# ABSTRACT


| | |
|---|---|
| Title of Document: | THE VEHICLE ROUTING PROBLEM WITH DEMAND RANGES |
| | Namrata Uppal Cornick, Master of Science, 2009 |
| Directed By: | Professor Bruce Golden,<br>Applied Mathematics and Scientific Computation<br>and the R.H. Smith School of Business |



The classic Capacitated Vehicle Routing Problem (CVRP) has been studied in the Operations Research field for over 5 decades. This thesis formulates the vehicle routing problem with a variation that has not been studied in detail. It is called the Vehicle Routing Problem with Demand Ranges (VRPDR). With increasing competition, corporations are looking to minimize costs. This problem aims to reduce the cost of distributing goods by allowing flexibility in the delivered or dropped off quantity. This benefits the customer as well, by reducing storage and other inventory costs. We solve the VRPDR problem where the customer gives the distributor a demand range. The distributor is rewarded for delivering more. A metaheuristic, record-to-record travel with demand range (RTRDR), is developed which is capable of solving large problem instances. The metaheuristic is a modification of a successful CVRP metaheuristic used in the past. In this thesis, we report results on problems ranging in size from 560 to 1200 customers. The developed metaheuristic uses the Clarke-Wright procedure to get initial solutions and then applies record-to-record travel in conjunction with two-opt

moves, one point moves, and two point moves. Since the problem has not been studied yet from a computational point of view, we have developed a comparison algorithm, which takes advantage of the demand range flexibility of this problem only after the algorithm has optimized for distance alone. We use the results from this algorithm as a benchmark to compare with our proposed metaheuristic RTRDR.

# VEHICLE ROUTING PROBLEM WITH DEMAND RANGES

By

Namrata Uppal Cornick

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2009

Advisory Committee:
Professor Bruce Golden, Chair
Professor Konstantina Trivisa
Associate Professor Zhi-Long Chen
Professor Edward Wasil

# Preface

This master's thesis is my final work in the Applied Mathematics and Scientific Computation program of the Department of Mathematics at University of Maryland, College Park. The thesis involves a variation of the Capacitated Vehicle Routing Problem (CVRP) commonly investigated in the Operations Research field. The variation, Vehicle Routing Problem with Demand Ranges (VRPDR) is like the CVRP, but for clients that have a demand window rather than a single demand.

This thesis describes and investigates the problem in seven chapters. The introduction discusses the intended practical use of the problem and the clientele base. One of the chapters solves the problem optimally for small problem sets. In a later chapter we develop a heuristic to solve the VRPDR. Since this problem is fairly new, we compared results of our algorithm with a slight variation on a common CVRP metaheuristic. We show that our algorithm outperforms the comparison algorithm.

I started research into this topic in June 2007 and wrote the algorithm at the end of August 2007 and was done by the end of November 2007. Most of December was spent running the results and assembling and tabulating the data.

# Dedication

I dedicate this thesis to the love of my life, my husband. My husband, Matthew Cornick, inspired me to believe that we never stop learning in life and the day we do we have ceased to live.

# Acknowledgements

Dr. Bruce Golden, my advisor, introduced me to the optimal world of Operations Research and I am grateful for that. Furthermore, he and Dr. Edward Wasil guided and advised me with good ideas and provided constructive criticism.

I would like to thank my friend Bijan Afsari, who helped in my earlier days when I was getting acquainted into the fun world of Mathematics.

I would like to acknowledge some other friends who stood by me - Sangeeta Parashar, Giridhar Nandikotur, Senni and Samata Pradhan.

Lastly, I would like to thank my family, my husband, Matthew Cornick, my mother-in-law, JoAnne Cornick, my father-in-law, Jon Cornick, and my parents, Ashok and Neelam Uppal for all their support and guidance.

# Table of Contents

# List of Tables

# List of Figures

# List of Graphs

# Chapter 1: Introduction

## ***Section 1.1:Introduction to the Vehicle Routing Problem with Demand Ranges***

The classic Capacitated Vehicle Routing Problem (CVRP) requires a fleet of vehicles to supply goods to customers [1,7]. Each vehicle has a limited capacity and each of the n customers has a fixed demand. All vehicles start at a given point (the depot) and the distance between customers is given. We seek to optimize the distance traveled by the vehicles with the constraint that all the customer demands must be met and vehicle capacity must not be exceeded. In the past, researchers have developed various algorithms to solve this problem. A summary is presented by Cordeau et al. (2005) [5]. We note that throughout this paper we use interchangeably the terms vehicle and truck.

VRP refers to the large class of problems that includes variations of the CVRP. Some of these variations of the CVRP are: the Split Delivery Vehicle Routing Problem (SDVRP)[2], the Vehicle Routing Problem with Time Windows (VRPTW), the VRP with Backhauls (VRPB)[12] and the Open Vehicle Routing Problem (OVRP)[11]. The SDVRP serves the demand of a set of customers with a fleet of capacitated vehicles at minimum cost, where a single customer can be served by more than one vehicle. The VRPTW seeks a set of least-cost routes such that each customer is visited within a pre-specified time interval [8]. The VRPB requires trucks to both deliver and pick up material from customers. The pick-up needs to be transported back to the depot. Moreover, all deliveries have to be made before any goods are picked up. In OVRP vehicles do not have to return to the depot after their final delivery.

The Vehicle Routing Problem with Demand Ranges (VRPDR) is a variation of the VRP, which has not yet received much attention, but has great practical appeal. This problem gives flexibility to the distributor to deliver goods within a demand range while optimizing the distance traveled and the amount dropped off. The demand range for each customer has a lower and upper limit, where the lower limit has to be met, and the upper limit cannot be exceeded. In order to discourage the distributor from only delivering the lower limit, for every unit dropped off the distributor is rewarded.

In this thesis, we first define the VRP and the VRPDR formally in Chapter 1. In Chapter 2, we formulate the VRPDR as an integer programming problem including the objective function to be minimized, and the constraints. In Chapter 3, we modify pre-existing algorithms such as Clarke-Wright (CW), as well as individual operations such as two-opt (TO) and two point move (TPM), to solve the VRPDR. In Chapter 4, we propose a metaheuristic that solves the VRPDR and we test it on problem instances from 560 to 1200 customers. Finally, in Chapter 6, we conclude with a discussion of future work and observations.

## ***Section 1.2: History of the VRP***

Among combinatorial optimization problems, one of the most popular problems is the Traveling Salesman Problem (TSP). The problem involves solving for the shortest closed route that connects a set of nodes. TSP-related problems were first looked at in the 1800s by the Irish Mathematician Sir William Rowan Hamilton, and by a British Mathematician, Thomas Kirkman [9]. In the 1930s, the TSP was studied by Karl Menger of Harvard.

In the 1950s, tools such as linear and integer programming were introduced that gave Operations Research much visibility. In 1959, Dantzig and Ramser introduced the Capacitated Vehicle Routing Problem (CVRP) [6]. The CVRP is an extension of the TSP and another optimization problem called the bin-packing problem (BPP) [4]. The objective of the BPP is to find the fewest number of fixed-size bins needed to hold a set of objects of various weights. Like the TSP and the BPP, the CVRP is a NP-hard problem.

### ***Section 1.3: Problem Motivation***

In real life, a customer's demand is not fixed. Demand varies in time according to climate, seasons, technological advancements, competition, sales, the economy, and several other complex factors. In general, a customer, retailer, or wholesaler would forecast the demand at the time of ordering. A fixed demand could lead to inefficiency, and reduce the level of collaboration between the distributor and the retailer.

In addition, retailers don't have to specify a fixed demand, as long as the average dropped off amount meets their average demand. The retailer is looking for a balance in the inventory flow such that they do not run out of goods, and at the same time, they do not have to store large amounts of inventory. The distributor can typically take advantage of the added flexibility of a demand range. Therefore, to serve the end users better and reduce warehouse storage costs, while keeping demand uncertainties low, the CVRP could be modified to include a demand range for each customer.

With increased competition, businesses look for ways to reduce cost; one of the biggest of these being inventory management. Large corporations with high sales turnover such as Walmart and Home Depot manage to keep cost down by controlling the inventory and reducing storage costs. Vendor Managed Inventory (VMI) is an approach that helps control, monitor, and forecast constant inventory flow. In VMI the vendor or the distributor monitors the retailer's demands and takes responsibility for maintaining a constant flow of stock such that there is no shortage or overstocking of any item. The information is delivered via Electronic Data Interchange (EDI). EDI helps the distributor forecast future demands. This system has dual benefits as both parties have an incentive to give better service to the end customers. VMI is closely related to the VRPDR since both apply to constant inventory flow of high sales turnover products. The goal of the VRPDR is to give control to the distributor, where the distributor has the freedom to deliver within a specified demand range. Therefore, the VRPDR is a form of VMI.

The VMI model is useful if the following elements are present:

- High sales turnover products: Goods such as clothes and perishables have high sales. However, sales of products with low turnover but high profit margins would not be appropriate. VMI works well if the retailers have multiple outlets with high sales turnover.
- Stable demand: Some businesses have drastic demand fluctuations based on competition or technological advancements and so the retailer might not be willing to forgo the control on the in-flow of inventory. An

example where VMI would not be appropriate is the computer industry where technology is changing rapidly and competition or innovation can drop or sky rocket demand.

- Economy products: If the product has low profit margins, such as grocery store items, losing sales due to inventory shortage will not affect the business drastically. However, if the product is a heavy-duty home appliance, usually a one-time buy, losing sales has a bigger impact than on economy products.
- Trust between distributor and retailers: It is imperative for both parties to develop a level of understanding of each other's demands and cost structure. Trust between the two parties will ensure a regular delivery flow for the retailers.

VMI has the following advantages:

- Better serve the end user: Receiving timely and consistent information directly from cash registers (EDI) can help the supplier better understand and predict the future needs of the end users.
- Reduce uncertainties: The distributor indirectly benefits from understanding the end users, and so a constant monitoring of the sales and buying patterns of the end users would reduce demand uncertainties.
- Reduce Warehouse or storage costs: Delivering goods when needed reduces the retailer's need to stock up. Furthermore, it reduces other uncertainties and costs associated with storage such as fire, theft, insurance, obsolesce of the product, and many more.

There are various business models similar to VMI that help reduce costs such as just-in- time (JIT) inventory, consignment stocks, scale-based management, and many more. The VMI model is composed of a demand forecasting component in conjunction with a route calculating component. The VRPDR addresses the route calculating component. From here onwards we will refer to the two parties as the customers (the end user or retailer) and the distributor (the delivering party).

### ***Section 1.4: Statement of the VRPDR Problem***

With the above incentives in mind, let us define the problem formally. The VRPDR problem may be defined on a graph G = {N, E} where N={0,1,2, …, n} is set of nodes and E is the set of edges. Nodes 1,2 …, n are customers and node 0 corresponds to the depot. The set $\{e_{ijv}\}$ is a set of decision variables where $e_{ijv}$ =1 if vehicle v travels from node i to node j and 0 otherwise. Note that $e_{ijv}$ is not symmetric in i and j, we keep track of the direction of travel (although the direction of the travel does not affect the objective function, we will use it in the post-processing stage). Each node i requests an item amount $d_i$ but allows the distributor to under or over deliver by a fraction α. Thus, the actual amount dropped off by vehicle v is $z_{iv}$ which must satisfy $l_i \leq z_{iv} \leq u_i$ for all v. Here $l_i = (1- \alpha)\, d_i$ is referred to as the lower demand and $u_i = (1+ \alpha)\, d_i$ is referred to as the upper demand. After the distributor delivers an amount $z_{iv}$ (for one vehicle v) to customer i, it receives an incentive reward $r_i$, where $r_i = b\, z_{iv}$ and b is the reward per dropped off unit.

Each edge (i,j) has a travel cost $c_{ij}$ associated with travel between node i and node j. The cost is $c_{ij} = a\, D_{ij}$ where a is the dollar cost per mile, and $D_{ij}$ is the

distance between node i to node j in miles. The cost matrix $c_{ij}$ is symmetric ($c_{ij} = c_{ji}$ for all i,j$\in$ N). Since we use Euclidean distances, the *triangle inequality* which states that $c_{ik} + c_{kj} \geq c_{ij}$ for all i, j, k $\in$ {0, 1, 2, …} is satisfied.

A fixed number n of identical vehicles starts from the depot with an identical known vehicle capacity, CAP. Each vehicle can traverse at most one route. For this reason, the total number of routes cannot exceed n. To ensure feasibility, we need the constraints $\sum_{i \in \text{Route v}}^{n} z_{iv} \leq \text{CAP}$ for all v = 1, 2, … , M.

Here Route v is the set of nodes in the route of vehicle v. The number of vehicles used (also the number of routes) is denoted M. In addition, there is a further constraint that the total distance traveled by a vehicle cannot exceed T, the maximum tour length.

The objective of this problem is to minimize cost while maximizing dropped off amount. The objective function to be minimized is cost minus revenue or (c - r) = $a\sum_{ijv} e_{ijv} D_{ij} - b \sum_{iv} z_{iv}$ , where c = $a\sum_{ijv} e_{ijv} D_{ij}$ and r = - b $\sum_{iv} z_{iv}$.

A classic CVRP problem has the same objective function as above with a = 1 and b = 0. The VRPDR can be made identical to the CVRP by setting $\alpha = 0$. The metaheuristic we develop in Chapter 4 solves the VRPDR problem by trying to minimize the above-mentioned objective function.

# Chapter 2: Formulating the VRPDR as an Integer Programming Problem.

Here we formulate the VRPDR problem as an integer linear programming problem. We will minimize the objective function (1) with a set of constraints (2) - (9).

## ***Section 2.1: Formulation, Objective Function, and Constraints***

Objective: minimize $(c - r) = (a \sum_{ijv} e_{ijv} D_{ij} - b \sum_{iv} z_{iv})$.

(1)

The choice variables ($e_{ijv}$ and $z_{iv}$) are subject to the following constraints:

$$\sum_{k=0,k\neq i} e_{kiv} - \sum_{j=0,j\neq i} e_{ijv} = 0 \qquad i = 0, \ldots, n \qquad v = 1, \ldots, M$$

(2)

$$\sum_{v=1}^{M} z_{iv} \geq (1-\alpha) d_i \qquad i = 1, \ldots, n$$

(3)

$$\sum_{j=0,j\neq i}^{n} (1+\alpha) d_i e_{ijv} \geq z_{iv} \qquad i = 1, \ldots, n \qquad v = 1, \ldots, M$$

(4)

$$\sum_{i=1}^{n} z_{iv} \leq CAP \qquad v = 1, \ldots, M$$

(5)

$$\sum_{j=0}^{n}\sum_{j\neq i}^{M}\sum_{v=1} e_{ijv} \leq 1 \qquad i = 1, \ldots, n$$

(6)

$$\sum_{v=1}^{M}\sum_{i\in S, j\in \bar{S}} e_{ijv} \geq \sum_{i\in S}\sum_{v=1}^{M} \frac{z_{iv}}{CAP} \qquad S \subseteq N \setminus \{0\},\ |S| \geq 2$$

(7)

$$e_{ijv} \in \{0,1\} \qquad i, j = 0, \ldots, n \ \ j \neq i \qquad v = 1, \ldots, M$$

(8)

$$Z_{iv} \in \{0, 1, 2, \ldots\} \qquad i = 1, \ldots, n \qquad v = 1, \ldots, M$$

(9)

$$0 \leq z_{iv} \leq (1+\alpha) d_i \qquad i = 1, \ldots, n \qquad v = 1, \ldots, M$$

(10)

This formulation is an extension of the formulation found in Campbell [1]. In this formulation, equation (2) is the flow balance constraint, meaning the number of times vehicle v arrives at node i is equal to the number times it leaves. This constraint is found in a typical VRP formulation. Equation (3) states that the total dropped off amount must be at least the lower demand and equation (4), although in a slightly different form, represents the constraint for the upper demand. Equation (5) is a feasibility constraint stating that for each vehicle, the sum of the dropped off amount should not exceed the vehicle capacity. Equation (6) states that only one vehicle can visit each node. Equation (7) is a sub-tour elimination constraint where S is a subset of the nodes excluding the depot where S has two or

more elements. Any set with Q elements has $2^Q - Q - 1$ subsets of cardinality 2 or more, where Q is the number of elements in the set. Thus, equation (7) represents $2^n - n - 1$ constraints. Note that $\bar{S}$ denotes the complement of S. Equation (8) says that the the variable $e_{ik}$ can be either 0 or 1, where 0 means that vehicle v did not traverse edge connecting node i and node j and 1 means that vehicle v traversed that edge. Equation (9) states that $Z_{iv}$ is an integer. Finally equation (10) defines the decision variable $Z_{iv}$ which cannot be less the lower demand, $(1- \alpha)\, d_i$ and cannot exceed the upper demand, $(1+ \alpha)\, d_i$.

The VRPDR problem has two types of decision variables, $z_{iv}$ and $e_{ijv}$, which represent Mn and $Mn^2$ decision variables, respectively. The constraint (7) makes the number of constraints exponential in problem size. Given that integer programs are hard to solve (NP-hard), there is a need to develop smart heuristics.

# Chapter 3: Constructive Methods

## ***Section 3.1: Clarke-Wright (CW) for Solving the CVRP***

One of the most widely used heuristics to get an initial solution to the CVRP is the Clarke-Wright (CW) algorithm (introduced in 1964 [3]). The algorithm is simple, greedy, and usually produces good results. We consider a case where there is one depot, which is the starting point for all vehicles. We begin the procedure by sending one vehicle to every customer, which travels back to the depot, traveling a total distance $2D_{0i}$, (the distance from the depot to customer i and back). Hence for n customers there should be n routes and n vehicles. The CW algorithm then merges routes to decrease distance as much as possible. It does this by finding the largest element of the *savings matrix* $s_{ij} = D_{0i} + D_{0j} - D_{ij}$ and merging the two routes associated with nodes i and j, which must not be interior to a route (the only nodes not interior to the route are the first and last node in the route). The algorithm continues to merge routes until no further mergers produce positive savings, or until there is no feasible merger remaining. A feasible merger is one that satisfies all constraints specific to the problem such as the vehicle capacity constraint.

For an example of how the savings formula is obtained, consider two nodes i and j with a vehicle traveling from the depot to each of the nodes and back. When these two routes are merged into a single route, then the new route formed is from the depot to i to j and then back to the depot. Thus, the change in distance can be computed as follows:

$$s_{ij} = 2D_{0i} + 2D_{0j} - (D_{0i} + D_{ij} + D_{j0}) = D_{0i} + D_{0j} - D_{ij}.$$

Occasionally, the algorithm is modified to include a general savings matrix that is as follows: $s_{ij} = D_{0i} + D_{0j} - \lambda D_{ij}$ where $\lambda$ is a parameter which is usually set close to one [13]. If $\lambda$ is greater than one, the $\lambda D_{ij}$ component is larger and $s_{ij}$ is smaller. Therefore, fewer mergers are made. If $\lambda$ is less than one, more mergers are made.

**Algorithm:**

1. Calculate the savings matrix $s_{ij} = D_{0i} + D_{0j} - \lambda D_{ij}$ for all i, j.
2. Determine largest savings element $s_{ij}$.
3. Provided it is feasible, perform the merger of i and j found in step 2.
4. Repeat step 2 and 3 until the largest feasible savings element is 0 or negative.

**Subsection 3.1.1: Modified Clarke-Wright (MCW) for solving the VRPDR**

We have developed a Modified Clarke-Wright (MCW) algorithm - a variation of the CW algorithm which considers the upper and lower demand constraint. Whereas CW merges routes that reduce distance the most, MCW merges routes that reduce the objective function (1) the most. We begin by noting that it is always lucrative for a vehicle to deliver as much as it can for a given route. In an optimal solution, each vehicle will either deliver the entire vehicle capacity CAP, or it will be constrained by the upper demands of the customers on its route. That is to say, no vehicle will return to the depot with undelivered goods unless every customer in its route was given his upper demand. The amount dropped off by vehicle v can then always be expressed as:

min (CAP, $\sum_{k \in \text{Route } v} u_k$ ). Recall that $u_k$ is the upper demand of node k.

The MCW procedure begins with calculating the savings matrix $s_{ij}$ and $R_{ij}$ where $R_{ij}$ is the drop off reduction as a result of merging routes i and j,

$$R_{ij} = \min\left(CAP, \sum_{k \in \text{Route } i} u_k\right) + \min\left(CAP, \sum_{k \in \text{Route } j} u_k\right) - \min\left(CAP, \sum_{k \in \text{Route } i + \text{Route } j} u_k\right)$$

where Route i + Route j is the new route formed by merging routes i and j. Note that the total dropped off amount can only go down (or stay the same) as the result of a merger because a single vehicle can never deliver more to a set of customers than two vehicles can.

We then compute the total savings per merger $T_{ij}$ where $T_{ij} = a\ s_{ij} - b\ R_{ij}$. The largest positive element in the $T_{ij}$ matrix is identified and checked for feasibility. If the route is feasible, the merger is made between routes i and j. A feasible merger is one that satisfies the following constraints:

- i and j cannot be interior to the route, i.e., they must come immediately after or before the depot in their previous routes.
- The sum of the lower demands on a route should be less than or equal to the vehicle capacity, $\sum_{k \in \text{Route } i} l_k \leq CAP$ for all i.

If the merger is feasible, edge (i, j) is added, while edges (i, 0) and (j, 0) are removed. Once the routes are merged, those elements in the $T_{ij}$ matrix are set to negative infinity so that they are not reconsidered. Then the $R_{ij}$ matrix is recomputed and the $T_{ij}$ matrix is updated. Next, the largest positive remaining element in the $T_{ij}$ matrix is identified to form the next merger. The algorithm loops until all the elements are infeasible, zero, or negative.

For an example of how the total saving matrix $T_{ij}$ is obtained, consider two nodes1 and 2 and two vehicles 1 and 2. Vehicle 1 travels from the depot to the node 1 and back and vehicle 2 travels from depot to node 2 and back. Before the merger, the two vehicles will deliver the most they can (e.g. vehicle 1 delivers min (CAP, $u_1$) while vehicle 2 delivers min (CAP, $u_2$)). However, once these two separate routes are merged, one truck delivers to both nodes, a total of min (CAP, $u_1 + u_2$). The total savings is the change in distance minus the amount of total delivery lost due to the merger. Another way to represent this is by considering the objective function before and after the merger, denoted $Obj_{before}$, and $Obj_{after}$, respectively. Below is an example of this two-node merge.

$Obj_{before} = c_{before} - r_{before} = a\,[2(D_{01} + D_{02})] - b\,[\min(CAP, u_1) + \min(CAP, u_2)]$ .

$Obj_{after} = c_{after} - r_{after} = a[D_{01} + D_{12} + D_{20}] - b\,[\min(CAP, u_1 + u_2)]$.

$T_{12} = Obj_{before} - Obj_{after} = a[D_{01} + D_{02} - D_{12}] - b[\min(CAP, u_1) + \min(CAP, u_2) - \min(CAP, u_1 + u_2)] = a\,s_{12} - b\,R_{12}$.

The parameter $\lambda$ is then introduced as well into the savings matrix as it was in the original CW method ($s_{ij} = D_{0i} + D_{0j} - \lambda D_{ij}$). Figure 1 is a complete graph, and we will use it to illustrate the MCW algorithm step-by-step.

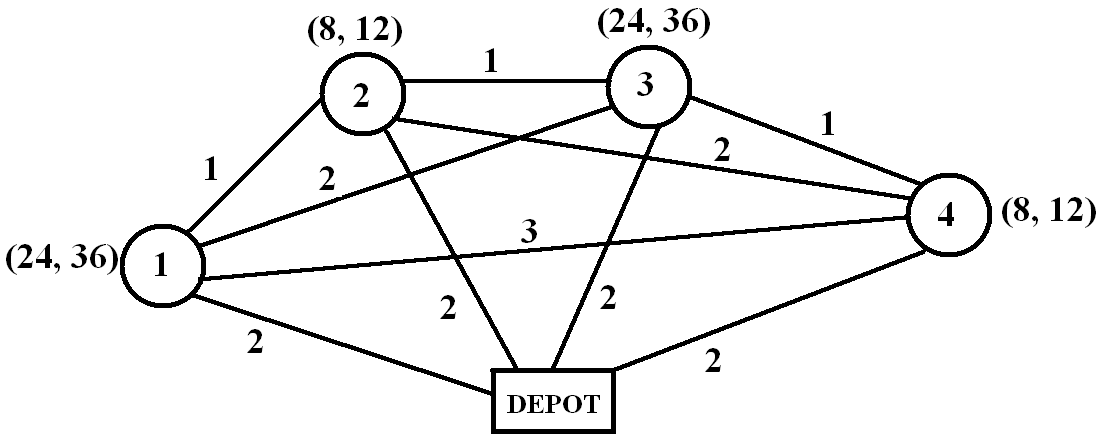



**Figure 1: A complete graph with four customers and a depot.**

In Figure 1 there are five nodes, 0, 1, 2, 3, and 4. Node 0 represents the depot and the other nodes are the customers. The numbers in parentheses next to each customer are their demand ranges. The demand range is determined as follows: if the customer demand is 30 and the α value is 0.2 (20%) then the dropped off amount can go as low as 24 or as high as 36. The number along each arc is the distance of that arc. There are four vehicles with capacity of 40 for each. Let us assume that for this specific example the ratio of b to a is b/a = 1, where a = 0.5 and b = 0.5.

Figure 2 is the first step of the MCW algorithm. A single truck will visit each customer, and since the truck capacity is greater than all the customers' upper demands, all the customers receive their upper demands.

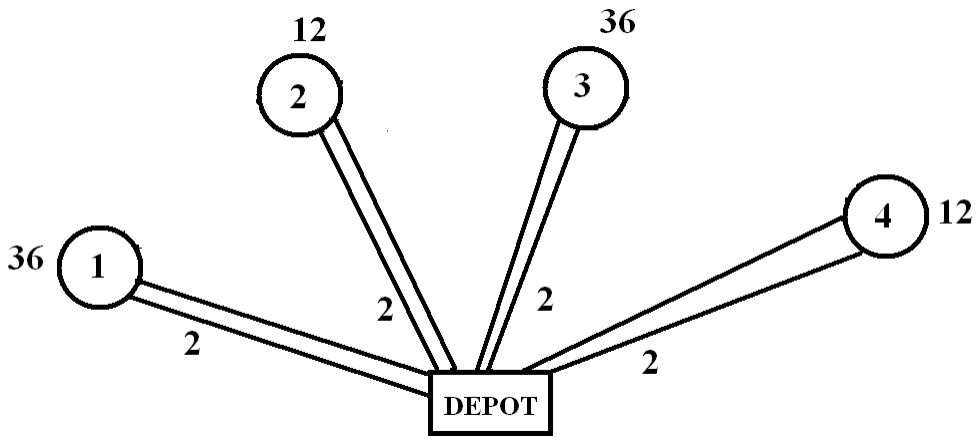



**Figure 2: A single truck visits each node.**

$$\begin{bmatrix} X & -2.5 & -15 & -3.5 \\ & X & -2.5 & 1 \\ & & X & -2.5 \\ & & & X \end{bmatrix} = \mathbf{a} \begin{bmatrix} X & 3 & 2 & 1 \\ & X & 3 & 2 \\ & & X & 3 \\ & & & X \end{bmatrix} - \mathbf{b} \begin{bmatrix} X & 8 & 32 & 8 \\ & X & 8 & 0 \\ & & X & 8 \\ & & & X \end{bmatrix}$$

**Total Savings = $T_{ij}$ = a $s_{ij}$ - b $R_{ij}$ = a x distance savings – b x drop off reduction.**

Above, the X represents an element of the matrix corresponding to an infeasible merger.

We choose the largest positive component in the $T_{ij}$ matrix and connect the two nodes. The element $T_{24} = 1$ is largest and, therefore, we connect nodes 2 and 4. This is shown in Figure 3.

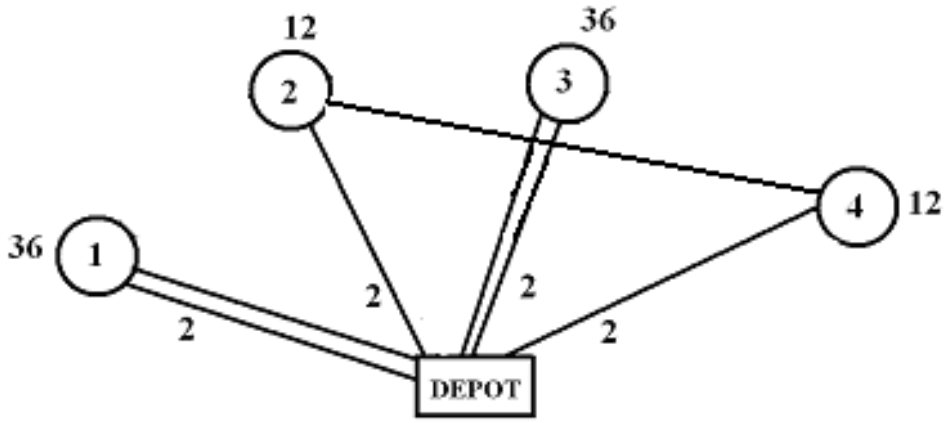



**Figure 3: MCW algorithm connects nodes 2 and 4.**

Since node 2 and node 4 are connected we will eliminate the merger from the matrix so that it not chosen again. The element $R_{24}$ is set to $\infty$.

$$\begin{bmatrix} X & -2.5 & -15 & -3.5 \\ & X & -2.5 & -\infty \\ & & X & -2.5 \\ & & & X \end{bmatrix} = \mathbf{a} \begin{bmatrix} X & 3 & 2 & 1 \\ & X & 3 & 2 \\ & & X & 3 \\ & & & X \end{bmatrix} - \mathbf{b} \begin{bmatrix} X & 8 & 32 & 8 \\ & X & 8 & \infty \\ & & X & 8 \\ & & & X \end{bmatrix}$$

**Total Savings = $T_{ij}$ = a $s_{ij}$ - b $R_{ij}$ = a x distance savings – b x drop off reduction.**

Now we re-calculate the $R_{ij}$ matrix as follows:

$R_{ij} = \begin{bmatrix} X & 20 & 32 & 20 \\ & X & 20 & \infty \\ & & X & 20 \\ & & & X \end{bmatrix}$ where X denotes that the components are infeasible due to the vehicle capacity constraint.

The new $T_{ij}$ matrix is then:

$T_{ij} = \begin{bmatrix} X & -8.5 & -15 & -9.5 \\ & X & -8.5 & -\infty \\ & & X & -8.5 \\ & & & X \end{bmatrix}$. There are no positive numbers remaining, so the procedure stops.

Hence the final routes are illustrated in Figure 4. We use a total of three vehicles.

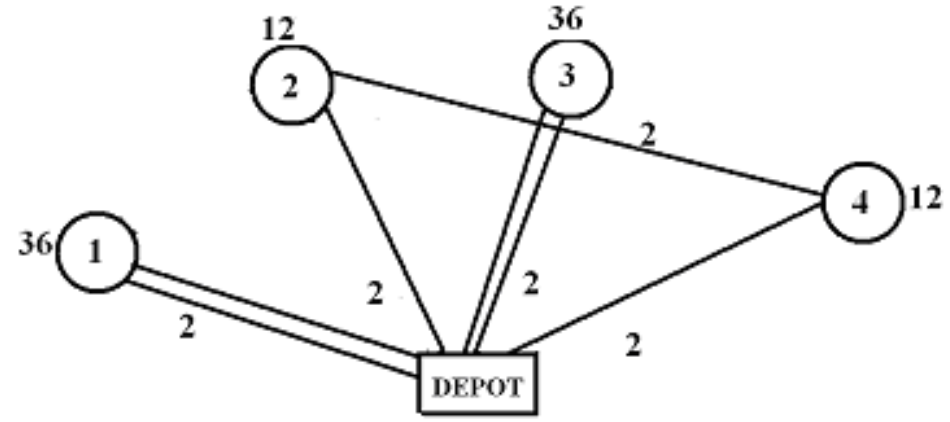



**Figure 4: MCW's final routes.**

Thus, the objective function = a x Distance –b x Dropped off amount = 0.5 x 14 – 0.5 x 96 = -41. If we solve this problem using the CW algorithm (which optimizes only distance) the routes would look as illustrated in Figure 5.

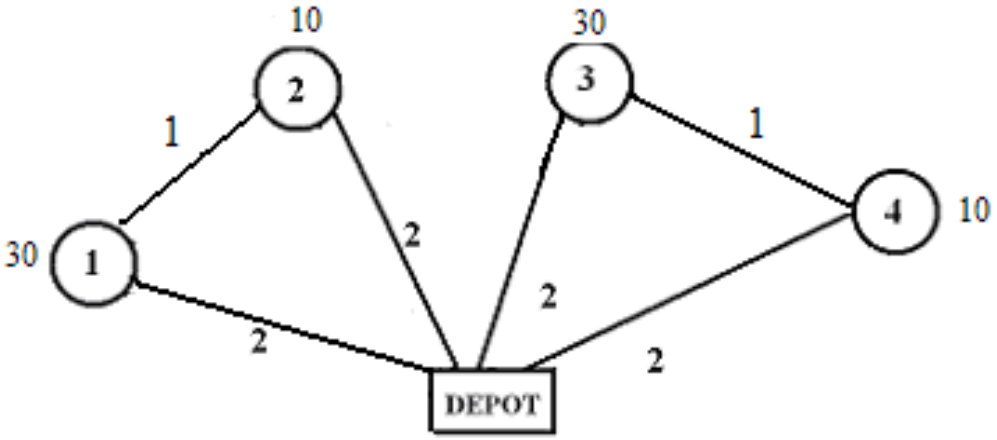



**Figure 5: CW final routes.**

Here, the objective function is 0.5 x 10 – 0.5 x 80 = -35. In this example MCW outperforms CW, as the objective function is lower.

**MCW Algorithm:**

1. We add the edges from the depot to node i, and from node i to the depot, (i,0) and (0,i), for all i.

2. Compute the savings matrix $s_{ij} = D_{0i} + D_{0j} - \lambda D_{ij}$.

3. Compute the R matrix

$R_{ij}$ = min (CAP, $\sum_{k \in \text{Route } i} u_k$ ) + min (CAP, $\sum_{k \in \text{Route } j} u_k$ ) - min (CAP, $\sum_{k \in \text{Route } i + \text{Route } j} u_k$ ), where Route i = set of all the nodes in the same route as node i.

4. Compute the Total Savings matrix $T_{ij} = a \times s_{ij} - b \times R_{ij}$

5. Look for the largest $T_{ij}$ and then check for feasibility constraints

( $\sum_{k \in \text{Route } i + \text{Route } j} l_k \leq$ CAP and i and j cannot be interior to the route). If it is feasible, add edge (i,j) and remove edges (0,i) and (0,j).

6. Go to 3 and repeat until $T_{ij}$ is infeasible or negative for all i and j.

We will now discuss ways to improve the output of the MCW algorithm. These techniques form the final metaheuristic to be discussed in Chapter 4.

***Section 3.2: Two Opt (TO)***

The Two opt (TO) operation is a way of improving an existing solution. It involves swapping a pair of edges between any four nodes to reduce the distance in the CVRP and objective function in the case of VRPDR. The algorithm involves looping through all pairs of edges. The first switch that reduces the objective function is accepted and the loop is ended. The swaps could be between routes or within a route. If the swap is within a route then the total drop off amount in the route remains the same, however, the order in which the vehicles visit every customer is changed. If the swap is between routes then the total drop

off amount in each route can change. Because of this, we have modified the TO algorithm to take into account the entire objective function. All the exchanges within a route are implemented through REVERSEPARTIALROUTES (RPR) and all exchanges between two routes are implemented through SWAPENDS (SE). These operations are illustrated below.

**REVERSEPARTIALROUTE (RPR)**

Figure 6 is an example of a route on which we will perform the RPR operation. In Figure 6, we show five customers and beside each customer in parenthesis is his lower and upper demands.

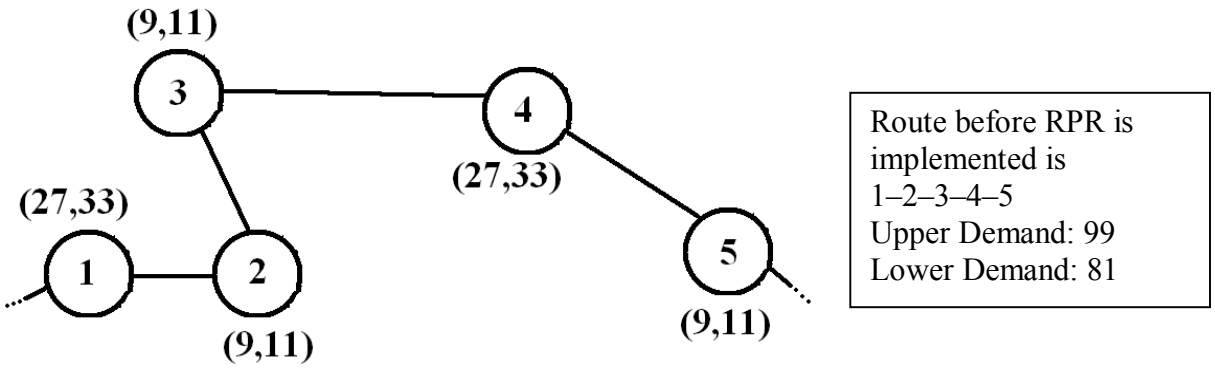



**Figure 6: RPR (Intra-Route), route before operation is performed.**

The illustration in Figure 6 shows the route before RPR is performed.

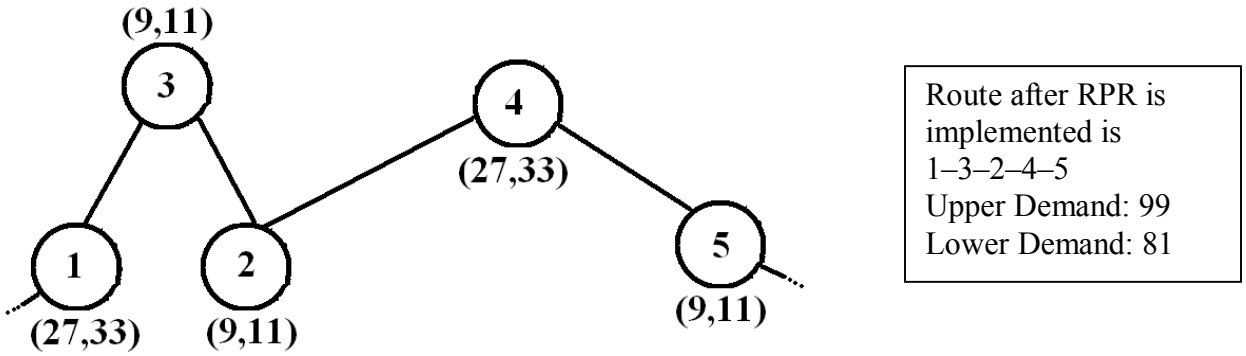



**Figure 7: RPR (Intra-Route), route after operation is performed.**

Note that in Figures 6 and 7, RPR has been performed within a single route.

**SWAPENDS (SE)**

Figures 8 and 9 show a SE operation occurring between two routes. Note that the route demands change when we perform a SE operation involving two routes. The SE operation swaps edges between routes. In Figure 8, we can see that route 1 has customers 1, 2, and 6, but after the SE operation, route 1 has customers 1, 2 and 3 (see Figure 9). This changes the upper and lower demands of route 1 from 99 to 77 and 81 to 63, respectively.

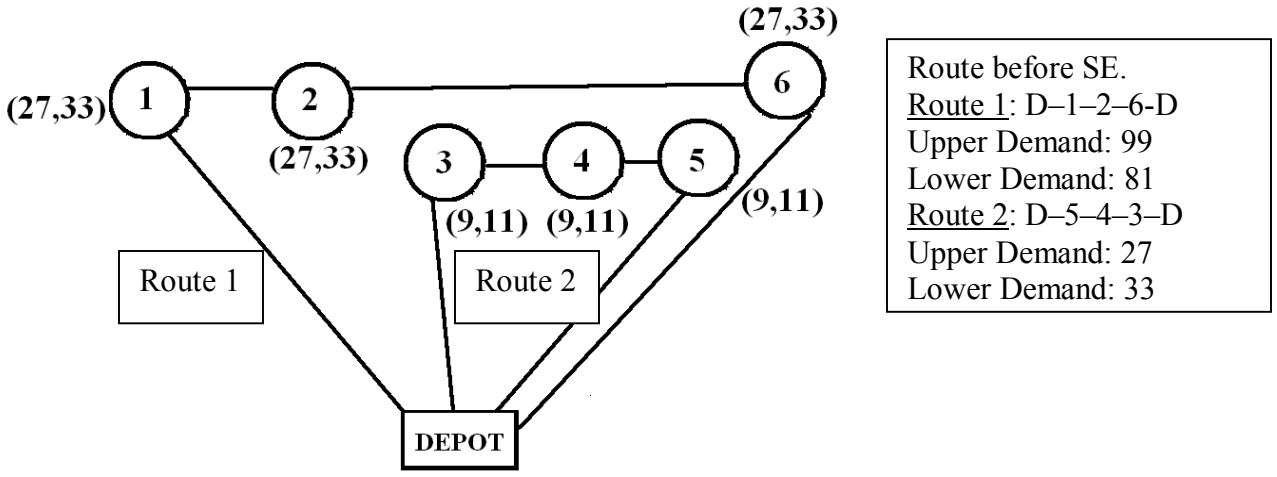



**Figure 8: SE (Inter-Route), route before operation is performed.**

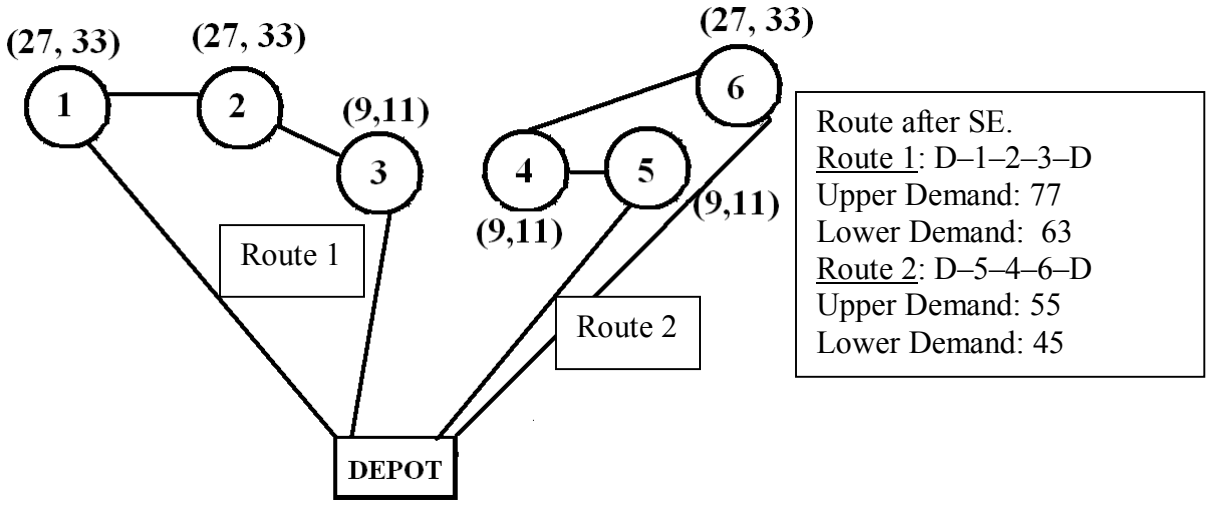



**Figure 9: SE (Inter-Route), route after operation is performed**

**Two-opt Algorithm:**

For each pair of edges (i,k) and (j,l):

1. Compute distance savings and change in drop off amounts (assuming that the switch is feasible) where the distance savings is $D_{ik} - D_{ij} + D_{jl} - D_{kl}$ for all i, j, k, l; and the change in drop off amounts is:

   $$\min\left(CAP, \sum_{p \in IR(i,k)} u_p\right) + \min\left(CAP, \sum_{p \in IR(j,l)} u_p\right) - \min\left(CAP, \sum_{p \in FR(i,j)} u_p\right) - \min\left(CAP, \sum_{p \in FR(k,l)} u_p\right)$$

   *FR(i,j): FinalRoute (i,j) = set of all the nodes in the same route as node i and j after performing the test switch from (i,k) to (i,j) and (j,l) to (k,l) .*

   *FR(k,l): FinalRoute (k,l) = set of all the nodes in the same route as node k and l after performing the test switch from (i,k) to (i,j) and (j,l) to (k,l).*

   *IR(i,k):InitialRoute (i,k) = set of all the nodes in the same route as node i and k before performing the test switch.*

*IR(j,l): InitialRoute (j,l) = set of all the nodes in the same route as node j and l before performing the test switch.*

2. If (a x savings – b x change in drop off) > 0 then switch edge (i,k) with (i,j) and edge (j,l) with (k,l), then exit the loop; otherwise go to 1.

Note: All the exchanges are implemented either thru SWAPENDS or REVERSEPARTIALROUTE

***Section 3.3: One Point Move (OPM)***

The One Point Move (OPM) is yet another operation used to improve upon an existing solution. The OPM involves adjusting the order in which the nodes are visited by placing one node after or before another node. Once again, the move could be inter-route or intra-route. As before, the move is made in the CVRP to reduce the travel distance, whereas in VRPDR, OPM aims to reduce the total objective function. OPM can be implemented either thru POSTSERT or PRESERT operations. Given two nodes A and B, POSTSERT would adjust the routes such that node B is visited just after node A, while PRESERT would insert node B just before node A. Both POSTSERT and PRESERT can be performed between or within a route. Figure 10 illustrates a route and in Figure 11 the

POSTSERT move is made.

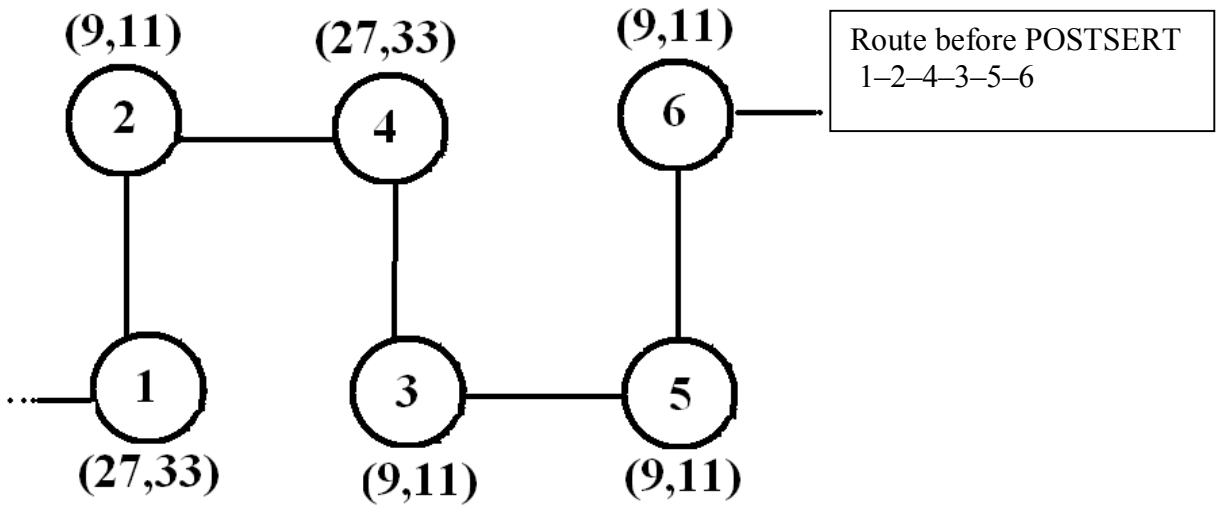



**Figure 10: POSTSERT (Intra-Route), route before operation is performed.**

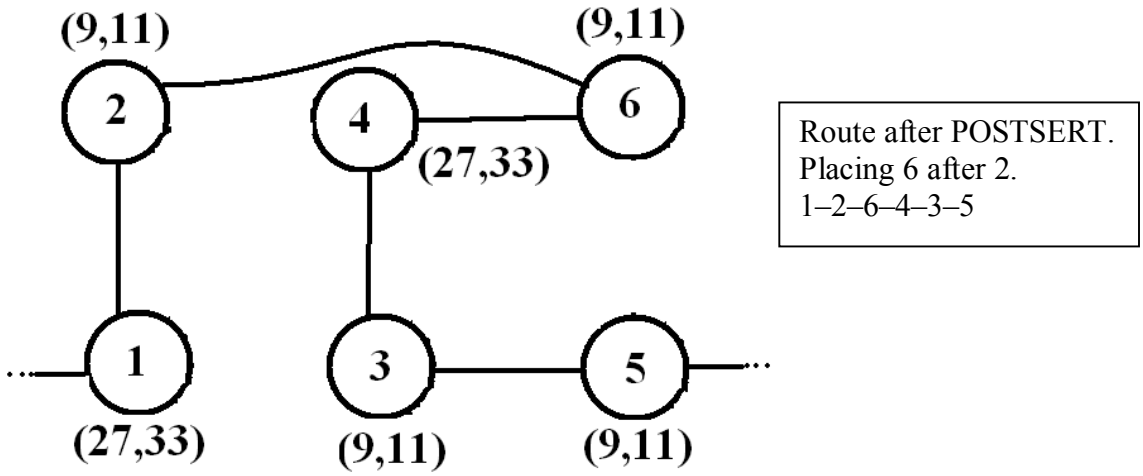



**Figure 11: POSTSERT (Intra-Route), route after operation is performed**.

Figure 12 shows two initial routes. In Figure 13 we implement POSTSERT between two nodes in separate routes.

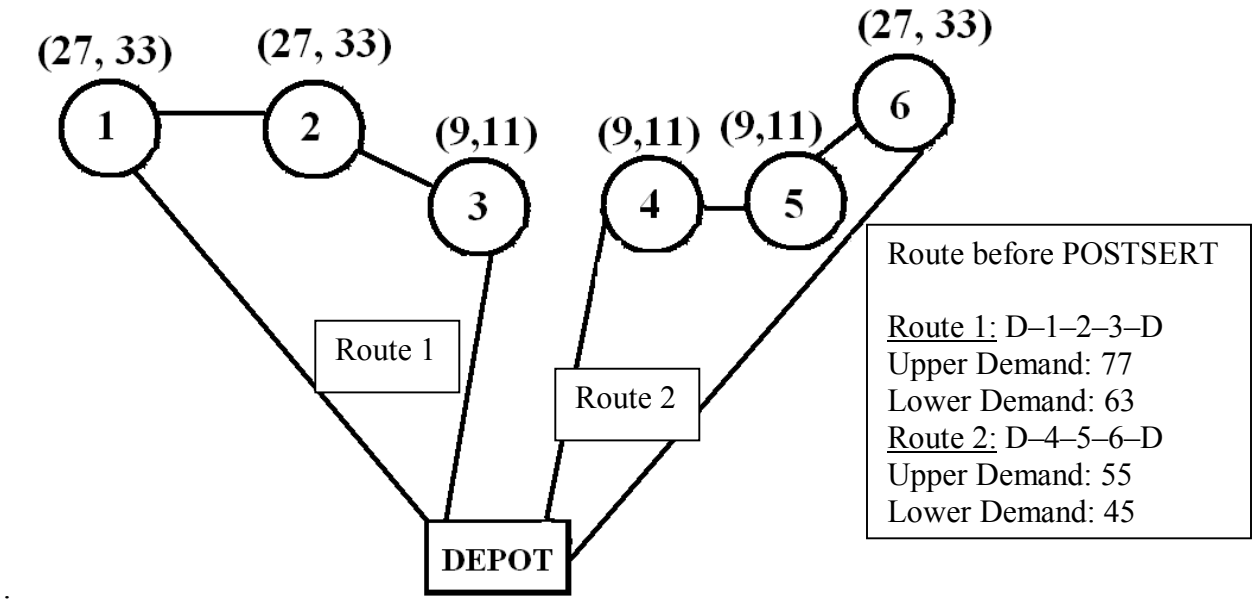



**Figure 12: POSTSERT (Inter-Route), routes before operation is performed.**

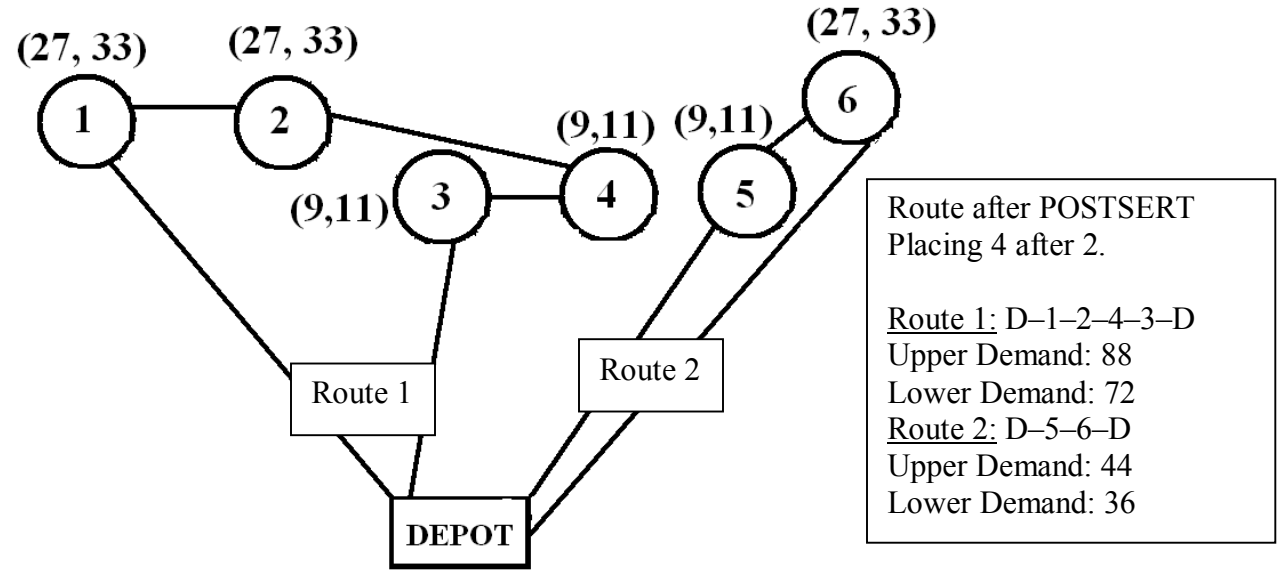



**Figure 13: POSTSERT (Inter-Route), routes after operation is performed.**

PRESERT performs the same operations as POSTERT, except it places a customer before another customer. As before, when the operation occurs between

routes, the total upper and lower demand of the routes may change. Let us assume that Figure 10 is the existing route and we will perform PRESERT on that.

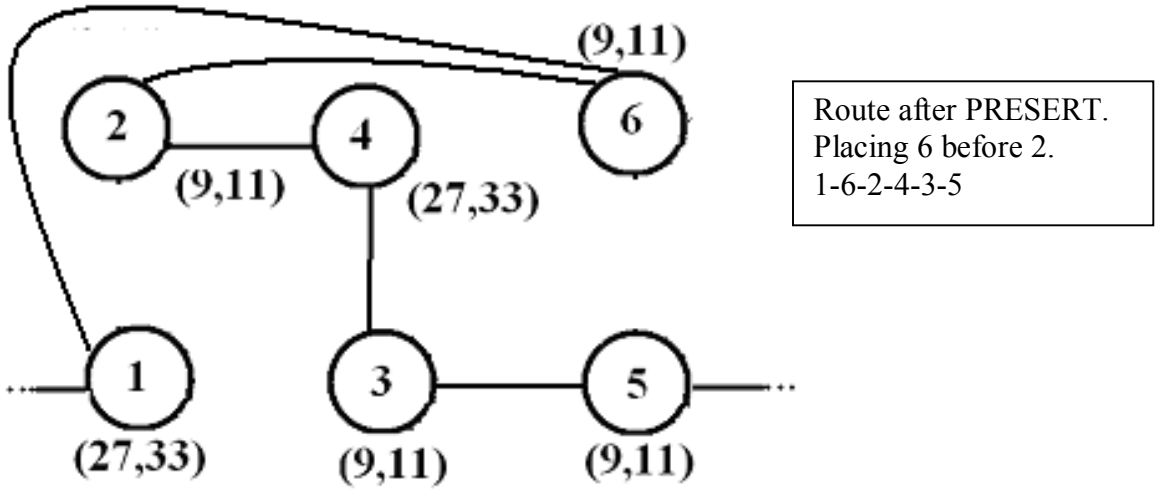



**Figure 14: PRESERT (Intra-Route), route after operation is performed**.

Now we will illustrate the PRESERT between routes and assume that Figure 12 were the existing routes.

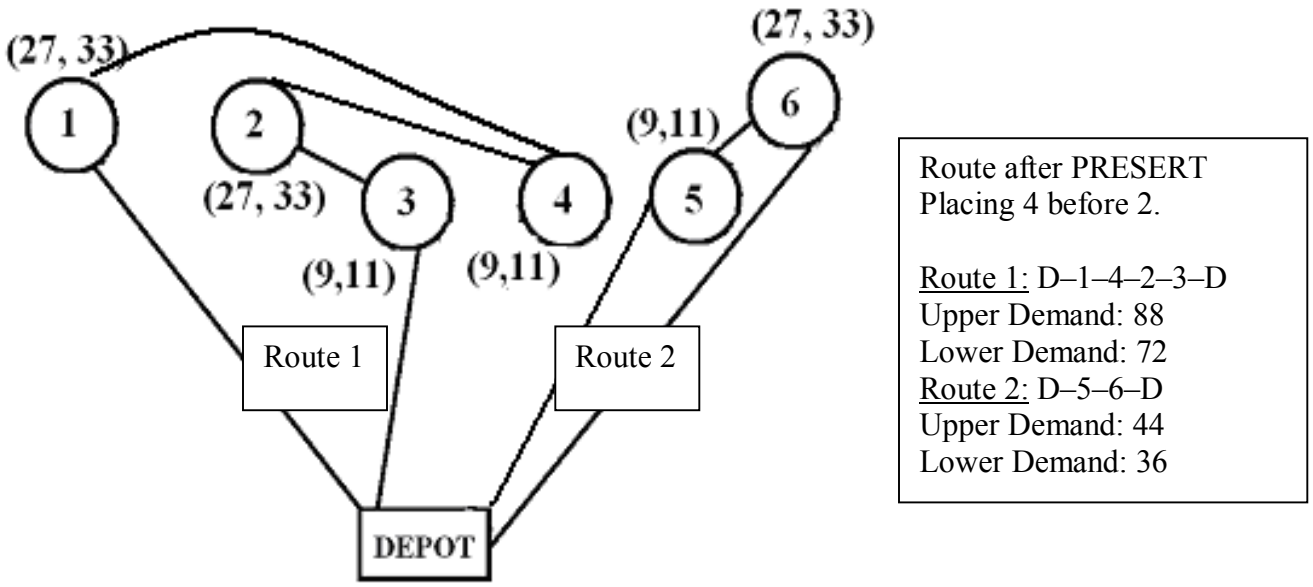



**Figure 15: PRESERT (Inter-Route), routes after operation is performed.**

**OPM Algorithm:**

For all j, v:

Let i = previous (j);

Let k = next (j);

Let u = previous (v);

Let w = next (v).

Here, the notation previous (i) is the node visited before node i, and next (i) is the node visited just after node i.

1. Compute distance savings and change in drop off amounts (assuming that the switch is feasible). The distance savings is $D_{uv} + D_{ij} + D_{vw} - D_{iv} - D_{uw} - D_{vj}$ and the change in drop off amounts is: $\min (CAP, \sum_{p \in IR(j)} u_p) + \min (CAP, \sum_{p \in IR(v)} u_p) - \min (CAP, \sum_{p \in FR(i,v,j,k)} u_p) - \min (CAP, \sum_{p \in FR(w,u)} u_p)$

*FR(i,v,j,k) : FinalRoute (i,v,j,k) = set of all the nodes in the same route as node i, j, k, and v after performing the test switch from (i,j) to (i,v) and (v,u) to (v,j) and (v,w) to (u,w).*

*FR(w,u): FinalRoute (w,u) = set of all the nodes in the same route as node w and u after performing the test switch from (i,j) to (i,v) and (v,u) to (v,j) and (v,w) to (u,w).*

*IR(j): InitialRoute (j) = set of all the nodes in the same route as node j before performing the test switch.*

*IR(v): InitialRoute (v) = set of all the nodes in the same route as node v before performing the test switch.*

2. If (a x savings – b x change in drop off) > 0 then switch edge (i,j) to (i,v) and (v,u) to (v,j) and (v,w) to (u,w), then exit the loop or else go to 1.

***Section 3.4: Two Point Move (TPM)***

Two Point Move (TPM) is the third heuristic used to improve existing solutions by swapping two customers within a route or between two routes. The swap is made to reduce distance in the CVRP. In VRPDR, the swap is performed to reduce the total objective function. These moves are accomplished via the SWAP operation. If the SWAP operation is performed intra-route, the drop off for that route does not change. We will use Figure 10 as the initial route and then perform SWAP between nodes 2 and 5.

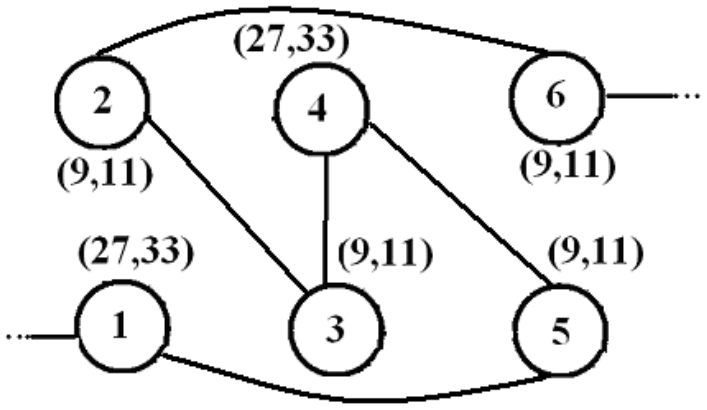



**Figure 16: SWAP (Intra-Route), routes after operation is performed.**

Now we will perform SWAP between nodes 2 and 5 on Figure 12. This time the drop off amounts will change since the SWAP is inter-route.

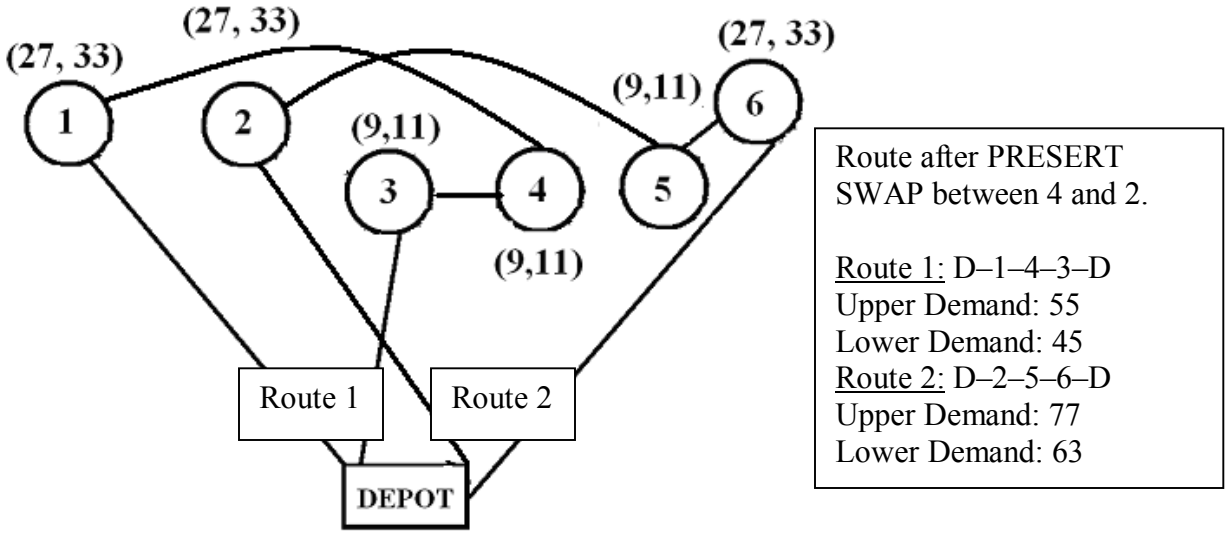



**Figure 17: SWAP (Inter-Route), routes after operation is performed.**

**TPM Algorithm:**

For all j, v;

Let i = previous (j);

Let k = next (j);

Let u = previous (v);

Let w = next (v).

1. Compute distance savings and change in drop off amounts (assuming that the switch is feasible), where the distance savings is $D_{jk} + D_{ij} + D_{uv} + D_{vw} - D_{iv} - D_{vk} - D_{uj} - D_{jw}$ for all i, j, k, u, v and w; and the change in drop off amounts is: min (CAP, $\sum_{p \in IR(i,j,k)} u_p$ ) + min (CAP, $\sum_{p \in IR(u,v,w)} u_p$ ) - min (CAP, $\sum_{p \in FR(i,v,k)} u_p$ ) - min (CAP, $\sum_{p \in FR(u,j,w)} u_p$ ).

*FR(i,v,k) : FinalRoute (i,v,k) = set of all the nodes in the same route as node i , v, and k after performing the test switch from (i,j) to (i,v) and (j,k) to (v,k) and (u,v) to (u,j) and (v,w) to (j,w).*

*FR(j,w,u): FinalRoute (j,w,u) = set of all the nodes in the same route as node j, u and w after performing the test switch (i,j) to (i,v) and (j,k) to (v,k) and (u,v) to (u,j) and (v,w) to (j,w).*

*IR(i,j,k): InitialRoute (i,j,k) = set of all the nodes in the same route as node i, j and k before performing the test switch.*

*IR(u,v,w) : InitialRoute (u,v,w) = set of all the nodes in the same route as node u,v, and w before performing the test switch.*

2. If (a x savings – b x change in drop off) > 0 then switch edge (i,j) to (i,v) and (j,k) to (v,k) and (u,v) to (u,j) and (v,w) to (j,w), then exit the loop or else go to 1.

# Chapter 4: Metaheuristics

A metaheuristic is a high-level family of heuristics. In this section, we apply a well-known metaheuristic called RECORD TO RECORD (RTR) to solve the VRPDR.

## ***Section 4.1: Record-to-Record (RTR)***

RECORD TO RECORD (RTR) is a metaheuristic developed by Dueck [7] that combines the improvement techniques mentioned in Chapter 3. RTR iterates to find a solution better than the existing one by doing moves such as TO, OPM, and TPM. RTR iterates by making both uphill (increasing objective function) and downhill (decreasing objective function) moves. In doing so, the metaheuristic also alleviates the problem of a solution getting stuck at local optimum. Furthermore, the uphill moves are performed in a controlled manner to explore the local solution space. By doing this it accepts short term increases in the objective function in order to better search the solution space. The reason why this algorithm has been termed RTR, is because the algorithm records the best solution found thus far, denoted $S_R$ (the "record"). Typically, RTR reduces the distance in the classic CVRP. In our problem, VRPDR, RTR reduces the objective function. We denote the demand range version of the RTR algorithm as RTRDR.

A more mathematical way to explain RTRDR would be as follows: Let the current solution be S and let N(S) be the neighborhood of solution S. N(S) would contain alternative solutions in the vicinity of S. RTR would allow a random

selection of a solution S' within the neighborhood N(S). In the downhill phase, if $O(S')<O(S)$ (with O(S) indicating the objective function of solution S), then the move to S' is made and the better solution is recorded. In the uphill phase, if $O(S') \leq O(S_R)$ + Deviation then the move is made where Deviation is usually set as a fixed fraction of $O(S_R)$ (here we use Deviation = 1% of $O(S_R)$. This process is repeated to explore better solutions than the existing record $S_R$.

***Section 4.2: RTRDR Metaheuristic Algorithm***

The proposed metaheuristic is explained in the next steps. This algorithm modifies Li's algorithm to solve the VRPDR [10].

Step 0: Initialization.

Parameters are I, and $\lambda$.

Set I = 30, and loop through each of $\lambda \in \{0.6, 1.4, 1.6\}$

Step 1: Starting solution.

Generate an initial feasible solution using the modified Clarke-Wright algorithm with parameter $\lambda$.

Set $O(S_R)$ = objective value of the current solution.

Set Deviation = 0.01 x $O(S_R)$.

Step 2: Perform uphill moves.

For i = 1 to I

Perform OPM with record-to-record travel.

Perform TPM with record-to-record travel.

Perform TO with record-to-record travel.

If no feasible record-to-record travel move is made in any of these three steps, jump to Step 3.

If a new record is produced, update $O(S_R)$ and Deviation.

End i loop.

Step 3: Perform downhill moves.

While better solution found

Perform OPM with record-to-record travel.

End while.

While better solution found

Perform TPM with record-to-record travel.

End while.

While better solution found

Perform TO with record-to-record travel.

End while.

For all three, only downhill moves are allowed.

If a new record is produced, update $O(S_R)$ and Deviation.

Step 4: Repeat iterations until solution no longer changes.

If no new record has been produced in the past 5 uphill and downhill moves, perturb the solution, as described below, and go to Step 2. This perturb step happens only once, the second time that no new record is produced after 5 uphill and downhill moves, go to Step 6.

Step 6: Go to Step 1 and select the next value of $\lambda$.

Note: TO, OPM, and TPM are the same as described in Section 3.2, 3.3, and 3.4 with the exception now that, in the uphill phase, it accepts moves provided the objective function does not exceed $O(S_R)$ +Deviation.

Algorithm for perturbing the feasible solution

For each node i, define $r(i) = d_i / s(i)$. Where $d_i$ is the (middle) demand of customer i and $s(i) = D_{ij} + D_{ik} - D_{jk}$, where for all i , j = previous ( i ) and k = next ( i ).

Sort the r(i) values in ascending order and select the first M nodes where M = min (20, n/10), where n is the number of nodes. Try to add these nodes, one by one, into a new location on a tour using least cost inserting while maintaining feasibility.

Algorithm for drop off distribution among customers

We note that the objective function is indifferent to how the drop off amounts are distributed within a route, so the output of the algorithm above only needs to record the total drop off amount for each route. To determine how the drop off amounts are distributed within a route, we use the following algorithm.

Step 1: Initialize the drop off amount (for each customer) to their lower demand.

Step 2: Repeatedly loop through all customers adding one to the drop off amount for each customer. If the upper demand of a customer is attained and the vehicle capacity is not exceeded, then continue to increment the drop off

amount one at a time, but skip those customers whose delivery amounts is their upper demand. Once the vehicle capacity is satisfied for a given route, that entire route is skipped for future iterations.

***Section 4.3: Comparison Algorithms***

The algorithms described in this section provide a benchmark which will be compared to results obtained by the RTRDR algorithm in Section 4.2. The comparison algorithm performs the RTRDR algorithm above (with b=0) so that it initially optimizes only for distance. We proceed to describe the three comparison heuristics, which differ only in the initial drop off amounts. They are:

- Lower Demand (VRPL).
- Middle Demand (VRPM).
- Upper Demand (VRPU).

When performing VRPL, the demand for each customer was fixed to be their lower demands from the demand range. Once the demand is set for each customer, the RTR algorithm is performed with $\alpha = 0$ and $b = 0$. When this algorithm is completed, in order to ensure that all vehicles drop off the most possible, we restore $\alpha$ to its non-zero value and set the drop off of each route i to min (CAP, $\sum_{k \in \text{Route i}} u_k$ ). Then we allocate to each customer their drop off amounts via the drop off distribution among customers algorithm, explained in Section 4.2. VRPM and VRPU are performed in the same way, but begin with the middle demand and upper demand as the initial drop off amounts, respectively.

One could argue that there are several algorithms that could be developed to compare results with the proposed algorithm in Section 4.2. However, we chose this procedure to show the importance of reducing distance and maximizing drop off together, as opposed to just reducing distance initially and then maximizing the drop off amount.

# Chapter 5: Optimal Results

In this section, we will briefly discuss the results from the optimal formulation in Chapter 2. The optimal results were computed via the optimizing software Xpress.

The table below shows us the results for number of nodes or customers as large as 9. Table 1 has six columns: number of nodes, objective function, distance, drop off, routes/trucks used, and the computation time needed for the various node sizes. The optimal solution was run for a single b and α value of 0.025 and 0.2 respectively, and the truck capacity was 100.

| # of Nodes | Objective Fctn | Distance | Drop Off | Trucks | Time (s) |
|---|---|---|---|---|---|
| 3 | 0.479627 | 5.87963 | 108 | 2 | 0.3 |
| 4 | -1.2908 | 5.9092 | 144 | 2 | 1.5 |
| 5 | -4.85126 | 3.74874 | 172 | 2 | 3.8 |
| 6 | -6.00121 | 4.39879 | 208 | 3 | 32.1 |
| 7 | -5.06527 | 7.53473 | 252 | 4 | 229 |
| 9 | -8.37335 | 7.02665 | 308 | 4 | 137167.5 |

**Table 1: Optimal solutions with a=0.5, b = 0.025, α = 0.2, CAP =100.**

It is evident from the computation time that it grows exponentially. Hence a node size larger than 9 would have taken days to compute. We tested the MCW algorithm on these small node sizes. If we recall from Chapter 3, MCW was used to achieve an initial solution for our metaheuristic. The results in Table 2 show us that it did as well as the optimal solution for node size as large as 7. For graphs with 9 nodes, the objective function is 0.05 above the optimal solution. However, it is worth noting that the computational times were all below one second.

| # of Nodes | Objective Fctn | Distance | Drop Off | Trucks | Time |
|---|---|---|---|---|---|
| 3 | 0.479627 | 5.87963 | 108 | 2 | <1 sec |
| 4 | -1.2908 | 5.9092 | 144 | 2 | <1 sec |
| 5 | -4.85126 | 3.74874 | 172 | 2 | <1 sec |
| 6 | -6.00121 | 4.39879 | 208 | 3 | <1 sec |
| 7 | -5.06527 | 7.53473 | 252 | 4 | <1 sec |
| 9 | -8.320646 | 7.879353 | 324 | 5 | <1 sec |

**Table 2: Modified Clarke-Wright solution with a=0.5, b = 0.025, α = 0.2, CAP =100.**

| # of Nodes | Objective Fctn. | Distance | Drop Off | Trucks | Time | Abs Diff. |
|---|---|---|---|---|---|---|
| 3 | 0.621843 | 5.62183 | 100 | 1 | <1 sec | 0.296514 |
| 4 | 0.233155 | 5.233155 | 100 | 1 | <1 sec | 1.180628 |
| 5 | -3.44279 | 3.35721 | 136 | 2 | <1 sec | 0.290331 |
| 6 | -4.67986 | 3.920144 | 172 | 2 | <1 sec | 0.220181 |
| 7 | -4.19815 | 5.801852 | 200 | 2 | <1 sec | 0.171189 |
| 9 | -5.4317 | 6.368279 | 236 | 2 | <1 sec | 0.351311 |

**Table 3: Comparison Algorithm (CWL) with a=0.5, b = 0.025, α = 0.2, CAP =100.**

To compare the optimal results we perform Clarke Wright using customer's lower demands. Then we check for remaining vehicle capacity and distribute the remaining amount as much as possible i.e. till their upper demands are met or vehicle capacity is satisfied. Table 3 shows us that since the objective function in this algorithm did not take drop off into account, the objective function could not reduce to the levels seen in the optimal solution, even for small node sizes. Based

on these results, we can say that drop off is an important part of the equation to consider during optimization, not just as an afterthought. Ignoring drop off can cause the solution to suffer by a substantial amount.

# Chapter 6: Computational Results

In this section, we will discuss and compare the computational results of the RTRDR algorithm explained in Section 4.2 with the comparison algorithms explained in Section 4.3. The algorithms have been developed to solve the same set of graphs used by Li [10] with large sizes ranging from 560 to 1200 customers. The customers have a demand of either 10 or 30. However, this variant of the problem has demand ranges and so the demand ranges will spread based on the α value provided. Each graph was run with 19 different b values ranging from 0.005 to 2.5 and with three α values of 0.1, 0.2 and 0.3. The b values were broken into three categories: high (0.16, 0.18, 0.2, 0.4, 1, 2.5), medium (0.06, 0.07, 0.08, 0.09, 0.1, 0.12, 0.14), and low (0.005, 0.01, 0.02, 0.03, 0.04, 0.05) rewards. The results from each category were averaged to come up with average results for the three categories of high, medium, and low b values. Below are some examples of the solution using RTRDR.

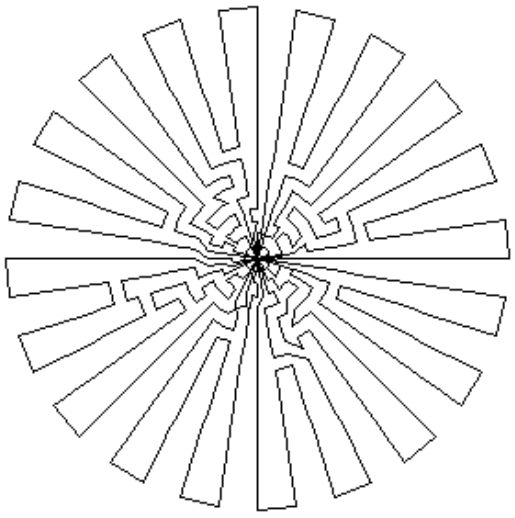


**Figure 18: Example solution of a graph with 1200 customers.**

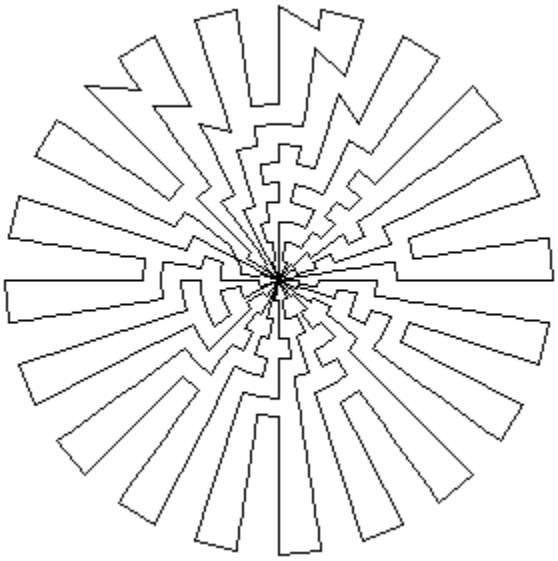


**Figure 19: Example solution of a graph with 560 customers.**

Tables 4a to 4k show detailed results from the runs for RTRDR, VRL, VRPM, and VRPU averaged over the b values for problem sizes ranging from 560 to 1200 nodes. Before we analyze the results, it would be useful to go over what each column means. All three categories are run for three different α values. The first column identifies the respective b group and α values and the associated nodes sizes. The next column is the objective function based on the distance, drop off, a, and b values. The third and fourth columns are total distance traveled and the total drop off amount. The next nine columns are objective function, distance and drop off for the respective comparison algorithms VRPL, VRPM, and VRPU.

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 7986.60 | 16602.12 | 12099.5 | 15951.97 | 16586.9 | 12289 | 15984.06 | 16608.3 | 12082 | 15988.77 | 16625.3 | 12320 |
| MEDIUM | 7188.58 | 16693.06 | 12269.7 | 14269.55 | 16586.9 | 12289 | 14329.98 | 16608.3 | 12082 | 14302.10 | 16625.3 | 12320 |
| HIGH | -738.00 | 16757.58 | 12320.0 | -1600.82 | 16586.9 | 12289 | -1273.06 | 16608.3 | 12082 | -1608.30 | 16625.3 | 12320 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 7969.93 | 16610.52 | 12870.0 | 15966.64 | 16569.9 | 11676 | 15962.26 | 16608.3 | 12504 | 16012.40 | 16706.8 | 13440 |
| MEDIUM | 7135.54 | 16770.36 | 13248.0 | 14368.14 | 16569.9 | 11676 | 14250.40 | 16608.3 | 12504 | 14172.40 | 16706.8 | 13440 |
| HIGH | -1494.08 | 16890.23 | 13400.0 | -710.58 | 16569.9 | 11676 | -1897.62 | 16608.3 | 12504 | -3184.40 | 16706.8 | 13440 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 8024.58 | 16726.87 | 13099.5 | 15933.95 | 16588.0 | 12659 | 15950.89 | 16608.3 | 12724 | 16351.93 | 17104.2 | 14560 |
| MEDIUM | 7097.99 | 16865.93 | 14098.1 | 14200.87 | 16588.0 | 12659 | 14208.92 | 16608.3 | 12724 | 14358.60 | 17104.2 | 14560 |
| HIGH | -2262.58 | 16998.97 | 14491.2 | -2147.32 | 16588.0 | 12659 | -2223.22 | 16608.3 | 12724 | -4444.60 | 17104.2 | 14560 |

**Table 4a: Results for 560 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 7101.26 | 14883.28 | 13143.5 | 14209.72 | 14878.6 | 12946 | 14233.65 | 14904.8 | 12990 | 14211.70 | 14893.7 | 13200 |
| MEDIUM | 6204.15 | 14896.66 | 13195.4 | 12437.35 | 14878.6 | 12946 | 12455.26 | 14904.8 | 12990 | 12404.56 | 14893.7 | 13200 |
| HIGH | -2318.80 | 14898.17 | 13199.3 | -4281.48 | 14878.6 | 12946 | -4320.40 | 14904.8 | 12990 | -4642.30 | 14893.7 | 13200 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 7085.34 | 14896.37 | 13990.0 | 14191.36 | 14883.9 | 13404 | 14199.86 | 14904.8 | 13644 | 14265.10 | 15009.1 | 14400 |
| MEDIUM | 6118.18 | 14930.81 | 14271.4 | 12356.29 | 14883.9 | 13404 | 12331.93 | 14904.8 | 13644 | 12293.67 | 15009.1 | 14400 |
| HIGH | -3154.89 | 15000.28 | 14394.0 | -4954.02 | 14883.9 | 13404 | -5288.32 | 14904.8 | 13644 | -6302.90 | 15009.1 | 14400 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 7074.00 | 14889.12 | 14263.3 | 14204.28 | 14901.0 | 13485 | 14182.91 | 14904.8 | 13972 | 14378.60 | 15184.6 | 15600 |
| MEDIUM | 6084.58 | 15063.10 | 15301.0 | 12358.11 | 14901.0 | 13485 | 12270.08 | 14904.8 | 13972 | 12242.89 | 15184.6 | 15600 |
| HIGH | -3951.96 | 15180.57 | 15589.5 | -5056.80 | 14901.0 | 13485 | -5773.76 | 14904.8 | 13972 | -7903.40 | 15184.6 | 15600 |

**Table 4b: Results for 600 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 9229.13 | 19164.68 | 13754.5 | 18378.85 | 19068.6 | 13350 | 18390.90 | 19085.2 | 13438 | 18481.93 | 19209.4 | 14080 |
| MEDIUM | 8266.61 | 19170.86 | 13973.3 | 16551.17 | 19068.6 | 13350 | 16551.18 | 19085.2 | 13438 | 16554.31 | 19209.4 | 14080 |
| HIGH | -871.06 | 19090.58 | 14064.2 | -689.40 | 19068.6 | 13350 | -803.04 | 19085.2 | 13438 | -1629.00 | 19209.4 | 14080 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 9224.84 | 19201.35 | 14237.3 | 18507.37 | 19252.4 | 14420 | 18380.88 | 19085.2 | 13632 | 18688.00 | 19481.6 | 15360 |
| MEDIUM | 8179.01 | 19182.06 | 14882.3 | 16533.20 | 19252.4 | 14420 | 16514.59 | 19085.2 | 13632 | 16585.14 | 19481.6 | 15360 |
| HIGH | -1720.46 | 19287.57 | 15350.7 | -2089.20 | 19252.4 | 14420 | -1090.16 | 19085.2 | 13632 | -3251.20 | 19481.6 | 15360 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 9128.56 | 19037.40 | 14792.0 | 18366.15 | 19072.9 | 13679 | 18376.44 | 19085.2 | 13718 | 18777.77 | 19637.5 | 16640 |
| MEDIUM | 8139.36 | 19250.83 | 15655.6 | 16493.43 | 19072.9 | 13679 | 16498.38 | 19085.2 | 13718 | 16499.67 | 19637.5 | 16640 |
| HIGH | -2516.99 | 19585.90 | 16620.8 | -1172.02 | 19072.9 | 13679 | -1217.44 | 19085.2 | 13718 | -4989.70 | 19637.5 | 16640 |

**Table 4c: Results for 640 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 10518.89 | 21842.58 | 15459.5 | 21129.30 | 21923.0 | 15362 | 21143.93 | 21927.4 | 15164 | 21301.20 | 22119.6 | 15840 |
| MEDIUM | 9470.18 | 21920.01 | 15793.0 | 19026.17 | 21923.0 | 15362 | 19067.90 | 21927.4 | 15164 | 19132.63 | 22119.6 | 15840 |
| HIGH | -718.57 | 22004.92 | 15836.5 | -812.76 | 21923.0 | 15362 | -515.32 | 21927.4 | 15164 | -1323.60 | 22119.6 | 15840 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 10546.18 | 21931.32 | 16146.0 | 21125.58 | 21935.3 | 15672 | 21122.02 | 21927.4 | 15588 | 21350.70 | 22243.5 | 17280 |
| MEDIUM | 9437.77 | 22047.74 | 16770.9 | 18980.01 | 21935.3 | 15672 | 18987.95 | 21927.4 | 15588 | 18984.99 | 22243.5 | 17280 |
| HIGH | -1688.28 | 22193.68 | 17268.0 | -1259.26 | 21935.3 | 15672 | -1142.84 | 21927.4 | 15588 | -3330.90 | 22243.5 | 17280 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 10553.22 | 21969.87 | 16403.3 | 21029.16 | 21852.0 | 15926 | 21103.63 | 21927.4 | 15944 | 21580.10 | 22547.3 | 18720 |
| MEDIUM | 9394.59 | 22166.44 | 17884.7 | 18848.81 | 21852.0 | 15926 | 18920.82 | 21927.4 | 15944 | 19017.24 | 22547.3 | 18720 |
| HIGH | -2611.71 | 22457.45 | 18659.7 | -1718.48 | 21852.0 | 15926 | -1669.72 | 21927.4 | 15944 | -5158.30 | 22547.3 | 18720 |

**Table 4d: Results for 720 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 8146.56 | 17148.37 | 16433.7 | 16324.51 | 17150.4 | 15985 | 16289.70 | 17129.8 | 16260 | 16378.93 | 17242.8 | 16720 |
| MEDIUM | 7024.06 | 17197.01 | 16690.6 | 14136.09 | 17150.4 | 15985 | 14063.63 | 17129.8 | 16260 | 14089.89 | 17242.8 | 16720 |
| HIGH | -3750.88 | 17243.23 | 16718.7 | -6507.40 | 17150.4 | 15985 | -6935.00 | 17129.8 | 16260 | -7502.80 | 17242.8 | 16720 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 8131.23 | 17152.83 | 17038.0 | 16270.84 | 17125.2 | 16536 | 16261.80 | 17129.8 | 16800 | 16519.20 | 17461.6 | 18240 |
| MEDIUM | 6938.01 | 17298.90 | 18128.6 | 14006.98 | 17125.2 | 16536 | 13961.80 | 17129.8 | 16800 | 14022.06 | 17461.6 | 18240 |
| HIGH | -4793.72 | 17407.75 | 18240.0 | -7348.08 | 17125.2 | 16536 | -7734.20 | 17129.8 | 16800 | -9533.60 | 17461.6 | 18240 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 8129.41 | 17183.35 | 17556.8 | 16232.89 | 17113.6 | 17046 | 16244.91 | 17129.8 | 17127 | 16689.87 | 17710.8 | 19760 |
| MEDIUM | 6882.14 | 17402.47 | 19227.1 | 13899.21 | 17113.6 | 17046 | 13900.14 | 17129.8 | 17127 | 13984.63 | 17710.8 | 19760 |
| HIGH | -5785.01 | 17672.25 | 19753.3 | -8114.48 | 17113.6 | 17046 | -8218.16 | 17129.8 | 17127 | -11534.00 | 17710.8 | 19760 |

**Table 4e: Results for 760 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 11642.06 | 24187.15 | 17462.5 | 23131.42 | 24003.4 | 16877 | 23295.02 | 24177.8 | 17086 | 23308.57 | 24217.9 | 17600 |
| MEDIUM | 10436.99 | 24188.60 | 17576.7 | 20820.88 | 24003.4 | 16877 | 20955.87 | 24177.8 | 17086 | 20899.04 | 24217.9 | 17600 |
| HIGH | -901.90 | 24243.27 | 17598.3 | -974.56 | 24003.4 | 16877 | -1109.48 | 24177.8 | 17086 | -1830.10 | 24217.9 | 17600 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 11641.84 | 24223.50 | 18057.3 | 23125.07 | 24003.4 | 17000 | 23272.60 | 24177.8 | 17520 | 23518.10 | 24510.1 | 19200 |
| MEDIUM | 10402.31 | 24400.36 | 19021.7 | 20797.69 | 24003.4 | 17000 | 20874.03 | 24177.8 | 17520 | 20889.53 | 24510.1 | 19200 |
| HIGH | -1978.11 | 24456.87 | 19192.7 | -1156.60 | 24003.4 | 17000 | -1751.80 | 24177.8 | 17520 | -3905.90 | 24510.1 | 19200 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 11630.81 | 24227.78 | 18416.2 | 23121.97 | 24000.3 | 17000 | 23263.04 | 24177.8 | 17705 | 23781.03 | 24855.7 | 20800 |
| MEDIUM | 10337.57 | 24467.50 | 20047.0 | 20794.59 | 24000.3 | 17000 | 20839.14 | 24177.8 | 17705 | 20933.41 | 24855.7 | 20800 |
| HIGH | -3004.35 | 24731.70 | 20677.5 | -1159.70 | 24000.3 | 17000 | -2025.60 | 24177.8 | 17705 | -5928.30 | 24855.7 | 20800 |

**Table 4f: Results for 800 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 13162.38 | 27316.58 | 19204.0 | 26329.64 | 27284.7 | 18485 | 26321.31 | 27286.6 | 18683 | 26742.53 | 27742.8 | 19360 |
| MEDIUM | 11818.23 | 27571.80 | 20823.4 | 23798.96 | 27284.7 | 18485 | 23763.52 | 27286.6 | 18683 | 24092.06 | 27742.8 | 19360 |
| HIGH | -1798.17 | 27654.30 | 21100.0 | -73.10 | 27284.7 | 18485 | -364.24 | 27286.6 | 18683 | -910.00 | 27742.8 | 19360 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 13162.38 | 27316.58 | 19204.0 | 26329.64 | 27284.7 | 18485 | 26321.31 | 27286.6 | 18683 | 26742.53 | 27742.8 | 19360 |
| MEDIUM | 11818.23 | 27571.80 | 20823.4 | 23798.96 | 27284.7 | 18485 | 23763.52 | 27286.6 | 18683 | 24092.06 | 27742.8 | 19360 |
| HIGH | -1798.17 | 27654.30 | 21100.0 | -73.10 | 27284.7 | 18485 | -364.24 | 27286.6 | 18683 | -910.00 | 27742.8 | 19360 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 13180.85 | 27425.33 | 20197.7 | 26296.28 | 27284.4 | 19125 | 26286.39 | 27286.6 | 19359 | 26895.87 | 28078.0 | 22880 |
| MEDIUM | 11750.78 | 27617.51 | 21752.3 | 23677.97 | 27284.4 | 19125 | 23636.05 | 27286.6 | 19359 | 23763.49 | 28078.0 | 22880 |
| HIGH | -2908.17 | 27953.35 | 22603.8 | -1020.60 | 27284.4 | 19125 | -1364.72 | 27286.6 | 19359 | -5784.40 | 28078.0 | 22880 |

**Table 4g: Results for 880 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 14145.25 | 29365.23 | 20638.2 | 28312.64 | 29363.9 | 20347 | 28296.63 | 29354.5 | 20475 | 28478.50 | 29569.7 | 21120 |
| MEDIUM | 12736.67 | 29437.23 | 20996.6 | 25527.04 | 29363.9 | 20347 | 25493.50 | 29354.5 | 20475 | 25587.07 | 29569.7 | 21120 |
| HIGH | -851.35 | 29554.53 | 21119.0 | -749.66 | 29363.9 | 20347 | -948.50 | 29354.5 | 20475 | -1687.90 | 29569.7 | 21120 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 14129.68 | 29366.70 | 21092.7 | 28300.60 | 29363.9 | 20580 | 28285.00 | 29354.5 | 20700 | 28611.20 | 29801.6 | 23040 |
| MEDIUM | 12681.76 | 29652.59 | 22692.6 | 25483.10 | 29363.9 | 20580 | 25451.07 | 29354.5 | 20700 | 25456.91 | 29801.6 | 23040 |
| HIGH | -2155.34 | 29780.55 | 23018.7 | -1094.50 | 29363.9 | 20580 | -1281.50 | 29354.5 | 20700 | -4297.60 | 29801.6 | 23040 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 14139.45 | 29402.15 | 21529.8 | 28152.07 | 29185.4 | 20000 | 28273.38 | 29354.5 | 20925 | 28931.90 | 30221.5 | 24960 |
| MEDIUM | 12613.46 | 29714.99 | 23765.1 | 25413.97 | 29185.4 | 20000 | 25408.64 | 29354.5 | 20925 | 25514.76 | 30221.5 | 24960 |
| HIGH | -3389.89 | 30080.68 | 24732.7 | -414.60 | 29185.4 | 20000 | -1614.50 | 29354.5 | 20925 | -6719.30 | 30221.5 | 24960 |

**Table 4h: Results for 960 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 15450.76 | 32058.18 | 22177.7 | 31003.25 | 32133.3 | 21872 | 30835.24 | 31947.0 | 21518 | 31099.17 | 32281.3 | 22880 |
| MEDIUM | 13947.74 | 32190.59 | 22756.9 | 28008.87 | 32133.3 | 21872 | 27889.32 | 31947.0 | 21518 | 27966.79 | 32281.3 | 22880 |
| HIGH | -811.74 | 32236.30 | 22872.7 | -237.26 | 32133.3 | 21872 | 100.36 | 31947.0 | 21518 | -1581.10 | 32281.3 | 22880 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 15439.35 | 32064.08 | 22642.0 | 30850.40 | 31969.5 | 21660 | 30822.94 | 31947.0 | 21756 | 31646.90 | 32936.5 | 24960 |
| MEDIUM | 13849.95 | 32291.41 | 24264.0 | 27885.04 | 31969.5 | 21660 | 27844.44 | 31947.0 | 21756 | 28229.76 | 32936.5 | 24960 |
| HIGH | -2156.39 | 32590.95 | 24858.0 | -87.30 | 31969.5 | 21660 | -251.88 | 31947.0 | 21756 | -4004.30 | 32936.5 | 24960 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 15425.18 | 32067.82 | 23123.7 | 30958.20 | 32143.9 | 22949 | 30810.64 | 31947.0 | 21994 | 32189.43 | 33586.5 | 27040 |
| MEDIUM | 13784.19 | 32284.13 | 24963.6 | 27816.37 | 32143.9 | 22949 | 27799.56 | 31947.0 | 21994 | 28487.53 | 33586.5 | 27040 |
| HIGH | -3440.51 | 33001.78 | 26665.3 | -1820.62 | 32143.9 | 22949 | -604.12 | 31947.0 | 21994 | -6432.70 | 33586.5 | 27040 |

**Table 4i: Results for 1040 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 17083.79 | 35413.03 | 24011.2 | 34124.29 | 35347.7 | 23679 | 34116.60 | 35336.6 | 23613 | 34341.63 | 35614.7 | 24640 |
| MEDIUM | 15431.47 | 35484.44 | 24480.9 | 30882.52 | 35347.7 | 23679 | 30883.86 | 35336.6 | 23613 | 30968.30 | 35614.7 | 24640 |
| HIGH | -435.00 | 35594.80 | 24633.3 | 302.78 | 35347.7 | 23679 | 389.36 | 35336.6 | 23613 | -852.50 | 35614.7 | 24640 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 17079.42 | 35456.02 | 24760.7 | 34126.31 | 35356.8 | 23816 | 34096.60 | 35336.6 | 24000 | 34591.80 | 35980.6 | 26880 |
| MEDIUM | 15387.86 | 35770.21 | 26400.0 | 30865.78 | 35356.8 | 23816 | 30810.89 | 35336.6 | 24000 | 30911.80 | 35980.6 | 26880 |
| HIGH | -1918.21 | 35934.65 | 26846.7 | 109.12 | 35356.8 | 23816 | -183.40 | 35336.6 | 24000 | -3801.80 | 35980.6 | 26880 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 17078.62 | 35486.55 | 25189.5 | 34102.24 | 35356.4 | 24274 | 34082.39 | 35336.6 | 24275 | 34874.37 | 36378.9 | 29120 |
| MEDIUM | 15295.47 | 35926.04 | 28179.0 | 30779.02 | 35356.4 | 24274 | 30759.03 | 35336.6 | 24275 | 30887.70 | 36378.9 | 29120 |
| HIGH | -3429.68 | 36219.18 | 29072.5 | -569.12 | 35356.4 | 24274 | -590.40 | 35336.6 | 24275 | -6718.70 | 36378.9 | 29120 |

**Table 4j: Results for 1120 node graphs.**

| | RTRDR | | | VRP L | | | VRP M | | | VRP U | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** | **Objective** | **Distance** | **Load** |
| **α= 0.1** | | | | | | | | | | | | |
| LOW | 18009.87 | 37380.83 | 26348.8 | 36071.47 | 37427.0 | 26236 | 36172.47 | 37531.1 | 26296 | 36088.90 | 37452.9 | 26400 |
| MEDIUM | 16215.04 | 37407.36 | 26392.7 | 32479.64 | 37427.0 | 26236 | 32572.43 | 37531.1 | 26296 | 32474.61 | 37452.9 | 26400 |
| HIGH | -822.38 | 37426.83 | 26398.7 | -1402.28 | 37427.0 | 26236 | -1386.98 | 37531.1 | 26296 | -1619.10 | 37452.9 | 26400 |
| **α= 0.2** | | | | | | | | | | | | |
| LOW | 17995.80 | 37423.30 | 27548.7 | 36002.63 | 37409.2 | 27224 | 36113.57 | 37531.1 | 27436 | 36365.70 | 37853.7 | 28800 |
| MEDIUM | 16088.53 | 37537.57 | 28398.9 | 32275.53 | 37409.2 | 27224 | 32357.45 | 37531.1 | 27436 | 32422.84 | 37853.7 | 28800 |
| HIGH | -2439.98 | 37723.42 | 28739.3 | -2882.32 | 37409.2 | 27224 | -3074.18 | 37531.1 | 27436 | -4770.30 | 37853.7 | 28800 |
| **α= 0.3** | | | | | | | | | | | | |
| LOW | 17974.54 | 37425.57 | 28406.0 | 35962.42 | 37426.5 | 28337 | 36070.79 | 37531.1 | 28264 | 36763.30 | 38375.3 | 31200 |
| MEDIUM | 16027.79 | 37743.10 | 30049.0 | 32082.95 | 37426.5 | 28337 | 32201.32 | 37531.1 | 28264 | 32491.87 | 38375.3 | 31200 |
| HIGH | -4014.13 | 38100.77 | 31073.0 | -4512.26 | 37426.5 | 28337 | -4299.62 | 37531.1 | 28264 | -7800.70 | 38375.3 | 31200 |

**Table 4k: Results for 1200 node graphs.**

Before we compare the results of the algorithms it would be useful to analyze the data below in order to intuit how the algorithm works in relation with the drop off amounts. Tables 5a and 5b show the relationship between vehicles and truck filling efficiency. The second column is the average number of vehicles used; it can also be interpreted as the average number of routes in the solution for a given number of nodes. The next column (% filled) is = Total drop off / (truck capacity x vehicles used). In words, it is the average percentage of space utilized in the vehicles. The higher the percentage, the lesser the wastage of space. If the percentage is very low, then there is a possibility that the number of routes could be reduced, as it indicates trucks have a lot of slack.

| | Average Number of Vehicles | %Filled | | | Average Number of Vehicles | %Filled |
|---|---|---|---|---|---|---|
| **560** | | | | **720** | | |
| **α= 0.1** | | | | **α= 0.1** | | |
| LOW | 10.7 | 0.95 | | LOW | 10.8 | 0.95 |
| MEDIUM | 10.9 | 0.94 | | MEDIUM | 11.1 | 0.95 |
| HIGH | 11 | 0.93 | | HIGH | 11 | 0.96 |
| **α= 0.2** | | | | **α= 0.2** | | |
| LOW | 11 | 0.98 | | LOW | 11 | 0.98 |
| MEDIUM | 11.2 | 0.98 | | MEDIUM | 11.4 | 0.98 |
| HIGH | 11.8 | 0.94 | | HIGH | 12 | 0.96 |
| **α= 0.3** | | | | **α= 0.3** | | |
| LOW | 11.2 | 0.98 | | LOW | 11.2 | 0.98 |
| MEDIUM | 11.9 | 0.99 | | MEDIUM | 12 | 0.99 |
| HIGH | 12.7 | 0.95 | | HIGH | 13 | 0.96 |
| **600** | | | | **760** | | |
| **α= 0.1** | | | | **α= 0.1** | | |
| LOW | 16 | 0.91 | | LOW | 20 | 0.91 |
| MEDIUM | 16 | 0.92 | | MEDIUM | 20.1 | 0.92 |
| HIGH | 16 | 0.92 | | HIGH | 20.5 | 0.91 |
| **α= 0.2** | | | | **α= 0.2** | | |
| LOW | 16 | 0.97 | | LOW | 20.2 | 0.94 |
| MEDIUM | 16.3 | 0.97 | | MEDIUM | 21.1 | 0.95 |
| HIGH | 16.8 | 0.95 | | HIGH | 21.7 | 0.94 |
| **α= 0.3** | | | | **α= 0.3** | | |
| LOW | 16 | 0.99 | | LOW | 20.5 | 0.95 |
| MEDIUM | 17.4 | 0.98 | | MEDIUM | 22.3 | 0.96 |
| HIGH | 18 | 0.96 | | HIGH | 23.3 | 0.94 |
| **640** | | | | **800** | | |
| **α= 0.1** | | | | **α= 0.1** | | |
| LOW | 10.3 | 0.95 | | LOW | 11 | 0.93 |
| MEDIUM | 10.4 | 0.96 | | MEDIUM | 11 | 0.94 |
| HIGH | 10.8 | 0.93 | | HIGH | 11.2 | 0.93 |
| **α= 0.2** | | | | **α= 0.2** | | |
| LOW | 10.3 | 0.98 | | LOW | 11.2 | 0.95 |
| MEDIUM | 10.7 | 0.99 | | MEDIUM | 11.9 | 0.94 |
| HIGH | 11.7 | 0.94 | | HIGH | 12 | 0.94 |
| **α= 0.3** | | | | **α= 0.3** | | |
| LOW | 10.7 | 0.99 | | LOW | 11.2 | 0.97 |
| MEDIUM | 11.3 | 0.99 | | MEDIUM | 12 | 0.98 |
| HIGH | 12 | 0.99 | | HIGH | 12.8 | 0.95 |

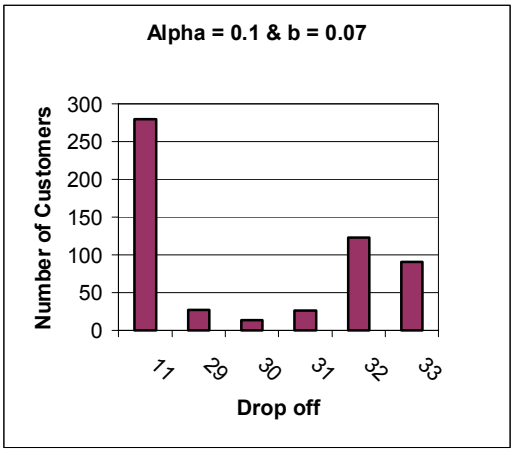
**Table 5a: Average number of vehicles and vehicle filling efficiency for problem sizes ranging from 560-800 nodes (results of the RTRDR algorithm).**

| | Average Number of Vehicles | %Filled | | | Average Number of Vehicles | %Filled |
|---|---|---|---|---|---|---|
| **880** | | | | **1120** | | |
| **α= 0.1** | | | | **α= 0.1** | | |
| LOW | 11.5 | 0.92 | | LOW | 11.3 | 0.92 |
| MEDIUM | 11.7 | 0.92 | | MEDIUM | 11.6 | 0.92 |
| HIGH | 12 | 0.9 | | HIGH | 12 | 0.89 |
| **α= 0.2** | | | | **α= 0.2** | | |
| LOW | 11.2 | 0.96 | | LOW | 11.5 | 0.94 |
| MEDIUM | 12 | 0.96 | | MEDIUM | 12.9 | 0.89 |
| HIGH | 12 | 0.98 | | HIGH | 13 | 0.9 |
| **α= 0.3** | | | | **α= 0.3** | | |
| LOW | 11.7 | 0.96 | | LOW | 11.7 | 0.94 |
| MEDIUM | 12.3 | 0.98 | | MEDIUM | 13 | 0.94 |
| HIGH | 12.8 | 0.98 | | HIGH | 13.2 | 0.96 |
| **960** | | | | **1200** | | |
| **α= 0.1** | | | | **α= 0.1** | | |
| LOW | 11 | 0.94 | | LOW | 12 | 0.88 |
| MEDIUM | 11.3 | 0.93 | | MEDIUM | 12 | 0.88 |
| HIGH | 11.8 | 0.89 | | HIGH | 12 | 0.88 |
| **α= 0.2** | | | | **α= 0.2** | | |
| LOW | 11 | 0.96 | | LOW | 12 | 0.92 |
| MEDIUM | 12 | 0.95 | | MEDIUM | 12.1 | 0.94 |
| HIGH | 12 | 0.96 | | HIGH | 12.5 | 0.92 |
| **α= 0.3** | | | | **α= 0.3** | | |
| LOW | 11.2 | 0.96 | | LOW | 12 | 0.95 |
| MEDIUM | 12 | 0.99 | | MEDIUM | 12.9 | 0.94 |
| HIGH | 12.8 | 0.96 | | HIGH | 13 | 0.96 |
| **1040** | | | | | | |
| **α= 0.1** | | | | | | |
| LOW | 11.5 | 0.92 | | | | |
| MEDIUM | 12 | 0.9 | | | | |
| HIGH | 12 | 0.91 | | | | |
| **α= 0.2** | | | | | | |
| LOW | 11.5 | 0.94 | | | | |
| MEDIUM | 12 | 0.96 | | | | |
| HIGH | 12.2 | 0.97 | | | | |
| **α= 0.3** | | | | | | |
| LOW | 11.5 | 0.96 | | | | |
| MEDIUM | 12 | 0.99 | | | | |
| HIGH | 13 | 0.98 | | | | |

**Table 5b: Average number of vehicles and vehicle filling efficiency for problem sizes ranging from 880-1200 nodes (results of the RTRDR algorithm).**

From tables 5a and 5b we can see that the % filled ranges from 91% to 100%. On average, trucks are 95% filled, the reason being that the feasibility constraint associated with the customer's upper demand is binding. More than 99% of the time the upper demands of the customers are met (rather than meeting the vehicle capacity ceiling). In other words, the slack of 5% was really a waste of space since most of the times the customer's upper demand was satisfied.

Figures 20a, 20b, and 20c show a typical distribution of drop off among the customers for lower b values. The b value of 0.07 was arbitrarily chosen. We see that customers with a demand of 10 always receive their upper demands, while there is more variability in customers with a demand of 30.



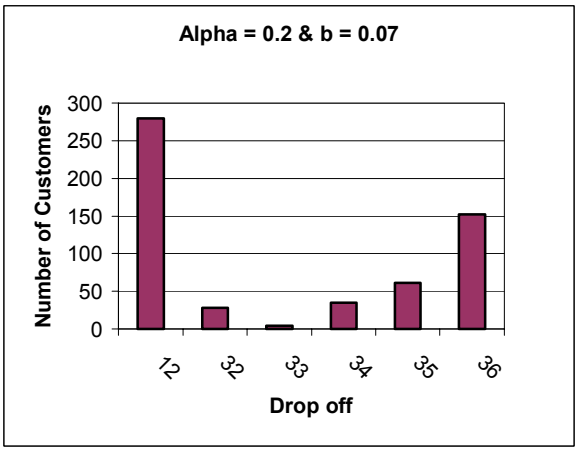




**Figure 20a: Drop off spread with α = 0.1**

**Figure 20b: Drop off spread with α = 0.2**

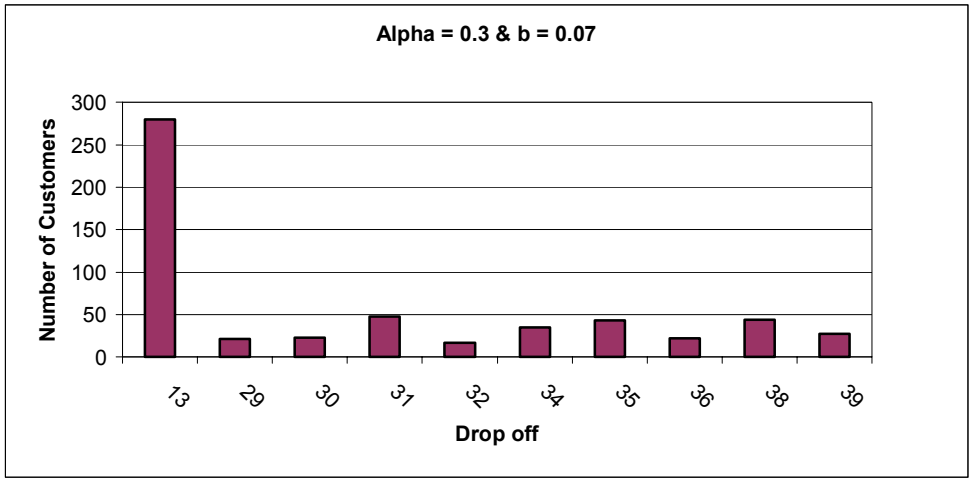
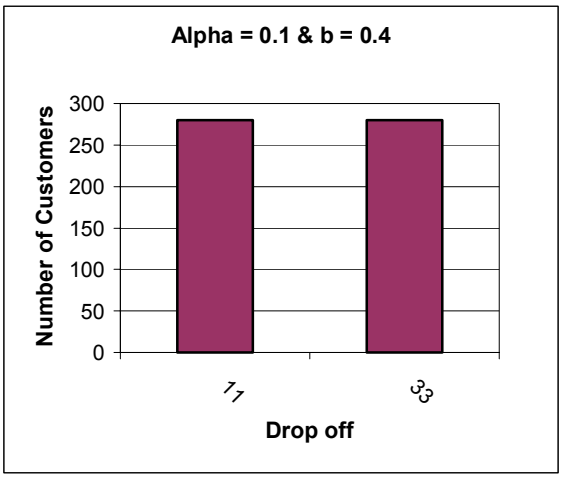



**Figure 20c: Drop off spread with α = 0.3**

However, the variability in the drop off distribution reduces as b increases. The reason being that as the reward increases it becomes more profitable to deliver more while compromising distance.

Figures 21a, 21b, and 21c show that when b is high, the distributor delivers only the upper demands. In this (high b) case, the drop off in the objective function has more weight over the distance; hence the algorithm will try to deliver the upper demands of the customers while compromising distance. Again, a high b value of 0.4 was chosen arbitrarily to demonstrate this effect.



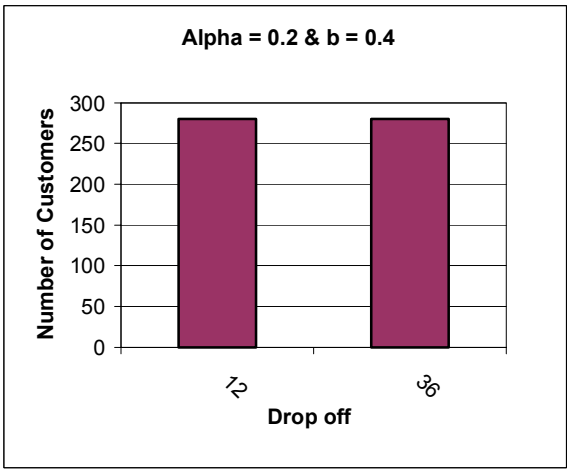




**Figure 21a: Drop off spread with α = 0.1**

**Figure 21b: Drop off spread with α = 0.2**

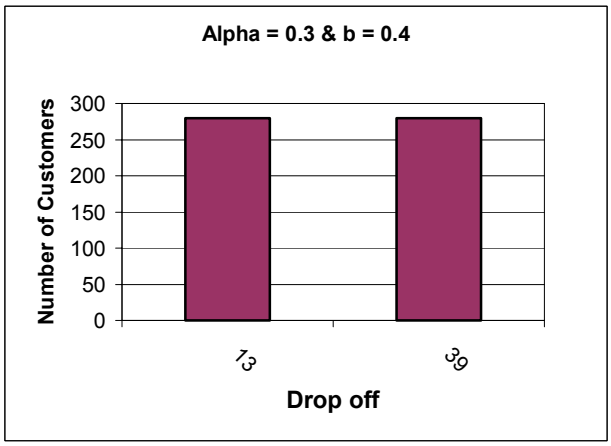



**Figure 21c: Drop off spread with α = 0.3**

Another interesting aspect of the results was the relationship between drop off amounts and the distances. The two variables play opposite roles in minimizing the objective function, meaning high drop off and low distances help in reducing the objective function. The graph below shows us the balance between the two variables that the RTRDR algorithm has found. The relationship between drop off and distance for the three α values, 0.1, 0.2, and 0.3, are presented in Figures 22, 23, and 24. These results have been averaged over all cases shown in Table 4a through 4k. The RTRDR result is further broken down into three b values of high, medium and low, while the comparison algorithm is broken into VRPU, VRPM, and VRPL.

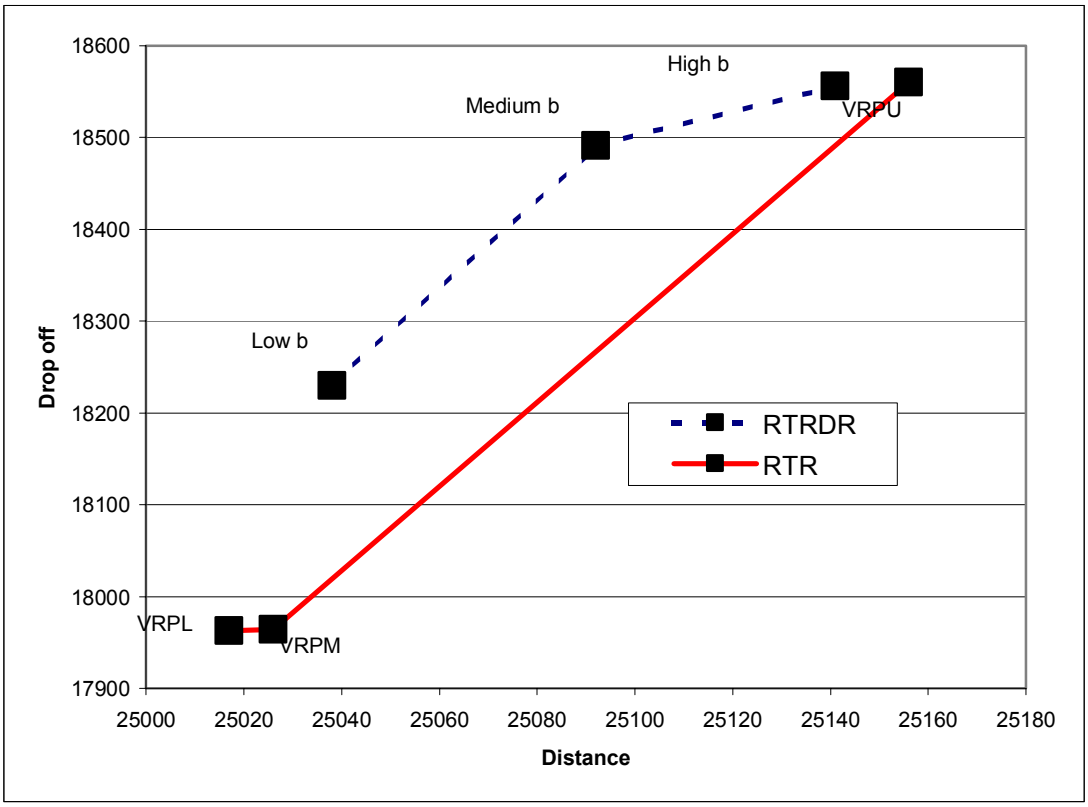



**Figure 22: Drop off vs. Distance with α = 0.1**

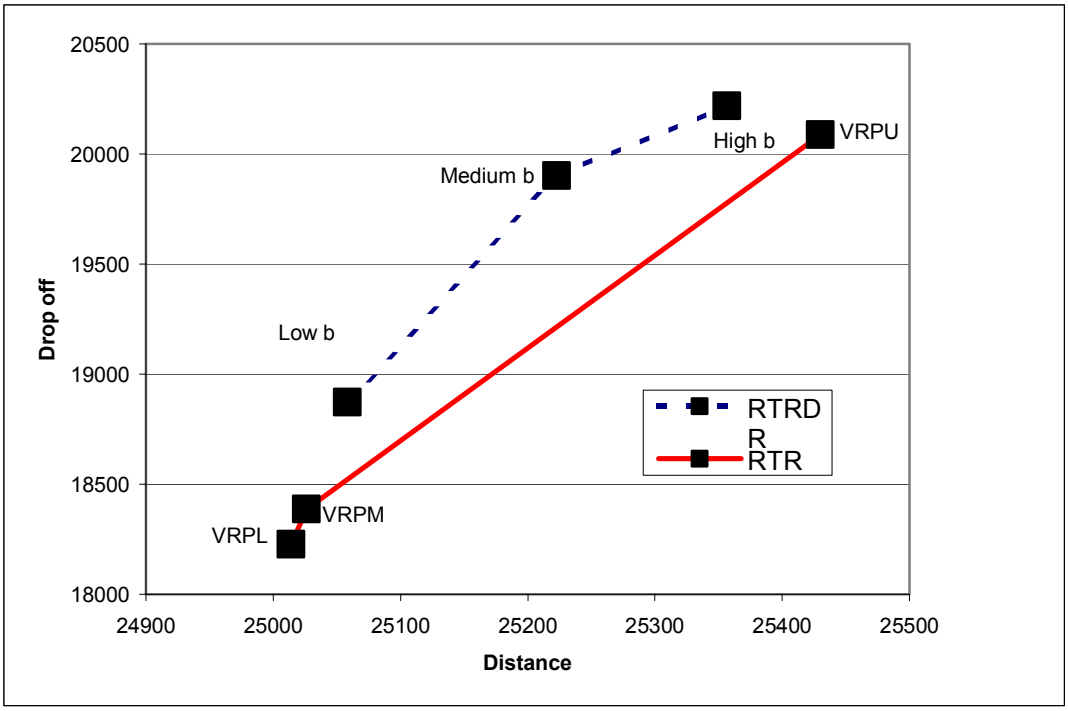



**Figure 23: Drop off vs. Distance with $\alpha = 0.2$**

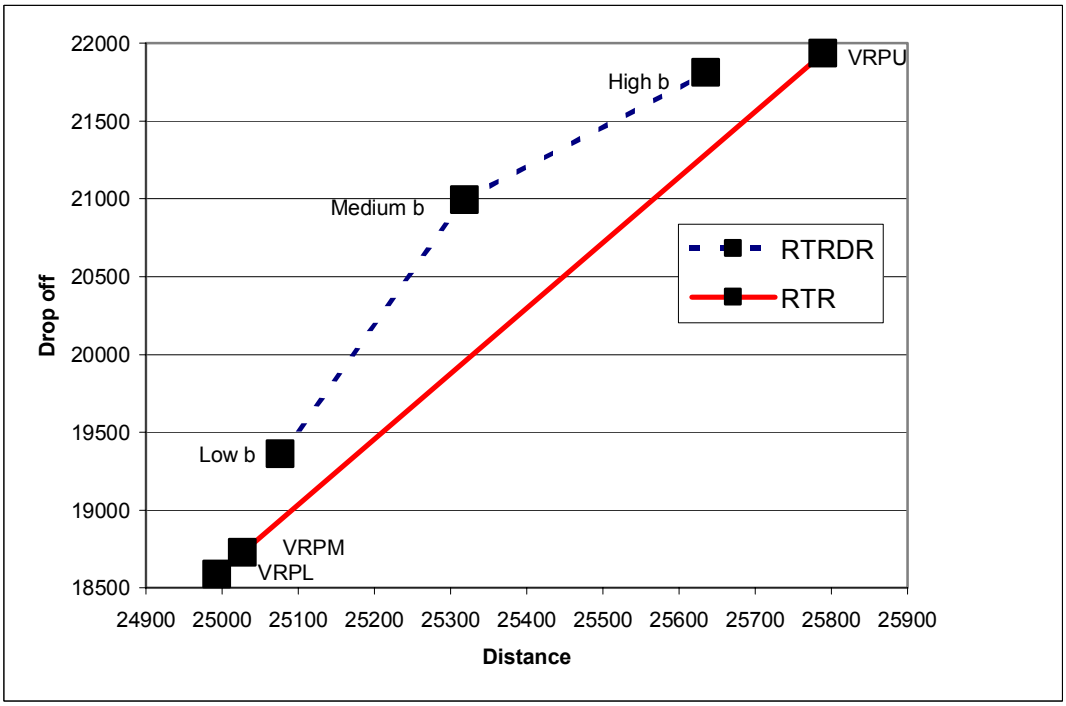



**Figure 24: Drop off vs. Distance with $\alpha = 0.3$**

All three graphs have similar shapes and tell a similar story. Our main purpose in the problem was to minimize the objective function. The most desirable position on the figure is the upper left hand corner of the graph, i.e., reducing distance and increasing drop off amounts (without violating the feasibility constraints). The comparison algorithm never crosses the RTRDR algorithm. Hence, RTRDR clearly outperforms the comparison algorithms in terms of balancing the two variables in the objective function.

It is evident from all the figures that RTRDR is achieving better results than the comparison algorithms. The RTRDR results form a concave function in the Drop off /Distance plane. In effect, distance is being traded for drop off amounts. We can see that the law of diminishing returns is in effect here; by sacrificing distance, we can increase drop off. However, this trade off, between distance and drop off, has to have a balance, meaning, if we compromise the distance to large extend to maximize drop off alone the objective function will not achieve good results.

The comparison algorithm approximately forms a line in Figures 22, 23, and 24. An interesting fact to note is that VRPU is so far from VRPL and VRPM (which perform about the same); the reason being that VRPU starts off with upper demands so it delivers the most possible from the very beginning of the algorithm. This makes it less flexible than VRPL and VRPM. VRPU generally gives higher drop off amounts than RTRDR but distance traveled is higher too. Tables 6 and 7 provide us with the averages of distance and drop off (averaged over different node sizes). Table 6 has been divided into three columns where each column

represents an α value and each row represents the grouped b values. For example, the first numbers in the table are 25038 and 18230. The two numbers represent distance and drop off respectively for α value of 10% and a low b value.

**RTRDR**

| Reward | 10% demand range | 20% demand range | 30% demand range |
|---|---|---|---|
| | Distance, Drop off | Distance, Drop off | Distance, Drop off |
| Low | 25038, 18230 | 25058, 18871 | 25076, 19361 |
| Med | 25092, 18491 | 25223, 19900 | 25318, 20992 |
| High | 25141, 18556 | 19216, 20218 | 25634, 21812 |

**Table 6: Average Distance and drop off for RTRDR.**

Table 7 is similar to Table 6 and provides us with average distance and drop off for the comparison algorithms: VRPL, VRPM, and VRPU.

**Comparison Algorithms**

| | | | |
|---|---|---|---|
| | Distance, Drop off | Distance, Drop off | Distance, Drop off |
| VRPL | 25017, 17963 | 25062, 17964 | 25156, 18560 |
| VRPM | 25014, 18224 | 25062, 18387 | 25429, 20087 |
| VRPU | 24993, 18589 | 25062, 18727 | 25789, 21935 |

**Table 7: Average Distance and drop off for comparison algorithm.**

Before presenting the results summary we will show the computation time for graphs with different numbers of nodes (customers) for the RTRDR algorithm. The following results are averaged over all α and b values. Figure 25 displays the relationship between time (in seconds) and node size. The computation time increases approximately quadratically.

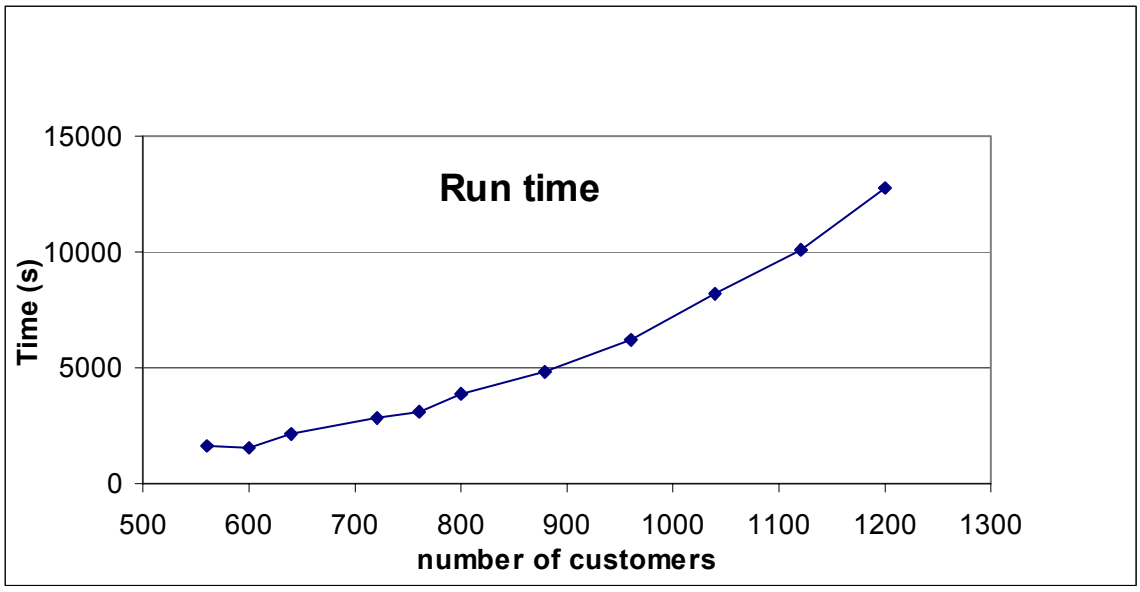



**Figure 25: Time to run the RTRDR algorithm versus number of customers**

The analysis is not complete without a total performance comparison between the algorithms. Below is a table that was grouped by the three b values and then averaged over number of nodes. The averages are over a single α value. Therefore the three columns represent the three α values 0.1, 0.2, and 0.3. Table 8 shows the percentage difference in objective functions of the three comparison algorithms with RTRDR. To give an example, the first number shows that the objective function of VRPL is 0.032% lower than RTRDR. For all numbers that are positive, the comparison algorithm is doing better than the RTRDR algorithm. We can see there are only two instances where the comparison does better than RTRDR, VRPL for low α, and demand range at 10% and 30%.

| Reward | 10% demand range | 20% demand range | 30% demand range |
|---|---|---|---|
| **Low** | **VRPL :** 0.032%<br>**VRPM :** -0.020 %<br>**VRPU :** -0.407 % | **VRPL :** -0.014 %<br>**VRPM :** -0.009 %<br>**VRPU :** -1.252 % | **VRPL :** 0.12 %<br>**VRPM :** -0.012 %<br>**VRPU :** -2.4 % |
| **Med** | **VRPL :** -0.107 %<br>**VRPM :** -0.156 %<br>**VRPU :** -0.204 % | **VRPL :** -0.628 %<br>**VRPM :** -0.497 %<br>**VRPU :** -0.785 % | **VRPL :** -0.752 %<br>**VRPM :** -0.75 %<br>**VRPU :** -1.3636 % |
| **High** | **VRPL :** -3.087 %<br>**VRPM :** -3.12 %<br>**VRPU :** 0.00 % | **VRPL :** -9.96 %<br>**VRPM :** -8.90 %<br>**VRPU :** -1.15 % | **VRPL :** -15.56 %<br>**VRPM :** -14.69 %<br>**VRPU :** -0.642 % |

**Table 8: Percentage difference in objective function between comparison & RTRDR.**

In general, RTRDR outperforms the comparison algorithms. Another important aspect of the analysis worth noting is that as the α values increase, especially at α=30%, the algorithm outperforms the comparison algorithm by larger margins. This brings us back to the point that the more the flexibility the distributor has in terms of delivery, the more options are available for searching the space to find a low objective function.

Another point to note is the relationship between the reward (b value) and the objective function. As the reward increases, difference in objective functions between RTRDR and the comparison algorithms increases drastically. In general we can conclude that RTRDR gives superior results.

# Chapter 7: Conclusions

VRPDR, a variant of CVRP, is a fairly new problem and has not been studied much in the past. We found optimal solutions for the problem in Chapter 5 for small graphs not exceeding 9 nodes. However, it is known that CVRP is NP hard and so we developed a heuristic to solve the problem. We adapted and modeled a version of the Record-to-record travel algorithm to fit our specific problem, motivated from Li et. al. [10]. Since the algorithm has not been studied in depth, we needed a benchmark with which to compare our results. We developed benchmark algorithms: VRPL, VRPM, and VRPU. The comparison algorithm was designed to mimic as closely as possible the CVRP Record-to-record travel algorithm that considered a fixed drop off amount. However, the drop off amount was not part of the optimization process and was an element considered separately after the optimization for distance was already performed. On the other hand, RTRDR considered distance and drop off a part of the same equation that needed to be optimized.

The results in Chapter 5 reveal the importance of distance and drop off as considered together. These two elements are critical components for the bottom line of any logistic organization and retailers that have a regular flow of merchandise. This problem is mainly targeting customers with regular demands who can accept a moderate amount of variation in merchandise inflow.

For further research, we could look at combining other variations on CVRP with a demand range. For example, time windows, split delivery, or multiple depots. There is also the possibility of solving the VRPDR problem with

other algorithms, which have been proven effective for CVRP such as genetic algorithms, simulated annealing, or tabu search.

**Bibliography**

[1] A. Campbell, "The Vehicle Routing Problem with Demand Range", *Annals of Operations Research*, Vol. 144, pp. 99-110 (2006)

[2] S. Chen, B. Golden, and E. Wasil, "The Split Delivery Vehicle Routing Problem: Applications, Algorithms, Test Problems, and Computational Results", *Networks*, Vol. 49, Issue 4, pp. 318–329 (2007)

[3] G. Clarke and J. Wright, "Scheduling of Vehicles From a Central Depot to a Number of Delivery Points", *Operations Research*, Vol. 12, pp. 568–81 (1964)

[4] E. Coffman, Jr., M. Garey, and D. Johnson, *Bin Packing Approximation Algorithms: A Survey in Approximation Algorithms for NP-Hard Problems* (edited by D. Hochbaum), PWS Publishing Co., pp. 46–93 (1996)

[5] J. Cordeau, M. Gendreau, A. Hertz, G. Laporte, and J. Sormany, "New Heuristics for the Vehicle Routing Problem", *Logistic System: Design and Optimization* (edited by A. Langevin and D. Riopel), Springer, pp. 207–297 (2005)

[6] G. Dantzig, and J. Ramser, "The Truck Dispatching Problem", *Management Science*, Vol. 6, pp. 80–91 (1959)

[7] G. Dueck, "New Optimization Heuristics", *Journal of Computational Physics*, Vol. 104, pp. 86-92 (1993)

[8] M. Gendreau, "Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms", *Transportation Science*, Vol. 39, Issue 1, pp. 104–118 (2005)

[9] T. Hankins, *Sir William Rowan Hamilton*, The Johns Hopkins University Press (1980)

[10] F. Li, B. Golden, and E. Wasil, "Very Large Scale Vehicle Routing: New Test Problems, Algorithms, and Results", *Computers & Operations Research*, Vol. 33, Issue 5, pp. 1165–1179 (2005)

[11] F. Li, B. Golden, and E. Wasil, "The Open Vehicle Routing Problem: Algorithms, Large-Scale Test Problems, and Computational Results", *Computers & Operations Research*, Vol. 33, Issue 10, pp. 2918–2930 (2007)

[12] P. Toth and D. Vigo, "The Vehicle Routing Problem", *SIAM*, pp. 195–224 (2001)

[13] P. Yellow, “A Computational Modification to the Savings Method of Vehicle Scheduling”, *Operations Research Quaterly*, Vol. 21, pp. 281–283 (1970)