n^2 neurons. With carefully selected threshold value and weights, neurons can change their status and the objective function based on cost items and neurons' status will be decreased correspondingly. The final network configuration can be used to construct the actual TSP route. However, this approach is not popular in the literature because of its complexity, excessive memory and computational requirements, infeasibility, and local minima effect. While Hopfield networks try to configure themselves from the problem data, the self-organizing neural networks use unsupervised learning to form routes. At the beginning of the self-organizing process, several rings consisting of nodes will be

generated. Then the problem data will be repetitively presented to those nodes. For each presented data, which usually is the location of a customer, a node will be declared winner according to a certain measurement. This node and those nodes in its neighborhood will adjust their positions, intending to be pulled toward the location of the presented customer. Finally, all customers can be found in or near one ring and those rings represent the actual routes. Some recent studies in self-organizing neural networks can be found in Toure, Rabelo, and Velasco (1993), Torki, Somhon, and Enkawa (1997), Somhom, Modares, and Enkawa (1999) and Soylu, Ozdemirel, and Kayaligil (2000).

2.3.2.3 Simulated annealing

Simulated annealing was first introduced to model the physical annealing of solids. Later on, this method was used to deal with optimization problem. In the analogy, the different states of substance correspond to different feasible solutions to the combinatorial optimization problem, and the energy of the system corresponds to the function to be minimized. A typical simulated annealing process needs to first set up an initial value for the objective function, then continuously reduce this objective function by moving to another feasible solution. Usually, the algorithm needs to specify the reduction rate, the number of repetition, and the stopping criterion. Alfa, Heragu, and Chen (1991) develop a solution procedure for the VRP by combining simulated annealing with the 3-opt procedure. Breedam (1995) reports an improvement heuristic based on simulated annealing. Ree and Yoon (1996) use a simulated annealing algorithm to provide solutions to a newspaper delivery problem.

2.3.2.4 Evolutionary metaheuristics

Genetic Algorithms, along with a similar technique called Evolutionary Strategies, form the class of evolutionary metaheuristics. Genetic Algorithms are stochastic search methods managing a population of simultaneous search positions. A conventional genetic algorithm consists of three essential elements: a coding of the optimization problem, a mutation operator, and a set of information exchange operators. The coding of the optimization problem produces the required discretization of the variable values and makes their simple management in a population of search points possible. The mutation operator determines the probability with which the data structures are modified. The information exchange operators control the recombination of the search points in order to generate a new, better population of points at each iteration step. Evolutionary strategy is different in that there is no encoding of individuals. The evolution process is simulated directly on the level of problem solution, not coded individuals. Recent research of the application of evolutionary metaheuristics in the VRP can be found in Malmborg (1996), Homberger and Gehring (1999), Chen, Hay, and Ke (2001), and Gehring and Homberger (2001).

Table 2-1 Classification of VRP

Element	Sub-element	Options
Objective Function		
	Cost	Traveling cost, penalty cost, vehicle cost
	Function type	MinSum, minMax
Depot		
	Number	1, multiple
	Type	Regular, satellite
	Capacity	Limited, unlimited
Customer		
	Location	Edge, node
	Service type	Delivery, pick-up
	Demand	Deterministic, stochastic
	Temporal	Time windows
Vehicle		
	Fleet size	Fixed, upper bound, lower bound
	Vehicle type	Homogeneous, heterogeneous
	Capacity	Limited, compartment
	Temporal	Time window, route duration
	Serving strategy	Demand split, backhauling, multiple shifts
Relationship		
	Customer to customer	Precedence, origin-destination, inclusion, exclusion
	Customer to vehicle	Customer-vehicle binding
	Customer to depot	Customer-depot binding
	Vehicle to vehicle	Vehicle synchronization
	Vehicle to depot	Vehicle-depot binding

Chapter III

MODELING SERVICE LEVEL BASED VEHICLE ROUTING PROBLEM

3.1 Basic models

Waiting time is used as the measurement for the service level. We define the waiting time of a customer as the time between the time point the vehicle leaves the depot and the time point the vehicle reaches this customer. Note that the actual waiting time of a customer is the time between the time point the customer places an order and the time point this customer receives service. The difference between our definition and the actual waiting time is that we have ignored the time period from placing the order to dispatching the vehicle. This difference will have no effect on optimization since the ignored time periods are constant terms in our objective function. For example, if a customer places an order at time minus 10 and we start to dispatch a vehicle at time 0, the actual waiting time is our waiting time plus 10 and the objective function can be rewritten as the original objective function plus 10. Since 10 is a constant term, the optimization process will be the same as before.

If assuming vehicles have constant speed, the waiting time of a customer will be proportional to the traveled distance of the arrived vehicle. In this study, we will directly use the traveled distance as the representative of the waiting time. Our objective will be minimizing the total waiting time of all customers.

For our basic models, we have the following assumptions:

a. There is one depot.

b. There are n customers.

c. There are m vehicles.

d. All vehicles originally stay at the depot.

e. All vehicles are identical.

f. There is no capacity limit for the vehicles, which implies either that customers only request pure service or that the demand is so small that the capacity is large enough for any possible route.

g. A route is defined as the traveling of a vehicle that starts from the depot, visits some customers, and then returns to the depot.

h. One vehicle serves one and only one route.

i. Each customer can be served by any one of those m vehicles.

j. The service time is negligible.

The first model is a three-index model. We use the decision variable x_{ijk} to denote if the kth vehicle first visits the ith node, then visits the jth node without stopping at any other nodes. Here, the set of nodes includes the depot and the customers. The depot is node 0 and the customers are node 1 to node n. Decision variable d_{ki} is used to record the waiting time of the ith node if the kth vehicle visits this node. The decision variable d_{k0} needs special treatment. The kth vehicle will be at the depot twice since it not only starts from the depot but also returns to the depot. Because we assume all vehicles originally stay at the depot, only the returning is worthy of being modeled; therefore, we let d_{k0} record the total traveled distance of the kth vehicle. Following is the first model.

Notation

Customer set C={1, 2, ..., n}

Depot set V={0}

Node set N=C∪V

$x_{ijk}=\begin{cases}1 & \text{if the kth vehicle visits the jth node directly after visiting the ith node}\\ 0 & \text{otherwise}\end{cases}$

d_{ki} the traveled distance of the kth vehicle when it reaches ith customer

d_{ko} the traveled distance of the kth vehicle when it returns the depot

c_{ij} the distance between the ith node and the jth node

D a very large positive number

Model I

Min

$$\sum_{i=1}^{n}\sum_{k=1}^{m} d_{ki} \qquad (1)$$

s.t.

$$\sum_{j=1}^{n} x_{0jk} = 1 \qquad k = 1, 2, \ldots, m \qquad (2)$$

$$\sum_{j=1}^{n} x_{j0k} = 1 \qquad k = 1, 2, \ldots, m \qquad (3)$$

$$\sum_{k=1}^{m}\sum_{j=0, j\neq i}^{n} x_{ijk} = 1 \qquad i = 1, 2, \ldots, n \qquad (4)$$

$$\sum_{k=1}^{m}\sum_{i=0, i\neq j}^{n} x_{ijk} = 1 \qquad j = 1, 2, \ldots, n \qquad (5)$$

$$\sum_{j=0, j\neq i}^{n} x_{ijk} - \sum_{j=0, j\neq i}^{n} x_{jik} = 0 \qquad i = 1, 2, \ldots, n$$

$$k = 1, 2, \ldots, m \qquad (6)$$

$$d_{ki} \geq c_{0i} * x_{0ik} \qquad i = 1, 2, \ldots, n$$

$$k = 1, 2, \ldots, m \qquad (7)$$

$$d_{kj} - d_{ki} + D \geq (c_{ij} + D) * x_{ijk} \qquad i = 1, 2, \ldots, n$$

$$j = 0, 1, 2, \ldots, n, \quad j \neq i$$

$$k = 1, 2, \ldots, m \qquad (8)$$

Each customer i is associated with a set of traveled distance variable d_{ki}, k=1, 2, ..., m. As mentioned before, variable d_{ki} represents the traveled distance of vehicle k while it visits customer i. Since only one vehicle visits customer i, only one out of those m variables should have meaningful value. In the model, we have constraints enforcing that variable to take the value of the actual traveled distance. No constraint is imposed on other variables. Since this is a minimization model, other variables will take the value 0. Note that in this study we directly use distance to represent time. So the waiting time of customer i can be expressed as $\sum_{k=1}^{m} d_{ki}$ and the objective function (1) is the summation of each customer's waiting time. Constraints (2) and (3) make sure that each vehicle leaves and returns to the depot. Constraints (4), (5) and (6) guarantee that each customer is visited by a vehicle and that this vehicle leaves the same customer. Constraints (7) and (8) define the traveled distance when a vehicle reaches a node. Here, a node can be a customer or the depot. Constraint (7) sets the value of d_{ki} if customer i is the first customer in a route. If so, suppose the visiting vehicle is the kth vehicle, variable x_{0ik} equals 1, and constraint (7) becomes $d_{ki} \geq c_{0i}$. This constraint will force d_{ki} to be no less than c_{0i}, the distance between depot and customer i. Since this is a minimization model, d_{ki} will take the value of c_{0i}. In other cases, i.e., customer i is not the first one in a route or vehicle k does not visit customer i, and constraint (7) only forces d_{ki} to be nonnegative. Constraint (8) sets the value of d_{kj} if customer j is not the first one in a route. Suppose trip from i to j happens and the vehicle traveling this trip is the kth vehicle; therefore, variable x_{ijk} equals 1 and constraint (8) becomes $d_{kj} - d_{ki} \geq c_{ij}$. The traveled distance of j will be updated as the traveled distance of i plus the distance between i and j. Since we

have already set the traveled distance of the customers who are the first ones in a route, the traveled distance of other customers can be updated according to constraint (8). In other cases, variable x_{ijk} equals to 0 and constraint (8) becomes $d_{kj} - d_{ki} + D \geq 0$. This constraint will always be satisfied since D is a very large positive number.

The above model is a three-index model. Three-index variables can show detailed information about which vehicle visits which customer. This detailed information is very useful in modeling complex vehicle routing systems. For our basic case, we can develop a two-index model, which has less size and can be solved more efficiently.

In the two-index model, our decision variables will be x_{ij}, which will be 1 if the customer i is directly ahead of the customer j to be served by the same vehicle. Even though we do not explicitly mention which vehicle visits which customer, the model can guarantee there will be one vehicle in and one vehicle out for each customer, implying that the incoming vehicle is the same as the outgoing vehicle. Another group of variables is d_i, which records the traveled distance when one vehicle visits the customer i. Following is the detailed model.

Model II

$$\text{Min} \sum_{i=1}^{n} d_i \qquad (9)$$

s.t.

$$\sum_{j=1}^{n} x_{0j} = m \qquad (10)$$

$$\sum_{j=1}^{n} x_{j0} = m \qquad (11)$$

$$\sum_{j=0, j\neq i}^{n} x_{ij} = 1 \qquad i = 1, 2, \ldots, n \qquad (12)$$

$$\sum_{i=0, i\neq j}^{n} x_{ij} = 1 \qquad j = 1, 2, \ldots, n \qquad (13)$$

$$d_i \geq c_{0i} \qquad i = 1, 2, \ldots, n \qquad (14)$$

$$d_j - d_i + D \geq (c_{ij} + D) * x_{ij} \qquad i = 1, 2, \ldots, n$$
$$j = 1, 2, \ldots, n \qquad (15)$$

The objective function (9) is still the summation of the waiting time of each customer. Instead of using a set of traveled distance variables d_{ki} to record the waiting time of the ith customer as in the first model, we have only one traveled distance variable d_i for the ith customer in this model. Constraint (10) makes sure there are m vehicles that leave the depot and constraint (11) makes sure that those m vehicles return to the depot. Constraint (12) guarantees that each customer is visited by a vehicle, and constraint (13) guarantees there is a vehicle that leaves each customer. These two constraints imply that the vehicle that visits a customer is the same as the vehicle that leaves this customer. Constraint (14) sets up a lower bound for each traveled distance variable d_i. This lower bound is set as the traveled distance from the depot to the location of the ith customer. If the ith customer is the first one in a route, this constraint assigns d_i exactly what it should have, the distance between the depot and the customer. If the ith customer is not the first one in a route, this constraint is still valid because of the triangle inequality. Constraint (15) sets the value of traveled distance variables for customers who are not the first ones in the routes. If a trip from i to j happens, the value of d_j is set as d_i plus the distance between i and j.

The above two models use traffic flows to model the vehicle routing system. The solution will be several chains of nodes. Each chain represents a route. We need to decode those chains to get each customer's position in the route. In the third model, we take a different approach. Instead of modeling the traffic flow, we construct the model by directly tackling the customers' positions.

Table 3-1 is an entity vs. position table that can be used to demonstrate our approach to directly tackle the customers' positions. Corresponding to each cell (i, k), a

decision variable x_{ik} is used to denote if the ith entity takes the kth position. The rows represent the entities. We have n customers and construct an artificial depot for each one of the m vehicles. The customers are indexed from 1 to n and the artificial depots are from n+1 to n+m. The columns represent the positions. The m*n columns can be broken into m disjoint segments. Each segment corresponds to a route and consists of n consecutive columns. We prepare n positions for a route because of the following reason. If we do not consider the single vehicle case, which actually is the traveling salesman problem (TSP), a route can have at most n-1 customers; otherwise, the other routes will be empty. Although it is obvious that each route starts from and returns to the depot, we will still reserve one position for the artificial depot, which will be useful in the modeling. So a route will be represented by up to n-1 customers and one artificial depot and we need to prepare n positions for the most crowded route.

Each artificial depot needs to be assigned a position indicating the end of the route. The m*n positions have been divided into m segments. For convenience, we assume that the jth artificial depot will be assigned a position out of the jth segment. The assigned position cannot be the first one in the segment. If we do so, since the depot is the last stop of the vehicle, we construct an empty route. After the artificial depot is assigned a position, the number of customers in this route is determined. If we write the depot position index as (k-1)*n+j, where k shows the kth segment or kth vehicle and j is the actual position in the route, the number of positions assigned for the customers in this route will be j-1. If we call this j-1 as the valid number of positions for this route, the total valid number of positions should be n since we have n customers. We can guarantee that each customer be assigned a valid position by making sure that each customer takes a

position, each position is assigned to no more than one entity, and in each segment of positions, no position after the associated artificial depot's position is assigned to the entities.

The objective function will be defined over individual cost items. Each cost item is the length of a trip from one entity to another entity. The lengths of the trips at the beginning part of the route will be counted more times than the lengths of the trips at the ending part of the route. For example, given one route depot 0 - customer 1 - customer 2 - customer 3 - depot 0, there will be three trips involved in the calculation of the waiting time. The lengths of those three trips are c_{01}, c_{12}, and c_{23}. The waiting time of the customer 1 is c_{01}. The waiting time of customer 2 is $c_{01}+c_{12}$. And the waiting time of customer 3 is $c_{01}+c_{12}+c_{23}$. Therefore, the total waiting time is $3c_{01}+2c_{12}+c_{23}$. The weight of the length of a trip depends on the total number of customers in the route and the positions of the entities associated with this trip. Suppose the total number of customers in the route is T and the position of the ending entity of the trip is t. The weight can be calculated as T-t+1. The t can be figured out from the local information, but we need global information to obtain the T. In the above-mentioned example, for the trip from the depot to the customer 1, the available local information is only that x_{11} equals to 1 if assuming that route is the first route. We can use that local information to figure out that c_{01} should be included in the objective function and the position of the ending entity is 0. To figure out the T, assuming the artificial depot is the fourth entity, we need to know x_{44} equals to 1, which is a piece of global information. We do not want to include global information in the model because the modeling will become difficult. To overcome this obstacle, we observe that the T-t+1 of a trip is exactly the same as the position of the

beginning entity of the trip in the reversed route. Since the position of an entity is local information, we are able to calculate the total waiting time of the customers in a route through the reversed route. For the route depot 0 - customer 1 - customer 2 - customer 3 - depot 0, we can write down $c_{12}+2c_{23}+3c_{30}$ based only on local information, which is the total waiting time for the route depot 0 - customer 3 - customer 2 -customer 1 - depot 0. So we can write down the objective function in this way as long as we keep in mind that the calculated total waiting time is for the reversed routes.

The validity of our objective function can be explained as follows. A feasible solution consists of m routes. If we reverse each route, we get a different feasible solution. These two solutions will be called the original solution and the reversed solution respectively in the following text. Note that if we reverse those m routes twice, we get back to the original solution. Note also that the reversed solution is unique to the original solution and vice versa. If we formulate the objective function as described before, our optimal solution is a feasible solution, denoted by O_1, whose reversed solution has the shortest total waiting time. We can claim that this reversed solution, denoted by R_1, is the feasible solution with shortest total waiting time, i.e. the optimal solution we are looking for. If not, suppose that there is a different feasible solution, denoted by R_2, with shorter total waiting time. Since R_2 can be viewed as the reversed solution of a feasible solution, denoted by O_2, O_2 should be better than O_1 in terms of our objective function, which is a contradiction. So R_1 is the feasible solution with the shortest total waiting time.

Conceptually, the major concern for our treatment of the objective function is that it seems the constraints define a feasible solution, but the objective function is for a

different feasible solution. This concern is not necessary because of the following. Given is a set of decision variables $\{x_{ik}, i=1, 2, \ldots, n+m, k=1, 2, \ldots, n*m\}$. If it defines a feasible solution, it implicitly defines the reversed solution at the same time. Since each feasible solution can be viewed as the reversed solution of an original solution, the set of all reversed solutions equals the feasible solution set. Therefore, we are able to say that the constraints define a feasible solution and the objective function is the total waiting time of that feasible solution. Our optimal solution out of the optimization process is just the feasible solution with the shortest total waiting time. While we decode the decision variables, we need to keep in mind that we want to get the reversed solution implicitly defined by those decision variables. Typically, the position k is behind the position k+1. For example, if decision variables x_{11}, x_{22}, x_{33} and x_{44} are equal to 1, where entities 1, 2, and 3 are customers and entity 4 is the artificial depot, we need to decode those decision variables as the route 0 - 3 - 2 - 1 - 0. In our basic situation, both the original solution and the reversed solution are feasible if either one is feasible; therefore, we can formulate the constraints with the intention to make either solution feasible. If we add some side constraints, such as the time windows and the capacity constraints, the feasibility may not be held for both solutions at the same time. Since the objective function is formulated as the total waiting time of the reversed solution, we need to make sure that the reversed solution is feasible.

Based on the above discussion, we can construct the following model.

Notation

Customer set C={1, 2, ..., n}

Depot set V={n+1, ..., n+m}

Entity set E=C ∪ V

Position set P={1,2, ..., n, n+1, n+2, ..., n+n, ..., (m-1)*n+1, (m-1)*n+2, ..., (m-1)*n+n}

Node set N=E × P

c_{ij} the traveling distance between ith entity and jth entity

$$x_{ik}=\begin{cases}1 & \text{if ith entity is located at kth position}\\ 0 & \text{otherwise}\end{cases}$$

Model III

$$Min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{m}\sum_{l=1}^{n-1} 1 * c_{ij} * x_{i,(k-1)*n+l} * x_{j,(k-1)*n+l+1} \qquad (16)$$

s.t.

$$\sum_{l=2}^{n} x_{n+k,(k-1)*n+l} = 1 \qquad k = 1, 2, \ldots, m \qquad (17)$$

$$x_{n+k,(k-1)*n+1} = 0 \qquad k = 1, 2, \ldots, m \qquad (18)$$

$$\sum_{j=1, j\neq k}^{m}\sum_{l=1}^{n} x_{n+k,(j-1)*n+l} = 0 \qquad k = 1, 2, \ldots, m \qquad (19)$$

$$\sum_{k=1}^{m}\sum_{l=2}^{n}(l-1)x_{n+k,(k-1)*n+l} = n \qquad (20)$$

$$\sum_{j=1}^{n*m} x_{ij} = 1 \qquad i = 1, 2, \ldots, n \qquad (21)$$

$$\sum_{i=1}^{n+m} x_{ij} \leq 1 \qquad j = 1, 2, \ldots, n * m \qquad (22)$$

$$\sum_{i=1}^{n+m} x_{i,(k-1)*n+j} \leq \sum_{i=1}^{n+m} x_{i,(k-1)*n+j-1} \qquad j = 2, 3, \ldots, n$$

$$k = 1, 2, \ldots, m \qquad (23)$$

Objective function (16) is nonlinear and it is calculated as the total waiting time of the reversed routes. Constraint (17) assigns one position for each artificial depot. Constraint (18) makes the empty route impossible. Constraint (19) prevents the artificial depots from taking positions in position segments other than the associated position segment. Constraint (20) guarantees there are n valid positions. If an artificial depot takes the *l*th position in its associated segment, there are *l*-1 valid positions in that segment. Constraint (20) just sets the total number of valid positions to be the number of customers. Constraint (21) is the row constraint, which assigns a position for each customer. Constraint (22) is the column constraint, which makes sure that each position is assigned to at most one entity. Constraint (23) guarantees that no invalid positions are assigned to the customers and the detailed reasoning is discussed next. A position is invalid if it is behind the position taken by the artificial depot. The summation of all decision variables in one column, called column value, is either 1 or 0. For a feasible route, the column values should be {1, 1, ..., 1, 0, ..., 0}. The last 1 is the column value for the position taken by the artificial depot. Since the total number of assigned positions are fixed to n+m, if an invalid position is assigned to an entity, some valid position will not be assigned. Therefore, the column values will be something like {1, ..., 1, 0, 1, ..., 1, 0, ..., 0}. The chain of 1s is interrupted by 0s, which is prevented by constraint (23).

3.2 Varieties of basic models

In the previous section, we have developed three mathematical models for the basic case of the SLB-VRP. Since flexibility is an essential feature we want the SLB-VRP to have, we will discuss some varieties of the basic models in this section.

Although we formulate the objective function as the minimization of the total waiting time in our basic models, we should prepare for the situation that the vehicle routing system has a different objective. In this section, we will discuss how to model two other common objectives in the literature of the VRP: the minimization of the total traveled distance and the minimization of the maximum waiting time. In practical situations, the vehicle routing system usually has some side constraints. The most popular two are the time windows and the capacity constraints. We will also introduce the modeling techniques for these two constraints in this section.

3.2.1 The minimization of the total traveled distance

In Model I, we have variable d_{ki} that records the traveled distance of the kth vehicle when it visits the ith node. Since d_{k0} is the traveled distance of the kth vehicle when it returns to the depot, we can easily model the total traveled distance as $\sum_{k=1}^{m} d_{k0}$.

In Model II, we do not explicitly specify which vehicle serves which customer; therefore, we cannot formulate the minimization of the total traveled distance as we have done in Model I. Since the variable d_i represents the traveled distance for the vehicle when it visits the ith customer, as long as we can figure out the last customer in a route, we are able to calculate the length of that route and then obtain the total traveled distance. In order to achieve this goal, we need to introduce a set of nonnegative variables p_i, i=1, 2, ..., n. By imposing the constraint $p_i - d_i + D \geq (D + c_{i0}) * x_{i0}$, if the ith customer is the last customer in the route, x_{i0} equals 1 and p_i will be d_i+c_{i0}, which is the length of this route. The minimization of the total traveled distance is $\sum_{i=1}^{n} p_i$.

In Model III, it is easier to model the total traveled distance than the total waiting time since each trip will be counted only once to calculate the total traveled distance. For a trip started from node i and ended at node j, the length is c_{ij} and the contribution of this trip to the objective is $c_{ij} * x_{i,k} * x_{j,k+1}$. This expression works for all trips except those that are the first trip in a route. For those trips, since we implicitly assume each route starts from the depot, it is not necessary to use the decision variable x_{0i}. The only decision variable we need is the one that attaches with the ending node of the trip, which is the first customer in that route. Suppose the ith customer is the first customer visited by a vehicle, the contribution of the first trip to the objective is simply $c_{0i} * x_{i,(l-1)n+1}$, where l belongs to the set {1, 2, ..., m} and is used to specify which route the ith customer is in. If we add all of these contributions together, we have the total traveled distance $\sum_{i=1}^{n-m}\sum_{j=1}^{n-m}\sum_{k=1}^{n*m-1} c_{ij} * x_{i,k} * x_{j,k+1} + \sum_{i=1}^{n}\sum_{l=1}^{m} c_{0i} * x_{i,(l-1)*n+1}$.

3.2.2 The minimization of the maximum waiting time

In Model I and Model II, we have variables that record the waiting time of each customer; therefore, the minimization of the maximum waiting time can be implemented as minimizing the maximum value of those variables. In Model I, the variable d_{ki} stores the waiting time of the ith customer if the kth vehicle visits this customer. We can introduce an additional variable d and impose constraints

$d \geq d_{ki}$ k = 1, 2, ..., m; i = 1, 2, ..., n. If we use the objective *Min d* in the model, d will be the maximum waiting time. In Model II, the variable d_i records the waiting time of the

ith customer, and we can implement the minimization of the maximum waiting time as $Min\ d$, along with the constraints $d \geq d_i \quad i = 1, 2, ..., n$.

In Model III, we have variables that represent the entity-position relationship. We need to first figure out the waiting time of the last customer in each route, which definitely is the maximum waiting time of all customers in the same route, and then find out the overall maximum waiting time out of the waiting times we have obtained at the first step. For the *l*th route, where *l*=1, 2, ..., m, the waiting time of the last customer is $\sum_{i=1}^{n+m}\sum_{j=1}^{n+m}\sum_{k=1}^{n-1} c_{ij} * x_{i,(l-1)n+k} * x_{j,(l-1)n+k+1}$. If we denote the previous expression as d_l and introduce an additional variable *d*, we can formulate the minimization of the maximum waiting time as $Min\ d$, along with the constraints $d \geq d_l \quad l = 1, 2, ..., n$.

3.2.3 The time windows

The time windows refer to the temporal constraints that the customers may require the service be provided in some specified time intervals. Since in this study we directly use traveled distance to represent the traveling time, in the following text we will directly use the traveled distance variables to model the time windows. For a given customer i, suppose the time interval is $[T_s, T_e]$, the assigned vehicle is required to visit this customer no earlier than T_s and no later than T_e. Since the variable d_{ki} records when the kth vehicle will visit the ith customer, in Model I, we can add constraints $d_{ki} \geq T_s$ and $d_{ki} \leq T_e$, where k=1, 2, ..., m, for the ith customer to impose this constraint. Similarly, in Model II, we add constraints $d_i \geq T_s$ and $d_i \leq T_e$. In Model III, we need to introduce an additional set of variables $\{d_{ik}, i=1, 2, ..., n+m, k=1, 2, ..., n*m\}$. The variable d_{ik}

records the time point that the ith node is served when this node takes the kth position. If the ith node is the depot, which means $i \in [n+1, n+m]$, d_{ik} is set to zero since each route starts from the depot. We can then figure out each d_{ik} in a propagation manner by imposing the constraints

$d_{i,k} - d_{j,k+1} + D \geq (c_{ij} + D) * x_{i,k} * x_{j,k+1}$, i ∈ [1, n], j ∈ [1, n + m], k ∈ [1, n * m - 1], where D is a large positive number. To understand these constraints, we need to keep in mind that in Model III the actual route follows the reversed order; therefore, the k+1 position actually is before the kth position. After we obtain the d_{ik}, we can impose the time windows as we have done in Model I and Model II.

If a customer is allowed to have multiple time windows, we need to introduce a set of indication variables to specify which time window is actually effective. Suppose customer i has q time windows $[T_{s1}, T_{e1}]$, $[T_{s2}, T_{e2}]$, ..., $[T_{sq}, T_{eq}]$. A set of 0-1 integer variable $\{I_1, I_2, \ldots, I_q\}$, along with the constraint $I_1+I_2+\ldots+I_q=1$, can specify in which time window the actual service happens. If I_j equals 1, the actual service happens in the time window $[T_{sj}, T_{ej}]$. Then the multiple time windows can be modeled by replacing the constraints $d \geq T_s$ and $d \leq T_e$ with the constraints $d + D \geq I_j * (T_{sj} + D)$ and $d - D \leq I_j * (T_{ej} - D)$, where d, I_j, T_{sj}, T_{ej} are the waiting time variable, the indication variable, and the start point of a time window and the end point of a time window respectively. The subscription needs to be added to the variables according to the actual model.

3.2.4 The capacity constraint

When the quantity issue is addressed in the requested service, we usually need to consider the capacity of the vehicles. If the vehicle will pick up something from the customer, we need to make sure that there is enough room in the vehicle to put what we will pick up. If the vehicle will deliver something to the customer, we need to guarantee there is something in the vehicle to be delivered. In other words, the requested resource should not be used up in the previous deliveries. To impose the capacity constraint, we need to introduce a set of variables to record the capacity status of the vehicles. In Model I, we use the variable q_{ik} to represent the capacity of the kth vehicle after it serves the ith customer. The q_{0k} will be set to the initial capacity of the kth vehicle and the values of the other variables will be set by the constraint

$q_{jk} - q_{ik} = p_j * x_{ijk}, i \in [0, n], j \in [1, n], k \in [1, m]$, where p_j is the service quantity requested by the jth customer. In the pickup case, p_j is a positive number, and in the delivery case, p_j is a negative number. If the trip from customer i to customer j is taken by the kth vehicle, x_{ijk} is 1 and we can see that the capacity of the kth vehicle is correctly updated form q_{ik} to $q_{ik}+p_j$. If that trip is not taken by the kth vehicle, x_{ijk} is 0 and the q_{jk} is set meaninglessly to q_{ik}. After we define these variables, we can impose the capacity constraint. Two typical cases will be that the capacity should not be negative and that the capacity should not exceed a specified maximum level. We can use $q_{ik} \geq 0$ and $q_{ik} \leq Q$, where Q is the specified maximum capacity level, to enforce these constraints. In Model II, we do not need to specify which vehicle the capacity variable represents since Model II is a two-index model. We will use q_i to denote the capacity of the vehicle immediately

after it servers the ith customer, and which vehicle serving the ith customer is not specified. Since q_0 represents the capacity of the vehicle when it stays at the depot, this variable should be set to the initial capacity of the vehicle. In case we have a heterogeneous fleet and the vehicles have different initial capacity, we may need to introduce artificial depots and the variable attached with those artificial depots that can be set to those different initial capacities. The other variables will be set by the constraint $q_j - q_i = p_j * x_{ij}, i \in [0, n], j \in [1, n]$. In Model III, we define a capacity variable for each entity the same way as in Model II. Since the depot and the artificial depots are indexed from n+1 to n+m, q_i, i=n+1, ..., n+m, will be set to the initial capacities of those vehicles, and the other variables will get their values by the constraint $q_j - q_i = p_j * \sum_{k=1}^{n+m-1} x_{jk} x_{i,k-1}, i \in [1, n+m], j \in [1, n]$. Note here that we are modeling the reversed route. In Model II and Model III, after we set the values of the variable q_i, we can impose the capacity constraints as we have done in Model I.

3.3 Difference between SLB-VRP and VRP

We have constructed several models for the SLB-VRP. One concern is whether it is worthwhile to do so. In other words, we want to know if there is a significant difference between the solution to the traditional VRP and the solution to the SLB-VRP. To answer this question, we test 10 problems. Each problem has 1 depot, 8 customers, and 2 vehicles. We set the depot's location at the origin, and randomly assign each customer's X and Y coordinates from a uniform distribution with 0 and 100 as ending points. After we get those locations, we then calculate their pairwise Euclidean distances. The Euclidean distance is used as the traveling distance. Two models are constructed for

each problem. One is a two-index SLB-VRP model. The other one is a two-index VRP model. We use CPLEX to solve these models. After we get the optimal solution of a model, we evaluate this solution by the alternative objective function as well. If there is no much difference between the total waiting time of the optimal solution to the VRP model and the optimal solution to the SLB-VRP model, it is not necessary to apply the SLB-VRP model for the vehicle routing system originally constructed using the VRP model. If the optimal solution to the SLB-VRP model has almost the same total traveled distance as the optimal solution to the VRP model, we may be able to claim that the SLB-VRP model can substitute the VRP model even when the vehicle routing system should be operated with the objective of minimizing the total traveled distance.

Example 3-1 Suppose the depot is located at the origin (0, 0) and 2 vehicles are used to serve 8 customers. The locations of customers are specified in Table 3-2 and the traveling distance matrix is in Table 3-3.

The optimal solution to the SLB-VRP model is:

Route 1 0 - 8 - 1 - 6 - 7 - 0

Route 2 0 - 5 - 4 - 2 - 3 - 0.

The total waiting time of this solution is 614.51 and the total traveled distance is 455.78. The optimal solution to the traditional VRP model is:

Route 1 0 - 8 - 0

Route 2 0 - 1 - 7 - 6 - 2 - 3 - 4 - 5 - 0.

The total waiting time of this solution is 1300 and the total traveled distance is 345.16. If this vehicle routing system should be operated with the objective of minimizing the total traveled distance, we need to apply the VRP model; otherwise, the total traveled distance will be increased by 32 percent ((455.78-345.16)/345.16=0.32). If this vehicle routing system aims to improve the service level that is measured by the total waiting time, we need to use the SLB-VRP model; otherwise, the total waiting time will be damaged by 111.6 percent ((1300-614.51)/614.51=1.116).

Table 3-4 contains the computational results for the comparison between the SLB-VRP model and the VRP model. If the objective is to minimize the total waiting time, the solution of SLB-VRP model is the optimal solution. The percentage increase in total waiting time of the corresponding solution of VRP model ranges from 43.1% to 141.0% and the average is 79.1%. If the objective is to minimize the total traveled distance, the solution of VRP model is the optimal solution. The percentage increase in

total traveled distance of the corresponding solution of SLB-VRP model ranges from 5.6% to 36.8% and the average is 22.8%. In both situations, there is huge difference between the outcomes of these models; therefore, we cannot use one model to substitute for the other one. Hence, we do need to develop models for the SLB-VRP if achieving better service level is the objective.

Table 3-1 Entity-Position Matrix

			Positions						
			1 2 ... n	...	(m-1)*n +1	(m-1)*n +2	...		(m-1)*n +n
Entities	Custome	1							
		2							
		...							
		n							
	Artificial Depots	n+1							
		n+2							
		...							
		n+m							

Table 3-2 Location file for Example 3-1

Customer	X	Y
1	33.45	33.74
2	72.77	20.49
3	82.50	5.05
4	58.18	1.07
5	52.32	1.98
6	83.88	63.09
7	83.68	97.95
8	5.83	17.09

Table 3-3 Traveling distance matrix for example 3-1

	0	1	2	3	4	5	6	7	8
0		47.51	75.6	82.65	58.19	52.36	104.95	128.82	18.05
1			41.48	56.81	40.97	36.93	58.34	81.52	32.25
2				18.24	24.29	27.58	44.01	78.22	67.02
3					24.64	30.32	58.04	92.9	77.6
4						5.92	67.12	100.17	54.75
5							68.76	100.95	48.88
6								34.86	90.59
7									112.24
8									

Table 3-4 Summary of the difference between SLB-VRP and VRP

	Total Waiting Time			Total Traveled Distance		
	SLB-VRP	VRP	%↑	VRP	SLB-VRP	%↑
Problem 1	614.51	1300.00	111.6	345.16	455.78	32.0
Problem 2	723.85	1097.80	51.7	386.29	451.05	16.8
Problem 3	663.05	1066.73	60.9	340.04	459.76	35.2
Problem 4	525.23	915.76	74.4	348.49	442.10	26.9
Problem 5	718.49	1157.67	61.1	398.71	437.85	9.8
Problem 6	588.28	1151.44	95.7	361.41	458.14	26.8
Problem 7	597.43	1038.47	73.8	313.16	428.44	36.8
Problem 8	834.60	1194.46	43.1	455.24	488.16	7.2
Problem 9	595.75	1058.61	77.7	348.14	455.63	30.9
Problem	537.98	1296.76	141.0	385.82	407.34	5.6
Average	639.92	1127.77	79.1	368.25	448.43	22.8

Chapter IV

AN ALGORITHM FOR SERVICE LEVEL BASED VEHICLE ROUTING PROBLEM

In this chapter, we develop a Hopfield networks based heuristic algorithm to solve the SLB-VRP. This chapter is organized as follows. In Section 4.1, we make an introduction to the Hopfield networks and discuss the application of the Hopfield networks to the TSP. In Section 4.2, we extend the application of the Hopfield networks to the SLB-VRP. We provide several procedures in Section 4.3 to improve our Hopfield networks based algorithm. An overview of the proposed algorithm is presented in Section 4.4.

4.1 Introduction to the Hopfield networks

The Hopfield networks can be used to optimize objective function having the following form: $-\frac{1}{2}\sum_i\sum_j w_{ij}x_i x_j + \sum_i \theta_i x_i$, where x_i is a 0-1 integer variable. Usually this function is called the energy function. The network is organized as follows. Corresponding to each variable x_i, a neuron is built. The weight between the ith neuron and the jth neuron is w_{ij} and the threshold value of the ith neuron is θ_i. At each step, one neuron will be chosen randomly. Then the activation is calculated. Suppose neuron i is the current chosen neuron. The activation will be $\sum_{j\neq i} w_{ij}x_j$. If activation is more than the threshold value, x_i will be 1; otherwise, x_i will be assigned 0. It can be proved that the

objective function will be decreased at each step, and this procedure will eventually reach the optimal point.

The above claim can be proved as follows (Rojas, 1996). Suppose neuron k changes status from x_k to x_k' at one step. Let us look at the change of the objective function. Only those items related with neuron k need to be considered. So the change is

$$-\sum_{j\neq k} w_{kj} x_j x_k' + \theta_k x_k' - (-\sum_{j\neq k} w_{kj} x_j x_k + \theta_k x_k) = (\sum_{j\neq k} w_{kj} x_j - \theta_k) * (x_k - x_k').$$

If the activation is larger than the threshold value, status is changed from 0 to 1. Then the change of the objective function is negative because the expression within the first parenthesis has positive value and the expression within the second parenthesis has negative value. If the activation is less than the threshold value, status is changed from 1 to 0. The change of the objective function is also negative because the expression within the first parenthesis has negative value and the expression within the second parenthesis has positive value. So, at any step, if one neuron changes its status, the objective function will be decreased.

For the TSP, supposing that we have one depot and n customers, let x_{ik} be the decision variable, where i=1, ...,n, n+1, denoting the depot and customers, k=1,...,n,n+1, denoting the traveling order. This design can be demonstrated in the following matrix:

i,k	1	2	...	n	n+1
1	x_{11}	x_{12}	...	x_{1n}	$x_{1,n+1}$
...					
n					
n+1	$x_{n+1,1}$	$x_{n+1,2}$	...	$x_{n+1,n}$	$x_{n+1,n+1}$

Each decision variable x_{ik} corresponds to a neuron, and x_{ik} is either 0 or 1. If it is 1, customer/depot i is visited at step k. The objective function can be written as $\frac{1}{2}\sum_{i,j,k}^{n+1} c_{ij} x_{ik} x_{j,k+1}$ where c_{ij} is the distance between i and j. This objective function represents the total traveling distance of a round trip; therefore, if k equals n+1, k+1 should be regarded as 1. Obviously, each row needs to have one and only one 1 and so does each column. These constraints can be implemented by adding penalty function to the objective function. The penalty function is $\sum_{j=1}^{n+1}(\sum_{i=1}^{n+1} x_{ij} - 1)^2 + \sum_{i=1}^{n+1}(\sum_{j=1}^{n+1} x_{ij} - 1)^2$ and we can multiply it by a large number $\gamma/2$ to enforce those embedded constraints. So the final objective function is

$$E = \frac{1}{2}\sum_{i,j,k}^{n+1} c_{ij} x_{ik} x_{j,k+1} + \frac{\gamma}{2}(\sum_{j=1}^{n+1}(\sum_{i=1}^{n+1} x_{ij} - 1)^2 + \sum_{i=1}^{n+1}(\sum_{j=1}^{n+1} x_{ij} - 1)^2).$$

By simply arranging the term, the objective function E becomes

$$E = \frac{1}{2}\sum_{i,j,k}^{n+1} c_{ij} x_{ik} x_{j,k+1} + \frac{\gamma}{2}\sum_{i,j_1,j_2}^{n+1} x_{i,j_1} x_{i,j_2} + \frac{\gamma}{2}\sum_{i_1,i_2,j}^{n+1} x_{i_1,j} x_{i_2,j} - \gamma\sum_{i,j}^{n+1} x_{ij} + 2(n+1)\gamma,$$

which has the form of the energy function of the Hopfield networks. Note that the last term of E is constant. Each neuron has a threshold of $-\gamma/2$. The weight between two neurons in the same row or column is $-\gamma$. For neuron (i, k) and neuron (j, k+1), there is one additional term $-c_{ij}$ in the weight w_{ij}.

4.2 Application of the Hopfield networks to SLB-VRP

In order to be solvable by the Hopfield networks, the SLB-VRP needs to be represented by an entity-position matrix and have a quadratic form objective function. We will build the Hopfield networks to the SLB-VRP based on the Model III and the following three issues need to be addressed:

a. Design one notation system to describe routes.

b. Transform the objective function to the form of the Energy function of the Hopfield networks.

c. Implement constraints to guarantee feasibility.

According to the entity-position matrix of Model III, the Hopfield networks should consist of (n+m)*n*m neurons. Each neuron is denoted as (i, k), where i=1, 2, ..., n+m is the index of the entity and k=1, 2, ..., n*m is the index of the position. The status of the neuron (i, k) is recorded by the variable x_{ik}. If x_{ik} equals 1, the neuron (i, k) is active, which means that the ith entity takes the kth position; if x_{ik} equals 0, the neuron (i, k) is inactive, which means that the ith entity does not take the kth position. The first n entities are the customers and the next m entities are the artificial depots. The n*m positions are divided into m disjoint segments. Each segment consists of n consecutive positions, and we assign a segment for each route.

Example 4-1 Given the SLB-VRP with 10 customers and 2 vehicles, the entity-position matrix consists of 12 rows and 20 columns.

		Positions							
Entities		1	2	...	10	11	12	...	20
	1								
	2								
	...								
	10								
	11								
	12								

The eleventh entity is the artificial depot assigned to the first route and the twelfth entity is the artificial depot assigned to the second route. The entities in the first route will take positions from position 1 to position 10 and the entities in the second route will take positions from position 11 to position 20. Suppose that the first route is depot - customer 1 - customer 2 - customer 3 - depot. To represent this route, the neurons (1, 1), (2, 2), (3, 3), (11, 4) will be active and the variables x_{11}, x_{22}, x_{33}, $x_{11,4}$ will have the value of 1.

The objective function of Model III $\sum_{i,j,k} kc_{ij} x_{ik} x_{j,k+1}$ has already been in the form of the energy function of the Hopfield networks, which means the weight between neuron (i, k) and neuron (j, k+1) should be set as $-kc_{ij}$. One thing needing to be kept in mind is that this function gives the total waiting time of the reversed routes.

We have three types of constraints in Model III. They are:

a. Each row must have one and only one 1.

b. Each column has at most one 1.

c. Each route must be ended at the depot.

The first constraint is easy to handle. It corresponds to constraint (21) in Model III. We simply add one penalty function $\lambda \sum_i (\sum_k x_{ik} - 1)^2$. If we break down this penalty function and use the relationship $x_{ik}^2 = x_{ik}$ if x_{ik} is a 0-1 integer variable, we get $2\lambda \sum_i \sum_k \sum_{l>k} x_{ik} x_{il} - \lambda \sum_i \sum_k x_{ik} + \lambda \sum_i 1$. If we drop the constant component $\lambda \sum_i 1$, we can recognize the penalty function as an energy function and realize that the threshold value of each neuron should be set as $-\lambda$ and the weight between a pair of neurons in the same row should be set as -2λ.

The second constraint is complicated because some positions may not be taken. In Model III, constraint (22) handles this situation. If one position is taken, which means one neuron in that column is set to 1, no more neurons in that column can be set to 1, but if that position is not taken yet, it is not necessary to have one 1 in that column. Our solution to this problem is that we will dynamically enforce this constraint. At the beginning we will not consider this constraint. As soon as one neuron is set to 1, suppose it is the neuron (i, k), the kth column needs to have that constraint and we will add one

penalty component $\lambda(\sum_i x_{ik} - 1)^2$ to the objective function. If we organize this function as the form of the energy function, we can realize that the weight between a pair of neurons in the kth column should be set as -2λ and the threshold value of each neuron in the kth column should be added -λ. And later on, if the active neuron in the kth column is deactivated, we need to drop the previous added penalty component from the objective function and reverse the previous change made to the threshold values and weights.

The third constraint is much more difficult to enforce. In Model III, we use constraints (17), (18), (19), (20), and (23) to make sure that each depot finds one position and there are n valid positions for customers. There is no straightforward approach to implement it. So we decide not to include this constraint in the model. Instead, we will programmatically enforce it while we computerize our algorithm. The idea is to let the artificial depots pick up their positions at the beginning of each run of the algorithm in a way that the total number of valid positions is n and only choose neurons associated with valid positions in the following computation.

We have configured the Hopfield networks according to the energy function. In the actual implementation, sometimes we need to manipulate the weights and threshold values in order to operate the network more effectively. In the following text, we will discuss how to set the value of parameter λ. Since the parameter λ is related only to the penalty components of the energy function, it is not necessary for us to set the related weights and threshold values strictly following the above-mentioned configuration as long as we can make sure the row and column constraints still hold.

As the coefficient in the penalty function, the parameter λ is used to avoid violation of row and column constraints. For two neurons in the same row, since the

weight between them is -2λ, if one neuron is active, the activation of the other one will be -2λ plus some value contributed by neurons in other rows. Because the threshold value is $-\lambda$, we can see that the other neuron will not be activated as long as λ is large enough to compensate the activation contributed by neurons in other rows. In our Hopfield networks, the activation contributed by neurons in other rows will always be negative since it is the product of a negative coefficient and a positive traveling distance; therefore, the row constraint will be satisfied no matter which value the parameter λ takes.

The parameter λ is also used to make sure the column constraints are held. For two neurons in the same column, if one neuron is active, the weight between them is the same as the weight between two neurons in the same row, which is -2λ, but the threshold value now is -2λ because the threshold value is added by $-\lambda$ while implementing the column constraint. In our Hopfield networks, the column constraint will hold because the activation contributed by neurons in other columns is negative, as we have explained before. In general Hopfield networks, the column constraint may not be effective if the activation contributed by neurons in other columns can be positive. The traditional approach in dealing with this problem is not to update the threshold value while implementing the column constraint. Because the threshold value has already been set as $-\lambda$ while implementing the row constraint, it is clear that the column constraint will be satisfied. In this approach, the value of λ is shared in the implementation of row constraint and the implementation of column constraint. This sharing is logical because the neuron will not be activated as long as either the row constraint or the column constraint is not satisfied. If we update the threshold value to -2λ, we are actually

modeling the situation that the neuron is not activated unless both the row constraint and the column constraint are not satisfied, which obviously is not a valid assumption. Although in our network the threshold value -2λ works, we will still not update the threshold value while implementing the column constraint in order to follow the tradition and be more flexible.

Another role that the parameter λ plays is to control the solution quality. Suppose neuron (p, k-1) and neuron (q, k+1) are active and we want to decide if neuron (i, k) should be active or not. Since the activation is $-[(k-1)*c_{pi}+k*c_{iq}]$ and the threshold value is $-\lambda$, we will activate the neuron (i, k) if $(k-1)*c_{pi}+k*c_{iq}$ is less than λ and then the total waiting time will be increased by $(k-1)*c_{pi}+k*c_{iq}$. If λ is large, we can see that some long trips will be allowed in the routes and the solution quality will not be good. On the other hand, if λ is small, only short trips will be in the routes and the solution quality will be good.

Since the row and column constraints will always be satisfied no matter what value the parameter λ takes, we will select the value of λ by focusing on how to effectively achieve good quality solutions. As mentioned above, the value of λ is used to evaluate if the waiting time $(k-1)*c_{pi}+k*c_{iq}$ is small enough. We can approximate $(k-1)*c_{pi}+k*c_{iq}$ as $(2k-1)*c$, where c is a positive parameter and can be regarded as the length of a general trip. This approximated waiting time varies with different value of k. To effectively play the role as the controller of the solution quality, the threshold value should be adjusted accordingly, which means the value of λ will be set differently in different columns. For the neuron associated with the kth position in a route, the value of λ is set as $(2k-1)*c$, where c is a positive parameter. This change has no effect on the

column constraints, but for the row constraints, since the neurons in the same row have different value of λ, the weights between a pair of neurons will not be symmetric any more.

To sum up, the network used to solve the SLB-VRP has (n+m)*n*m neurons. x_{ik} is the decision variable and each decision variable corresponds to a neuron. Subscription i is from 1 to n+m, where 1 to n refers to the customers and n+1 to n+m corresponds to the artificial depots. Subscription k is from 1 to n*m. The range of n*m is divided into m disjoint segments. Each segment corresponds to one vehicle. The initial objective function is $\sum_{i,j,k} kc_{ij}x_{ik}x_{j,k+1}$ and becomes $\sum_{i,j,k} kc_{ij}x_{ik}x_{j,k+1} + \lambda\sum_{i}(\sum_{k} x_{ik} - 1)^2$ after adding a penalty component, where λ is a large positive number. After simple algebraic arrangements and setting the value of λ as discussed before, the configuration of the neurons can be stated as follows. For neuron (i, k), the threshold is -(2k-1)*c, where c is a positive parameter and the weight connecting with another neuron in the same row is -2*(2k-1)*c. The weight between neuron (i, k) and neuron (j, k+1) is -kc_{ij}, and the weight between neuron (i, k) and neuron (j, k-1) is -(k-1)c_{ij}. If the neuron (i, k) changes status to 1, the kth column needs to enforce the column constraint, which means $\lambda(\sum_{i} x_{ik} - 1)^2$ needs to be added to the objective function. To implement this constraint, the weight between any two neurons in the kth column should be set as -2*(2k-1)*c and the threshold value of the neurons in the kth column should be set as -(2k-1)*c. Since the threshold value of each neuron has already been set as -(2k-1)*c due to the row constraint, we do not need to update the threshold value while we implement the column constraint.

The network configuration based on the above explanation is illustrated in Figure 4-1 and Figure 4-2. The stochastic computation process of the Hopfield networks is demonstrated in the following example.

Example 4-2 Use the Hopfield networks to solve a 5 customers 2 vehicles SLB-VRP. The locations of customers are specified in Table 4-1 and the traveling distance matrix is in Table 4-2.

The network is built according to the following entity-position matrix:

	1	2	3	4	5	6	7	8	9	10
1										
2										
3										
4										
5										
6										
7										

Entity 6 is the artificial depot for the first vehicle. Positions 1, 2, 3, 4, 5 are reserved for the first route. Entity 7 is the artificial depot for the second vehicle. Positions 6, 7, 8, 9, 10 are reserved for the second route. Totally, we have 70 neurons. The threshold value

of a neuron is -(2k-1)*c, where k is this neuron's position in the route it belongs to and c is a positive parameter. We set c as 70 in this example. For neurons in each row, their threshold value will be -70, -210, -350, -490, -630, -70, -210, -350, -490, -630, respectively. For a neuron (i, k), the related weights are:

-kc_{ij}, with neuron (j, k+1);

-(k-1)c_{ji}, with neuron (j, k-1);

-2*(2k-1)*70, with neuron (i, l);

-2*(2k-1)*70, with neuron (j, k) if column constraint is implemented.

A particular run of the algorithm can be illustrated as follows. The corresponding network states are demonstrated in Table 4-3 to Table 4-14.

Step1 Assign positions to the artificial depots.

We first randomly assign a position to the first artificial depot. Suppose neuron (6, 4) is picked up. We activate this neuron, which means that the artificial depot 6 takes the 4th position and there are 3 valid positions in the first route. Since the total valid positions should be 5, the valid position in the second route should be 2; therefore, the artificial depot 7 needs to take the 8th position, which is the third position in the second segment, and the neuron (7, 8) now is active.

After we initialize the artificial depots, the algorithm proceeds as follows. At each step, we randomly pick up one neuron to calculate its activation. Let us define that the neighborhood be all neurons connected with this neuron. Only active neurons in the neighborhood can contribute to the activation since other neurons either have 0 status

values or have 0 weight values. After we obtain the activation, we can compare it to the threshold value to determine the status value.

Step 2 Randomly pick up a neuron - neuron (3, 2).

Since no neuron in the neighborhood is active, the activation is 0, which is larger than the threshold value -210. We activate neuron (3, 2) and add the column constraint. Now there is a weight -420 assigned to any pair of neurons in the second column.

Step 3 Randomly pick up a neuron - neuron (1, 2).

In the neighborhood, neuron (3, 2) is active. The weight between neuron (1, 2) and neuron (3, 2) is -420. The activation is -420, which is smaller than the threshold value -210. We do not activate neuron (1, 2).

Step 4 Randomly pick up a neuron - neuron (3, 1).

In the neighborhood, neuron (3, 2) is active. The weight between neuron (3, 1) and neuron (3, 2) is -140. The activation is -140, which is smaller than the threshold value -70. We do not activate neuron (3, 1).

Step 5 Randomly pick up a neuron - neuron (4, 1).

In the neighborhood, neuron (3, 2) is active. The weight between neuron (4, 1) and neuron (3, 2) is -57.92. The activation is -57.92, which is larger than the threshold value -70. We activate neuron (4, 1) and add column constraint.

Step 6 Randomly pick up a neuron - neuron (1, 3).

In the neighborhood, neuron (3, 2) and neuron (6, 4) are active. The weight between neuron (1, 3) and neuron (3, 2) is -2*83.32=-166.64. The weight between neuron (1, 3) and neuron (6, 4) is -3*27.81=-83.43. The activation is -250.07, which is larger than the threshold value -350. We activate neuron (1, 3) and add column constraint.

Step 7 Randomly pick up a neuron - neuron (3, 2).

In the neighborhood, neuron (1, 3) and neuron (4, 1) are active. The weight between neuron (3, 2) and neuron (1, 3) is -2*83.32=-166.64. The weight between neuron (3, 2) and neuron (4, 1) is -57.92. The activation is -224.56, which is smaller than the threshold value -210. We deactivate neuron (3, 2) and remove column constraint.

Step 8 Randomly pick up a neuron - neuron (5, 2).

In the neighborhood, neuron (1, 3) and neuron (4, 1) are active. The weight between neuron (5, 2) and neuron (1, 3) is -2*81.32=-162.64. The weight between neuron (5, 2) and neuron (4, 1) is -30.4. The activation is -193.04, which is larger than the threshold value -210. We activate neuron (5, 2) and add column constraint.

Step 9 Randomly pick up a neuron - neuron (3, 7).

In the neighborhood, neuron (7, 8) is active. The weight between neuron (3, 7) and neuron (7, 8) is -2*96.24=-192.48. The activation is -192.48, which is larger than the threshold value -210. We activate neuron (3, 7) and add column constraint.

Step 10 Randomly pick up a neuron - neuron (2, 6).

In the neighborhood, neuron (3, 7) is active. The weight between neuron (2, 6) and neuron (3, 7) is -2*96.24=-32. The activation is -32, which is larger than the threshold value -70. We activate neuron (2, 6) and add column constraint.

Step 11 Output.

Now neurons (4, 1), (5, 2), (1, 3), (6, 4), (2, 6), (3, 7) and (7, 8) are active. We get a feasible solution. The actual routes are 0 - 1- 5 - 4 - 0 and 0 - 3 - 2 - 0. The total waiting time is 500.95.

4.3 Several improvement procedures

In this section, we will discuss several techniques we have applied to improve our Hopfield networks. To reduce the complexity of the Hopfield networks, we provide an estimation of the maximum number of customers in a route; therefore, we do not need to assume that each route takes up to n positions, and consequently we can decrease the number of neurons used in the network. As a byproduct, the minimum number of customers in a route is estimated as well, which can help us to narrow the solution space. We have also developed a guided sampling process that can save much computation time to reach a feasible solution. After a feasible solution is obtained, we will then apply two local search procedures to improve the solution. The first local search procedure is called the oscillating procedure. By manipulating the threshold values of the neurons up and down, we can replace some trips with shorter ones and maintain the feasibility at the

same time. The second local search procedure is a traditional exchanging procedure. We will implement a 2-opt exchanging procedure using the computational mechanism of the Hopfield networks.

4.3.1 Estimation of the maximum/minimum number of customers in a route

In Model III, the entity-position matrix has n+m rows and n*m columns. Correspondingly, we need (n+m)*n*m neurons in the Hopfield networks to solve the SLB-VRP. This scheme is designed with the assumption that each route can consist of up to n customers. Although this assumption is valid and makes modeling a more straightforward task, it overestimates the maximum number of customers in a route, which leads to some unnecessary neurons in the network. If we can obtain a good estimation of the number of customers in the most crowded route, which will be denoted as M (<n) in the following texts for simplicity, we only need M*m columns in the entity-position matrix and the number of neurons can be reduced to (n+m)*M*m. We will estimate M based on the following observation:

If a route is much longer than another one, by inserting the last customer of the longer route into the last position of the shorter route, we may end up with two better routes in terms of the total waiting time.

Example 4-3 Route 1: 0 - 1 - 2 - 3 - 0; Route 2: 0 - 4 - 0. The waiting time of customer 3 in route 1 is 100 time units. The waiting time of customer 4 in route 2 is 60 time units. The travel time from customer 4 to customer 3 is 20 units. If we insert customer 3 behind

customer 4, the waiting time of customer 3 is 80 time units. Since all other customers have the same waiting time, the total waiting time is better off by 20 time units.

Property 1 For the SLB-VRP with n customers and m vehicles, if the length of a trip follows a distribution with the mean of μ and the standard deviation of σ, the approximation of M is $\frac{n+(m-1)*(\sqrt{\frac{2n}{m}+1}*Z_\alpha*\frac{\sigma}{\mu}+1)}{m}$, where Z_α satisfies $\Phi(Z_\alpha)=1-\alpha$ and $\Phi(.)$ is the distribution function of the standard normal distribution.

Proof Suppose originally route 1 and route 2 consist of p and q customers, respectively. If we denote X_i, i=1, 2, ..., p, as the length of the p trips from the depot to the pth customer in route 1 and Y_j, j=1, 2, ..., q, as the length of the q trips from the depot to the qth customer in route 2, the total waiting time of customers in route 1 is $L_1=pX_1+(p-1)X_2+\ldots+X_p$ and the total waiting time of customers in route 2 is $L_2=qY_1+(q-1)Y_2+\ldots+Y_q$. Now let us take the pth customer out of route 1 and insert this customer into route 2 as the (q+1)th customer. The new total waiting time of customers in route 1 is $L_3=(p-1)X_1+(p-2)X_2+\ldots+X_{p-1}$ and the new total waiting time of customers in route 2 is $L_4=(q+1)Y_1+qY_2+\ldots+2Y_q+Y_{q+1}$, where Y_{q+1} is the length of the trip from the qth customer to the (q+1)th customer. The change of the total waiting time of these p+q customers is $L=L_3+L_4-L_1-L_2=Y_1+Y_2+\ldots+Y_q+Y_{q+1}-X_1-X_2-\ldots-X_p$. Since each X_j and Y_j follows the same distribution, by the Central Limit Theory, L follows the Normal distribution $N((q+1-p)\mu, (p+q+1)\sigma^2)$. Let the probability of L<0 equal to $1-\alpha$. Since

$\frac{L-(q+1-p)\mu}{\sqrt{p+q+1}\sigma}$ follows the standard normal distribution N(0,1).

Prob(L<0)=Prob($\frac{L-(q+1-p)\mu}{\sqrt{p+q+1}\sigma} < \frac{-(q+1-p)\mu}{\sqrt{p+q+1}\sigma}$)=1-α. Therefore, as long as

$\frac{-(q+1-p)\mu}{\sqrt{p+q+1}\sigma} \geq Z_\alpha$, in a significance level of α, the new total waiting time is less than the original total waiting time. In the optimal solution, we need to have

$\frac{-(q+1-p)\mu}{\sqrt{p+q+1}\sigma} < Z_\alpha$; otherwise, the solution can be improved by the previously mentioned adjustment. In the SLB-VRP with n customers and m vehicles, supposing the number of customers in the m routes to be $n_1, n_2, \ldots, n_m$ respectively and $n_1 \geq n_2 \geq \ldots \geq n_m$, we have the following relationship $\frac{(n_1-n_m-1)\mu}{\sqrt{n_1+n_m+1}\sigma} < Z_\alpha$. Since route 1 is the most crowded route and route m is the least crowded route, we can approximate n_1+n_m as 2*n/m; therefore, $n_1 - n_m < \sqrt{1+\frac{2n}{m}} Z_\alpha \frac{\sigma}{\mu} + 1$. Since $n_1+n_2+\ldots+n_m=n$, it is clear that n_1 takes the maximum value while $n_2=n_3=\ldots=n_m$ and we can deduct the largest possible value of n_1 is $\frac{n+(m-1)*(\sqrt{\frac{2n}{m}+1}*Z_\alpha*\frac{\sigma}{\mu}+1)}{m}$.

Q.E.D.

Similarly, we can provide an approximation for the minimum number of customers in a route. As we discussed before, the artificial depots will pick up their positions at the beginning of each run. After the artificial depots obtain their positions,

the number of customers in each route is fixed. With the approximation of the minimum number of customers in a route, we can reduce the number of possible positions the artificial depots can take and consequently narrow the solution space we need to search. This approximation is stated in Property 2.

Property 2 For the SLB-VRP with n customers and m vehicles, if the length of a trip follows a distribution with the mean of μ and the standard deviation of σ, the approximation of the minimum number of customers in a route is

$$\frac{n-(m-1)*(\sqrt{\frac{2n}{m}+1}*Z_\alpha*\frac{\sigma}{\mu}+1)}{m}$$, where Z_α satisfies $\Phi(Z_\alpha)=1-\alpha$ and $\Phi(.)$ is the distribution function of the standard normal distribution.

Proof In the SLB-VRP with n customers and m vehicles, supposing the number of customers in the m routes to be $n_1, n_2, \ldots, n_m$ respectively and $n_1 \geq n_2 \geq \ldots \geq n_m$, we have the following relationship $\frac{(n_1-n_m-1)\mu}{\sqrt{n_1+n_m+1}\sigma} < Z_\alpha$ that has been shown in the proof of Property 1. Since route 1 is the most crowded route and route m is the least crowded route, we can approximate n_1+n_m as 2*n/m; therefore, $n_1-n_m < \sqrt{1+\frac{2n}{m}}Z_\alpha\frac{\sigma}{\mu}+1$. Since $n_1+n_2+\ldots+n_m=n$, it is clear that n_m takes the minimum value while $n_1=n_2=\ldots=n_{m-1}$ and we can deduct the smallest possible value of n_m is $\frac{n-(m-1)*(\sqrt{\frac{2n}{m}+1}*Z_\alpha*\frac{\sigma}{\mu}+1)}{m}$.

Q.E.D.

An obvious range of M can be set up as follows. M cannot exceed n-m+1 since each of the other m-1 routes consists of at least 1 customer. On the other hand, a route consists of [n/m] customers at the average, and the most crowded route should have at least [n/m] customers, where [n/m] refers to the largest integer that is smaller than n/m. Therefore, the range of M is from [n/m] to n-m+1. After we calculate M according to Property 1, if M is less than [n/m], we round it up to [n/m]; if M is larger than n-m+1, we round it down to n-m+1. For the minimum number of customers in a route, the range is from 1 to [n/m]. If the approximation is out of this range, we need to round it to 1 or [n/m].

4.3.2 Guided sampling process

The Hopfield networks are operated stochastically. At each step, a neuron is selected randomly, and then the activation is calculated to decide the status. Because of the row and column constraints, at some steps, the selected neuron may have no chance to be set to 1; therefore, the computational time spent at those steps is wasted. To save the computational time, we implement a guided sampling process. If neuron (i,k) sets its status to 1, the neurons in both the ith row and the kth column will not be selected. One drawback to this scheme is that neuron (i,k) will not be selected again either, and we lost the capability to set its status back to 0. Fortunately, this problem can be alleviated by the oscillating procedure discussed next.

4.3.3 Oscillating procedure

The oscillating procedure is a local search procedure to replace trips with shorter ones by manipulating the threshold value up and down. After we obtain a feasible solution, we increase the threshold value and scan all of those active neurons again. Since the threshold value is increased, some neurons will be deactivated, which means that some customers now are not assigned to the routes and there are some unoccupied positions in the routes. We then focus on those neurons corresponding to these unassigned customers and unoccupied positions and try to activate some neurons to obtain a new feasible solution. Due to the increased threshold value, the new feasible solution will be better than the original one. If we can obtain a new feasible solution, we will increase the threshold value again and repeat the process; if we cannot obtain a new feasible solution, we will decrease the threshold value to get a feasible solution first and then increase the threshold value to see if we can get a better solution.

4.3.4 Exchanging procedure

The exchanging procedure is an ordinary 2-opt procedure. Suppose we have two route segments: i_1 - i - i_2 and j_1 - j - j_2. To see if it is worthwhile to exchange i and j, we need to study the difference in the total waiting time. Let us assume customer i takes the pth position and customer j takes the qth position. The total waiting time related with these two customers is $(p-1)c_{i_1 i} + pc_{ii_2} + (q-1)c_{j_1 j} + qc_{jj_2}$ before the exchange. After the exchange, the related waiting time is $(p-1)c_{i_1 j} + pc_{ji_2} + (q-1)c_{j_1 i} + qc_{ij_2}$. If the related waiting time is reduced, we should switch customer i and customer j. To implement this procedure in the Hopfield networks, each neuron is added to the

capability of calculating the waiting time related to itself in the current position or the other positions. For example, neuron (i, p) will know how to calculate $(p-1)c_{i_1 i} + pc_{i i_2}$ and $(q-1)c_{j_1 i} + qc_{i j_2}$.

4.4 Overview

Our Hopfield networks based heuristic includes two stages: the initialization stage and the running stage. Figure 4-3 is the flow-chart for the heuristic. We construct and configure the network in the initialization stage. We first do some statistical analysis about the traveling distances among all entities to derive the mean μ and the standard deviation σ of the distribution that we assume the length of a trip should follow. We then can estimate the maximum number of customers in a route and decide how many neurons need to be created. The minimum number of customers in a route is estimated as well, which is used as a lower bound for the number of customers in the routes. The parameter c is set as $\mu+p*\sigma$, where p is a parameter. The weights among neurons are calculated, based on the traveling distance matrix as we stated in section 4.2. The threshold value of neuron (i, k) is set as $(2k-1)*(\mu+p*\sigma)$. We can adjust the value of p to increase and decrease the threshold value. The running stage includes multiple iterations. In each iteration, we first conduct the guided sampling process to generate one feasible solution and then apply the oscillating procedure and the exchanging procedure to improve it. If the solution is better than the best solution among all previous found solutions, we will keep this solution as the best solution. The running stage will be stopped if the improvement is not significant any more. We will check the improvement every n/m*5

runs. The improvement is considered insignificant if the current best solution is less than 1 percent better than the previous best solution.

Table 4-1 Location file for Example 4-2

	X	Y
Depot 0	0	0
Customer 1	12.17	25.01
Customer 2	62.97	16.50
Customer 3	94.96	15.68
Customer 4	94.02	73.59
Customer 5	91.41	43.30

Table 4-2 Traveling distance matrix for Example 4-2

	0	1	2	3	4	5
0		27.81	65.09	96.24	119.39	101.14
1			51.51	83.32	95.18	81.32
2				32	64.99	39.07
3					57.92	27.84
4						30.4
5						

Table 4-3 Demonstration of Example 4-2: Configuration

	1	2	3	4	5	6	7	8	9	10
1										
2										
3										
4										
5										
6										
7										

For (i,k):
d=70
θ=-70(2k-1)
Weights:
$-kc_{ij}$ (j,k+1)
$-(k-1)c_{ij}$ (j,k-1)
-2d(2k-1) (i,q)
-2d(2k-1) (p, k)

Table 4-4 Demonstration of Example 4-2: Step 1

	1	2	3	4	5	6	7	8	9	10
1										
2										
3										
4										
5										
6				1						
7								1		

Assign positions to the artificial depots

Table 4-5 Demonstration of Example 4-2: Step 2

	1	2	3	4	5	6	7	8	9	10
1										
2										
3		1								
4										
5										
6				1						
7								1		

For (3,2):
active neighbor:
none
activation:
0
threshold:
-210

Table 4-6 Demonstration of Example 4-2: Step 3

	1	2	3	4	5	6	7	8	9	10
1										
2										
3		1								
4										
5										
6				1						
7								1		

For (1,2):
active neighbor:
(3,2)
activation:
-420
threshold:
-210

Table 4-7 Demonstration of Example 4-2: Step 4

	1	2	3	4	5	6	7	8	9	10
1										
2										
3		1								
4										
5										
6				1						
7								1		

For (3,1):
active neighbor:
(3,2)
activation:
-140
threshold:
-70

Table 4-8 Demonstration of Example 4-2: Step 5

	1	2	3	4	5	6	7	8	9	10
1										
2										
3		1								
4	1									
5										
6				1						
7								1		

For (4,1):
active neighbor:
(3,2)
activation:
-57.92
threshold:
-70

Table 4-9 Demonstration of Example 4-2: Step 6

	1	2	3	4	5	6	7	8	9	10
1			1							
2										
3		1								
4	1									
5										
6				1						
7								1		

For (1,3):
active neighbor:
(3,2),(6,4)
activation:
-250.07
threshold:
-350

Table 4-10 Demonstration of Example 4-2: Step 7

	1	2	3	4	5	6	7	8	9	10
1			1							
2										
3		0								
4	1									
5										
6				1						
7								1		

For (3,2):
active neighbor:
(4,1),(1,3)
activation:
-224.56
threshold:
-210

Table 4-11 Demonstration of Example 4-2: Step 8

	1	2	3	4	5	6	7	8	9	10
1			1							
2										
3										
4	1									
5		1								
6				1						
7								1		

For (5,2):
active neighbor:
(4,1),(1,3)
activation:
-193.04
threshold:
-210

Table 4-12 Demonstration of Example 4-2: Step 9

	1	2	3	4	5	6	7	8	9	10
1			1							
2										
3							1			
4	1									
5		1								
6				1						
7								1		

For (3,7):
active neighbor:
(7,8)
activation:
-192.48
threshold:
-210

Table 4-13 Demonstration of Example 4-2: Step 10

	1	2	3	4	5	6	7	8	9	10
1			1							
2						1				
3							1			
4	1									
5		1								
6				1						
7								1		

For (2,6):
active neighbor:
(3,7)
activation:
-32
threshold:
-210

Table 4-14 Demonstration of Example 4-2: Result

	1	2	3	4	5	6	7	8	9	10
1			1							
2						1				
3							1			
4	1									
5		1								
6				1						
7								1		

Routes:
0 - 1 - 5 - 4 - 0
0 - 3 - 2 - 0

Waiting time:
500.95

Figure 4-1. Network used to solve a SLB-VRP with n customers m vehicles

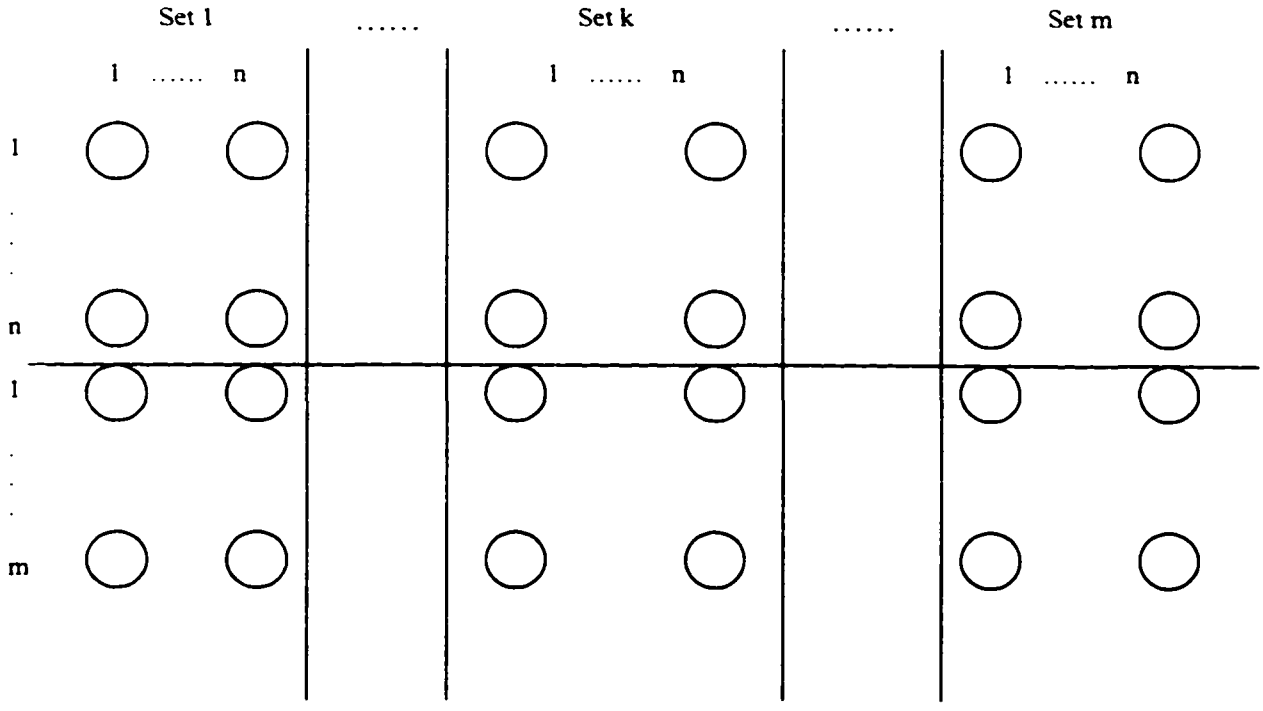


Figure 4-2. Network configuration

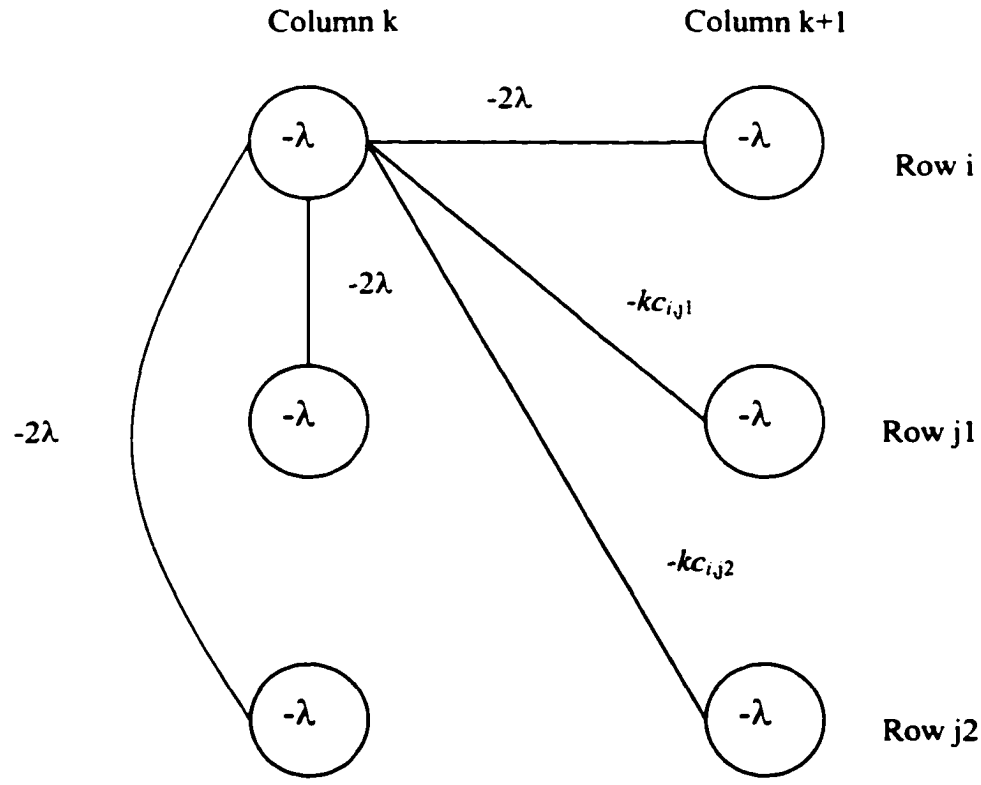
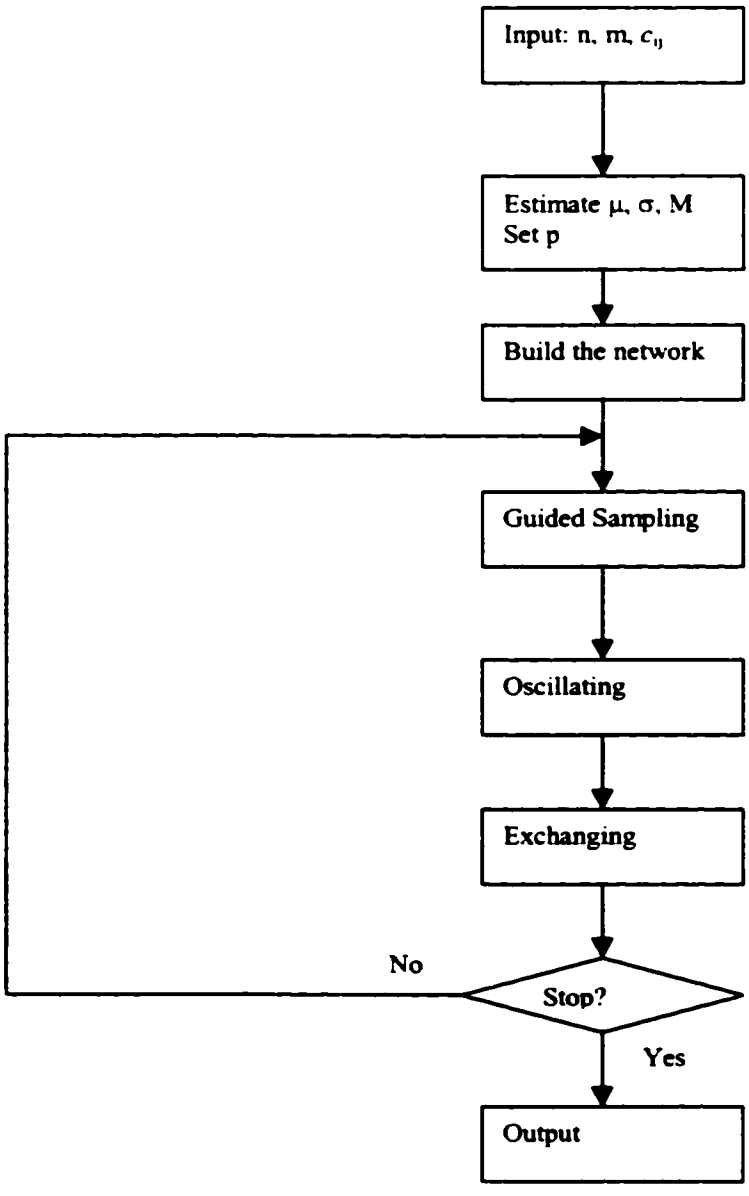


Figure 4-3. Flow-chart of Hopfield networks based heuristic for SLB-VRP



Chapter V

EXPERIMENTAL DESIGN AND COMPUTATIONAL RESULTS

In this study, experiments are designed for the following two purposes. The first one is to test the performance of our heuristic algorithm. Since SLB-VRP is NP-complete (Sahni and Gonzalez, 1976) and relatively new in the literature, no benchmark cases are available. So we need to measure the quality of our solutions by some indirect approaches. The second purpose is to study the operational implications of SLB-VRP. The operational implications we want to study include the effect of the shape of service region and the effect of the location of the depot. These implications can help us configure the warehousing and distributing system.

5.1 Experimental design

5.1.1 Performance measurements

The approach we use to measure the performance is to compare our solutions with some estimated optimal solutions. We have developed three estimations for the optimal solution. The first estimator is based on the order statistics of all of the solutions generated in our algorithm and then calculate the percentage error of our final solution. The optimal solution X^* is estimated as $2X(1)-X(2)$ (Dannebring, 1977), where $X(1)$ and $X(2)$ are the first and second order statistics of all obtained solutions. The other estimators are to generate some near optimal solutions by some other well-known good heuristic algorithms. We have modified the Savings algorithm and the Sweep algorithm

to solve the SLB-VRP. Details of these two algorithms are provided in Appendix A and Appendix B.

Another approach in evaluating the quality of our solutions can be described as follows. The total waiting time is defined upon those individual elements of the traveling distance matrix. We can regard solving this problem as looking for a set of those elements that corresponds to a feasible routing scheme with the shortest total waiting time. Therefore, one characteristic of the set consisting of those selected elements is that, in general, elements in this set have smaller values than elements not in this set. After we get one solution, if we can construct an indicator to show how small the selected elements' values are, we will be able to get some insight into how good that solution is.

Given one route consisting of n customers, n elements of the traveling distance matrix will be used to calculate the total waiting time. Actually, there are n*(n+1)/2 elements involved because the first element will be counted n times, the second elements will be counted n-1 times, and so on. If we divide the total waiting time by n*(n+1)/2, the result should be a good measurement for the average value of all elements involved. If there is more than one route, we can first get the total number of elements involved by considering those routes one by one, then calculate the average value by dividing the total waiting time by that number. To determine how small this average value is, we will measure it using the mean and standard deviation of all of the elements in the traveling distance matrix. We will divide the difference between the average value and the mean by the standard deviation. The result is the indicator that we will use to evaluate how small the values of those selected elements are. To illustrate this idea, an example is provided in Appendix C.

In order to obtain some insights about how to use this indictor, we have done a small experiment. This experiment is set up as follows. Since we can solve small size SLB-VRP by CPLEX (a mathematical programming software package), we can calculate this performance indicator for those optimal solutions and hence derive some preliminary conclusions about which value of this indicator indicates a good quality solution. We have randomly generated 720 problem instances. The number of customers for the problem instance used in this experiments is ranged from 5 to 8. The number of vehicles is ranged from 1 to 3. For each problem instance, we build the linear programming model (Model II) and solve it by CPLEX. We then calculate this indicator and have done a univariate analysis. The detailed result is in Appendix D. Based on the mean -0.735, standard deviation 0.342, 75% quantile -0.53 and 25% quantile -0.988, we construct the following tentative grading system to help us judge the performance of a solution:

Grade	Range	Explanation
A	<=-1.0	Performance is excellent
B	(-1.0, -0.75]	Performance is good
C	(-0.75, -0.5]	Performance is acceptable
D	(-0.5, 0]	Performance needs to be monitored
E	>0	Performance is not acceptable

5.1.2 Operational implications

The operational implications we want to study include the effect of the shape of service region and the effect of the location of a depot. We will study if service regions

with different shapes lead to different total waiting time. The result can guide the distribution manager in deciding how to divide the entire responsible service region into some smaller operating regions. And we will try to find the difference in total waiting time if we move the depot from the corner of the service region to the center of the service region. If the depot is located at the corner, several adjacent service regions can share a large depot, which usually means a less fixed cost compared to several smaller depots. We call this type of warehousing system the centralized system. On the other hand, if the depot is located at the center of the service region, we can expect shorter total waiting time compared to the depot at the corner case, but we need to have a depot for each service region. We call this type of warehousing system the decentralized system.

5.1.3 Experiments setup

To test the performance, the service region is set as a 100*100 square and the depot is located at the origin. Based on some trial runs, we have found that our current computing system (a Compaq Presario 5304, CPU Cyrix MII 366 MHz) can effectively handle only up to 100 customers; therefore, we consider the problems with 50, 60, 70, 80, 90 and 100 customers. The number of vehicles is chosen so that the ration between the number of customers (n) and the number of vehicles (m) is around 10. Let each combination of n and m specify a problem case. We will study the following 18 problem cases: (50, 5), (50, 6), (50, 7), (60, 6), (60, 7), (60, 8), (70, 7), (70, 8), (70, 9), (80, 8), (80, 9), (80, 10), (90, 9), (90, 10), (90, 11), (100, 10), (100, 11) and (100, 12). For each problem case, we generate 10 problem instances. We will solve these problem instances

using our Hopfield networks based heuristic, the modified Savings algorithm and the modified Sweep algorithm.

To study the operational implication, we consider 6 shapes of the service region and 2 locations of the depot. If we characterize the type of service region by the shape of the service region and the location of the depot, we have 12 types of service region. The problem case can be specified by three parameters: the service region type, the number of customers (n), and the number of vehicles (m). To ensure that the operational implications are derived from reliable experimental results, we will not include the n=100 case in our study. For each service region type, we consider 15 (n, m) combinations: (50, 5), (50, 6), (50, 7), (60, 6), (60, 7), (60, 8), (70, 7), (70, 8), (70, 9), (80, 8), (80, 9), (80, 10), (90, 9), (90, 10), and (90, 11). For each problem case, we generate 10 problem instances; therefore, we have 180 problem cases and 1800 problem instances to solve.

The details of the service region are as follows. The area of service region is fixed to 10,000. Without loss of generality, we do not mention the unit. One shape is square, two are rectangles, one is a circle, one is equilateral triangle, and one is a sector. A detailed description is in Table 5-1. Note that the Rectangle-Horizontal shape and the Rectangle-Vertical shape are symmetric. So we do expect these two shapes to have similar results in total waiting time. Note also that the Triangle shape is similar to the Sector shape. So we also expect these two shapes to have similar results in total waiting time. The rectangular-shaped and sectorial-shaped regions are well studied in the literature (Ballou and Agarwal, 1988; Chien, 1992). The rectangular-shaped service region occurs where the locations of customers are forced into a linear pattern usually due to some natural or man-made barriers. The sectorial-shaped service region results from

the manner in which the routes are formed throughout a week, or other time periods. One depot usually covers a service region around itself. This 360-degree service region is divided into sectors so that each sector will be served in a particular day, or other time units. We study the Triangle-shaped service region simply for the purpose of comparison.

The depot can be located at either the corner or the center of the service region. In the depot at corner case, square and rectangular-shaped service regions are put in the first quadrant, and all other service regions are put in the upper half-plane and symmetric to the Y-axis. In the depot at center case, all service regions are centered at the origin. The details of the layout are given in Table 5-2.

Our algorithm will have multiple runs. In each run, the computation process consists of two stages. We use our Hopfield networks to generate an initial solution at the first stage. If this initial solution is feasible, we will feed it into the local search procedures at the second stage. The computation process of our algorithm is controlled by parameter p. The parameter p sets up a threshold value $\mu+p\sigma$, where μ is the mean of the lengths of all trips and σ is the standard deviation of the lengths of all trips. Trips whose lengths exceed this threshold value will not be considered while forming a route. If p is small, only trips with short length can be included in our solution; therefore, the solution quality is good, but we may not be able to reach a feasible initial solution. On the other hand, if p is large, we can easily reach a feasible initial solution, but the quality cannot be guaranteed. The quality of a final solution depends on two factors. The first factor is the quality of the initial solution. With a better quality initial solution, it is more likely to find a better quality final solution. The second factor is the number of valid

runs, i.e., the number of runs that generate feasible initial solutions. Since our algorithm is, in essence, a partial enumeration algorithm, we will have more chance to reach a better final solution with more valid runs.

We have done a preliminary study to find a suitable value for p. In a Square-Corner service region, we have studied 9 problem cases. The (n, m) combinations are (20, 2), (20, 3), (20, 4), (25, 3), (25, 4), (25, 5), (30, 3), (30, 4), and (30, 5). We have solved 10 instances for each problem case with different p values and recorded the number of runs, the number of infeasible runs, the initial solution quality, and the final solution quality. Details are in Table 5-3. The results are consistent with our argument above. We find the number of valid runs is not a serious problem while p is more than 2. The percentage of invalid runs is 0.01% for p=3 and 0 for p=4. Except for the p=0 case, the initial solution quality and the final solution quality are similar among all p>0 cases. Since p=0 case has a too high percentage of invalid runs (93.37%), we do not think our algorithm will be robust with p=0. Based on this preliminary study, the parameter p is set to 3, which means that only trips whose length is less than 3 times standard deviation larger than the mean can be included in a solution.

5.2 Computational results

5.2.1 Performance

The computational result of performance measurement is presented in Table 5-4, Table 5-5, Table 5-6, and Table 5-7. We have used a Compaq Presario 5304 (CPU Cyrix MII 366 MHz) to solve all problem instances. The performance measurement based on order statistics is called %error. For a problem instance, suppose the best solution is x_1

and the second best solution is x_2. The optimal solution x^* is estimated as $2*x_1-x_2$ and %error is $(x_1-x^*)/x^*$. The performance measurements based on the solution of the modified Savings algorithm and the modified Sweep algorithm are called %diff1 and %diff2, respectively. For a problem instance, suppose x_{nn} is the solution from our algorithm, $x_{savings}$ is the solution from the modified Savings algorithm, and x_{sweep} is the solution from the modified Sweep algorithm. %diff1 is $(x_{nn}-x_{savings})/x_{savings}$ and %diff2 is $(x_{nn}-x_{sweep})/x_{sweep}$. The performance indicator is calculated by comparing the trips included in our solution to all of the available trips. Let us denote the weighted average length of the trips included in our solution as AvgCost, the mean of the length of all trips as mean, and the standard deviation of the length of all trips as sd. The performance indicator is calculated as (AvgCost - mean)/sd.

Table 5-4 provides the overall performance information. On the average, our algorithm spends 203 seconds to solve a problem instance. The range is from 19 seconds to 845 seconds. The %error is from 0 percent to 5.31 percent with a mean at 0.68 percent. Compared to the modified Savings algorithm, our algorithm is 6.21 percent better and the range is from 16.94 percent better to 1.17 percent worse. Compared to the modified Sweep algorithm, our algorithm is 2.35 percent better on the average and the range is from 18.77 percent better to 5.3 percent worse. The range of performance indicator is from -1.61 to -1.2 with a mean at -1.43. Considering the fact that the neural networks based algorithm usually needs massive parallel computing and we have used only a PC, the CPU time spent by our algorithm is acceptable. All performance measurements indicate that the performance of our algorithm is good. The %error is very small, which means our algorithm has generated a sequence of solutions that have little

improvement potential. The %diff1 and %diff2 indicate that our algorithm can produce better solutions than the modified Savings algorithm and the modified Sweep algorithm. The performance indicator is well below -1, which means that our solutions consist of some trips whose average length is more than 1 standard deviation below the average length of all trips.

Table 5-5 is the performance information for each category of n. Table 5-6 provides the performance information for each combination of (n, m). As the problem size increases, the solution space becomes bigger. Since our algorithm is an enumeration based algorithm, it will be more difficult for us to search the solution space and hence the performance of our algorithm shows a tendency of deteriorating that is indicated by the increase of %diff1 and %diff2. The phenomena that the modified Savings and the modified Sweep algorithms have relative better performance while the problem size becomes larger can be explained as follows. Both the modified Savings algorithm and the modified Sweep algorithm are designed to take advantage of some configuration information. They only search a subset of the solution space that is related to the configuration information they utilize. The reason why our algorithm can generate better solutions is that we do not restrict our search in that subset of the solution space. While problem size increases, our search becomes relatively difficult; therefore, the improvement of our algorithm over the modified Savings and the modified Sweep algorithms becomes relatively small. In other words, while problem size becomes larger, the modified Savings and the modified Sweep algorithms do not necessarily have better performance; it is more that our algorithm has a relatively worse performance. Note that %error is stable as the problem size increases, which can be explained as follows. %error

is based on order statistics. As long as we have a converged solution sequence, %error will be small. Even for large problem, our algorithm will generate a converged sequence of solution because the search is conducted in a sub-region around a local optimal solution Table 5-7 provides the performance indicator categorized by (n,m). For each problem case, we have shown the range and mean of the performance indicator. Based on our tentative grading system, our solutions can receive grade A in all problem cases.

5.2.2 Operational implications

5.2.2.1 Effect of the shape of service region

The overall average Total Waiting Time and the average Total Waiting Time for each problem case are reported in Table 5-8 to Table 5-11. In Table 5-8, the overall average Total Waiting Time of type Square-Center, Rectangle-Vertical-Center, Rectangle-Horizontal-Center, Circle-Center, Triangle-Center, and Sector-Center are 4123.19, 4680.36, 4753.53, 2941.53, 4068.22 and 4061.21, respectively. In Table 5-9, the overall average Total Waiting Time of type Square-Corner, Rectangle-Vertical-Corner, Rectangle-Horizontal-Corner, Circle-Corner, Triangle-Corner, and Sector-Corner are 5990.96, 7764.02, 7842.86, 4991.02, 5052.18 and 5129.44, respectively. The information in Table 5-8 and Table 5-9 reveals the following five facts:

1. Rectangle service regions have longer Total Waiting Time than non-rectangle service regions;
2. Among rectangle service regions, the square service region has the shortest Total Waiting Time;
3. Circle service region has the shortest Total Waiting Time;

4. Rectangle-Vertical and Rectangle-Horizontal service regions have almost the same total waiting time as expected; and
5. Triangle and Sector service regions have almost the same total waiting time as expected.

These findings are further supported by the Total Waiting Time for each problem case, which are reported in Table 5-10 and Table 5-11.

We have done an ANOVA analysis based on the Total Waiting Time of each problem instance; the results support our observation. In the depot at center case, those six shapes can be classified into three groups. Group I includes type Rectangle-Vertical-Center and type Rectangle-Horizontal-Center. Group II includes type Square-Center, type Triangle-Center, and type Sector-Center. Group III includes type Circle-Center. In the depot at corner case, those six shapes can be classified into three groups as well. Group I includes type Rectangle-Vertical-Corner and type Rectangle-Horizontal-Corner. Group II includes type Square-Corner. Group III includes type Circle-Corner, type Triangle-Corner, and type Sector-Corner. The ANOVA analysis results are reported in Table 5-12 and Table 5-13. Similar results can be obtained from the ANOVA analysis for each problem case. The grouping information is available in Tables 5-14 and 5-15.

5.2.2.2 Effect of the location of depot

To study the effect of the location of depot, we need to arrange a data set to compare the Total Waiting Time between the depot at center case and the depot at corner case. Since we have 15 problem cases and 6 shapes of service region, we have 90 scenarios for this comparison. Each scenario refers to the combination of one problem

case and one shape of service region. In our experiment, each shape corresponds to two types of service region, depending on the depot at center or at corner, and there are 10 problem instances for each combination of problem case and service region type. So totally we have 20 problem instances for each scenario. We generate a data record for each scenario. The data record includes two elements. The first element is the average Total Waiting Time of those 10 problem instances whose depots are at corner. The second element is the average Total Waiting Time of those 10 problem instances whose depots are at center. Then we run a simple regression using the first element (total waiting time for depot at corner case) as dependent variable and the second element (total waiting time for depot at center case) as independent variable. The regression result is reported in Table 5-16. The average Total Waiting Time in the depot at corner case is estimated to be 1.5 times of the average Total Waiting Time in the depot at center case.

The result obtained above can help the distribution manager in configuring a distribution system. If the depot is at center, the Total Waiting Time should be shorter than the depot at corner case since the average distance between customers and depot is shorter. If the depot is at corner, although the Total Waiting Time is longer as discussed before, we may be able to save some warehousing cost since several adjacent service regions can share a large depot. The location of depot depends on whether the savings in warehousing can compensate the extra cost incurred from longer Total Waiting Time.

Table 5-1 Description of service region shape

Shape	Characteristics
Square	Length of side: 100
Rectangle-Vertical	Length: 200; Width: 50; Vertical side is longer than horizontal side
Rectangle-Horizontal	Length: 200; Width: 50; Horizontal side is longer than vertical side
Circle	Radius: 56.42;
Triangle	Equilateral triangle; Length of side: 151.97
Sector	Radius: 138.2; Angle: 60 degree

Table 5-2 Description of service region type

	Type*	Layout in X-Y plane**
Type 1	Square-Center	Corner points: (50, 50), (50, -50), (-50, 50), (-50, -50)
Type 2	Square-Corner	Corner points: (0, 0), (0, 100), (100, 100), (100, 0)
Type 3	Rectangle-Vertical-Center	Corner points: (25, 100), (-25, 100), (-25, -100), (25, -100)
Type 4	Rectangle-Vertical-Corner	Corner points: (0, 0), (0, 200), (50, 200), (50, 0)
Type 5	Rectangle-Horizontal-Center	Corner points: (100, 25), (-100, 25), (-100,-25), (100, -25)
Type 6	Rectangle-Horizontal-Corner	Corner points: (0, 0), (0, 50), (200, 50), (200, 0)
Type 7	Circle-Center	Center: (0, 0), Radius: 56.42
Type 8	Circle-Corner	Center: (0, 56.42), Radius: 56.42
Type 9	Triangle-Center	Corner points: (0, -65.81), (-75.98, 65.81), (75.98, 65.81)
Type 10	Triangle-Corner	Corner points: (0, 0), (-75.98, 131.61), (75.98, 131.61)
Type 11	Sector-Center	Corner points: (0, -69.1), (-69.099, 50.58), (69.099, 50.58)
Type 12	Sector-Corner	Corner points: (0, 0), (-69.1, 119.68), (69.1, 119.68)

* Type name is given by shape name + "-" + location of depot

** Depot is always at (0,0)

Table 5-3 Effects of Parameter p

p	Run*	Invalid-Run*	Percentage	APE-Initial**	APE-Final***
0	315.56	293.53	93.37%	76.72%	19.46%
1	308.00	88.91	29.82%	123.10%	28.37%
2	317.44	4.07	1.45%	134.00%	28.70%
3	307.00	0.02	0.01%	136.88%	29.81%
4	338.44	0	0	136.45%	29.43%

* This number is an average number of all tested problem instances.

** Average percentage error for initial solution. In each run, the initial solution is compared with the best solution, which is the multiple run result.

*** Average percentage error for the final solution. In each run, the final solution is compared with the best solution, which is the multiple run result.

Table 5-4 Overall Performance*

	Min	Max	Mean
Cost	3687.33	8937.36	6398.65
Time	19.00	845.00	203.26
%error**	0	5.31	0.68
%diff1***	-16.94	1.17	-6.21
%diff2****	-18.77	5.30	-2.35
Performance indicator*****	-1.61	-1.20	-1.43

* Based on solutions for all 180 problem instances mentioned on Page 104.

** This value shows the performance of our algorithm based on order statistics. Among all solutions generated by our algorithm for one problem instance, suppose x_1 is the best solution and x_2 is the second best solution. The optimal solution is estimated as $x_{op}=2*x_1-x_2$. %error is defined as $(x_1-x_{op})/x_{op}$.

*** This value shows the difference between our algorithm and the modified Savings algorithm. Suppose x_{nn} is solution generated by our algorithm and $x_{savings}$ is the solution from the modified Savings algorithm. %diff1 is $(x_{nn}-x_{savings})/x_{savings}$.

**** This value shows the difference between our algorithm and the modified Sweep algorithm. Suppose x_{nn} is solution generated by our algorithm and x_{sweep} is the solution from the modified Sweep algorithm. %diff2 is $(x_{nn}-x_{sweep})/x_{sweep}$.

***** This value shows how small the average length of trips in our solution is, when compared to the overall average (μ) and standard deviation (sd) of the length of all elements in the traveling distance matrix. Suppose the average length of trips in our solution is AvgCost. This performance indicator is defined as (AvgCost-μ)/sd.

Table 5-5 Performance classified by n*

	n=50			n=60			n=70		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	3687.33	5241.58	4357.10	4662.38	5734.32	5224.07	5376.64	6713.18	5986.17
Time	19.00	221.00	36.57	32.00	205.00	99.33	49.00	297.00	147.93
%error	0.01	5.31	0.81	0.05	5.18	0.71	0.01	2.65	0.59
%diff1	-16.94	-0.89	-7.81	-14.99	-0.58	-6.87	-12.50	-1.19	-6.52
%diff2	-18.77	3.23	-4.73	-9.53	2.07	-3.77	-12.10	5.30	-2.76

	n=80			n=90			n=100		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	6193.16	7359.99	6665.36	6889.71	8488.89	7722.09	7905.09	8937.36	8437.10
Time	75.00	660.00	256.90	116.00	722.00	272.13	151.00	845.00	377.97
%error	0	2.04	0.58	0.04	2.77	0.72	0.02	3.05	0.67
%diff1	-15.06	1.17	-6.44	-10.94	-0.40	-4.85	-10.55	0.99	-4.74
%diff2	-11.52	5.01	-1.49	-5.28	3.75	-1.17	-4.63	3.61	-0.18

* 30 problem instances in each n category

Table 5-6 Performance classified by (n, m)*

	(50,5)			(50,6)			(50,7)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	3986.05	5241.58	4589.16	3794.27	4865.78	4319.46	3687.33	4575.85	4162.69
Time	20.00	221.00	70.70	19.00	111.00	63.70	37.00	106.00	61.40
%error	0.07	5.31	1.17	0.06	2.06	0.90	0.01	1.54	0.36
%diff1	-16.94	-5.48	-10.74	-16.13	-2.59	-8.13	-7.41	-0.89	-4.55
%diff2	-18.77	2.40	-8.32	-17.51	3.23	-4.20	-3.84	0.58	-1.66

	(60,6)			(60,7)			(60,8)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	4902.98	5734.32	5453.92	4803.25	5419.13	5187.33	4662.38	5248.69	5030.97
Time	34.00	205.00	117.50	64.00	192.00	103.40	32.00	162.00	77.10
%error	0.06	5.18	0.99	0.08	1.89	0.71	0.05	1.87	0.42
%diff1	-14.77	-5.06	-8.71	-14.40	-2.53	-7.50	-14.99	-0.58	-4.40
%diff2	-9.53	-0.90	-5.38	-6.07	2.07	-2.62	-7.30	-0.72	-3.32

	(70,7)			(70,8)			(70,9)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	5893.35	6713.18	6250.79	5624.58	6357.26	5950.84	5376.64	6066.77	5756.87
Time	53.00	277.00	137.20	52.00	258.00	169.00	49.00	297.00	137.60
%error	0.10	2.65	0.91	0.01	7.16	0.36	0.04	2.07	0.51
%diff1	-12.50	-2.21	-7.83	-9.75	-1.19	-5.74	-9.30	-3.72	-5.98
%diff2	-12.10	5.30	-6.25	-6.01	1.34	-1.63	-3.45	1.90	-0.40

* 10 problem instances in each (n, m) category

Table 5-6 Performance classified by (n, m) (continued)*

	(80,8)			(80,9)			(80,10)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	6554.57	7359.99	6895.80	6327.16	7031.39	6626.11	6193.16	6770.94	6474.16
Time	161.00	660.00	320.10	75.00	314.00	200.00	159.00	480.00	250.60
%error	0.02	1.27	0.41	0.37	2.04	0.72	0	1.90	0.61
%diff1	-15.06	1.17	-6.75	-10.68	-2.72	-7.57	-8.27	-1.06	-5.02
%diff2	-11.52	5.01	-1.93	-6.13	1.53	-1.51	-4.21	2.01	-1.03

	(90,9)			(90,10)			(90,11)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	7213.05	8488.89	7924.80	6912.86	8065.71	7690.39	6889.71	8040.93	7551.09
Time	116.00	722.00	262.20	120.00	480.00	308.00	119.00	351.00	249.20
%error	0.29	2.77	1.07	0.05	2.04	0.63	0.04	1.07	0.44
%diff1	-10.94	-2.31	-6.50	-9.45	-0.40	-5.12	-7.73	-1.10	-2.92
%diff2	-5.28	1.72	-2.11	-4.40	0.46	-1.06	-3.46	3.75	-0.33

	(100,10)			(100,11)			(100,12)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Cost	8248.15	8937.36	8613.42	8032.29	8676.48	8425.32	7905.09	8544.88	8272.57
Time	168.00	643.00	364.40	164.00	814.00	377.30	161.00	845.00	392.20
%error	0.23	3.05	1.35	0.19	7.20	0.24	0.06	0.95	0.42
%diff1	-10.55	-2.05	-6.29	-10.42	0.99	-4.27	-6.82	-0.51	-3.66
%diff2	-4.63	3.61	-0.90	-1.75	1.91	0.03	-2.43	2.55	0.33

* 10 problem instances in each (n, m) category.

Table 5-7 Performance indicator classified by (n, m)

n=50, mean of traveling distance matrix elements=53.18, standard deviation of traveling distance matrix elements=25.09									
	(50,5)			(50,6)			(50,7)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Average Length	14.23	18.85	16.4	15.67	20.61	18.1	17.89	22.21	20.22
Indicator	-1.61	-1.33	-1.47	-1.55	-1.26	-1.4	-1.48	-1.2	-1.31
Tentative Grade	A	A	A	A	A	A	A	A	A

n=60, mean of traveling distance matrix elements=53.27, standard deviation of traveling distance matrix elements=25.32									
	(60,6)			(60,7)			(60,8)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Average Length	14.63	17.81	16.25	16.17	18.36	17.63	18.14	20.02	19.69
Indicator	-1.57	-1.39	-1.46	-1.5	-1.33	-1.41	-1.42	-1.27	-1.34
Tentative Grade	A	A	A	A	A	A	A	A	A

n=70, mean of traveling distance matrix elements=51.92, standard deviation of traveling distance matrix elements=24.58									
	(70,7)			(70,8)			(70,9)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Average Length	15.03	19.63	15.99	15.93	18.00	17.05	17.01	19.63	18.46
Indicator	-1.51	-1.38	-1.46	-1.47	-1.34	-1.42	-1.41	-1.29	-1.36
Tentative Grade	A	A	A	A	A	A	A	A	A

Table 5-7 Performance indicator classified by (n, m) (continued)

n=80, mean of traveling distance matrix elements=52.46, standard deviation of traveling distance matrix elements=24.93									
	(80,8)			(80,9)			(80,10)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Average Length	14.67	16.35	15.41	15.68	17.44	16.39	16.64	18.34	17.52
Indicator	-1.55	-1.42	-1.49	-1.5	-1.37	-1.45	-1.46	-1.32	-1.4
Tentative Grade	A	A	A	A	A	A	A	A	A

n=90, mean of traveling distance matrix elements=52.65, standard deviation of traveling distance matrix elements=25.01									
	(90,9)			(90,10)			(90,11)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Average Length	14.28	16.94	15.71	15.06	17.53	16.72	16.05	18.91	17.70
Indicator	-1.56	-1.40	-1.48	-1.53	-1.37	-1.44	-1.49	-1.32	-1.40
Tentative Grade	A	A	A	A	A	A	A	A	A

n=100, mean of traveling distance matrix elements=52.54, standard deviation of traveling distance matrix elements=24.80									
	(100,10)			(100,11)			(100,12)		
	Min	Max	Mean	Min	Max	Mean	Min	Max	Mean
Average Length	14.67	15.98	15.36	15.59	16.81	16.35	16.40	18.02	17.30
Indicator	-1.54	-1.45	-1.50	-1.51	-1.42	-1.46	-1.48	-1.36	-1.42
Tentative Grade	A	A	A	A	A	A	A	A	A

Table 5-8 Total waiting time for each shape of service region - Depot at center

	Average	Minimum	Maximum
Square-Center	4123.19	2302.66	5864.49
Rectangle-Vertical-Center	4680.36	2854.96	6658.39
Rectangle-Horizontal-Center	4753.53	2728.09	6881.16
Circle-Center	2941.53	1625.98	4497.61
Triangle-Center	4068.22	2338.16	5890.29
Sector-Center	4061.21	2421.46	6088.53

Table 5-9 Total waiting time for each shape of service region - Depot at corner

	Average	Minimum	Maximum
Square-Corner	5990.96	3687.33	8488.89
Rectangle-Vertical-Corner	7764.02	5020.92	10861.60
Rectangle-Horizontal-Corner	7842.86	4958.89	10458.20
Circle-Corner	4991.02	3093.51	7076.48
Triangle-Corner	5052.18	3031.13	6912.75
Sector-Corner	5129.44	3090.51	7464.50

Table 5-10 Average total waiting time for each problem case - Depot at center

	Square-Center	Rectangle-Vertical-Center	Rectangle-Horizontal-Center	Circle-Center	Triangle-Center	Sector-Center
(50,5)	3455.63	3735.62	3855.12	2441.46	3420.78	3379.67
(50,6)	3006.27	3336.29	3436.80	2106.08	3022.82	2933.21
(50,7)	2683.01	3081.43	3133.81	1896.74	2760.89	2659.02
(60,6)	4109.35	4444.21	4459.59	2946.73	3876.08	3882.28
(60,7)	3658.29	4040.46	4066.66	2567.56	3467.04	3495.25
(60,8)	3296.23	3755.33	3792.31	2339.08	3192.38	3236.24
(70,7)	4479.23	5006.85	5213.10	3191.81	4503.62	4357.57
(70,8)	4042.96	4673.79	4752.37	2876.54	4062.83	3995.50
(70,9)	3761.02	4364.68	4454.98	2651.01	3815.48	3720.92
(80,8)	4996.37	5688.13	5697.54	3494.42	4943.24	4957.13
(80,9)	4555.98	5314.19	5319.88	3226.35	4576.44	4566.97
(80,10)	4254.68	5083.05	5052.20	2999.68	4269.41	4317.28
(90,9)	5561.41	6182.67	6385.50	4055.41	5361.61	5505.68
(90,10)	5151.57	5909.32	5970.25	3790.23	4988.01	5073.74
(90,11)	4835.84	5690.46	5712.85	3539.85	4762.62	4837.65

Table 5-11 Average total waiting time for each problem case - Depot at corner

	Square-Corner	Rectangle-Vertical-Corner	Rectangle-Horizontal-Corner	Circle-Corner	Triangle-Corner	Sector-Corner
(50,5)	4589.16	5766.34	5599.24	3787.86	3576.28	3852.25
(50,6)	4319.46	5633.57	5472.26	3510.43	3453.33	3716.53
(50,7)	4162.69	5546.33	5383.83	3369.28	3400.69	3637.01
(60,6)	5453.92	6923.61	6986.56	4477.68	4612.65	4412.90
(60,7)	5187.33	6796.04	6841.25	4224.85	4484.86	4305.89
(60,8)	5030.97	6699.63	6764.12	4046.57	4402.69	4224.97
(70,7)	6250.79	7884.20	7958.50	5252.11	5197.72	5278.34
(70,8)	5950.84	7764.44	7817.82	4986.25	5054.47	5141.56
(70,9)	5756.87	7680.64	7760.98	4807.48	4978.08	5054.54
(80,8)	6895.80	8871.27	9170.19	5983.47	5863.61	5752.36
(80,9)	6626.11	8733.88	9038.86	5788.51	5749.33	5636.16
(80,10)	6474.16	8651.30	8952.67	5563.27	5685.27	5570.82
(90,9)	7924.80	9946.46	10070.24	6582.14	6532.11	6885.53
(90,10)	7690.39	9817.78	9946.13	6325.27	6428.87	6777.50
(90,11)	7551.09	9744.89	9880.31	6160.11	6362.76	6695.18

Table 5-12 ANOVA for depot at center situation

The ANOVA Procedure

Dependent Variable: cost

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	316003384.4	63200676.9	87.46	<.0001
Error	894	645285480.5	722604.1		
Corrected Total	899	961288864.9			

Means with the same letter are not significantly different.

Duncan Grouping	Mean	N	type
I	4753.53	150	5
I			
I	4680.36	149	3
II	4123.19	150	1
II			
II	4068.22	150	9
II			
II	4061.21	150	11
III	2941.53	150	7

Table 5-13 ANOVA for depot at corner situation

Dependent Variable: cost

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	1362504817	272500963	164.17	<.0001
Error	894	1483899193	1659842		
Corrected Total	899	2846404010			

Means with the same letter are not significantly different.

Duncan Grouping	Mean	N	type
I	7842.9	150	6
I			
I	7764.0	150	4
II	5991.0	150	2
III	5129.4	150	12
III			
III	5052.2	150	10
III			
III	4991.0	150	8

Table 5-14 ANOVA grouping for each problem case - Depot at center

	Square-Center	Rectangle-Vertical-Center	Rectangle-Horizontal-Center	Circle-Center	Triangle-Center	Sector-Center
(50,5)	II	I	I	III	II	II
(50,6)	II	I	I	III	II	II
(50,7)	II	I	I	III	II	II
(60,6)	II	I	I	III	II	II
(60,7)	II	I	I	III	II	II
(60,8)	II	I	I	III	II	II
(70,7)	II	I	I	III	II	II
(70,8)	II	I	I	III	II	II
(70,9)	II	I	I	III	II	II
(80,8)	II	I	I	III	II	II
(80,9)	II	I	I	III	II	II
(80,10)	II	I	I	III	II	II
(90,9)	II	I	I	III	II	II
(90,10)	II	I	I	III	II	II
(90,11)	II	I	I	III	II	II

Table 5-15 ANOVA grouping for each problem case - Depot at corner

	Square-Corner	Rectangle-Vertical-Corner	Rectangle-Horizontal-Corner	Circle-Corner	Triangle-Corner	Sector-Corner
(50,5)	II	I	I	III	III	III
(50,6)	II	I	I	III	III	III
(50,7)	II	I	I	III	III	III
(60,6)	II	I	I	III	III	III
(60,7)	II	I	I	III	III	III
(60,8)	II	I	I	IV	III	IV
(70,7)	II	I	I	III	III	III
(70,8)	II	I	I	III	III	III
(70,9)	II	I	I	III	III	III
(80,8)	II	I	I	III	III	III
(80,9)	II	I	I	III	III	III
(80,10)	II	I	I	III	III	III
(90,9)	II	I	I	III	III	III
(90,10)	II	I	I	IV	IV	III
(90,11)	II	I	I	IV	III	III

Table 5-16 Center vs. corner regression analysis result

The REG Procedure

Model: MODEL1
Dependent Variable: m_cost1

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	212672711	212672711	300.11	<.0001
Error	88	62360630	708644		
Corrected Total	89	275033342			

Root MSE	841.80967	R-Square	0.7733
Dependent Mean	6128.41367	Adj R-Sq	0.7707
Coeff Var	13.73618		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-126.50467	371.80414	-0.34	0.7345
m_cost	1	1.52344	0.08794	17.32	<.0001

Chapter VI

SUMMARY, CONCLUSION AND FUTURE RESEARCH DIRECTIONS

6.1 Thesis summary

In this research, we extend the traditional Vehicle Routing Problem (VRP) to the Service Level Based Vehicle Routing Problem (SLB-VRP) by integrating the concern of the customer service level into the objective function. We construct several mathematical models and develop a neural networks based heuristic algorithm for the SLB-VRP. Furthermore, we study the operational implications related to the SLB-VRP, which can provide some guidelines while implementing a distribution/warehousing system.

In Chapter I, we define the SLB-VRP, and then discuss the research motivations and research methodology. We find that the introduction of SLB-VRP can be supported by several economical and social factors. Then several research questions are raised, including model construction, solution technique, and concerns for operational implication. We outline our research methodology for these research questions and discuss some expected research results.

In Chapter II, we provide a literature review. We first propose a classification scheme for VRP. In our classification scheme, a VRP model consists of five elements: objective function, depot, vehicle, customer, and relationship. Different combinations of options in those five elements lead to various different VRP models studied in the literature. Some popular solution techniques are summarized in this chapter. These solution techniques can be categorized as two groups. The first group consists of exact

algorithms. These solution techniques are further grouped as direct tree search method, dynamic programming method, and integer programming method. The second group of solution techniques includes heuristic algorithms. In addition to some classical heuristic algorithms, we also discuss several metaheuristics.

In Chapter III, we construct three mathematical models for SLB-VRP. The first and second models take the traditional approach in handling the routing problem by modeling the traffic flow. The first model is a three-index model. Its decision variable corresponds to a potential trip and specifies the beginning customer, the ending customer, and the vehicle. The second model is a compact version of the first model. It is a two-index model. Its decision variable specifies only the beginning customer and the ending customer. The third model deals with the routing problem by studying the sequence of customers in a route. Its decision variable x_{ik} indicates whether the ith customer takes the kth place in the sequence of customers. Since that a trip from customer i to customer j happens or not is determined by the value of $x_{ik}*x_{j,k+1}$, the third model is a quadratic model. In addition to modeling the basic SLB-VRP, we suggest the ways to model some variations for each mathematical model. The variations considered in this research include different objective functions, time windows, and capacity requirements. We also study the difference between the traditional VRP and SLB-VRP using the two-index model. The results reveal significant differences between VRP and SLB-VRP for both total traveled distance and total waiting time.

In Chapter IV, we develop a Hopfield networks based heuristic algorithm. Our algorithm is developed based on the third model because Hopfield networks require a quadratic objective function. For each decision variable x_{ik}, we build a neuron. The

status of this neuron indicates if the ith customer takes the kth place in a route. In order to fit the SLB-VRP into the Hopfield networks structure, we assume that each route can have up to n-1 customers, where n is the total number of customers, and we assign an artificial depot for each route to indicate the end of that route. Totally, we have (n+m)*m*n neurons and they are organized as a (n+m) row times m*n columns network, where n is the number of customers and m is the number of vehicles. Our algorithm follows the regular Hopfield networks computing process. At each step, a neuron is randomly picked up and its status is determined by the status of its neighbors. The solution that is represented by the status of neurons will gradually converge to one local optimal point.

To improve the efficiency of our algorithm, we estimate the maximum and minimum number of customers in a route. These estimations can help us to simplify the network structure and speed up the computing process. In Property 1, we prove that the maximum number of customers in a route is $\frac{n+(m-1)*(\sqrt{\frac{2n}{m}+1}*Z_{\alpha}*\frac{\sigma}{\mu}+1)}{m}$. We assume that the length of a trip follows a distribution with mean μ and standard deviation σ, where n is the number of customers, m is the number of vehicles and the area under the standard normal curve to the left of Z_{α} is $1-\alpha$. With this estimation, the network structure can be simplified since now we do not need to assume that each route can have up to n-1 customers. Similarly, we estimate the minimum number of customers in a route as $\frac{n-(m-1)*(\sqrt{\frac{2n}{m}+1}*Z_{\alpha}*\frac{\sigma}{\mu}+1)}{m}$ in Property 2. With this estimation, we do not

need to consider those routes whose number of customers is below this estimated lower bound; therefore, the computation time can further be reduced.

We also improve our algorithm by modifying the sampling process. At each step, if we turn on one neuron x_{ik}, the neurons in the ith row and kth column will be prevented from being picked up in the following steps. In other words, after we activate one neuron, we will not deactivate this neuron in the following steps; therefore, we can reach a feasible solution more quickly. To alleviate the lack of option to deactivate a neuron, we attach an oscillating process after we obtain a feasible solution. In the oscillating process, we manipulate the threshold value of neurons to deactivate some unfavorable ones and then start over the sampling process to generate another feasible solution.

Our algorithm is organized as a multiple run algorithm. In each run, we first use the Hopfield networks computing process to generate a feasible solution, then apply the oscillating process to delete some long trips and start the Hopfield networks computing process again. This procedure terminates when the oscillating process can not delete any trip from the generated solution. We then apply a two-opt exchanging procedure to do a local search for this solution. The resulted solution is recorded as the solution of this run. The best solution is checked after a certain number of runs. The whole algorithm stops if there is not much improvement in the best solution.

In Chapter V, we conduct experiments to test the performance of our algorithm. We consider 18 problem cases. Each problem case is specified by a combination of number of customers (n) and number of vehicles (m). The 18 (n, m) combinations are: (50, 5), (50, 6), (50, 7), (60, 6), (60, 7), (60, 8), (70, 7), (70, 8), (70, 9), (80, 8), (80, 9), (80, 10), (90, 9), (90, 10), (90, 11), (100, 10), (100, 11), and (100, 12). For each problem

case, we generate 10 problem instances in a 100*100 service region. The results of our algorithm are first measured by an order statistics based estimation of the optimal solution. Our algorithm has less than 1% error on average according to this measurement. We then compare our results with the results from a modified Savings algorithm and a modified Sweep algorithm. Our algorithm can produce better solutions in most problem cases. We also develop a performance grading indicator for SLB-VRP. The results of our algorithm show excellent performances based on this indicator.

We also study the operational implications of SLB-VRP in Chapter V. We are interested in studying the effects of service region shape and location of depot on the total waiting time. To study the effect of service region shape, we consider 6 shapes: Square, Rectangle-Vertical, Rectangle-Horizontal, Circle, Triangle, and Sector. To study the effect of the location of the depot, we consider the case that the depot is at center and the case that the depot is at corner. Therefore, we have 12 types of service region. The area of each service region is fixed to be 10,000. We consider 15 problem cases: (50, 5), (50, 6), (50, 7), (60, 6), (60, 7), (60, 8), (70, 7), (70, 8), (70, 9), (80, 8), (80, 9), (80, 10), (90, 9), (90, 10), and (90, 11). We generat 10 problem instances for each problem case. We find that the circle shape has the least total waiting time and that the depot at corner case has 50% longer total waiting time than the depot at center case.

6.2 Thesis conclusion

The first major contribution of this study is that we extend the VRP to the SLB-VRP. We build two linear programming models and one quadratic programming model for the SLB-VRP. We also show that some variations of the basic SLB-VRP, such

as different objective function and side-constraints, can be incorporated into our modeling framework.

The second major contribution is that we develop a Hopfield networks based heuristic algorithm to solve the SLB-VRP. The motivations for us to develop such an algorithm are partly due to the NP-completeness of SLB-VRP and partly due to our desire to develop a new solution technique in this field. In this study, we show that we can apply the Hopfield networks framework on SLB-VRP with a little bit of sacrifice in efficiency. The sacrificed efficiency is compensated later by several improvement procedures. The most important improvement procedure is based on our estimations of the maximum and minimum number of customers in a route. We use these estimations to reduce the complexity of our networks and hence save some computation time.

The third major contribution comes from our study of the operational implications of SLB-VRP. We consider two factors that should be concerned while designing a warehousing/distributing system. They are the shape of service region and the location of depot. We find that the effect of the shape of service region is significant, i.e., that different shapes of service region will lead to significant different total waiting time. Our finding suggests that potential performance improvement is possible if we take the shape factor into consideration while dividing service regions. To study the effect of the location of the depot, we consider two situations: depot at corner and depot at center. We obtain the total waiting time on both situations for each problem instance. The result of a regression analysis reveals that there exists strong correlation between the total waiting time in these two situations. In our experiments, the relationship is that the total waiting time if the depot is at the corner is about 50% more than the total waiting time if the

depot is set at the center. This finding can help us study the trade-off between a centralized warehousing system and a decentralized warehousing system. Besides, this finding can help us to have insight into the location-routing integration problem.

6.3 Future research directions

The most imminent research question is to study how to avoid being trapped in a local optimal solution. Our algorithm searches the solution space by updating the status of neurons. At each step, the status of one neuron will be checked and the value of the energy function will be decreased or unchanged. If the current solution is a local optimal solution, we will not be able to move away from this solution since its neighbors have higher values of the energy function. To address this concern, our algorithm adopts a multiple run mechanism. In each run, the initial solution is randomly picked up so that we can search different regions in the solution space. Although this mechanism lets us avoid being trapped in a limited region, it is time consuming and does not take advantage of the self-adaptation capability of neural networks. Ideally, we should let the neural networks escape the local optimal solution by themselves and accordingly be able to implement a single run mechanism in our algorithm.

To escape the local optimal solution, our neural networks should be able to move from one solution to a worse solution. The typical approach in handling this situation is to let neurons change their status according to a certain probability distribution; therefore, a neuron may be turned on even if its activation is less than its threshold value. Consequently, we can jump from a local optimal solution to one of its neighbors and have a chance to leave the local region. In the literature, this kind of neural networks is called

the Boltzman machine. For our SLB-VRP, the probability distribution can be determined based on the travelling distances among customers.

It is also an interesting research direction to extend the application of our algorithm to those variations of SLB-VRP. In Chapter III, we have discussed some variations of the basic type of SLB-VRP. For those variations with side constraints, there is a straightforward way to apply our algorithm. We can let the neurons carry out the feasibility check, i.e., while we turn on one neuron, we will require that those side constraints be satisfied. Furthermore, we might be able to derive some properties out of those side constraints and hence use them to reduce the solution space or facilitate the computation process. For those variations related to the objective function, we need to first study how to turn the objective function into the energy function form and then design the network accordingly.

Another interesting future research direction is to study the stochastic SLB-VRP. In the stochastic SLB-VRP, the demand arrives stochastically in terms of both location and quantity. Some statistical characteristics, such as the expected waiting time, and expected amount of demand, will be useful while designing and operating the warehousing/distributing system. How to estimate those characteristics will be an active research topic in this field. An effective algorithm will also be needed to quickly respond to the changing delivery requirements. To achieve the quick response in our neural networks based algorithm, some type of memory must be used for a neuron to recognize a similar environment and then decide its status accordingly. So we need to integrate a learning mechanism into our algorithm. During the learning process, some memory will be developed and kept in each neuron. How to design the learning process and use the

developed memory will be an interesting research question if we want to apply our algorithm on stochastic SLB-VRP.

Appendix A

Modified Savings Algorithm

Step 1: Initialize, form n routes, with each route containing only one customer

Step 2: Calculate savings for each pair of routes.

Suppose we have two routes: route 1 and route 2. If we merge these two routes by appending route 2 at the end of route 1, the savings is defined as: Waiting time of new route - (Waiting time of route 1 + Waiting time of route 2).

Note that the savings are not symmetric.

Step 3: Pick up the smallest savings and merge the associated two routes

Step 4: Repeat step 2 and step 3 until there are m routes.

Step 5: Within each route, exchange the position of customers by a 2-opt procedure to reduce the waiting time.

Step 6: Exchange customers in different routes to reduce the waiting time.

Step 7: Repeat step 5 and step 6 until no further improvement is available.

Appendix B

Modified Sweep Algorithm

Step 1: Calculate polar coordinates for each customer.

Step 2: Calculate the estimated maximum number of customers in a route and the estimated minimum number of customers in a route as stated in Property 1 and Property 2, Chapter IV.

Step 3: Randomly decide how many customers are in each of those m routes, using the result of step 2 as upper bound and lower bound.

Step 4: Form m routes with a 360 degree sweep. Note that we terminate a route while the number of customers in this route achieve the number assigned at step 3.

Step 5: Within each route, exchange the position of customers by a 2-opt procedure to reduce the waiting time.

Step 6: Exchange customers in two adjacent routes by a 2-opt procedure to reduce the waiting time.

Step 7: Repeat step 5 and step 6 until no further improvement is available.

Appendix C

An Example for Performance Indicator

EXAMPLE

There are 8 customers and 2 vehicles. The depot is at the origin (0, 0). The coordinates of those 8 customers are:

(33.454, 33.736), (72.77, 20.493),

(82.498, 5.054), (58.182, 1.069),

(52.324, 1.983), (83.877, 63.085),

(83.677, 97.949), (5.827, 17.09).

The traveling distance matrix is:

	0	1	2	3	4	5	6	7	8
0	0	47.51	75.6	82.65	58.19	52.36	104.95	128.82	18.05
1		0	41.48	56.81	40.97	36.93	58.34	81.52	32.25
2			0	18.24	24.29	27.58	44.01	78.22	67.02
3				0	24.64	30.32	58.04	92.9	77.6
4					0	5.92	67.12	100.17	54.75
5						0	68.76	100.95	48.88
6							0	34.86	90.59
7								0	112.24
8									0

Appendix C

An Example for Performance Indicator (continued)

The mean is 59.54 and the standard deviation is 29.34. The optimal solution consists of the following two routes:

Route 1: 0 - 8 - 1 - 6 - 7 - 0,

Route 2: 0 - 5 - 4 - 2 - 3 - 0.

The total waiting time is 614.51. Since each route consists of 4 customers, there are 4*(4+1)/2=10 traveling distance matrix elements for each route and the total number is 20. So the average value is 614.51/20=30.726 and the indicator is (30.726-59.54)/29.34=-0.982.

Appendix D

Univariate Analysis for Performance Indicator

The UNIVARIATE Procedure
Variable: indicator
Moments

N	720	Sum Weights	720
Mean	-0.7345782	Sum Observations	-528.89633
Std Deviation	0.34189516	Variance	0.1168923
Skewness	0.77842874	Kurtosis	0.53232588
Uncorrected SS	472.561304	Corrected SS	84.0455635
Coeff Variation	-46.543055	Std Error Mean	0.01274168

Basic Statistical Measures

Location		Variability	
Mean	-0.73458	Std Deviation	0.34190
Median	-0.78585	Variance	0.11689
Mode	.	Range	1.91054
		Interquartile Range	0.45824

Quantiles (Definition 5)

Quantile	Estimate
100% Max	0.5653355
99%	0.2172846
95%	-0.0966509
90%	-0.2860529

The UNIVARIATE Procedure
Variable: indicator

Quantile	Estimate
75% Q3	-0.5295905
50% Median	-0.7858476
25% Q1	-0.9878342
10%	-1.1344955
5%	-1.1894170
1%	-1.3018812
0% Min	-1.3452085

Appendix E

Examples

Example 1

Description:

Shape:	Square	**Depot:**	Center
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 5569.46

CPU Time: 582 seconds

Route 1	0 - 22 - 66 - 35 - 43 - 40 - 33 - 27 - 67 - 0
Route 2	0 - 89 - 9 - 1 - 64 - 87 - 65 - 85 - 74 - 78 - 39 - 0
Route 3	0 - 72 - 55 - 51 - 44 - 48 - 20 - 82 - 54 - 12 - 42 - 0
Route 4	0 - 50 - 34 - 86 - 80 - 28 - 84 - 60 - 88 - 29 - 59 - 24 - 0
Route 5	0 - 8 - 57 - 45 - 30 - 62 - 49 - 70 - 53 - 25 - 69 - 56 - 0
Route 6	0 - 71 - 5 - 75 - 79 - 31 - 47 - 10 - 63 - 2 - 0
Route 7	0 - 61 - 77 - 32 - 3 - 90 - 58 - 36 - 52 - 4 - 6 - 83 - 0
Route 8	0 - 68 - 37 - 18 - 26 - 73 - 21 - 7 - 17 - 13 - 81 - 0
Route 9	0 - 15 - 16 - 23 - 76 - 46 - 19 - 38 - 14 - 41 - 11 - 0

Appendix E

Examples (continued)

Example 1 - Customer Locations:

	X	Y		X	Y		X	Y
1	-37.58	41.78	**31**	2.89	27.69	**61**	47.36	39.80
2	4.86	2.08	**32**	13.22	30.63	**62**	31.97	43.52
3	12.32	45.13	**33**	18.50	-43.86	**63**	8.11	4.85
4	-9.23	12.81	**34**	16.12	-19.16	**64**	-40.85	31.19
5	-3.83	49.78	**35**	36.43	-37.49	**65**	-40.67	23.60
6	-11.35	-0.66	**36**	-14.43	24.18	**66**	43.34	-47.56
7	-29.69	-37.62	**37**	-39.19	-39.05	**67**	11.10	-28.56
8	43.85	18.56	**38**	-2.30	-27.35	**68**	-32.46	-30.03
9	-47.50	46.25	**39**	-14.93	9.48	**69**	15.31	22.77
10	11.29	20.13	**40**	22.75	-39.71	**70**	40.50	23.34
11	-10.66	-15.56	**41**	-6.18	-18.62	**71**	-24.00	46.28
12	-22.01	-19.32	**42**	-25.73	-8.60	**72**	-38.86	-45.38
13	-18.90	-24.31	**43**	32.77	-35.96	**73**	-13.15	-31.59
14	-4.08	-23.04	**44**	-43.83	-27.64	**74**	-31.52	22.18
15	44.68	15.83	**45**	24.14	49.25	**75**	-1.71	33.65
16	49.81	15.63	**46**	2.56	-33.13	**76**	20.08	-31.58
17	-21.95	-27.40	**47**	9.59	31.69	**77**	42.89	40.01
18	-30.80	-43.33	**48**	-38.57	-18.59	**78**	-29.15	20.50
19	0.66	-29.06	**49**	35.76	37.64	**79**	2.87	30.33
20	-43.53	-9.83	**50**	8.35	-43.36	**80**	29.82	19.49
21	-16.51	-32.86	**51**	-41.69	-35.09	**81**	-13.83	-24.34
22	43.39	-47.72	**52**	-11.85	14.05	**82**	-42.69	-6.61
23	49.69	5.46	**53**	29.93	26.51	**83**	-11.19	-4.40
24	7.62	-5.02	**54**	-26.03	-17.72	**84**	27.46	11.59
25	22.15	18.94	**55**	-45.40	-44.10	**85**	-39.75	22.43
26	-22.60	-43.51	**56**	1.46	6.99	**86**	21.72	-14.96
27	17.82	-42.88	**57**	36.08	49.28	**87**	-44.15	30.50
28	28.02	14.45	**58**	-16.97	30.34	**88**	37.78	4.48
29	40.44	-2.15	**59**	22.10	-8.06	**89**	-44.16	20.99
30	26.58	42.26	**60**	32.34	10.83	**90**	-4.51	42.17

Appendix E

Examples (continued)

Example 2

Description:

Shape:	Square	**Depot:**	Corner
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 8134.30

CPU Time: 244 seconds

Route 1	0 - 62 - 24 - 11 - 5 - 33 - 76 - 19 - 83 - 6 - 84 - 88 - 2 - 0
Route 2	0 - 66 - 50 - 10 - 42 - 77 - 12 - 28 - 4 - 64 - 39 - 0
Route 3	0 - 71 - 55 - 68 - 41 - 59 - 85 - 14 - 26 - 43 - 0
Route 4	0 - 69 - 56 - 31 - 8 - 90 - 48 - 70 - 89 - 73 - 61 - 0
Route 5	0 - 51 - 29 - 13 - 30 - 16 - 82 - 52 - 54 - 49 - 60 - 0
Route 6	0 - 32 - 21 - 79 - 15 - 3 - 35 - 22 - 44 - 17 - 37 - 72 - 23 - 0
Route 7	0 - 45 - 34 - 46 - 9 - 86 - 20 - 47 - 57 - 53 - 0
Route 8	0 - 36 - 40 - 25 - 1 - 38 - 78 - 80 - 87 - 58 - 81 - 0
Route 9	0 - 65 - 18 - 74 - 27 - 67 - 7 - 75 - 63 - 0

Appendix E

Examples (continued)

Example 2 - Customer Locations:

	X	Y		X	Y		X	Y
1	83.36	76.40	**31**	52.16	74.72	**61**	13.56	31.67
2	2.18	56.53	**32**	95.32	85.85	**62**	71.43	97.43
3	92.01	53.46	**33**	36.94	84.07	**63**	36.77	11.32
4	53.64	5.30	**34**	10.21	67.49	**64**	45.90	3.70
5	39.78	86.79	**35**	90.49	47.32	**65**	85.79	80.91
6	6.38	66.99	**36**	83.96	95.24	**66**	61.89	67.47
7	84.19	18.53	**37**	74.84	23.73	**67**	89.57	17.01
8	47.09	77.76	**38**	80.01	74.06	**68**	52.52	63.47
9	41.11	48.91	**39**	37.36	1.32	**69**	75.96	97.11
10	64.63	39.33	**40**	80.87	87.45	**70**	33.59	74.98
11	43.58	89.87	**41**	41.51	63.87	**71**	76.86	89.60
12	61.60	23.44	**42**	66.79	28.51	**72**	22.64	15.32
13	99.28	11.22	**43**	9.61	31.95	**73**	19.88	58.16
14	23.74	51.84	**44**	86.21	32.00	**74**	96.42	48.23
15	93.36	63.59	**45**	8.60	61.56	**75**	49.68	16.16
16	94.27	7.17	**46**	22.99	72.53	**76**	27.47	85.67
17	78.47	30.29	**47**	26.74	48.30	**77**	61.39	28.37
18	97.12	62.12	**48**	33.77	73.38	**78**	81.11	67.58
19	17.60	81.64	**49**	34.98	3.69	**79**	95.89	72.29
20	28.22	49.74	**50**	76.11	52.52	**80**	72.89	56.97
21	96.76	79.48	**51**	99.87	40.88	**81**	41.89	39.61
22	86.90	35.74	**52**	73.01	17.42	**82**	83.63	15.87
23	4.63	7.08	**53**	16.88	31.76	**83**	13.58	79.12
24	51.33	95.63	**54**	62.67	8.97	**84**	4.27	63.16
25	83.92	80.14	**55**	62.63	74.62	**85**	24.01	57.18
26	10.91	43.70	**56**	63.65	86.14	**86**	35.92	53.27
27	95.31	28.46	**57**	22.33	42.22	**87**	54.47	38.96
28	56.42	9.00	**58**	45.15	37.53	**88**	1.82	56.64
29	99.15	24.45	**59**	35.10	63.88	**89**	27.63	74.33
30	94.28	3.97	**60**	29.75	4.05	**90**	37.64	67.19

Appendix E

Examples (continued)

Example 3

Description:

Shape:	Rectangle-Vertical	**Depot:**	Center
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 5973.95

CPU Time: 762 seconds

Route 1	0 - 29 - 45 - 89 - 60 - 8 - 32 - 40 - 7 - 80 - 15 - 84 - 28 - 85 - 0
Route 2	0 - 86 - 63 - 39 - 35 - 73 - 1 - 81 - 43 - 0
Route 3	0 - 37 - 22 - 25 - 3 - 78 - 62 - 75 - 11 - 61 - 16 - 36 - 66 - 88 - 0
Route 4	0 - 48 - 76 - 69 - 19 - 46 - 33 - 74 - 50 - 6 - 41 - 47 - 0
Route 5	0 - 51 - 26 - 38 - 54 - 70 - 30 - 49 - 0
Route 6	0 - 13 - 68 - 53 - 65 - 52 - 72 - 87 - 18 - 44 - 10 - 31 - 64 - 0
Route 7	0 - 90 - 17 - 82 - 4 - 9 - 59 - 12 - 27 - 2 - 0
Route 8	0 - 24 - 21 - 23 - 56 - 42 - 20 - 34 - 55 - 0
Route 9	0 - 67 - 77 - 57 - 5 - 58 - 83 - 71 - 79 - 14 - 0

Appendix E

Examples (continued)

Example 3 - Customer Locations:

	X	Y		X	Y		X	Y
1	20.47	43.43	**31**	-5.06	27.66	**61**	17.69	16.18
2	4.15	-4.06	**32**	-15.08	-46.91	**62**	23.51	-28.06
3	22.78	-69.52	**33**	-19.98	-60.03	**63**	-17.07	71.21
4	-8.43	-52.27	**34**	9.63	-38.63	**64**	-5.53	16.35
5	24.31	61.84	**35**	5.46	58.47	**65**	9.20	93.11
6	-18.12	0.26	**36**	7.76	15.05	**66**	5.55	11.34
7	-12.32	-26.05	**37**	22.35	-93.54	**67**	1.65	80.73
8	-19.32	-51.18	**38**	4.68	-76.18	**68**	-1.20	90.52
9	5.34	-37.09	**39**	-11.92	75.17	**69**	-1.78	-92.80
10	-6.82	36.06	**40**	-15.03	-38.68	**70**	-3.56	-36.55
11	21.73	2.60	**41**	-20.06	0.32	**71**	15.23	62.60
12	18.79	-17.05	**42**	14.87	-53.48	**72**	5.44	75.53
13	-23.79	95.33	**43**	12.59	8.73	**73**	16.58	46.52
14	11.74	21.74	**44**	-3.74	39.43	**74**	-18.86	-35.00
15	-1.89	-13.21	**45**	-23.57	-60.19	**75**	23.59	-5.31
16	17.48	17.52	**46**	-16.00	-67.09	**76**	-6.02	-99.77
17	-11.05	-61.94	**47**	-20.96	1.65	**77**	8.72	72.76
18	-0.93	53.97	**48**	-10.10	-95.19	**78**	19.69	-43.35
19	-4.73	-85.33	**49**	0.67	-17.24	**79**	5.62	25.26
20	11.93	-45.47	**50**	-24.25	-15.43	**80**	-10.68	-23.73
21	7.35	-73.22	**51**	17.86	-92.46	**81**	22.68	28.49
22	17.74	-88.00	**52**	6.92	83.40	**82**	-10.13	-58.45
23	15.27	-66.08	**53**	6.31	97.60	**83**	14.71	64.04
24	0.68	-92.58	**54**	-4.11	-60.80	**84**	0.37	-8.78
25	14.06	-85.74	**55**	5.19	-19.39	**85**	-0.34	2.80
26	9.81	-86.47	**56**	11.89	-64.69	**86**	-19.06	7.07
27	23.84	-9.50	**57**	10.45	74.67	**87**	3.50	69.45
28	1.57	-1.45	**58**	18.86	66.39	**88**	4.38	9.12
29	-20.74	-96.74	**59**	13.84	-31.55	**89**	-23.52	-56.49
30	-4.44	-27.28	**60**	-20.98	-54.63	**90**	-19.24	-89.51

Appendix E

Examples (continued)

Example 4

Description:

Shape:	Rectangle-Vertical	**Depot:**	Corner
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 9870.99

CPU Time: 127 seconds

Route 1	0 - 48 - 86 - 50 - 84 - 89 - 27 - 1 - 25 - 67 - 39 - 73 - 16 - 61 - 0
Route 2	0 - 17 - 28 - 44 - 59 - 6 - 10 - 18 - 0
Route 3	0 - 4 - 2 - 29 - 26 - 13 - 83 - 21 - 49 - 66 - 45 - 72 - 76 - 34 - 0
Route 4	0 - 33 - 71 - 68 - 47 - 22 - 46 - 56 - 15 - 38 - 23 - 63 - 19 - 0
Route 5	0 - 75 - 51 - 88 - 78 - 32 - 36 - 80 - 70 - 77 - 55 - 0
Route 6	0 - 3 - 9 - 40 - 82 - 12 - 69 - 5 - 31 - 42 - 90 - 11 - 0
Route 7	0 - 41 - 64 - 62 - 60 - 53 - 87 - 8 - 0
Route 8	0 - 35 - 81 - 24 - 37 - 14 - 85 - 20 - 65 - 0
Route 9	0 - 7 - 30 - 54 - 43 - 79 - 57 - 74 - 58 - 52 - 0

Appendix E

Examples (continued)

Example 4 - Customer Locations:

	X	Y		X	Y		X	Y
1	7.44	118.97	**31**	42.09	73.73	**61**	3.75	27.12
2	14.10	175.41	**32**	20.07	178.44	**62**	46.33	112.70
3	39.99	197.86	**33**	28.34	170.72	**63**	16.49	19.90
4	29.39	179.57	**34**	6.02	14.09	**64**	42.07	125.95
5	38.96	108.25	**35**	32.65	140.27	**65**	25.20	6.48
6	44.41	45.75	**36**	18.19	178.37	**66**	4.77	65.18
7	11.01	192.11	**37**	45.64	12.39	**67**	14.83	89.86
8	18.59	28.85	**38**	25.53	25.68	**68**	30.33	110.97
9	48.63	188.60	**39**	18.57	76.90	**69**	38.29	116.79
10	41.46	27.93	**40**	44.57	172.71	**70**	17.51	158.97
11	20.34	5.42	**41**	37.15	166.83	**71**	28.71	169.23
12	38.53	132.95	**42**	41.88	64.52	**72**	13.48	34.31
13	3.80	98.18	**43**	22.73	161.74	**73**	16.42	54.79
14	43.61	2.77	**44**	41.53	46.00	**74**	21.56	80.68
15	26.22	53.08	**45**	4.70	53.92	**75**	38.28	193.51
16	8.43	36.57	**46**	24.68	70.80	**76**	13.80	32.92
17	36.40	153.60	**47**	29.28	100.72	**77**	9.20	72.49
18	31.22	15.16	**48**	5.24	188.19	**78**	23.73	183.56
19	7.33	4.99	**49**	4.96	72.07	**79**	21.62	119.45
20	36.08	9.27	**50**	1.69	162.52	**80**	16.67	161.69
21	6.38	88.36	**51**	34.06	194.77	**81**	49.00	32.55
22	24.05	78.35	**52**	0.22	0.17	**82**	45.83	157.01
23	22.50	21.53	**53**	43.39	93.85	**83**	3.69	96.19
24	45.01	16.86	**54**	18.22	186.07	**84**	2.64	142.15
25	6.56	114.56	**55**	4.26	35.93	**85**	37.88	11.05
26	6.68	143.87	**56**	30.67	64.53	**86**	8.78	173.34
27	11.08	125.58	**57**	23.54	90.99	**87**	28.53	71.68
28	43.31	77.55	**58**	13.86	38.55	**88**	25.47	182.45
29	10.57	145.94	**59**	42.98	45.87	**89**	4.03	135.87
30	12.17	187.81	**60**	42.50	107.23	**90**	23.02	12.51

Appendix E

Examples (continued)

Example 5

Description:

Shape: Rectangle-Horizontal **Depot:** Center

Customers: 90 **Vehicles:** 9

Solution:

Total Waiting Time: 6930.66

CPU Time: 500 seconds

Route 1 0 - 15 - 12 - 31 - 69 - 45 - 84 - 66 - 58 - 24 - 21 - 90 - 9 - 40 - 0

Route 2 0 - 52 - 20 - 11 - 76 - 36 - 37 - 43 - 68 - 56 - 19 - 27 - 70 - 2 - 0

Route 3 0 - 47 - 1 - 81 - 17 - 86 - 78 - 14 - 33 - 54 - 65 - 63 - 0

Route 4 0 - 30 - 5 - 89 - 10 - 28 - 71 - 18 - 82 - 75 - 0

Route 5 0 - 16 - 39 - 38 - 35 - 73 - 67 - 25 - 59 - 0

Route 6 0 - 32 - 80 - 57 - 6 - 23 - 51 - 60 - 0

Route 7 0 - 26 - 46 - 61 - 53 - 29 - 44 - 4 - 64 - 72 - 0

Route 8 0 - 42 - 62 - 50 - 55 - 7 - 83 - 88 - 0

Route 9 0 - 8 - 77 - 41 - 85 - 34 - 79 - 3 - 13 - 48 - 49 - 74 - 87 - 22 - 0

Appendix E

Examples (continued)

Example 5 - Customer Locations:

	X	Y		X	Y		X	Y
1	-78.33	-7.72	**31**	72.52	23.23	**61**	-12.84	-24.35
2	37.74	17.05	**32**	87.21	-6.30	**62**	-12.23	-19.89
3	-71.25	-10.79	**33**	-17.77	18.91	**63**	5.68	17.22
4	-36.32	6.71	**34**	-76.40	-15.40	**64**	-35.64	15.05
5	-87.57	8.13	**35**	93.70	-2.40	**65**	-7.79	24.56
6	40.41	-17.38	**36**	54.60	18.67	**66**	77.87	-2.55
7	12.77	-19.85	**37**	57.88	16.17	**67**	83.02	5.75
8	-98.62	19.78	**38**	93.49	-3.58	**68**	66.43	19.71
9	21.59	-5.53	**39**	98.86	-12.44	**69**	72.34	16.00
10	-80.69	13.51	**40**	9.07	-6.89	**70**	49.96	16.57
11	28.72	18.23	**41**	-89.80	-2.50	**71**	-65.23	17.46
12	77.32	21.65	**42**	-45.23	-8.88	**72**	-27.62	19.82
13	-68.18	-5.76	**43**	57.90	20.67	**73**	94.71	7.91
14	-39.28	16.43	**44**	-34.70	5.74	**74**	-41.55	-4.31
15	98.96	6.35	**45**	72.82	6.73	**75**	-24.99	-3.81
16	96.97	-19.54	**46**	-11.11	-24.66	**76**	50.01	23.36
17	-61.06	4.59	**47**	-75.25	7.65	**77**	-88.96	1.80
18	-59.57	16.12	**48**	-64.29	-5.07	**78**	-57.37	5.16
19	56.53	11.96	**49**	-63.48	-4.11	**79**	-74.55	-17.80
20	23.57	24.11	**50**	-7.66	-18.10	**80**	56.69	-18.25
21	48.03	-9.70	**51**	18.17	-23.07	**81**	-75.04	-5.69
22	-22.18	-5.28	**52**	-23.05	24.15	**82**	-54.46	10.87
23	27.09	-24.62	**53**	-20.19	-18.97	**83**	2.63	-24.18
24	65.02	-4.99	**54**	-8.35	23.89	**84**	68.76	4.24
25	61.71	0.29	**55**	16.34	-23.92	**85**	-92.95	-11.95
26	-45.86	-23.96	**56**	65.90	14.63	**86**	-59.95	5.46
27	50.25	11.84	**57**	47.89	-22.27	**87**	-24.96	-5.43
28	-71.54	17.05	**58**	78.99	-4.82	**88**	-3.35	-18.99
29	-32.38	-3.25	**59**	47.05	-6.34	**89**	-85.44	8.46
30	-85.46	17.46	**60**	-0.30	-8.06	**90**	31.69	-12.43

Appendix E

Examples (continued)

Example 6

Description:

Shape: Rectangle-Horizontal **Depot:** Corner

Customers: 90 **Vehicles:** 9

Solution:

Total Waiting Time: 10457.80

CPU Time: 381 seconds

Route 1 0 - 11 - 58 - 53 - 61 - 76 - 65 - 88 - 48 - 0

Route 2 0 - 8 - 46 - 59 - 71 - 35 - 79 - 34 - 68 - 28 - 52 - 60 - 7 - 0

Route 3 0 - 20 - 89 - 17 - 9 - 4 - 40 - 85 - 29 - 74 - 63 - 26 - 0

Route 4 0 - 75 - 67 - 10 - 54 - 25 - 24 - 18 - 62 - 31 - 33 - 41 - 82 - 37 - 0

Route 5 0 - 78 - 55 - 12 - 15 - 83 - 39 - 56 - 0

Route 6 0 - 49 - 16 - 32 - 70 - 22 - 19 - 30 - 6 - 1 - 0

Route 7 0 - 3 - 27 - 5 - 80 - 64 - 77 - 50 - 0

Route 8 0 - 21 - 69 - 2 - 45 - 72 - 23 - 84 - 57 - 86 - 43 - 81 - 14 - 0

Route 9 0 - 47 - 42 - 87 - 38 - 51 - 73 - 44 - 90 - 36 - 13 - 66 - 0

Appendix E

Examples (continued)

Example 6 - Customer Locations:

	X	Y		X	Y		X	Y
1	32.95	4.52	**31**	114.84	32.38	**61**	55.28	37.36
2	186.53	40.41	**32**	174.92	1.39	**62**	117.44	37.98
3	164.54	18.65	**33**	86.21	25.91	**63**	16.81	6.13
4	78.87	29.53	**34**	178.50	7.72	**64**	27.01	28.96
5	74.01	23.76	**35**	181.73	11.83	**65**	50.15	27.56
6	69.33	16.36	**36**	93.62	17.25	**66**	52.52	5.82
7	37.74	9.60	**37**	62.65	16.70	**67**	175.44	27.78
8	195.45	14.95	**38**	161.83	14.18	**68**	165.49	7.71
9	89.63	45.62	**39**	7.55	17.62	**69**	189.67	42.40
10	169.02	31.39	**40**	63.89	25.66	**70**	163.31	4.64
11	78.72	44.59	**41**	82.52	27.15	**71**	182.58	15.84
12	9.81	41.33	**42**	179.76	17.59	**72**	180.20	43.99
13	85.27	12.15	**43**	118.32	29.18	**73**	130.11	20.89
14	107.82	23.44	**44**	117.02	20.58	**74**	31.70	13.10
15	16.56	37.68	**45**	184.41	45.66	**75**	180.30	29.81
16	186.07	2.29	**46**	195.19	19.89	**76**	51.72	33.03
17	94.52	40.88	**47**	184.26	21.67	**77**	24.32	23.62
18	129.99	45.13	**48**	11.08	6.95	**78**	62.96	36.37
19	94.20	24.14	**49**	187.01	0.22	**79**	180.26	9.26
20	99.83	49.77	**50**	4.64	6.87	**80**	59.08	21.79
21	195.73	42.76	**51**	157.86	18.84	**81**	108.02	22.98
22	148.59	4.74	**52**	94.39	13.36	**82**	70.82	18.74
23	171.62	43.91	**53**	52.71	47.05	**83**	20.59	34.02
24	131.04	49.01	**54**	158.54	31.95	**84**	138.71	38.64
25	147.36	40.60	**55**	2.23	47.81	**85**	60.43	24.81
26	8.14	5.50	**56**	0.13	3.20	**86**	121.32	32.73
27	145.17	28.51	**57**	126.25	35.97	**87**	169.39	12.71
28	126.29	11.72	**58**	53.73	49.83	**88**	30.62	23.05
29	49.17	22.26	**59**	191.71	23.84	**89**	96.02	43.38
30	82.64	21.17	**60**	66.49	15.40	**90**	105.11	19.64

Appendix E

Examples (continued)

Example 7

Description:

Shape:	Circle	**Depot:**	Center
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 4291.67

CPU Time: 490 seconds

Route 1	0 - 31 - 60 - 44 - 83 - 38 - 4 - 86 - 70 - 18 - 0
Route 2	0 - 41 - 54 - 48 - 74 - 49 - 81 - 63 - 52 - 89 - 43 - 0
Route 3	0 - 37 - 75 - 32 - 62 - 29 - 6 - 90 - 16 - 21 - 82 - 67 - 30 - 0
Route 4	0 - 12 - 66 - 39 - 50 - 59 - 8 - 3 - 84 - 45 - 17 - 13 - 26 - 0
Route 5	0 - 20 - 34 - 14 - 55 - 7 - 5 - 19 - 68 - 77 - 10 - 0
Route 6	0 - 85 - 23 - 1 - 40 - 78 - 25 - 11 - 72 - 36 - 65 - 15 - 69 - 0
Route 7	0 - 27 - 61 - 56 - 80 - 71 - 53 - 76 - 33 - 0
Route 8	0 - 46 - 9 - 79 - 58 - 57 - 42 - 64 - 47 - 0
Route 9	0 - 51 - 28 - 35 - 87 - 73 - 24 - 88 - 2 - 22 - 0

Appendix E

Examples (continued)

Example 7 - Customer Locations:

	X	Y		X	Y		X	Y
1	5.71	-30.59	**31**	-46.07	-31.63	**61**	-31.51	-45.15
2	10.31	15.71	**32**	17.99	-35.68	**62**	2.01	-34.96
3	33.96	7.19	**33**	-21.87	0.96	**63**	-5.82	13.82
4	4.71	-17.96	**34**	-7.80	54.86	**64**	-6.88	-0.70
5	5.70	37.14	**35**	25.42	46.82	**65**	4.39	9.48
6	-10.99	-30.88	**36**	5.41	9.38	**66**	47.22	-25.34
7	-13.96	23.27	**37**	33.02	-36.69	**67**	-5.75	-6.65
8	36.72	7.25	**38**	3.32	-19.91	**68**	5.93	22.10
9	-49.01	-4.12	**39**	45.81	23.55	**69**	3.03	1.39
10	3.83	18.33	**40**	6.85	-26.79	**70**	12.90	-7.35
11	13.11	2.94	**41**	-42.72	35.15	**71**	-37.45	-20.28
12	49.16	-25.26	**42**	-5.15	8.08	**72**	5.64	7.27
13	13.91	-14.15	**43**	0.10	0.13	**73**	23.63	41.14
14	-20.31	35.53	**44**	-21.28	-12.75	**74**	-30.47	35.00
15	4.92	5.15	**45**	20.24	-21.37	**75**	30.72	-27.67
16	-15.39	-25.15	**46**	-46.54	-20.65	**76**	-26.70	-7.62
17	17.90	-14.57	**47**	-0.35	-0.05	**77**	4.40	20.13
18	5.22	1.99	**48**	-27.15	36.98	**78**	7.77	-24.16
19	10.32	34.03	**49**	-25.09	17.19	**79**	-31.13	2.26
20	-3.08	55.09	**50**	43.66	20.90	**80**	-32.60	-22.79
21	-16.52	-22.13	**51**	28.43	38.68	**81**	-12.94	13.23
22	10.56	14.31	**52**	-4.63	14.38	**82**	-18.18	-21.81
23	5.56	-40.15	**53**	-27.42	-11.36	**83**	-5.40	-19.94
24	19.21	40.05	**54**	-33.33	37.35	**84**	28.78	-23.91
25	20.41	-4.33	**55**	-15.27	27.68	**85**	-20.00	-51.12
26	12.90	-12.22	**56**	-31.02	-34.93	**86**	8.99	-11.66
27	-27.52	-49.19	**57**	-19.29	5.54	**87**	23.32	47.66
28	31.71	44.63	**58**	-20.81	4.50	**88**	20.70	32.31
29	-4.12	-32.56	**59**	35.15	13.41	**89**	0.01	0.48
30	0.41	-1.20	**60**	-45.50	-23.44	**90**	-18.40	-27.84

Appendix E

Examples (continued)

Example 8

Description:

Shape:	Circle	**Depot:**	Corner
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 6643.77

CPU Time: 124 seconds

Route 1	0 - 33 - 69 - 10 - 89 - 87 - 14 - 56 - 72 - 22 - 61 - 34 - 0
Route 2	0 - 52 - 37 - 40 - 53 - 39 - 11 - 29 - 23 - 18 - 66 - 49 - 74 - 0
Route 3	0 - 58 - 36 - 59 - 68 - 64 - 76 - 75 - 82 - 38 - 0
Route 4	0 - 16 - 6 - 35 - 17 - 3 - 9 - 45 - 73 - 2 - 20 - 57 - 7 - 0
Route 5	0 - 42 - 31 - 50 - 71 - 81 - 60 - 30 - 12 - 13 - 86 - 0
Route 6	0 - 55 - 85 - 47 - 51 - 28 - 44 - 19 - 26 - 46 - 77 - 27 - 0
Route 7	0 - 90 - 43 - 78 - 8 - 5 - 63 - 48 - 25 - 0
Route 8	0 - 1 - 62 - 15 - 70 - 67 - 84 - 79 - 24 - 0
Route 9	0 - 41 - 54 - 88 - 21 - 4 - 65 - 83 - 80 - 32 - 0

Appendix E

Examples (continued)

Example 8 - Customer Locations:

	X	Y		X	Y		X	Y
1	-8.85	94.79	31	-15.75	80.55	61	11.83	44.43
2	-22.35	47.05	32	15.71	34.45	62	-8.17	64.67
3	-11.50	74.00	33	-35.75	96.10	63	14.41	62.30
4	12.30	64.76	34	9.94	34.66	64	43.22	51.63
5	14.80	64.73	35	-3.07	83.64	65	26.64	55.08
6	-4.67	88.91	36	45.32	83.53	66	0.73	54.06
7	-7.94	16.55	37	4.92	83.90	67	-12.29	48.50
8	18.36	65.15	38	25.02	18.71	68	50.22	64.62
9	-12.72	64.80	39	-0.17	59.54	69	-44.02	50.52
10	-32.43	50.54	40	7.62	71.74	70	-10.45	58.11
11	-0.93	58.83	41	-51.10	57.03	71	-5.28	76.78
12	-2.17	53.47	42	-17.38	83.21	72	-3.94	47.80
13	-1.69	52.70	43	14.38	93.94	73	-22.53	49.12
14	-15.69	55.65	44	-28.19	43.75	74	-0.65	45.88
15	-4.66	62.78	45	-21.40	51.91	75	46.47	34.40
16	3.59	105.52	46	-24.93	26.73	76	51.57	40.64
17	-8.73	74.32	47	-18.44	81.29	77	-30.61	20.57
18	0.08	54.92	48	13.21	51.15	78	19.19	67.66
19	-32.05	35.40	49	-0.07	50.61	79	-11.89	14.62
20	-20.60	44.40	50	-12.66	79.45	80	21.21	41.99
21	9.25	65.61	51	-20.54	63.70	81	4.10	70.80
22	6.54	47.89	52	3.74	87.54	82	44.02	31.51
23	0.48	55.21	53	8.71	66.31	83	32.08	50.66
24	-7.38	11.17	54	-52.99	41.22	84	-16.89	30.76
25	1.99	30.53	55	-33.00	91.88	85	-32.04	87.33
26	-25.69	29.64	56	-9.46	50.39	86	-3.15	46.87
27	-30.61	17.61	57	-16.57	37.71	87	-17.73	57.41
28	-27.60	54.71	58	30.20	70.75	88	-39.51	31.94
29	0.67	55.59	59	46.52	75.65	89	-19.81	57.87
30	2.25	56.70	60	4.33	65.60	90	15.95	103.95

Appendix E

Examples (continued)

Example 9

Description:

Shape:	Triangle	**Depot:**	Center
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 5520.98

CPU Time: 507 seconds

Route 1 0 - 47 - 17 - 30 - 77 - 12 - 61 - 60 - 3 - 55 - 44 - 36 - 42 - 0

Route 2 0 - 29 - 73 - 50 - 16 - 2 - 33 - 49 - 74 - 90 - 78 - 27 - 0

Route 3 0 - 1 - 19 - 37 - 51 - 28 - 20 - 39 - 70 - 54 - 0

Route 4 0 - 57 - 66 - 81 - 40 - 53 - 80 - 84 - 38 - 23 - 0

Route 5 0 - 34 - 71 - 82 - 83 - 52 - 89 - 48 - 15 - 88 - 5 - 0

Route 6 0 - 87 - 75 - 21 - 31 - 24 - 35 - 65 - 69 - 4 - 0

Route 7 0 - 68 - 56 - 62 - 10 - 76 - 25 - 41 - 63 - 0

Route 8 0 - 22 - 11 - 59 - 6 - 72 - 85 - 26 - 64 - 9 - 43 - 79 - 0

Route 9 0 - 13 - 58 - 7 - 14 - 67 - 32 - 86 - 45 - 18 - 8 - 46 - 0

Appendix E

Examples (continued)

Example 9 - Customer Locations:

	X	Y		X	Y		X	Y
1	-0.67	64.70	**31**	-15.02	42.97	**61**	44.91	52.07
2	-41.51	19.60	**32**	0.19	-64.06	**62**	-14.82	-33.97
3	12.13	58.31	**33**	-44.81	13.34	**63**	2.37	-3.64
4	-4.14	3.59	**34**	3.88	-45.35	**64**	37.45	10.70
5	9.47	-14.80	**35**	-28.07	19.64	**65**	-17.96	14.83
6	48.78	35.01	**36**	5.87	30.45	**66**	-2.00	-57.95
7	-0.22	-62.83	**37**	35.66	63.16	**67**	-0.03	-64.11
8	2.53	-45.48	**38**	-4.06	-31.29	**68**	-63.50	64.37
9	33.68	-3.49	**39**	13.95	29.75	**69**	-1.29	9.65
10	-13.97	-17.06	**40**	-3.66	-53.23	**70**	14.34	12.59
11	46.19	45.53	**41**	-7.37	-2.52	**71**	10.97	-43.69
12	47.75	51.45	**42**	5.73	15.76	**72**	50.98	35.67
13	0.36	-58.75	**43**	12.80	4.72	**73**	-35.47	48.67
14	-0.20	-65.06	**44**	-4.98	45.68	**74**	-33.43	3.80
15	6.85	-20.78	**45**	0.90	-54.08	**75**	-11.14	50.11
16	-36.04	14.79	**46**	3.44	-26.13	**76**	-8.43	-14.33
17	51.88	60.42	**47**	72.63	64.00	**77**	51.69	49.85
18	2.47	-46.99	**48**	13.19	-28.49	**78**	-11.56	4.52
19	2.89	61.95	**49**	-41.85	11.23	**79**	10.78	-2.03
20	27.68	21.17	**50**	-27.63	33.06	**80**	-8.64	-45.11
21	-6.51	41.63	**51**	33.87	50.58	**81**	-5.12	-55.68
22	64.91	52.73	**52**	3.81	-33.37	**82**	11.26	-39.79
23	-6.03	-25.21	**53**	-7.00	-46.48	**83**	2.98	-34.03
24	-14.85	36.64	**54**	7.74	4.06	**84**	-3.44	-36.55
25	-9.56	-8.41	**55**	-4.05	48.80	**85**	49.77	29.40
26	49.30	25.25	**56**	-48.62	45.87	**86**	2.00	-60.45
27	-11.80	1.15	**57**	6.22	-52.58	**87**	-13.16	49.20
28	32.57	32.95	**58**	-1.71	-62.33	**88**	8.38	-19.24
29	-43.74	51.57	**59**	37.95	34.15	**89**	13.82	-35.83
30	46.00	56.27	**60**	20.37	58.14	**90**	-23.16	4.84

Appendix E

Examples (continued)

Example 10

Description:

Shape:	Triangle	**Depot:**	Corner
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 6709.34

CPU Time: 381 seconds

Route 1	0 - 88 - 33 - 43 - 86 - 3 - 40 - 49 - 59 - 50 - 44 - 38 - 80 - 0
Route 2	0 - 89 - 42 - 60 - 30 - 61 - 32 - 57 - 90 - 0
Route 3	0 - 17 - 58 - 53 - 54 - 66 -87 - 41 - 22 - 64 - 0
Route 4	0 - 18 - 16 - 77 - 69 - 20 - 51 - 82 - 46 - 73 - 74 - 81 - 67 - 0
Route 5	0 - 83 - 9 - 47 - 15 - 39 - 37 - 75 - 79 - 0
Route 6	0 - 10 - 2 - 7 - 36 - 1 - 12 - 14 - 48 - 71 - 13 - 0
Route 7	0 - 29 - 23 - 34 - 63 - 85 - 35 - 8 - 31 - 6 - 56 - 5 - 21 - 0
Route 8	0 - 68 - 70 - 28 - 24 - 84 - 19 - 25 - 11 - 52 - 4 - 0
Route 9	0 - 72 - 26 - 62 - 78 - 27 - 65 - 55 - 76 - 45 - 0

Appendix E

Examples (continued)

Example 10 - Customer Locations:

	X	Y		X	Y		X	Y
1	10.06	63.98	**31**	18.37	97.55	**61**	-24.88	54.61
2	4.74	102.70	**32**	-24.55	43.64	**62**	35.37	85.09
3	37.63	105.04	**33**	39.52	128.49	**63**	17.31	120.50
4	-0.42	2.41	**34**	1.11	122.53	**64**	-2.10	7.70
5	3.60	49.05	**35**	19.36	108.82	**65**	14.53	47.20
6	16.72	90.05	**36**	10.21	67.08	**66**	-25.44	86.22
7	11.52	93.20	**37**	10.42	22.87	**67**	-3.73	8.93
8	18.45	98.30	**38**	11.52	20.30	**68**	-6.35	105.40
9	27.98	51.13	**39**	9.77	24.84	**69**	-8.27	59.93
10	-0.55	111.25	**40**	29.88	77.41	**70**	-15.70	110.33
11	1.30	57.77	**41**	-14.06	44.86	**71**	1.10	18.74
12	13.39	57.68	**42**	-45.16	83.11	**72**	71.69	127.32
13	0.51	6.47	**43**	48.32	117.43	**73**	-5.02	32.35
14	11.04	46.74	**44**	17.04	33.97	**74**	-6.97	15.76
15	12.12	36.48	**45**	0.84	2.54	**75**	1.24	2.83
16	-14.08	89.63	**46**	0.02	42.15	**76**	1.89	7.50
17	-68.32	131.60	**47**	18.86	41.36	**77**	-18.58	84.47
18	-48.83	92.41	**48**	5.74	34.83	**78**	27.82	85.79
19	2.51	72.74	**49**	30.31	75.30	**79**	0.03	0.30
20	-6.31	60.30	**50**	25.27	60.00	**80**	4.89	8.95
21	0.30	28.30	**51**	-4.23	58.90	**81**	-7.37	13.57
22	-14.16	39.47	**52**	-1.05	5.89	**82**	0.35	45.27
23	-10.52	123.96	**53**	-48.49	102.39	**83**	19.98	96.16
24	-14.18	96.84	**54**	-27.57	88.27	**84**	-4.29	93.54
25	-0.66	67.71	**55**	5.35	15.10	**85**	21.79	115.39
26	56.60	100.52	**56**	15.44	88.44	**86**	38.02	108.11
27	26.59	77.59	**57**	-15.08	36.10	**87**	-17.69	75.82
28	-15.10	109.76	**58**	-54.07	117.36	**88**	26.74	126.13
29	-39.21	126.92	**59**	31.30	70.05	**89**	-46.44	81.22
30	-32.65	74.14	**60**	-36.46	80.89	**90**	-12.62	25.71

Appendix E

Examples (continued)

Example 11

Description:

Shape: Sector **Depot:** Center

Customers: 90 **Vehicles:** 9

Solution:

Total Waiting Time: 5545.77

CPU Time: 498 seconds

Route 1 0 - 83 - 33 - 15 - 49 - 18 - 26 - 39 - 30 - 1 - 12 - 27 - 86 - 40 - 0

Route 2 0 - 4 - 65 - 56 - 43 - 63 - 32 - 46 - 89 - 50 - 0

Route 3 0 - 20 - 70 - 38 - 6 - 9 - 51 - 44 - 37 - 82 - 62 - 25 - 53 - 0

Route 4 0 - 87 - 11 - 69 - 5 - 57 - 36 - 19 - 0

Route 5 0 - 71 - 48 - 54 - 75 - 73 - 10 - 28 - 78 - 81 - 0

Route 6 0 - 76 - 67 - 23 - 22 - 29 - 2 - 34 - 16 - 79 - 0

Route 7 0 - 3 - 72 - 7 - 60 - 13 - 77 - 41 - 45 - 66 - 0

Route 8 0 - 35 - 64 - 47 - 21 - 52 - 24 - 90 - 42 - 59 - 0

Route 9 0 - 68 - 8 - 80 - 85 - 14 - 84 - 58 - 61 - 74 - 88 - 31 - 55 - 17 - 0

Appendix E

Examples (continued)

Example 11 - Customer Locations:

	X	Y		X	Y		X	Y
1	-8.55	-47.89	**31**	2.93	-27.19	**61**	-7.28	-43.38
2	-15.12	-20.22	**32**	-2.21	58.05	**62**	-9.76	-22.94
3	1.04	-64.94	**33**	-1.13	-64.59	**63**	-11.61	57.42
4	-56.12	51.29	**34**	-10.93	-23.34	**64**	-44.96	33.70
5	24.94	-20.29	**35**	-43.15	41.57	**65**	-55.97	53.48
6	1.31	-44.41	**36**	29.88	-9.20	**66**	6.18	-4.03
7	4.09	-50.56	**37**	-11.64	-36.14	**67**	26.86	51.81
8	-5.02	-59.09	**38**	-20.56	-23.92	**68**	-3.72	-59.48
9	2.19	-46.62	**39**	0.21	-54.77	**69**	18.70	-31.36
10	34.32	11.97	**40**	1.22	-8.04	**70**	-19.87	-23.05
11	16.82	-37.48	**41**	9.50	-21.85	**71**	59.80	37.89
12	-13.04	-43.55	**42**	-21.81	-10.09	**72**	5.47	-56.55
13	11.58	-45.50	**43**	-19.26	57.78	**73**	35.52	6.32
14	-7.68	-52.79	**44**	-9.88	-41.76	**74**	-4.32	-39.34
15	-0.43	-64.73	**45**	8.71	-16.86	**75**	36.27	2.76
16	-0.38	-27.73	**46**	-7.97	34.03	**76**	44.34	58.66
17	5.93	-19.64	**47**	-54.14	28.26	**77**	11.07	-32.50
18	2.17	-58.58	**48**	27.05	21.26	**78**	33.80	44.74
19	12.70	-12.35	**49**	3.13	-60.66	**79**	1.76	-19.14
20	-20.48	14.73	**50**	-2.86	13.32	**80**	-5.24	-55.66
21	-33.73	10.99	**51**	-3.32	-46.37	**81**	18.07	5.73
22	-15.70	29.84	**52**	-30.51	7.71	**82**	-12.22	-28.42
23	2.93	42.61	**53**	-5.02	-15.64	**83**	-0.26	-68.20
24	-33.31	-5.57	**54**	41.86	6.54	**84**	-6.48	-52.20
25	-5.15	-18.38	**55**	5.06	-20.74	**85**	-6.69	-54.37
26	0.79	-55.77	**56**	-23.66	56.98	**86**	-7.38	-29.33
27	-12.96	-41.15	**57**	30.36	-8.64	**87**	-25.89	34.76
28	41.24	32.02	**58**	-9.17	-43.73	**88**	-4.11	-36.73
29	-13.32	17.15	**59**	1.21	-0.56	**89**	-4.46	25.59
30	-4.46	-50.36	**60**	10.07	-45.65	**90**	-24.15	-5.60

Appendix E

Examples (continued)

Example 12

Description:

Shape:	Sector	**Depot:**	Corner
Customers:	90	**Vehicles:**	9

Solution:

Total Waiting Time: 6912.8

CPU Time: 391 seconds

Route 1 0 - 54 - 38 - 66 - 56 - 33 - 36 - 74 - 16 - 64 - 19 - 0

Route 2 0 - 81 - 29 - 76 - 45 - 3 - 58 - 60 - 27 - 0

Route 3 0 - 77 - 51 - 72 - 8 - 75 - 12 - 85 - 53 - 80 - 9 - 0

Route 4 0 - 17 - 34 - 68 - 15 - 78 - 62 - 28 - 83 - 73 - 88 - 0

Route 5 0 - 41 - 25 - 65 - 23 - 90 - 1 - 82 - 18 - 21 - 49 - 0

Route 6 0 - 31 - 70 - 46 - 69 - 59 - 2 - 30 - 40 - 13 - 42 - 0

Route 7 0 - 71 - 52 - 26 - 86 - 10 - 6 - 24 - 22 - 84 - 5 - 79 - 43 - 0

Route 8 0 - 20 - 61- 7 - 67 - 39 - 50 - 89 - 63 - 87 - 0

Route 9 0 - 4 - 11 - 48 - 14 - 37 - 57 - 47 - 35 - 55 - 44 - 32 - 0

Appendix E

Examples (continued)

Example 12 - Customer Locations:

	X	Y		X	Y		X	Y
1	-14.79	77.93	**31**	59.81	115.48	**61**	25.92	109.25
2	11.21	53.95	**32**	-0.27	11.19	**62**	22.34	67.79
3	22.32	49.46	**33**	-36.29	66.31	**63**	-2.81	48.31
4	-46.90	117.10	**34**	18.97	92.36	**64**	-5.21	10.86
5	9.84	26.95	**35**	8.52	59.04	**65**	-30.98	107.48
6	2.51	81.92	**36**	-21.34	37.13	**66**	-18.18	69.47
7	21.60	108.37	**37**	-11.11	101.51	**67**	18.72	97.51
8	30.68	83.94	**38**	-36.47	104.01	**68**	19.97	92.19
9	-0.06	0.13	**39**	14.61	89.45	**69**	26.21	86.16
10	0.54	94.38	**40**	6.60	32.52	**70**	49.96	107.17
11	-18.70	110.82	**41**	-18.20	127.34	**71**	15.76	127.59
12	27.20	69.32	**42**	0.90	5.04	**72**	27.42	96.04
13	2.93	16.84	**43**	6.88	18.25	**73**	11.33	24.13
14	-14.50	113.28	**44**	3.44	19.95	**74**	-14.10	39.12
15	16.38	77.74	**45**	29.31	61.84	**75**	33.69	80.88
16	-16.31	30.23	**46**	45.16	103.13	**76**	39.14	67.84
17	6.52	130.94	**47**	-0.74	67.03	**77**	32.39	128.75
18	-10.96	41.49	**48**	-14.28	115.79	**78**	20.31	75.27
19	-1.39	3.20	**49**	-1.17	2.58	**79**	7.12	19.41
20	19.13	124.92	**50**	4.74	75.01	**80**	2.52	9.35
21	-10.28	33.42	**51**	30.04	101.87	**81**	43.47	116.74
22	16.10	62.16	**52**	15.59	118.08	**82**	-10.41	65.18
23	-32.97	93.97	**53**	13.07	39.36	**83**	17.12	34.79
24	10.90	73.33	**54**	-54.56	103.71	**84**	13.31	55.69
25	-22.96	116.94	**55**	4.66	41.45	**85**	19.24	53.26
26	12.92	118.56	**56**	-25.78	65.71	**86**	6.12	104.19
27	2.73	6.13	**57**	-7.28	75.40	**87**	-5.34	32.82
28	21.82	48.13	**58**	23.40	40.99	**88**	6.58	14.40
29	42.93	88.48	**59**	24.70	72.24	**89**	-6.19	49.94
30	9.19	47.65	**60**	17.20	31.80	**90**	-20.70	80.56

REFERENCES

Achuthan, N. R., L. Caccetta, and S. Hill (1996). A new subtour elimination constraint for the vehicle routing problem. *European Journal of Operational Research*, 91(3), 573-586.

Achuthan, N. R., L. Caccetta, and S. Hill (1998). Capacitated vehicle routing problem: Some new cutting planes. *Asia - Pacific Journal of Operational Research*, 15(1), 109-123.

Agarwal, Y., K. Mathur, and H. M. Salkin (1989). A set-partitioning-based algorithm for the vehicle routing problem. *Networks*, 19, 731-750.

Alfa, A. S., S. S. Heragu, and M. Chen (1991). A 3-opt based simulated annealing algorithm for vehicle routing problems. *Computers and Industrial Engineering*, 21, 635-639.

Anily, S., and A. Federgrun (1990a). A class of Euclidean routing problems with general route cost functions. *Mathematics of Operations Research*, 15(2), 268-285.

Anily, S., and A. Federgrun (1990b). One warehouse multiple retailer systems with vehicle routing. *Management Science*, 36(1), 92-114.

Assad, A. A., and B. L. Golden (1995). Arc routing methods and applications. In: M. O. Ball, T. L. Magnanti, C. L. Monma, G. L. Nemhauser (Eds.), *Handbooks in Operations Research and Management Science*, Vol. 8, Network Routing, Elsevier, Amsterdam, 375-483.

Augerat, P., J. M. Belenguer, E. Benavent, A. Corberan, and D. Naddef (1998). Separating capacity constraints in the CVRP using tabu search. *European Journal of Operational Research*, 106, 546-557.

Baker, B. M., and J. Sheasby (1999). Extensions to the generalised assignment heuristic for vehicle routing. *European Journal of Operational Research*, 119, 147-157.

Balakrishnan, N. (1993). Simple heuristics for the vehicle routing problem with soft time windows. *The Journal of the Operational Research Society*, 44(3), 279-287.

Balinski, M., and R. Quandt (1964). On an integer program for a delivery problem. *Operations Research*, 12, 300-304.

Ballou, R. H. (1990). A continued comparison of several popular algorithms for vehicle routing and scheduling. *Journal of Business Logistics*, 11(1), 111-126.

Ballou, R. H., and Y. K. Agarwal (1988). A performance comparison of several popular algorithms for vehicle routing and scheduling. *Journal of Business Logistics*, 9(1), 51-65.

Barbarosoglu, G., and D. Ozgur (1999). A tabu search algorithm for the vehicle routing problem. *Computers and Operations Research*, 26, 255-270.

Bard, J. F., L. Huang, M. Dror, and P. Jaillet (1998). A branch and bound algorithm for the VRP with satellite facilities. *IIE Transactions*, 30, 821-834.

Basnet, C., L. R. Foulds, and J. M. Wilson (1999). Heuristics for vehicle routing on tree-like networks. *The Journal of the Operational Research*, 50, 627-635.

Belenguer, J. M., M. C. Martinez, and E. Mota (2000). A lower bound for the split delivery vehicle routing problem. *Operations Research*, 48(5), 801-810.

Benton, W. C., and M. D. Rossetti (1992). The vehicle scheduling problem with intermittent customer demands. *Computers and Operations Research*, 19(6), 521-531.

Bertsimas, D. J. (1992). A vehicle routing problem with stochastic demand. *Operations Research*, 40(3), 574-585.

Bodin, L. D., and B. L. Golden (1981). Classification in vehicle routing and scheduling. *Networks*, 11, 97-108.

Bodin, L. D., B. L. Golden, A. A. Assad, and M. O. Ball (1983). Routing and scheduling of vehicles and crews: The state of the art. *Computers and Operations Research*, 10, 69-211.

Brandao, J., and A. Mercer (1997). A tabu search algorithm for the multi-trip vehicle routing and scheduling problem. *European Journal of Operations Research*, 100(1), 180-191.

Brandao, J., and A. Mercer (1998), The multi-trip vehicle routing problem, *The Journal of the Operational Research Society*, 1998, 49(8), 799-805.

Breedam, A. V. (1995). Improvement heuristics for the vehicle routing problem based on simulated annealing. *European Journal of Operational Research*, 86(3), 480-490.

Butt, S. E., and D. M. Ryan (1999). An optimal solution procedure for the multiple tour maximum collection problem using column generation. *Computers and Operations Research*, 26, 427-441.

Chan, Y. (2001). A multiple-depot, multiple-vehicle, location-routing problem with stochastically processed demands. *Computers and Operations Research*, 28(8), 803-826.

Chao, I. M., B. Golden, and E. Wasil (1999). A computational study of a new heuristic for the site-dependent vehicle routing problem. *INFOR*, 37(3), 319-336.

Chen, T. K., L. L. Hay, and O. Ke (2001). Hybrid genetic algorithms in solving vehicle routing problems with time window constraints. *Asia - Pacific Journal of Operational Research*, 18, 121-130.

Chen, X., W. Wan, and X. Xu (1998). Modeling rolling batch planning as vehicle routing problem with time windows. *Computers and Operations Research*, 25(12), 1127-1136.

Chien, T. W. (1992). Operational estimator for the length of a traveling salesman tour. *Computers and Operations Research*, 19(6), 469-478.

Christofides, N., A. Mingozzi, and P. Toth (1979). The vehicle routing problem. In: N. Christofides, A. Mingozzi, P. Toth and C. Sandi (Eds.), *Combinatorial Optimization*, Wiley, Chichester, 315-338.

Christofides, N., A. Mingozzi, and P. Toth (1981a). Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxation. *Mathematical Programming*, 20, 255-282.

Christofides, N., A. Mingozzi, and P. Toth (1981b). State space relaxation procedures for the computation of bounds to routing problems. *Networks*, 11, 145-164.

Chung, H. K., and J. P. Norback (1991). A clustering and insertion heuristic applied to a large routing problem in food distribution. *The Journal of the Operational Research*, 42(7), 555-564.

Clarke, G., and J. W. Wright (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12, 568-581.

Cordeau, J. F., and G. Laporte (2001). A tabu search algorithm for the site dependent vehicle routing problem with time windows. *INFOR*, 39(3), 292-298.

Cordeau, J. F., G. Laporte, and A. Mercier (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *The Journal of the Operational Research Society*, 52(8), 928-936.

Dannenbring, D. G. (1977). Procedures for estimating optimal solution values for large combinatorial problems. *Management Science*, 23, 1273-1283.

Dantzig, G., and J. Ramser (1959). The truck dispatching problem. *Management Science*, October 1959, 81-91.

Desrochers, M., J. Desrosiers, and M. Solomon (1992). A new optimization algorithm for the vehicle routing problem. *Operations Research*, 40(2), 342-354.

Desrochers, M., J. K. Lenstra, and M. W. P. Savelsbergh (1990). A classification scheme for vehicle routing and scheduling problems. *European Journal of Operations Research*, 46, 322-332.

Desrochers, M., and T. W. Verhoog (1991). A new heuristic for the fleet size and mix vehicle routing problem. *Computers and Operations Research*, 18(3), 263-274.

Desrosiers, J., F. Soumis, and M. Desrochers (1984). Routing with time windows by column generation. *Networks*, 14, 545-565.

Dumas, Y., J. Desrosiers, and F. Soumis (1991). The pickup and delivery problem with time windows. *European Journal of Operations Research*, 54(1), 7-22.

Eilon, S., C. D. T. Watsin-Gandy, and N. Christofides (1971). *Distribution Management: Mathematical Modeling and Practical Analysis*, Griffin, London.

Eiselt, H. A., M. Gendreau, and G. Laport (1995a). Arc routing problems. Part I: The Chinese postman problem. *Operations Research*, 43, 231-242.

Eiselt, H. A., M. Gendreau, and G. Laport (1995a). Arc routing problems. Part II: The rural postman problem. *Operations Research*, 43, 399-414.

Fisher, M. L. (1994). Optimal solution of vehicle routing problems using minimum K-tree. *Operations Research*, 42(4), 626-642.

Fisher, M. L., and R. Jaikumar (1981). A generalized assignment heuristic for vehicle routing. *Networks*, 11, 109-124.

Fisher, M. L., K. O. Jornstem, and O. B. G. Madsen (1997). Vehicle routing with time windows: two optimization algorithms. *Operations Research*, 45(3), 488-492.

Fishetti, M., P. Toth, and D. Vigo (1994). A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs. *Operations Research*, 42(5), 846-859.

Foster, B., and D. Ryan (1976). An integer programming approach to the vehicle scheduling problem. *Operations Research Quarterly*, 27, 367-384.

Frizzell, P. W., and J. W. Giffin (1995). The split delivery vehicle scheduling problem with time windows and grid network distance. *Computers and Operations Research*, 22(6), 655-667.

Garcia, B. L., J. Y. Potvin, and J. M. Rousseau (1994). A parallel implementation of the Tabu search heuristic for vehicle routing problems with time window constraints. *Computers and Operations Research*, 21(9), 1025-1033.

Garey, M. R., and D. S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-complete*, Freeman, San Francisco.

Gehring, H., and J. Homberger (2001). A parallel two-phase metaheuristic for routing problems with time windows. *Asia - Pacific Journal of Operational Research*, 18, 35-47.

Gendreau, M., A. Hertz, and G. Laporte (1994). A tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10), 1276-1290.

Gendreau, M., G. Laporte, C. Musaraganyi, and E. D. Taillard (1999). A tabu search heuristic for the heterogeneous fleet vehicle routing problem. *Computers and Operations Research*, 26, 1153-1173.

Gendreau, M., G. Laporte, and R. Seguin (1996). A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3), 501-509.

Ghiani, G., and G. Improta (2000). An efficient transformation of the generalized vehicle routing problem. *European Journal of Operational Research*, 122, 11-17.

Gillett, B., and L. Miller (1974). A heuristic algorithm for the vehicle dispatch problem. *Operations Research*, 22, 340-349.

Golden, B. L., G. Laporte, and E. Taillard (1997). An adaptive memory heuristic for a class of vehicle routing problems with minmax objective. *Computers and Operations Research*, 24(5), 445-452.

Golden, B. L., T. L. Magnanti, and H. Q. Nguyen (1977). Implementing vehicle routing algorithms. *Networks*, 7, 113-148.

Hadjiconstantinou, E. (1998). A multi-depot period vehicle routing problem arising in the utility sector. *The Journal of the Operational Research Society*, 49(12), 1239-1248.

Herer, Y. T., and R. Levy (1997). The metered inventory routing problem, an integrative heuristic algorithm. *International Journal of Production Economics*, 51, 69-81.

Homberger, J., and H. Gehring (1999). Two evolutionary metaheuristics for the vehicle routing problem with time windows. *INFOR*, 37(3), 297-318.

Hong, S. C., and Y. B. Park (1999). A heuristic for bi-objective vehicle routing with time window constraints. *International Journal of Production Economics*, 62(3), 249-259.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of National Academy of Science*, 79, 2554-2558.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of National Academy of Science*, 81, 3088-3092.

Hopfield, J. J., and D. Tank (1985). Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52, 141-152.

Jansen, K. (1992). An approximation algorithm for the general routing problem. *Information Processing Letters*, 41(6), 333-339.

Karkazis, J., and T. B. Boffey (1995). Optimal location of routes for vehicles transporting hazardous materials. *European Journal of Operations Research*, 86(2), 201-215.

Kohl, N., and O. B. G. Madsen (1997). An optimization algorithm for the vehicle routing problem with time windows based on Lagrangean relaxation. *Operations Research*, 45(3), 395-406.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43, 59-69.

Laport, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operations Research*, 59, 345-358.

Laporte, G., H. Mercure, and Y. Nobert (1986). An exact algorithm for the asymmetrical capacitated vehicle routing problem. *Networks*, 16, 33-46.

Laporte, G., and Y. Nobert (1987). Exact algorithms for the vehicle routing problem. In: S. Martello, G. Laporte, and C. Ribeiro (Eds.), *Surveys in Combinatorial Optimization*, North-Holland, Amsterdam, 147-184.

Laporte, G., Y. Nobert, and M. Desrochers (1985). Optimal routing under capacity and distance restrictions. *Operations Research*, 33, 1050-1073.

Laport, G., and I. H. Osman (1995). Routing problems: A bibliography. *Annals of Operations Research*, 61, 227-262.

Lenstra, J. K., and A. H. G. Rinnooy Kan (1975). Some simple applications of the traveling salesman problem. *Operations Research Quarterly*, 26, 1050-1073.

Liu, F. H., and S. Y. Shen (1999). The fleet size and mix vehicle routing problem with time windows. *The Journal of the Operational Research Society*, 50, 721-732.

Malmborg, C. J. (1996). A genetic algorithm for service level based vehicle scheduling. *European Journal of Operations Research*, 93(1), 121-134.

Min, H., J. Current, and D. Schilling (1992). The multiple depot vehicle routing problem with backhauling. *Journal of Business Logistics*, 13, 259-288.

Mosheiov, G. (1998). Vehicle routing with pickup and delivery: tour-partitioning heuristics. *Computers and Industrial Engineering*, 34(3), 669-684.

Ong, H. L., B. W. Ang, T. N. Goh, and C. C. Deng (1997). A vehicle routing and scheduling problem with time windows and stochastic demand constraints. *Asia - Pacific Journal of Operational Research*, 14(1), 1-17.

Or, I. (1976). Traveling salesman type combinatorial problems and their relation to the logistics of blood banking, Ph. D. dissertation, Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL.

Orloff, C. (1976). Route-constraint fleet scheduling. *Transportation Science*, 10, 149-168.

Paessens, H. (1988). The savings algorithm for the vehicle routing problem. *European Journal of Operational Research*, 34(3), 336-344.

Park, Y. B., and C. P. Koelling (1989). An interactive computerized algorithm for multicriteria vehicle routing problems. *Computers and Industrial Engineering*, 16(4), 477-490.

Potvin, J. Y., and J. M. Rousseau (1993). a parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3), 331-340.

Potvin, J. Y., and J. M. Rousseau (1995). An exchange heuristic for routing problems with time windows. *The Journal of the Operational Research Society*, 46(12), 1433-1446.

Psaraftis, H. N. (1995). Dynamic vehicle routing. *Annals of Operations Research*, 61, 143-164.

Rao, M. R., and S. Zionts (1968). Allocation of transportation units to alternative trips - A column generation scheme with out-of-kilter subproblems. *Operations Research*, 16, 52-63.

Ree, S., and B. Yoon (1996). A two-stage heuristic approach for the newspaper delivery problem. *Computers and Industrial Engineering*, 30(3), 501-509.

Rego, C., and C. Roucairol (1995). Using Tabu search for solving a dynamic multi-terminal truck dispatching problem. *European Journal of Operational Research*, 83(2), 411-429.

Rego, C. (1998). A subpath ejection method for the vehicle routing problem. *Management Science*, 44(10), 1447-1459.

Renaud, J., G. Laporte, and F. F. Boctor (1996). A tabu search heuristic for the multi-depot vehicle routing problem. *Computers and Operations Research*, 23(3), 229-235.

Rojas, R. (1996). *Neural networks: a systematic introduction*. Springer-Verlag, Berlin.

Sahni, S., and T. Gonzalez (1976). P-complete approximation problems. *Journal of the Association for Computing Machinery*, 23, 555-565.

Salhi, S., and G. Nagy (1999). A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *The Journal of the Operational Research Society*, 50(10), 1034-1042.

Salhi, S., and M. Sari (1997). A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operations Research*, 103(1), 95-112.

Sariklis, D., and S. Powell (2000). A heuristic method for the open vehicle routing problem. *The Journal of the Operational Research Society*, 51(5), 564-573.

Secomandi, N. (2000). Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers and Operations Research*, 27(11), 1201-1225.

Secomandi, N. (2001). A rollout policy for the vehicle routing problem with stochastic demands. *Operations Research*, 49(5), 796-802.

Shang, J. S., and C. K. Cuff (1996). Multicriteria pickup and delivery problem with transfer opportunity. *Computers and Industrial Engineering*, 30(4), 631-645.

Somhom, S., A. Modares, and T. Enkawa (1999). Competition-based neural network for the multiple traveling salesman problem with minmax objective. *Computers and Operations Research*, 395-407.

Soylu, M., N. E. Ozdemirel, and S. Kayaligil (2000). A self-organizing neural network approach for the single AGV routing problem. *European Journal of Operational Research*, 121, 124-137.

Srivastava, R., and W. C. Benton (1990). The location-routing problem: Considerations in physical distribution system design. *Computers and Operations Research*, 17, 427-435.

Sumichrast, R. T., and I. S. Markham (1995). A heuristic and lower bound for a multi-depot routing problem. *Computers and Operations Research*, 22(10), 1047-1056.

Swihart, M. R., and J. D. Papastavrou (1999). A stochastic and dynamic model for the single-vehicle pick-up and delivery problem. *European Journal of Operations Research*, 114(3), 447-464.

Taillard, E. D., G. Laporte, and M. Gendreau (1996). Vehicle routing with multiple use of vehicles. *The Journal of the Operational Research Society*, 47(8), 1065-1070.

Thangian, S. R., J. Y. Potvin, and T. Sun (1996). Heuristic approaches to the vehicle routing with backhauls and time windows. *Computers and Operations Research*, 25(11), 1043-1057.

Torki, A., S. Somhon, and T. Enkawa (1997). A competitive neural network algorithm for solving vehicle routing problem. *Computers and Industrial Engineering*, 33, 473-476.

Toth, P., and D. Vigo (1999). A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls. *European Journal of Operations Research*, 113(3), 528-543.

Toure, S., L. Rabelo, and T. Velasco (1993). Artificial neural networks for flexible manufacturing systems scheduling. *Computers and Industrial Engineering*, 25, 385-388.

Vaidyanathan, B. S., J. O. Matson, D. M. Miller, and J. E. Matson (1999). A capacitated vehicle routing problem for just-in-time delivery. *IIE Transactions*, 31, 1083-1092.

Van Landeghem, H. R. G. (1988). A bi-criteria heuristic for the vehicle routing problem with time windows. *European Journal of Operational Research*, 36(2), 217-226.

Vigo, D. A (1996). Heuristic algorithm for the asymmetric capacitated vehicle routing problem. *European Journal of Operational Research*, 89(1), 108-126.

Weintraub, A., J. Aboud, C. Fernandez, G. Laporte, and E. Ramirez (1999). An emergency vehicle dispatching system for an electric utility in Chile. *Journal of the Operational Research Society*, 50, 690-696.

Wijeratne, A. B., M. A. Turnquist, and P. B. Mirchandani (1993). Multiobjective routing of hazardous materials in stochastic networks. *European Journal of Operational Research*, 65, 33-42.