

https://dspace.org/

The assignment routing problem with nominated delivery days (ARPNDD): definition and solution heuristics

Rademeyer, Angela L

2008-11-11T07:20:04Z
http://hdl.handle.net/10539/5832

Downloaded from a DSpace repository. https://dspace.org/

Dissertation for the degree of Master of Science
University of the Witwatersrand,
Johannesburg

The assignment routing problem with nominated delivery days (ARPNDD): Definition and solution heuristics

Angela L. Rademeyer

May 2008

Supervisor: Prof. D. Lubinsky

Declaration

I declare that this dissertation is my own unaided work. It is being submitted for the degree of Master of Science in the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other University.

A.L. Rademeyer

day of May 2008

Abstract

The Assignment Routing Problem with Nominated Delivery Days (ARP-NDD) is a newly defined, complex optimization problem in the area of logistics and operations research. The objective of the problem is to reduce the distribution costs and increase the optimization of a company's fleet, given a set of customers who receive deliveries a specified number of times in a delivery period. The customers must be clustered into routes and delivery days assigned if necessary, to smooth the workload. The restriction that each customer must remain in the same delivery group (cluster) for the entire period must be obeyed. An effective and efficient heuristic is developed which produces very satisfactory results on large problems. Detailed routing, with additional constraints, is done using scheduling software as a post-process.

In memory of my

Grandmother

Acknowledgements

I would like to thank David Lubinsky, my supervisor and boss, for his tremendous assistance throughout the compilation of this dissertation. He made valuable suggestions for using clustering and graphic techniques in the design of the heuristic developed in *R*. In addition, he provided me with an opportunity to do my research while still gaining valuable work experience at *OPSI Systems*.

I also acknowledge *Clover Pty Ltd South Africa*, for allowing me to use their sales data and customer information in the study.

Contents

1 Introduction **1**
- 1.1 The problem . . . 1
- 1.2 Case study . . . 5
- 1.3 Data and assumptions . . . 7
- 1.4 Outputs . . . 9
- 1.5 Summary of the remainder of this dissertation . . . 10

2 Literature Review **12**
- 2.1 Scheduling theory . . . 17
- 2.2 Seminal Papers . . . 19
- 2.3 Fixed Routes . . . 28
- 2.4 The Periodic VRP . . . 30
- 2.5 Applications of the VRP . . . 44
  - 2.5.1 Logistics decisions . . . 44
  - 2.5.2 Deadline VRP . . . 45
  - 2.5.3 Probabilistic VRP . . . 47
- 2.6 Clustering Techniques . . . 48
- 2.7 Genetic Algorithms . . . 51
  - 2.7.1 Introduction . . . 51
  - 2.7.2 GA's in the literature . . . 52

3 Method **62**
- 3.1 Route-first Cluster-second heuristic . . . 63
  - 3.1.1 Other complexities . . . 65
- 3.2 Cluster-first Route-second heuristic . . . 65
- 3.3 Formulation using mixed integer programming . . . 68
- 3.4 Cluster based approach . . . 72
  - 3.4.1 Method details . . . 73

4 Results **81**
4.1 Comparison of the Route-first and Cluster-first heuristics . . . 82
4.2 Mathematical model formulation 84
4.2.1 Model runs 84
4.3 Cluster based approach 89
4.3.1 Alternative use of the cluster based approach 91
4.3.2 Test problem cases 91
4.3.3 Extensions 98

5 Conclusions **100**

Appendix **98**

Bibliography **117**

List of Figures

1.1 VRP tree 3

2.1 Clarke & Wright savings algorithm 22
2.2 Basing service days on geographic location 33
2.3 Solution approach for the PVRP-SC. 40

3.1 Grouping methods 66
3.2 Routing grid 67
3.3 Graduation method 68
3.4 Distance estimate 72
3.5 K-means clustering 75
3.6 Clustering algorithm inner loop 76
3.7 Clustering algorithm results 77

4.1 Plot 3.1 (a) 93
4.2 Plot 3.1 (b) 93
4.3 Plot 3.1 (c) 94
4.4 Plot 3.1 (d) 95
4.5 Plot 3.1 (e) 96
4.6 Plot 3.1 (f) 96

List of Tables

1.1 Service schedule menu 4

2.1 Optimization metaheuristics 14
2.6 Proximity measures 49

3.1 ARPNDD formulation 71

4.1 Customer-schedule type summary 81
4.2 Comparison of the route schedules for the two methods under consideration. 82
4.3 *LINGO's* model analysis for the ARPNDD models considered with 10 customers. 86
4.4 Comparison of solution times for 2 *LINGO* models 87
4.5 Comparison of routes for 2 *LINGO* models 89
4.6 Comparison of the weekly schedule statistics for the cluster-first heuristic and the R function. 90

Chapter 1

Introduction

1.1 The problem

Noronha and Sarma [NS91] describe scheduling as *'a rich area demanding the application of efficient methods to tackle the combinatorial explosion that results in real-world applications'*. The demands of consumers for scheduling products and consultation are ever increasing as businesses strive for more transparent and efficient methods of operation. With rising fuel prices and increased competition, managers need to harness the benefits of powerful mathematical algorithms to drive various areas of their organizations. There are increasing numbers of success stories where advances in optimization techniques, which leverage available computing power, have resulted in decision problems being solved better than by any means previously available.

With the growth in the variety of optimization techniques, has come a rise in the number of different routing problems which need solving. One such new problem is what will be referred to as the assignment routing problem with nominated delivery days (ARPNDD). This problem is a specific case of the period routing problem in which routes are created over an m-day period. For each customer, the service frequency over the period is given, but not necessarily which combination of days the customer will be serviced on. An additional feature of this problem is that customers must remain in the same delivery group[1] each day on which they are serviced. The output of the model includes assigning customers to delivery groups as well as service

[1] A delivery group is defined as a set of customers. Only customers in the same delivery group can be placed together on a route. Each trip will thus be made up of customers from a single delivery group; no mixing of delivery groups on a truck is permitted. This ensures that customer groups are fixed over the planning period and the same customers are serviced together even if their service frequencies differ.

day combinations in such a way that cost (fixed vehicle cost for each vehicle used, cost per kilometer traveled and staff cost per hour) is minimized. The parameters used in the model, as well as a mathematical representation of the objective function and constraints for the ARPNDD, are given in Section 3.3.

Most routing problems stem from the vehicle routing problem (VRP) which is a natural extension of the traveling salesman problem (TSP). In the VRP, customers are not simply being visited but an amount of product is being collected and/or delivered at each stop. So it is not only the visit sequence that must be optimized but also, an appropriate capacitated vehicle must be assigned to each route. Another useful extension to the VRP is the inclusion of delivery windows. The vehicle routing problem with time windows (VRPTW) involves routing a fleet of vehicles, with limited capacities and travel times, from a central depot to a set of geographically dispersed customers with known demands within specified time windows. The route cost of a vehicle is the total of the traveling time and distance, waiting time and service time taken to visit a set of customers.

The periodic vehicle routing problem (PVRP) involves designing a set of routes which minimizes costs for each day of a given T-day period. Each customer has a visit frequency for the period and can have at most one visit per day. The relationship between these different models is shown in Figure 1.1. Those authors and researchers who have managed to find heuristic solutions to small, simple cases of the assignment routing problem[2] or problems involving NDD's have done so by adapting 'traditional' optimization methods used to solve the VRPTW or the PVRP.

The concept of NDD's is usually used in the context of the PVRP where routes are planned for more than one day at a time. If a customer's service level agreement[3] stipulates only the number of times per week they will be visited but not which days then an assignment routing problem must be solved to determine the NDD's. If the model is not making any decisions about what days to service at least some of the customers (given their service level agreement), then the problem is merely one of building fixed daily

[2]The term 'allocation' and/or 'assignment' used in the context of routing problems has taken on a variety of meanings. Here it is used to mean the allocation of customers to a particular set of NDD's whereas in other contexts it may mean assigning a customer to a warehouse facility for example.

[3]Also known as a service pack, this is the initial delivery obligation of the supplier to the customer. It usually stipulates the number of times a customer will be serviced in the period, potential delivery days, approximate visit times, average volumes etc.

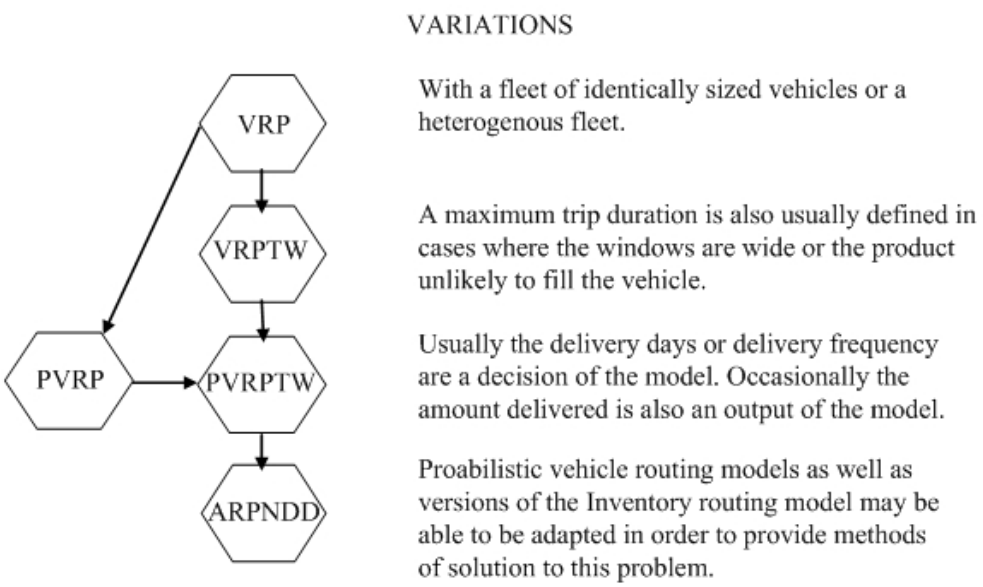


Figure 1.1: VRP tree showing how the ARPNDD model fits into the structure.

routes based on customers' NDD's, i.e. no assignment routing variables are required.

The additional requirement concerning delivery groups mentioned above, makes our problem much more complicated. If any customer in a particular delivery group receives a delivery on a particular day then all the other customers in that delivery group who are also serviced on that day must be on the same route. So, customers in a particular delivery group on one day have to remain in that delivery group for all other days on which they are serviced. This constraint means that if customer A is in a delivery group with customers B, C, D, E, F and G on Monday for example, then if customer A requires a delivery on Wednesday, it can only be grouped with those customers in its existing delivery group. Simply put, each customer is in a fixed delivery group for the duration of the delivery period (typically a week). This additional constraint makes the study and allocation of nominated delivery days to customers more restrictive. In addition, the designed set of routes has to allow for reasonably level fleet utilization over the week while ensuring that each customer's nominated delivery days do not violate any rules.

A study of this sort is usually based on rules for allocating delivery days to customers according to the number of drops specified in their service agreement. Table 1.1 shows an example of a menu of different possible service

Schedule	No. of drops per week	Days which can be scheduled
1	1	Any weekday
2	2	a) M, Th
		b) Tu, Th
		c) Tu, F
		d) W, F
3	2	a) Tu, Sa
		b) W, Sa
4	3	M, W, F
5	3	Tu, Th, Sa
6	4	M, Tu, Th, F
7	5	M, Tu, W, Th, F
8	6	M, Tu, W, Th, F, Sa

Table 1.1: Each customer must be allocated to one service schedule which dictates the customer's nominated delivery days (NDD's).

levels used in a case study (see Section 1.2 for more detail).

Generally, customers prefer a delivery early in the week (Monday or Tuesday) to replenish stock for the week and another later in the week (usually Friday) to stock up for weekend sales. The working week can thus be divided into two parts: Monday-Wednesday and Thursday-Friday. Saturdays are normally used to make-up lost time during the week, for re-deliveries or if volumes increase towards month-end. Many stores do not have staff available at their receiving bays on the weekend. Typical rules for delivery scheduling include:

Customers on a 2- or 3-day per week schedule should not receive deliveries on consecutive days (here Monday, Friday or Monday, Saturday may also be regarded as consecutive).

The 4-day per week customers do not get stock for three days running.

Also, the fewer delivery days there are for a customer, the more flexibility there is with regard to which days to service them.

There are a number of different variations of this problem which can be examined. Three factors differentiate the combinations of problem types:

1. Whether or not to apply the constraint described above: customers in

a particular delivery group on one day must remain in that delivery group for all other days on which they are serviced.

2. Whether to use the existing fleet of vehicles or to incorporate a fleet sizing exercise into the problem.

3. Whether or not customers with different visit frequencies can be placed together on a route. If they cannot, then the routes for each customer will be identical for each day on which they are serviced.

The application of the first constraint makes the problem very restrictive as routes on one day become dependent on other days. If the existing fleet configuration is assumed in the simulations, sub-optimal results may be produced. Changing a fleet though, is much more costly, time consuming and difficult than re-assigning customers to delivery groups.

It must be noted that a particular delivery group may have one route on some days and two or more on others, i.e. more than one vehicle servicing a delivery group on any particular day. This will occur if, for example, 3-day a week and 5-day a week customers are placed in the same delivery group. Also, the total number of delivery groups on any day cannot be greater than the number of vehicles available unless second loads on a vehicle for a different delivery group are permitted. If customers order loads larger than the largest vehicle then more than one truck would be used to service that customer's delivery group. It must be decided what the maximum number of vehicles that can be allowed to service a delivery group is. A suggestion is to look at historical job data and find into how many parts the largest order needed to be split in order to be delivered by the fleet. The maximum number of vehicles per delivery group per day can be limited to this amount.

1.2 Case study

The ARPNDD problem was motivated by a study done for *Clover Pty Ltd* [S.A06]. Clover is South Africa's largest dairy company and one of the leading manufacturers and marketers of food products in Southern Africa. The company collects *'some 30% of South Africa's milk and processes it in thirteen factories and distributes its range of well-known dairy and related products through twenty three national distribution depots and seven large agents'.* Clover realized that there was value in applying optimization techniques to their real-world routing and scheduling problems. The ARPNDD problem

would be applicable for any such company making cyclic deliveries with complex service levels.

Clover requested assistance in creating new routes because they had decided to integrate two of their depots. Previously, one depot only supplied fresh products while the other also delivered concentrated product to its database of customers. It was requested that a study be performed to help integrate the routes (and fleet) as certain customers, who had accounts at both depots, were being serviced twice on a single day (once for fresh and once for concentrated product) or more frequently than their service level agreement stipulated because their delivery days at the two depots were different. These customers could not simply be moved across to the depot which supplied both product types as the existing routes could not cope with the increased demand. New routes had to be designed to incorporate existing customers as well as those from the fresh-product-only depot. The existing routes were mainly suburb based (intra-city routes) and it seemed reasonable to assume that some of the new routes would contain a lot of similarities. This is because most routing algorithms group customers based, amongst other things, on their geographic location, which produces reasonable routes. Some inter-city routes, which had previously not been considered, would also probably arise.

Customers were currently assigned to days of the week but this requirement was often not met. This is one of the reasons why this study was needed as it was evident that certain days had too many customers and others too few and that when the service level agreement was drawn up for a new customer, existing volumes on each day were not examined. Delivery windows are very broad for most customers and it is a set of days which is offered to the customer for deliveries rather than a day and time.

The stores were grouped into delivery groups as a result of the simulations so that the multi-drop routes each week were fairly static (milk routes). Clover prefers to keep its operations as is with regard to keeping customers in the same delivery group each day even though this constraint is very restrictive. Reasons they give for doing so include: ease of picking and sorting, simpler invoicing and accounting practices. Driver familiarity of routes and also the relationships between customers and drivers are important. While Clover argues that this is better for their brand, others argue that drivers can become too familiar with 'other stops' on their routes or theft is more likely because of relationships with receiving checkers etc.

1.3 Data and assumptions

The following information is required to solve a routing and scheduling problem of this type:

A list of customers with the following details for each customer to whom they deliver:

Location	Physical address which is translated into longitude and latitude (i.e. a geocode)
Delivery window	Earliest time and latest time at which a truck can arrive and expect to be offloaded *Note:* These windows can also be set up to ensure a truck does not arrive at a customer when the loading bays are very congested.
Time at customer	Comprised of waiting (queue) time, offload time and time spent doing paperwork etc. A fixed time rather than a time/unit was given because of the heterogeneity of the orders and because waiting (queue) time is usually longer than offload time
Vehicle restriction	Largest vehicle which the customer can accept onto the premises
Priority constraint	If a customer needs to be first on route, last on route or alone on route (single trip)
Linked accounts	A supermarket may sell fresh milk but also order milk to use in their own bakery. Each of these two departments has a separate account and thus places separate orders. Here the orders were manually combined in the preprocessing phase to ensure that queue time was not doubled for such customers.
Service package	Number of times a customer is serviced per period. If certain delivery days are pre-assigned these must also be specified.

To obtain the size and nature of the fleet, the following information is required for each vehicle:

Payload	Measured in the same unit as the product orders
Running costs	Cost/km and cost/day
Start time	Earliest time the vehicle is available for departure

The problem is specified for the case where all vehicles operate from a single depot; they leave and return to this location after each route. Hire-in vehicles can be used if the dedicated fleet is unable to meet delivery requirements for the day. It is assumed that production is ready at the time that trucks are loaded and that dynamic[4] scheduling is used to ensure staggered departure times for the trucks so that loading bays are not congested. It is also assumed that all vehicles can travel on any roads and that they 'mass out' first, i.e. because of the density of the product a truck will never be overloaded volumetrically. Alternatively, the calculations can all be done on volume instead. No other loading constraints will be considered.

Driver and assistant costs per hour and also overtime costs must be supplied. A separate sleep-out cost is also used should such an event occur. There is no maximum or minimum on duty time allowed for for the individual drivers. This is controlled by the expected time on the road (length of workday). One driver and one assistant for each vehicle are assumed - if drivers are not present at work a relief driver is hired in. All drivers meet at the depot and would be told the night before what time to arrive for work the next day based on the route they would be driving. No breaks are explicitly allowed for and all drivers can do multiple trips (second loads) in a day and/or sleep out if necessary. The location of the depot and a reload time at the depot for second loads must also be provided.

The problem is based on the following assumptions:

- The company runs a pure delivery operation (no return loads).
- Customers must be served according to their (weekly) service level agreement.
- None of the customers are in an existing delivery group.
- Delivery sizes do not vary greatly from week to week for any particular customer on any particular day, so an average for each customer for each day on which that customer is served is used. (A brief description of the data compilation method used is explained below).

[4]Under dynamic scheduling, the departure time for each truck is calculated by subtracting the travel time to the first customer from that customer's opening time.

- No product level detail is used since all goods need to be transported in the same type of truck. (If the product is not heterogenous and ambient and refrigerated trucks or rigid and dropside trucks are needed for example, then more variables would be required.)
- Partial filling of orders is not allowed.

Note that delivery sizes for a customer with more than one job per week need not be the same. For a 2-day per week client in schedule type 2 for example, 40% of their total mass may typically be received on Tuesday or Wednesday with the remaining 60% being delivered on Friday. Also, two customers on schedule 2 for example may have different proportions of their total weekly mass delivered on each schedule day (i.e. the schedule type does not dictate that a certain proportion be delivered on specific days).

1.4 Outputs

Given the above macro-level information, the following output is typically provided:

- A partitioning of the customer list into delivery groups
- The delivery schedule for those customers whose nominated delivery days are determined by the model
- Expected total mass for each day of the week
- Number of drops for each day of the week
- Average vehicle space and time utilization per day
- Average time on the road per trip per day
- Average distance traveled per trip per day

An implementable and accurate result ensures that stores who do not wish to keep high inventories get the reliable deliveries that they demand. If customers have fixed delivery days then consolidated deliveries can be made the day after a direct order has been placed. Other questions that could also be answered from the results of an ARPNDD simulation are: is it necessary to acquire or deploy vehicles and what reasonable cost and

efficiency benchmarks can be used to assess the current delivery operations. Other spinoffs expected from the new routes derived include increased vehicle and time utilization through better routing.

1.5 Summary of the remainder of this dissertation

The rest of this dissertation is divided into four chapters which contain the following detail:

Literature Review
Discusses the routing problems that form a unifying framework for understanding the papers presented in the literature. This chapter also covers clustering techniques and genetic algorithms.

Method
Four different techniques are discussed here, namely:

1. Route-first cluster-second heuristic developed in conjunction with the use of commercial routing software. This method follows logical procedures but gave the least optimal results.

2. Cluster-first route-second heuristic. This method used a new idea of visual clustering on an uneven grid and was significantly better than the route-first method.

3. Using an optimization solver to solve the mathematically formulated description of the ARPNDD problem. This attempt failed due to the large size and complex nature of the problem.

4. Using a clustering based method. This method was much faster than the first two and produced results comparable to method 2.

Results
The results of each method, using the Clover data set, are examined, criticized and compared. Smaller test problems are used to assess the final method used. Extensions and improvements to this model are also discussed.

Conclusions
A robust method which runs in very good time and allows for a number of constraints to be applied was found (Method 4). This method enables the

user of the program to set parameters as well as treat customers in different sales segments with different levels of priority. The results are not only a set of fixed routes with customer delivery days but can also be used for determining the specifications of a fleet.

Chapter 2

Literature Review

Saul Gass, in his introduction to the book *Vehicle Routing: Methods and Studies* [AG88], points out that VRP's are solved operationally every day - the world's economies could not operate if it wasn't for the fact that goods are picked up and delivered daily with reasonable adherence to customer service level requirements. *'Competition and the desire to improve profits'* have led practitioners to make improvements by non-optimal, directed investigation *'into the mathematical and computational structures that describe VRPs'* [AG88]. The heuristics that they develop *'seek and find improved solutions that can be implemented'*. Problem specific constraints and complications though, have made *'theoretical analysis a far from trivial task'* [HS88]. Gass [AG88] suggests that we should not abandon hope of finding improved solution methods to the VRP and its variants; we know a solution exists because Santa Claus does it successfully every year!

It has been said of solution methods applicable to the VRP that *'the best methods combine population search and local search, thus providing at the same time breadth and depth in the solution space exploration. What is now needed is greater emphasis on simplicity and flexibility'* [CS05]. Luckily for industry progression and thanks to vast amounts of current research, *'the field of VRP heuristics is very active, as witnessed by the large number of recent articles'* [CS05]. As mentioned before, logistics problems are evolving into increasingly complicated problems to model as demands on planners and decision makers in business increase. New and versatile heuristics are being developed because *'in contrast to exact algorithms, heuristics are better suited to the solution of VRP variants involving side constraints such as time windows, pickups and deliveries, periodic visits, etc.'* [CS05]. At a macro-level, heuristics for the NP-hard VRP and its variants (for problem cases of a useful size) combine some of the following four components:

1. Construction of an initial solution
2. Improvement procedures
3. Population mechanisms
4. Learning mechanisms

What makes for an effective vehicle routing solution is not necessarily a mathematically complex algorithm, but rather one which captures as many real-world characteristics as possible as a result of careful modeling.

No papers currently available describe the constrained problem (ARP-NDD) where customers must be in the same delivery group each day. Two papers appeared in the 1970's describing assignment routing problems and only in the last few years have more papers, which consider nominated delivery days and scheduling over periods of more than a day, emerged. A number of texts describing the VRP are also discussed but work would be required to adapt them into formulations with service day allocations. Some of the newest optimization techniques, such as genetic algorithms, are being tried in problem instances which are variations of the VRP but no significant contributions have yet been made.

Table 2.1 below shows the four broad types of metaheuristics in common use as outlined in *Logistics Systems* [CS05]. A brief explanation of each as well as examples in the literature have been added.

Type	Description/comment	Examples
Constructive heuristics	These involve building a feasible solution from scratch. A greedy-like algorithm is an example. Constructive heuristics are usually followed by an improvement phase.	**Clarke and Wright savings concept** [RI79], [BB74] Sweep mechanism **Route-first cluster-second**[a] [BSL97], [BB74], [HS88], [JO88] **Cluster-first route-second**[b] [RI79], [BB74], [Bal88], [NS88], [Tha95]
Improvement heuristics	The heuristic attempts to improve its value by starting from a feasible solution. Often this involves moving to neighbour solutions. The most common post-optimization scheme for routing problems involves applying the TSP to each route.	**λ-Interchange mechanisms** [Tha95] **Edge exchange schemes** [RI79], [BC84] Ejection chains
Population mechanisms	Offspring are constructed from good parent routes by combining strong features.	**Genetic algorithms** [Tha95], [BP02] Memetic algorithms
Learning mechanisms	A learning feedback loop enables the process to restart with different rules and/or parameter settings.	Ant algorithms Neural networks

[a & b] Also known as tour partitioning and region partitioning heuristics respectively.

Table 2.1: A variety of heuristics have been applied to the VRP over the years. The TSP is often used to test the relative performance of new combinatorial optimization heuristics. If successful, these are usually adapted to other problems in the area of routing.

Those examples in bold type are discussed in this proposal. No relevant papers applying memetic algorithms or neural networks to the VRP have

been found.

The texts to be discussed have been broken down into the following sections:

Scheduling Theory

- Operations Research in Transportation Systems; Belenky, A.S. [Bel98]

Seminal Papers

- An assignment routing problem; Russell, R. and Igo, W. [RI79]
- Networks and vehicle routing for municipal waste collection; Bodin, L. and Beltrami, E. [BB74]
- The period routing problem; Beasley, J.E. and Christofides, N. [BC84]
- A heuristic algorithm for the period vehicle routing problem; Tan, C.R.R. and Beasley, J.E. [TB84]

Fixed Routes

- Fixed Routes; Beasley, J.E. [Bea84]

The Periodic VRP

- Allocation/Routing: Models and algorithms; Ball, M.O. [Bal88]
- The period vehicle routing problem with service choice; Francis, P., Smilowitz, K. and Tzur, M. [FT05]
- Modeling techniques for periodic vehicle routing problems; Francis, P. and Smilowitz, K. [FS06]
- Flexibility and complexity in periodic distribution problems; Francis, P., Smilowitz, K. and Tzur, M. [FT06]

Applications of the VRP

- The Logic of Logistics; Bramel, J. and Simchi-Levi, D. [BSL97]

- The network of logistics decisions; Langevin, A., Riopel, D. and Campbell, J.F. [LC05]
- New heuristics for the vehicle routing problem; Cordeau, J.F., Gendreau, M., Hertz, A., Laporte, G. and Sormany, J.S. [CS05]
- Operational research methods for efficient warehousing; Cormier, G. [Cor05]
- Generalized assignment methods for the deadline vehicle routing problem; Nygard, K.E., Greenberg, P., Bolkan, W.E. and Swenson, E.J. [NS88]
- The probabilistic vehicle routing problem; Jaillet, P. and Odoni, A.R. [JO88]

Clustering Techniques

- Design of multiple-vehicle delivery tours - I: A ring-radial network; Newell, G.F. and Daganzo, C.F. [ND86]
- Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane; Karp, R.M. [Kar77]
- Hierarchical Vehicle Routing Problems; Marchetti-Spaccamela, A., Rinnooy Kan, A.H.G. and Stougie, L. [MSS84]

Genetic Algorithms

- Coordinating the distribution chain: New models for new challenges; Balakrishnan, A., Geunes, J. and Pangburn, M.S. [BP02]
- Vehicle routing with time windows using genetic algorithms; Thangiah, S.R. [Tha95]
- Handbook of Genetic Algorithms; Davis, L. [Dav91]
- The Design of Innovation; Goldberg, D.E. [Gol02]
- Genetic Algorithms in Search, Optimization and Machine Learning; Goldberg, D.E. [Gol97]

2.1 Scheduling theory

To increase the activity and interest in mathematical modeling and the detection of new transport problems that may be formalized and researched is an ongoing and intensive process. In the course of this process, new optimization methods within the framework of known classes of problems, as well as new formulations of problems necessitating the use of different existing mathematical tools or working out the new ones, appear. Belenky [Bel98] believes that one of the directions that research will follow in the future *'is associated with analyzing transport as a large-scale system, functioning in the interconnection with other large-scale systems of the national economy complex and the environment'* [Bel98].

Belenky [Bel98] states that the degree of employing the tools for analysis and decision making in economic and technical systems in transportation still remains low. Available systems do not allow users to easily incorporate *'practical issues arising in strategic planning and operations management'*. This has resulted in a large number of people believing that *'decision making in practical situations does not require any serious mathematical analysis and modeling and should be exclusively based on practical experience of particular people in the field'* [Bel98]. The best way to *'convince people of the efficacy of mathematical modeling and operations research methods was to present them with test examples demonstrating how a company could lose profit and even become bankrupt (under conditions of competition) if it did not use mathematical tools for analysis of its potential'* [Bel98]. One of the foremost reasons for inadequate and insufficient utilization of operations research methods in industry is *'the existing gap between the level of mathematical education of transportation managers and graduates from engineering colleges in the transportation field and that necessary for understanding the potential and substance of optimization methods'* [Bel98]. Traditionally, the majority of companies just make-do with hand-made procedures which are often developed as a result of the 'psychological mistrust' of optimization models. The author feels that those who make the financing decisions in the areas of strategic development in a company need to first become knowledgeable on the applicability and benefits of available techniques.

Many routing problems, including the one being studied, are difficult to express in terms of general statements as they have enormous numbers of variables and constraints. Some complex problems can be reduced or reformulated into *'optimization problems for which there exist either effective algorithms for their solving or developed heuristic approaches to their solving*

based upon well-known properties of the problems' [Bel98]. The ARPNDD problem being solved here is an extension of well known pattern routing problems described by Belenky [Bel98].

One such problem is the delivery (vehicle routing) problem with one base: from a base point, where loads and p transportation means are located, it is necessary to deliver these loads to N points of destination by the available transportation means which after ending the delivery, must return to the base point. For each transportation means, one knows its cargo-carrying capacity, cargo-holding capacity, time of work, and time interval during which the load should be delivered to each of the points. It is necessary to design routes for p transportation means subject to the mentioned restrictions and providing the minimal summary run of the transportation means, the minimal delivery time, etc.

The restrictions for the ARPNDD are the service day rules which define the customers' NDD's as well as their delivery group allocation. On each day, a variant of the p rural postman problem must be solved. This problem is considered for a non-orientated transportation network and consists of finding routes for p transportation means passing through a certain subset of the network edges and having minimal length.

Often a solution to a difficult problem can be found by combining solutions of simpler problems into which the original problem can be de-composed. Algebraic transformations, in particular polynomial reducibility, have proven to be useful. Belenky mentions that *'scheduling theory, as a branch of applied mathematics that studies models and methods of scheduling, relates to discrete optimization. At the same time, the scheduling theory has narrow-specialized methods aimed at solving narrow classes of problems and even separate problems of the scheduling theory proper that are constructed based on combinatorial features of problems solved rather than on general ideas of discrete optimization'* [Bel98]. The former methods in the *'arsenal of ones for the scheduling theory'* [Bel98] include implicit enumeration, equivalent transformations, local optimization, dual approach, de-composition and approximate methods with guaranteed estimates. The second group consists of permutation and combinatorial techniques, priority methods, cyclograms, functional analysis and methods of the theory of numbers.

2.2 Seminal Papers

One of the earliest, yet still the best, papers on the allocation of NDD's is *An assignment routing problem* by Russell and Igo [RI79]. Different heuristic methods which provide approximate solutions to the problem are discussed. The objective is to *'assign customer demand points to days of the week in order to solve the resulting node routing problems over the entire week most effectively'*. The biggest difference between this and an ordinary vehicle routing problem, is that most VRP's treat the assignment of demand points to days of the week as being fixed, whereas here, this is a decision of the model.

The notation and constraints used in the paper are summarized below:

S_i no. of days per week customer i is serviced, $1 \leq S_i \leq 7$ (input value, not a decision variable)
V number of vehicles available daily
C_k load capacity of vehicle k
D_k maximum distance (proportional to time) allowed for vehicle k on any route
Q_{id} demand at node i on day d
U_d maximum allowable load on day d
d_{ij} distance (proportional to time) from node i to node j
P_i set of permissable day assignments for node i

It is *'assumed that the resulting node routing problem on each day of the week is a single depot vehicle dispatch problem whose objective is to minimize the distance or time required to service customer demand points and to minimize the number of vehicles required'* [RI79]. By balancing the number of vehicles required, the volume of work done on each day is also balanced.

The following constraints are specified:

1. The total demand of the points assigned to day d does not exceed U_d
2. No more than V vehicles are required each day
3. The demand load for each vehicle does not exceed C_k for any route

4. The total distance traveled for any vehicle does not exceed D_k for any route

5. Each point is serviced by only one vehicle on each day of the week

6. The assignment of points requiring service more than once per week satisfies certain day assignment spacings (e.g. insisting that at least one day without service elapses between the first and last day of service).

For customers requiring service more than once a week, multiple copies of that point are made. The problem size is now expanded to the total number of drops over the week. *'Fortunately, the assignment of points requiring service 5 or 6 times per week is combinatorially simpler than the assignment of points requiring service 3 times per week in that there are fewer combinations of assignment'* [RI79]. After expanding the number of points, a subproblem can be created by clustering groups of points together. *'Points are grouped according to service frequency and are clustered only if their frequency of required service is identical'* [RI79]. The iterative clustering procedure is based on proximity; starting with points 0.1 miles apart and increasing the distance until the problem is reduced to an acceptable size. *'The point nearest the centroid of the cluster assumes the combined demand load of the points clustered'* [RI79].

Three heuristics are used to tackle this assignment routing problem. The first practical procedure works well on problems with a large number of customers (the authors have managed to solve a problem with more than 700 points) as it makes an intractable problem (like the one at hand) much easier to handle. Firstly, compact clusters of points are generated. Initially points requiring service six days per week or points requiring service on specific days are assigned. *'The resulting nuclei of clusters on each day of the week act as magnets in attracting other unassigned points. Classes of points are assigned sequentially in [decreasing] order of their frequency of service. Three statistics are calculated in order to determine the combination of days to which a particular point should be assigned'* [RI79]. The three measures are:

1. Average distance to the assigned nucleus of points on each day combination

2. Variance in this average

3. Average distance to the nearest point in each nucleus on each day

A point is assigned to the day combination with the lowest average distance. If this average distance is not at least 10% less than the next smallest distance then the day combination with the lowest variance in this distance is chosen. If the variance of the one distance is not at least 40% less than the other, then the decision is made based on the third statistic. The 10% and 40% rules were determined during empirical testing.

After this initial pass, subsequent passes generate feasible solutions according to the constraints above and balance the workload by considering reassignments. Total travel distance is not reduced much as the routes are sensitive to the initial nuclei. These passes prepare the solution for the next stage. The second heuristic stage uses a modification of the MTOUR algorithm (a generalization of the Lin and Kernighan traveling salesman algorithm for M salesman) developed by Russell. It considers both the assignment and routing aspects of the problem to reduce the total distance and time of the routes.

The approach works as follows: find from S, the set of all links (two connected nodes), a subset T that forms M distinct routes that satisfy all side constraints and minimizes distance traveled. Links y_i are identified in $S - T$ to replace links x_i in T, the current feasible set of M routes. Any exchanges, k, are explored as long as the gain criterion $G_k = \sum_{i=1}^{k} \{|x_i| - |y_i|\} > 0$ is satisfied. A promising link exchange is only implemented if the side constraints are met. MTOUR has not been found to be useful for problems with over 300 points.

Finally, the third heuristic, a modification of the widely used Clarke and Wright savings algorithm, is used, which checks for the load, distance and also the spacing constraint between service days. Initially all nodes are assumed to be connected directly to the depot. A saving, s_{ij}, is calculated for each pair of nodes i and j. This is the distance saved if the nodes were to be connected on the same route: $s_{ij} = d_{i1} + d_{1j} - d_{ij}$ (see Figure 2.1 below). This exchange algorithm then adds and deletes links for links with the largest distance savings and routes are built through successive iterations. To reduce computation time, a limited number of nearest neighbours are considered in the linking process.

Since the algorithm minimizes total weekly distance, the total distance on some days may increase when the 'before' and 'after' scenarios are compared. The potential savings are also limited if the 5- and 6-day a week customers are widely dispersed geographically as savings cannot be realized by allocat-

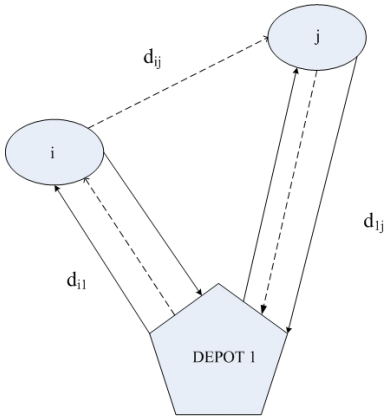


Figure 2.1: The Clarke and Wright savings concept works as follows: instead of traveling back to the depot after visiting each store, travel can be reduced by linking stores together.

ing customers in the same region to the same day. So, for a given point i, a smaller set P_i will allow fewer combinations of day assignments. The authors suggest generating a number of starting points from the first heuristic when attempting such a problem. The benefits derived from an assignment routing analysis are thus dependent on the composition of the data points. The solution described in the paper is limited as it only allows for groups of customers with the same service frequency to be grouped together on a route.

Another early paper in this area of logistics research, *Networks and vehicle routing for municipal waste collection*, also brings to light a number of interesting findings. The problem described by Beltrami and Bodin [BB74] where municipal waste collection vehicles collect refuse and terminate a route by visiting a dumpsite, can be adapted to the delivery version of the problem by treating the dump as a depot. The authors point out that this problem, like many others, does not fit into a mould and that *'one must be satisfied with near optimal solutions as obtained by combining formal arguments with heuristic reasoning'*. The routing is described as being done over the nodes and not the branches in the network since it is of interest to stop at the customers and not to cover certain roads like in a street sweeping exercise. The problem is thus known as a discrete or node routing problem (*in which case the branches indicate the nodes which are adjacent to each node* [BB74]).

An interesting point is made in the above mentioned paper about deciding whether to cluster the customer nodes first and route second or vice versa.

The book *The Logic of Logistics* [BSL97] describes this as the Optimal Partitioning (OP) heuristic defined by Beasley. Here, a traveling salesman tour is constructed through all the points and this is then split up into routes with a limit to the number of customers on each route. Such partitioning heuristics have been said to *'hardly exploit the topological structure of the Euclidean plane in which the points are located'* [HS88]. The technique is usually used *'if there are few routes to be formed with many pickup points on each route. Then it is generally more effective to form a giant tour and then partition this tour into smaller segments'* [BB74]. This is not true if there are many routes with few points on each because the stops are not ordered with regard to the time to go from the drop points to the depot.

In the latter case, a cluster-first route-second approach is preferred. Many researchers have suggested that the following criterion be used as a first step in data aggregation when such a heuristic is used: customers located in close proximity to each other are aggregated using a grid network or other clustering technique. All customers within a single cell or cluster are replaced by a single customer located at the centroid of the cell or cluster (customer zone). Another effective technique is aggregation according to postal code. A suggested guideline is to ensure that each zone has roughly equal demand which means zones may vary a lot in size. In South Africa, postal codes do not necessarily group customers in the same location and may cover vast and unusually shaped regions - particularly in rural areas. Bramel and Simchi-Levi [BSL97] state that for some customers *'in rural or isolated areas, it is harder to satisfy the same service level as most other customers'.*

The waste collection sites in this example required service either three days per week (Monday, Wednesday and Friday or Tuesday, Thursday and Saturday) or six days per week (Monday through Saturday). Two approaches are investigated. The first forms tours which are then assigned to different service day combinations while the second pre-assigns sites to days before routes are generated. In the first approach the Clarke and Wright savings procedure was used with the 6-time a week customers appearing twice in the network. In forming tours, *'a site and its image were not allowed to appear on the same tour'* [BB74].

Once these tours are formed they must be assigned to different service day combinations which can pose a difficulty. *'It is not obvious that from a given collection of tours a feasible day assignment can be found'* [BB74]. This example illustrates the problem: if three tours are generated with drop points {1,2,3}, {2,4,5} and {1,4,6} respectively (assuming points 1, 2 and 4

require service 6 time a week) then the first tour can be assigned to Monday, Wednesday and Friday and the second to Tuesday, Thursday and Saturday. This works well since point 2 is common to both routes. The second and third tours have point 4 in common so the third tour should be assigned to Monday, Wednesday and Friday. This leads to a contradiction because the first and third tours have point 1 in common and thus cannot both be assigned to the service day combination beginning with Monday. The authors managed to include a check in their algorithm to prevent such cycles but this becomes more difficult (and less optimal) as the number of 6-day a week points increase.

The second approach applies the Clarke and Wright algorithm separately to the points assigned to the Monday combination and those assigned to the Tuesday combination. The random assignment of customers to days can be repeated with the best result after the application of the savings algorithm being used. This is not a particularly strong approach, especially for problems of a practical size.

Beltrami and Bodin [BB74] have found from experience that by imposing additional constraints, certain tours causing the pathology in method 1 above, are blocked. Better results can sometimes be yielded because of this since the heuristic algorithm may not get locked onto local optima but be forced to find the global optimum. Finally, day schedules must be formed for each of the trucks. Several tours can be hooked together on one truck as long as crew time constraints, for example, are not violated.

Another study done in the area of waste disposal incorporated the decision of how much to collect (deliver) at each point. A given customer site (particulary those who almost fill the truck with the size of their load) can be allowed to be visited more than once a day. By allowing key nodes to be visited by different trucks on different routes, substantial savings can be realized. The authors [BB74] caution that the number of different possibilities is now much greater as there is a choice of how many visits to be made as well as how much to collect at each visit. *'In a general case when there are many nodes, randomizing a few of the large capacity nodes allow these demand points to act as slack variables in generating tours; the unused capacity of any tour being filled by part of the demand at these points'* [BB74]. The solution method proposed combines a randomized search procedure with the Clarke and Wright algorithm.

Suppose nodes i and j are encountered in the savings list and that each

node has m_i and m_j unassigned units to still be collected (delivered) respectively. The first check involves seeing whether the time constraints of the vehicle are violated if these two nodes are joined. If such a union is possible then two random integers (n_i) and (n_j) (one between 1 and m_i and another between 1 and m_j) are generated to represent the amount of units serviced at each node at that visit. This generation continues until a feasible load combination results (one that does not violate capacity constraints). Now, node i has m_i - n_i unassigned units and similarly for node j. The procedure is repeated until either or both points have their requirements satisfied. Once both items are depleted, the next item on the savings list is examined. If it occurs that one but not both points is exhausted, then the unexhausted node (say j) is linked to a tour with i as endpoint (as long as that tour does not contain j already). The amount to assign to m_j is again determined randomly as above. This continues until $m_j = 0$ or no more tours can be found to place j on. The next item on the savings list is then considered to increase the number and/or size of the tours. Usually a number of repetitions are done and the best retained until it becomes too costly to continue or subsequent iterations do not show much change or improvement.

The authors also made reference to graph theory in order to explain the problem characteristics. Many large scale problems in the area of graph theory still remain open and are NP-hard. Future development in this area should spark new attempts to find feasible solutions to routing problems with the aid of graph theory.

Christofides and Beasley [BC84] attempt to solve the period routing problem with heuristics that speed up the computational time of the solution method. All customers initially have a set of feasible delivery days assigned to them. What follows is an interchange procedure to improve the routing costs for each day. Instead of solving a VRP for each day when a change is considered, simpler calculations are done to determine if an improvement has been made or not. This allows more combinations of customers to days and routes to be considered because computation time is reduced for each case. Once the final day assignments have been made, standard VRPs are solved for each day.

'It is not easy to solve exactly a PVRP of nontrivial size, not least because for a given choice of customer-delivery-day combinations the resulting daily VRPs are difficult to solve' [BC84]. The authors derived two different heuristic subproblems related to the VRP, but easier to solve, in a response to the need for more efficient solution methods. They note that *'it*

is computationally expensive to evaluate the effect of each day-combination interchange by re-solving (even heuristically) the VRPs for each day' [BC84].

The PVRP problem can be seen as an instance of a more general problem: *'choose a combination for each customer so as to minimize the total cost over the period, there being a separate subproblem for each day of the period'* [BC84]. The general problem is stated as follows:

Let S_i be the set of allowable combinations for customer i.
Let n be the number of customers served in the T day period.
Let $x_{ik} = 1$ if the kth combination is chosen for i and 0 otherwise.

Minimize the total cost: $\sum_{t=1}^{T} z(P_t(x))$
Such that:
$\sum_{k \epsilon S_i} x_{ik} = 1$, $i = 1...n$
$C(P_t(x)) \geq 0$, $t = 1...T$
$x_{ik} \epsilon (0, 1)$, $i = 1...n$, $\forall\ k \epsilon S_i$

The second constraint ensures that the subproblem $P_t(x)$ is feasible with respect to the set of inequalities $C(P_t(x)) \geq 0$ (detailed constraints not given here). The PVRP corresponds to the above general problem where $P_t(x)$ is the VRP for day t involving only the customers serviced on day t.

The two relaxations of the VRP which are considered are a median problem (MP) and a traveling salesman problem. Christofides has shown in previous works that the expected length of vehicle routes in a VRP is monotonically related to the sum of the radial distances of the customers from the centre. By *'minimizing the sum of the radial distances of the customers from a centre chosen for each day of the period'* [BC84] it is expected that the underlying PVRP would also be minimized. A similar approach is taken with the TSP. By minimizing the TSP tour, the corresponding VRP is also assumed to be minimized because of the monotonic relationship between these problems.

The general heuristic algorithm for the PVRP in the paper has two parts:
A. Initial allocation

1. Place the customers in a list ordered by importance; first those with a fixed delivery combination, followed by the remainder in descending order of demand size. This ordering is appropriate where capacity constraints are tight and avoids feasibility problems later.

2. Working from the top of the list, calculate the total increase in cost of the affected routes (with only the customers higher in the list) when a day combination is chosen for a customer. Choose the feasible combination that is the cheapest.

3. Apply a local optimization procedure to each route for each day. For the MP algorithm, this means finding the best centre for the day. For the TSP, the best TSP solution for each day is found.

B. Interchanges

1. A family of small subsets of customers are chosen. For the procedure to be computationally efficient, a restricted number of such subsets is selected.

2. Choose a subset and remove the customers in that subset from the current day assignments.

3. All delivery combination possibilities are enumerated for this subset.

4. If an improvement in cost occurs for any of the possibilities in the previous step then the local optimization procedure (see last point in step A) is applied. Another subset from the family of subsets is then chosen and the interchange procedure begins again.

5. Stop when the maximum number of iterations has been reached or no improvement has been made.

The authors found that, on the biggest problem solved (with 126 customers), there was an improvement of over 10% in routing costs from the previous best known solution. This was attributed to the TSP heuristic which was found to be more effective than the MP heuristic.

Tan and Beasley [TB84] have developed a heuristic for solving the PVRP based on the daily VRP algorithm developed by Fisher and Jaikumar (as quoted in [TB84]). The above mentioned VRP algorithm is first extended to the PVRP. This resulting PVRP is a large, complex, zero-one integer formulation. Subsequently, a smaller, simpler zero-one integer program is developed which can be solved computationally.

Let n be the number of customers and K the number of vehicles available each day. The Fisher and Jaikumar formulation is thus *'a linear (n by K) generalized assignment problem, the solution to which defines a (capacity)*

feasible assignment of customers to vehicles. The delivery sequence for each vehicle can be determined by applying any (heuristic or optimal) traveling salesman algorithm to the customers assigned to the vehicle' [TB84]. When this problem is extended to T days it is not a linear (n by KT) generalized assignment problem as might be supposed. This is because a customer must first be assigned an allowable day combination and then routed on a vehicle for each day in that combination. The integer program has O(n+nT+KT) constraints and O(nKT) variables.

To make this formulation much simpler, we neglect the assignment of customers to vehicles and place a constraint on the total demand on any day in the period. Once all customers have been assigned to delivery days, a VRP is solved for each day. This simpler PVRP program has (n+T) constraints and (n * average number of delivery combinations per customer) variables. The formulation involves the use of a measure D_{it}; the contribution of a customer to the routes on a particular day. A linear programming relaxation is used to assign customers to day combinations.

The results from using this method, appeared to be competitive with the solutions provided in [BC84] with respect to both speed and quality. The authors briefly mention that the formulation could be also be adapted to cases where the amount to be delivered depends on the day of delivery.

2.3 Fixed Routes

Beasley shows in his paper, *Fixed routes* [Bea84], how standard vehicle routing algorithms can be adapted to solve the problem of designing routes which can be used repeatedly. Three types of vehicle routing problems are defined:

1. Daily Routing: where a set of vehicle routes are developed for a single day's deliveries.

2. Period Routing: where a set of vehicle routes have to be developed for a certain period to meet customer service level requirements (not all customers requiring delivery on every day in the period). So a set of delivery days from allowable combinations needs to be decided upon for the customers. This is the ARPNDD problem without the requirement that customers who are serviced together on any one route must be serviced together on all other days on which they receive a delivery.

The daily routing problem, as pointed out, is a special case of the period problem.

3. Fixed Routes: where a set of vehicle routes has to be developed that can be operated unchanged for a given period of time.

Beasley [Bea84] points out that problem 2 described above *'has been given little attention in the literature'* but that *'research indicates that substantial benefits can be obtained by applying a systematic method to the choice of delivery days for customers to meet service level requirements'.*

Let T be the number of days in the period

n the total number of customers

q_{it} be the demand of customer i on day t.

Feasible routes must thus be designed for the T days. The demand quantities may be obtained from historical job data or *'sampled from a projected distribution of demand data'* [Bea84]. If the objective is to first minimize the number of vehicles used and then the distance traveled, then the constraint that all the routes are feasible may be relaxed. There may only be one day in the week where x vehicles are used and $x - 1$ on all other days. Instead of having x vehicles, the company may be able to hire in the extra vehicle on the one day so that they don't have an idle vehicle on the four days with lower volume. If T_f is the number of days for which any fixed route has to be feasible, the problem is to design routes where at least $T_f \leq T$ routes are feasible with respect to quantity delivered, distance traveled, etc.

If infeasible routes are to be accepted, they must be dealt with without having to completely restructure the fixed routes. Three options are identified in the paper:

1. Ignore the infeasibility (if the capacity constraint is slightly violated for example)

2. Schedule a second vehicle (by splitting the route in two and maintaining the drop sequence)

3. Drop customers from the fixed route until it becomes feasible. The dropped customers can then be:

   (a) left until another day

(b) moved onto other routes

(c) loaded onto another vehicle (an extra vehicle being used here)

The choice of a, b or c will affect the choice of which customers to remove.

Beasley [Bea84] mentions two papers in his research: one by O'Brien and the other by Christofides. O'Brien suggests designing fixed routes by first routing the outlying customers and then adding on those closer to the depot. Christofides only puts forward a proposal for designing fixed routes with daily repetition. It involves *'solving a number of daily vehicle routing problems based on typical customer demand data and then using the frequency of occurrence of each inter-customer link as a basis for forming fixed routes'* [Bea84].

Beasley's computational results for the daily repetition problem show that the additional costs of running fixed routes, rather than designing new routes every day, are negligible. The simplicity of having fixed routes and the time savings far outweigh a slight drop in profits.

2.4 The Periodic VRP

Michael Ball describes allocation/routing problems as follows : they *'involve determining a set of routes for a fleet of vehicles over a multiple day time horizon. Thus, these problems can be viewed as containing two components, one that allocates deliveries to days of the week and a second that forms routes over each day of the week'* [Bal88]. These models were initially designed for use in the waste collection industry as mentioned earlier.

The period routing model described by Ball [Bal88] includes a variable to assign a customer to a service pattern. Each customer has a set of possible patterns which can be used to describe his demand. Each pattern specifies the feasible delivery days and the proportion of the total weekly demand delivered on each of those days. *'While several patterns might be feasible to a particular customer, there was a definite preference amongst them.'* This is particularly true in a driver-sell situation where additional product could be sold when a delivery is made. It is preferable in such a situation to deliver on a day when the client has the most warehouse space available.

Notation used in the paper is shown below:

M is the set of customer locations

n is the number of days of the planning horizon
The horizon is the time over which the problem will be solved and over which it will subsequently be used on a regular basis.

d_{ih} customer i's delivery size on day h

P is the set of patterns

f_{ph} is the fraction of total customer demand allocated to day h by pattern p

$$e_{ph} = \begin{cases} 1 & \text{if } f_{ph} > 0 \\ 0 & \text{otherwise} \end{cases}$$

d'_i is the total demand for customer i for the planning horizon

g_{ip} is the value of customer i/pattern p combination

$$z_{ip} = \begin{cases} 1 & \text{if pattern } p \text{ is assigned to customer } i \\ 0 & \text{otherwise} \end{cases}$$

$$y_{ih}{}^{1} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to day } h \\ 0 & \text{otherwise} \end{cases}$$

x_{ih} is the amount delivered to customer i on day h

S_h is the set of customers serviced on day h

The Period Routing Problem is now formulated as:

Min $\sum_{h=1}^{n} VRP(S_h, h) - \sum_{i \epsilon M} \sum_{p \epsilon P} g_{ip} z_{ip}$

Such that:

[1]The original paper used y_{ikh} (where k represented a vehicle) but this varible was absent in the constraints.

$\sum_{p\epsilon P} z_{ip} = 1 \ \forall \ i \ \epsilon \ M$ (i)

$d_{ih} = \sum_{p\epsilon P}(f_{ph}d'_i)z_{ip}$ **for** $h = 1, 2...n$ **and** $i \ \epsilon \ M$ (ii)

$y_{ih} = \sum_{p\epsilon P} e_{ph}z_{ip}$ **for** $h = 1, 2...n$ **and** $i \ \epsilon \ M$ (iii)

$S_h = \{i \ \epsilon \ M : y_{ih} = 1$ **for** $h = 1, 2...n\}$ (iv)

$y_{ih}, z_{ip} \ \epsilon \ 0, 1 \ \forall \ i, h$ **and** p (v)

$d_{ih} \geq 0 \ \forall \ i$ **and** h**.** (vi)

Constraint (i) ensures that each customer will be assigned one pattern. The daily demand implied by the pattern allocation is captured in constraint (ii). Constraint (iii) ensures that if a delivery (pattern) is assigned to day h and that delivery is assigned to a customer, then that customer is visited on day h. (iv) defines the customer set that will be serviced on each day.

A p-median approach has been suggested to first cluster the customers before solving the routing aspect of the problem just described. The p-median problem is the problem of locating p 'facilities' relative to a set of customers such that the sum of the shortest demand weighted distance between customers and 'facilities' is minimized. In this case the 'facilities' are the centre of a cluster of customers. It involves a similar concept to that of a seed in the generalized assignment problem. A customer/day surrogate assignment cost is applied. The *'expected total vehicle routing costs increase as the total customer radial distances from a centre increase'* [Bal88]. By *'associating a centre with each day'* the distance to the centre can then be used *'as a surrogate for the cost of assigning a customer to a day'* [Bal88]. What tends to happen as a result, is that each day's routes are clustered in a certain geographical area. If a centre is chosen in advance for each day then the following problem results:

Min $\sum_{i\epsilon M}\sum_{h=1}^{n} b_{ih}y_{ih} - \sum_{i\epsilon M}\sum_{p\epsilon P} g_{ip}z_{ip}$

Such that:

All previous constraints hold and

$\sum_{i\epsilon M} d_{ih} \leq q*$ **for** $h = 1, 2...n$

where $q*$ is an artificial daily capacity equal to the total load the vehicles

could handle in a day.

b_{ih} is the cost of assigning customer i to the centre associated with day h.

One of the most recent contributions to the PVRP literature has come from Francis, Smilowitz and Tzur [FT05]. Their first paper introduces an extension of the original PVRP, the PVRP-SC; the periodic vehicle routing problem with service choice. This model allows for visit frequency to each node to be a decision of the model. This is because operational efficiency may be gained when nodes are serviced more frequently than required. Consider the following simple case put forward in the paper: if two outlying nodes have different service schedules then servicing the node with fewer visits per week to be served on the same days as the other node with a higher pre-set visit frequency may result in cost savings.

In Figure 2.2 below, it would be more efficient to service node A on Monday, Wednesday and Friday since there would be a truck in the area visiting node B. This would mean that a truck does not have to travel so far out of town five days a week.

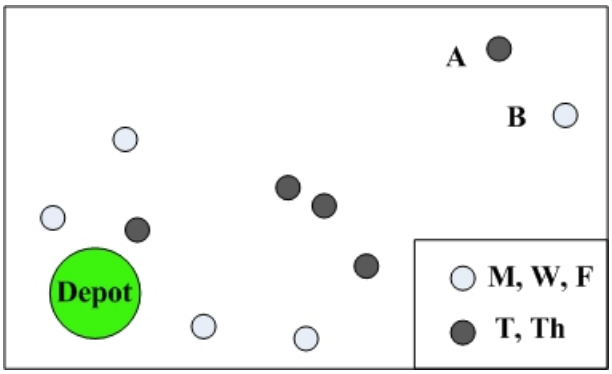


Figure 2.2: When determining a customer's nominated delivery days (NDD's) the location of customers on similar service days should be considered.

The PVRP-SC is defined in *The period vehicle routing problem with service choice* [FT05] as follows:

GIVEN: A set of nodes with known demand and minimum visit frequency requiring service over the planning period; a fleet of capacitated vehicles; a set of service schedules with head-ways and service benefits; and a network with travel times.

FIND: An assignment of nodes to service schedules and a set of vehicle routes for each day of the planning period that minimizes the total routing cost incurred net of the service benefit.

The customer and/or the system benefit from higher visit frequencies, which is accounted for in the objective function. The benefit from higher service frequency may be attributed to lower holding costs or the customer's willingness to pay for enhanced (more regular) service. The PVRP-SC thus *'exploits possible inefficiencies from combined routing and service decisions'* [FT05]. The authors also point out that the inventory routing problem (IRP), like the PVRP-SC, determines visit frequency and route configuration simultaneously. With the PVRP-SC, the amount delivered is the amount accumulated since the last visit (i.e. it is dependent on the schedule chosen) while in the IRP the amount delivered is a decision independent of visit frequency.

The notation below is used in the formulation of the PVRP-SC model:

N set of demand nodes

$N = \{1, \ldots n\}$; node 0 represents the depot

A set of network arcs

$A = (i, j) : i, j \; \epsilon \; N \cup 0$

K set of vehicles

C vehicle capacity; (items per vehicle)

t represents the length of the period

T set of days $T = \{1 \ldots t\}$

S set of service schedules; $S = 1 \ldots |S|$

$s \; \epsilon \; S$ is a subset of T

t_{ij} travel cost on arc $(i, j) \; \epsilon \; A$; (dollars)

w_i demand at node $i \; \epsilon \; N$;
(items per day, where each item is assumed to occupy the same amount of space)

f_i minimum visit frequency at node $i \; \epsilon \; N$; (number of days per period)

γ^s service frequency for schedule $s \epsilon S$; (number of days)

τ_s^i stopping cost at node $i \epsilon N$ when served by schedule $s \epsilon S$; (dollars/stop)

α^s service benefit for schedule $s \epsilon S$; (dollars/item)

β^s demand accumulation adjustment factor for schedule $s \epsilon S$

Notes:
τ_s^i the stopping cost at a node is a function of the demand at the node and the frequency of the schedule the node is on

α^s can be interpreted as the perceived utility of a customer who is serviced more frequently than f_i

β^s is an adjustment factor and is estimated as the maximum number of days between visits on schedule s. This estimate is conservative and guarantees that vehicles are not overfilled but it may preclude better solutions.

The following sets of decision variables are used:

$$yik^s = \begin{cases} 1 & \text{if node } i \epsilon N \text{ is visited by vehicle } k \epsilon K \text{ on schedule } s \epsilon S \\ 0 & \text{otherwise} \end{cases}$$

This ensures that nodes are visited by the same vehicle each time.

$$x_{ijk}^d = \begin{cases} 1 & \text{if vehicle } k \epsilon K \text{ traverses the arc } (i,j) \epsilon A \text{ on day } d \epsilon T \\ 0 & \text{otherwise} \end{cases}$$

The discrete PVRP-SC can now be stated formally as:

$$Z^\star = min \sum_{k \epsilon K} [\sum_{d \epsilon T} \sum_{(i,j) \epsilon A} t_{ij} x_{ijk}^d + \sum_{s \epsilon S} \sum_{i \epsilon N} (\gamma^s \tau_i^s - \omega_i \alpha^s) y_{ik}^s]$$

Subject to:

$$\sum_{s\epsilon S}\sum_{k\epsilon K}\gamma^s y^s_{ik} \geq f^i \ \forall\ i\ \epsilon\ N \tag{i}$$

$$\sum_{s\epsilon S}\sum_{k\epsilon K} y^s_{ik} \leq 1 \ \forall\ i\ \epsilon\ N \tag{ii}$$

$$\sum_{s\epsilon S}\sum_{i\epsilon N}(\beta^s\omega_i)a_{sd}y^s_{ik} \leq C \ \forall\ k\ \epsilon\ K;\ \forall\ d\ \epsilon\ T \tag{iii}$$

$$\sum_{j\epsilon N\cup\{0\}} x^d_{ijk} = \sum_{s\epsilon S} a_{sd}y^s_{ik} \ \forall\ i\ \epsilon\ N;\ \forall\ k\ \epsilon\ K;\ \forall\ d\ \epsilon\ T \tag{iv}$$

$$\sum_{j\epsilon N\cup\{0\}} x^d_{ijk} = \sum_{j\epsilon N\cup\{0\}} x^d_{jik} \ \forall\ i\ \epsilon\ N\cup\{0\};\ \forall\ k\ \epsilon\ K;\ \forall\ d\ \epsilon\ T \tag{v}$$

$$\sum_{i,j\epsilon Q} x^d_{ijk} \leq |Q| - 1 \ \forall\ Q \subseteq N;\ \forall\ k\ \epsilon\ K;\ \forall\ d\ \epsilon\ T \tag{vi}$$

$$y^s_{ik}\ \epsilon\ \{0,1\} \ \forall\ i\ \epsilon\ N;\ \forall\ k\ \epsilon\ K;\ \forall\ s\ \epsilon\ S \tag{vii}$$

$$x^d_{ijk}\ \epsilon\ \{0,1\} \ \forall\ (i,j)\ \epsilon\ A;\ \forall\ k\ \epsilon\ K;\ \forall\ d\ \epsilon\ T \tag{viii}$$

The objective function balances travel time and service benefit. The first term represents arc travel times, the second term node stopping costs and finally demand-weighted service benefit. By adjusting the value of α_s the emphasis of the service benefit can be controlled.

Constraints (i) enforce the minimum frequency of visits for each node. Constraints (ii) ensure that one vehicle and one schedule are chosen for each demand node. Constraints (iii) represent vehicle capacity constraints. Constraints (iv) link the x and y variables for the demand nodes. Constraints (v) ensure flow conservation at each node. Constraints (vi) are subtour elimination constraints and ensure that all tours contain a visit to the depot. Constraints (vii) and (viii) define the binary assignment and routing variables respectively.

To make the problem more tractable and the routes more implementable, it is possible to limit the number of routes performed by each driver. If a set of schedules satisfies the following lemma then the number of different routes does not depend on t.

LEMMA 1: *Assume that the set of schedules S includes $|S|-1$ disjoint schedules (schedules which do not share any common days) and a schedule $|S|$ that is the union of all disjoint schedules. Then the set of nodes served on a route on a certain day included in disjoint schedule s is always visited on the same route each day in schedule s. This results in at most $|S|-1$ different routes for each vehicle.*

This results in each node that is being visited by a disjoint schedule, being visited by the same vehicle at the same time each day. If the set of schedules is as defined in Lemma 1 then the number of routing variables x_{ijk} can be reduced.

Let U be the set of disjoint schedules $1 \ldots |S| - 1$. The x variables are now defined by unique delivery days, x^d_{ijk}; $u \, \epsilon \, U$. Also, γ^s and a_{sd} are indexed by u rather than d with:

$$a_{su} = \begin{cases} 1 & \text{for } s = u, s \, \epsilon \, S, \, u \, \epsilon \, U \\ 1 & \text{for s= } |S|, u \, \epsilon \, U \\ 0 & \text{otherwise} \end{cases}$$

The objective function is now:

$$Z^\star = min \sum_{k \epsilon K} [\sum_{u \epsilon U} \sum_{(i,j) \epsilon A} \gamma^u t_{ij} x^u_{ijk} + \sum_{s \epsilon S} \sum_{i \epsilon N} (\gamma^s \tau^s_i - \omega_i \alpha^s) y^s_{ik}]$$

The authors use a Lagrangian relaxation to find a solution to the PVRP-SC. Firstly the constraints linking the $x's$ (routing variables) and the $y's$ (assignment variables) are relaxed. Lagrangian multipliers are introduced for the relaxed constraints; λ^u_{ik} for $i \, \epsilon \, N; k \, \epsilon \, K; u \, \epsilon \, U$ with $\lambda^0_{ku} = 0$. The following Lagrangian function is obtained:

$$\text{LR}(\lambda) = \sum_{k \epsilon K} [\sum_{u \epsilon U} \sum_{(i,j) \epsilon A} \gamma^u t_{ij} x^u_{ijk} + \sum_{s \epsilon S} \sum_{i \epsilon N} (\gamma^s \tau^s_i - \omega_i \alpha^s) y^s_{ik}] + \sum_{u \epsilon U} \sum_{k \epsilon K} \sum_{i \epsilon N} \lambda^u_{ik} (\sum_{j \epsilon N \cup 0} x^u_{ijk} - \sum_{s \epsilon S} a_{su} y^s_{ik}) =$$

$$\sum_{k \epsilon K} \sum_{u \epsilon U} \sum_{(i,j) \epsilon A} (\gamma^u t_{ij} + \lambda^u_{ij}) x^u_{ijk} + \sum_{k \epsilon K} \sum_{s \epsilon S} \sum_{i \epsilon N} (\gamma^s \tau^s_i - \omega_i \alpha^s) y^s_{ik} - \sum_{k \epsilon K} \sum_{u \epsilon U} \sum_{i \epsilon N} \sum_{s \epsilon S} \lambda^u_{ik} a_{su} y^S_{ik}$$

For a given λ vector: $Z_\lambda(x, y) = min_{x,y} LR(\lambda)$ subject to all the remaining constraints. The Lagrangian function is minimized by the solutions to the following two independent subproblems where $Z_\lambda(x, y) = Z_\lambda(x) + Z_\lambda(y)$.

Assignment subproblem:

$$Z_\lambda(y) = min \sum_{k \epsilon K} \sum_{s \epsilon S} \sum_{i \epsilon N} (\gamma^s \tau^s_i - \omega_i \alpha^s) y^s_{ik} - \sum_{k \epsilon K} \sum_{u \epsilon U} \sum_{i \epsilon N} \sum_{s \epsilon S} \lambda^u_{ik} a_{su} y^S_{ik}$$

Subject to:

$$\sum_{s \epsilon S} \sum_{k \epsilon K} \gamma^s y^s_{ik} \geq f^i \; \forall \; i \, \epsilon \, N$$

$\sum_{s\epsilon S}\sum_{k\epsilon K} y_{ik}^{s} \leq 1 \ \forall \ i \ \epsilon \ N$

$\sum_{s\epsilon S}\sum_{i\epsilon N}(\beta^{s}\omega_{i})a_{su}y_{ik}^{s} \leq C \ \forall \ k \ \epsilon \ K; u \ \epsilon \ U$

$y_{ik}^{s} \ \epsilon \ \{0,1\} \ \forall \ i \ \epsilon \ N; k \ \epsilon \ K; s \ \epsilon \ S$

Routing subproblem:

$Z_{\lambda}(x) = min \sum_{k\epsilon K}\sum_{u\epsilon U}\sum_{(i,j)\epsilon A}(\gamma^{u}t_{ij} + \lambda_{ik}^{u})x_{ijk}^{u}$

Subject to:

$\sum_{j\epsilon N\cup\{0\}} x_{ijk}^{u} = \sum_{j\epsilon N\cup\{0\}} x_{jik}^{u} \ \forall \ i \ \epsilon \ N \cup \{0\}; k \ \epsilon \ K; u \ \epsilon \ U$

$\sum_{i,j\epsilon Q} x_{ijk}^{u} \leq |Q| - 1 \ \forall \ Q \subseteq N; k \ \epsilon \ K; u \ \epsilon \ U$

$x_{ijk}^{u} \ \epsilon \ \{0,1\} \ \forall \ (i,j) \ \epsilon \ A; k \ \epsilon \ K; u \ \epsilon \ U$

The latter (routing) subproblem de-composes further by (k, u) pairs for each vehicle/delivery day combination. This is an example of the TSP with profits where it is desired to find a route of minimum cost and maximum profit (each node has a profit associated with it) without visiting each node. This problem is very difficult to solve and bounds are embedded within the Lagrangian iterations. After each iteration of the algorithm, the *'multipliers are updated according to a standard subgradient optimization procedure'* [FT05]. The iteration's lower bound is set to the sum of the optimal solution of the assignment problem and the lower bound of the routing problem. At the end of the algorithm, the best feasible upper bound represents the suggested solution. If the upper and lower bounds converge to the same value then the optimal solution has been found. If not, a branch and bound (B&B) algorithm is used to narrow the gap.

The authors used a test case with 50 delivery points and 3 vehicles over a period of 3 days. They admit that finding solutions to larger problem sizes usually results in bottlenecks at the routing subproblem phase. They sug-

gest controlling this using a 'time budget'. Alternatively, the B&B procedure could be terminated when *'the lower bound is greater than* $(1-\delta)$ *of the upper bound'* [FT05].

B&B is not typically used for hard problems like the PVRP-SC. Two reasons justify its use in this problem though; the upper bound from the Lagrangian relaxation phase is usually not too far from the optimal (*'significant parts of the branch and bound tree are thus truncated'* [FT05]) and solutions from the subproblems eliminate the need for repetitive computation (they can be used to guide the B&B procedure as well).

Branching is only done over the assignment decision variables with an aggregate decision variable being introduced:

$$z_i^s = \sum_{k \epsilon K} y_{ik}^s = \begin{cases} 1 & \text{if node } i \text{ is assigned to schedule } s \\ 0 & \text{otherwise} \end{cases}$$

This is done since the vehicle index assigned to each route is arbitrary. *'Further down the branch and bound tree more z-variables become fixed and both subproblems become more restricted'* [FT05].

The authors mention that route length constraints may be useful if driver shifts are a factor in the business to which the model is being applied. Such constraints would create another link between the x and y variables since travel time is a function of the x variables and stopping cost a function of the $y's$.

Also, in a more general case (as in the problem being investigated on the Clover data) schedules may overlap and nodes may be visited by different vehicles. The accumulation parameter would thus be β^{sd} and would be more precise than the conservative estimate used in this formulation.

Figure 2.3 depicts the three options for implementation of the solution method:

1. Terminating after Lagrangian Relaxation (LR). This method finishes with a gap (G_1) between the feasible solution (Z^{LR}) and the final lower bound (LB^{LR}).

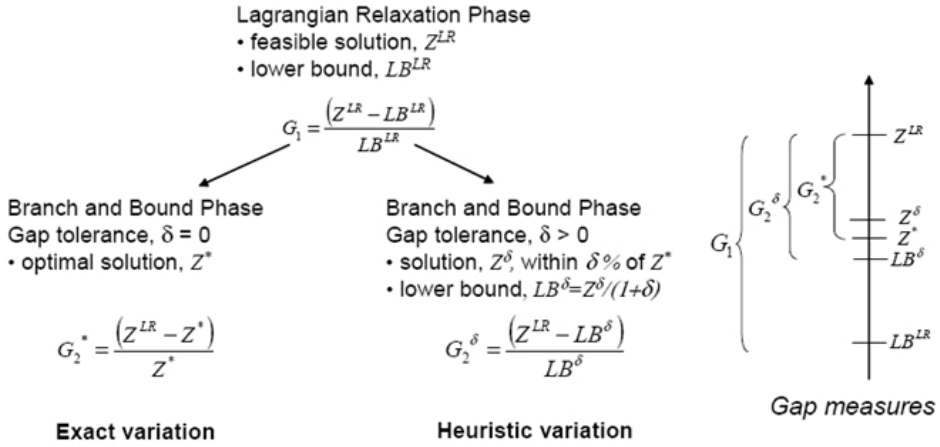


Figure 2.3: Solution approach for the PVRP-SC.

2. Continuing with exact branch and bound (B&B) until an optimal solution is found.

3. Continuing with heuristic B&B until a solution within δ% of the optimal is found.

The authors suggest using methods 2 or 3 for smaller problem instances and in this way finding an estimate of the gap (G_1) for larger problem instances. If the exact solution is available, the quality of Z^{LR} is assessed relative to Z^* with G_2^*. If a solution with δ optimality is available, the quality of Z_{LR} is assessed relative to a lower bound on the optimal solution, $LB^\delta = Z^\delta/(1+\delta)$, with a gap G_2^δ.

Using LR can result in savings in computational time. In the 2-phase approach, LR *'significantly reduces the gap before performing B&B'* [FT05]. Two trends were clear from the simulations done by the authors: G_1 was larger for larger sets of nodes and smaller with additional vehicle capacity.

Three variations of the PVRP were run on several test cases; the normal PVRP, the PVRP-SC with service benefit not included in the objective but changes in visit frequency to improve routing efficiency allowed for, and the PVRP-SC with service benefit in the objective. The value of the objective in case 1 is an upper bound for case 2. The optimal value of the routing component in case 2 is a lower bound for the same component in case 3. The improvement from using case 3 often comes more from the service benefit rather than reduced routing costs. It is the *'geographic distribution of the*

highest minimum frequency nodes' which *'impacts the extent to which service choice can reduce the travel distances between remote nodes and the depot. The potential for savings is greater when high frequency nodes are closer to the depot. The move to greater service levels also depends on the stopping costs'* [FT05].

The continuous PVRP model, developed by the same researchers, can *'result in more efficient vehicle tours and/or greater service benefit to customers'* [FS06] than the discrete model. The results from the simpler model formulation can also be used as guidelines for constructing solutions to the discrete PVRP model and for making strategic and tactical decisions. *'This research provides practitioners with a tool to analyze efficiencies in distribution operations arising from service choice, without requiring extensive computations and detailed data collection typical of discrete models for periodic vehicle routing problems'* [FS06].

To assess whether operational improvements, due to the inclusion of service choice, warrant the increased model complexity we must turn to the results of the continuous case.

The notation used in the development of the continuous PVRP with service choice (PVRP-SC) is:

Let $\delta^i(x)$ denote the spatial density of nodes with minimum service schedule $i \epsilon S$ about a point x, measured in nodes per unit area.

For a subregion A of R, let $N(A) = \sum_{i \epsilon S}(\int_{x \epsilon A} \delta^i(x)dx)$

Let $\lambda^i(x)$ denote the demand density rate about a point x of nodes with minimum service schedule $i \epsilon S$, measured in items per unit time-area.

Let $f^{is}(x)$ denote the fraction of nodes about a point x with minimum schedule i being served by schedule s.

Let $\Delta^d(x)$ denote the spatial density of nodes about a point x to be visited on day d, measured in nodes per unit area.
The effective density is: $\Sigma_{s \epsilon S} a_{sd} \Sigma_{i \epsilon S} \delta^i(x) f^{is}(x)$

Let $\Lambda^d(x)$ denote the demand density on day d, measured in demand per unit area.
The effective demand density collected is: $\Sigma_{s \epsilon S} a_{sd} H^s \Sigma_{i \epsilon S} \lambda^i(x) f^{is}(x)$

Let $n^d(x)$ denote the number of stops on a route on day d.

Let $v^d(x)$ denote the shipment size collected at a node.

Let $r(x)$ denote the distance from the depot to a point $x \epsilon R$.

The average distance between nodes is approximated by $(\Delta^d(x))^{-1/2}$ and a metric-dependent constant $\hat{k}$.

Let $\bar{c}$ denote the average cost per distance.

Let $\dot{\tau}$ and $\ddot{\tau}$ denote the fixed and variable stopping costs respectively.

Let $z^d(x) = \frac{2r(x)\bar{c}}{n^d(x)v^d(x)} + \frac{\bar{c}\hat{k}(\Delta^d(x))^{-1/2} + \dot{\tau} + \ddot{\tau}v^d(x)}{v^d(x)}$ denote the daily routing cost on day $d \epsilon D$.
The first term represents the line-haul travel between the depot and nodes. The second term represents the cost of visiting demand nodes.

Let Z_R denote the total cost over the planning period.

The continuous PVRP-SC can now be stated formally as:

Min $Z_R = \int_{x \epsilon R}(\sum_{d \epsilon D} \Lambda^d(x) z^d(x) - \sum_{s \epsilon S} \alpha^s \sum_{i \epsilon S} \lambda^i(x) f^{is}(x))dx$

Subject to:

$$n^d(x)v^d(x) \leq C \ \forall \ d \ \epsilon \ D, x \ \epsilon \ R \tag{i}$$

$$v^d(x) = \frac{\Lambda^d(x)}{\Delta^d(x)} \ \forall \ d \ \epsilon \ D, x \ \epsilon \ R \tag{ii}$$

$$\Delta^d(x) = \sum_{s \epsilon S} a_{sd} \sum_{i \epsilon S} \delta^i(x) f^{is}(x) \ \forall \ d \ \epsilon \ D, x \epsilon R \tag{iii}$$

$$\Lambda^d(x) = \sum_{s \epsilon S} a_{sd} H^s \sum_{i \epsilon S} \lambda^i(x) f^{is}(x) \ \forall \ d \ \epsilon \ D, x \ \epsilon \ R \tag{iv}$$

$$\sum_{s \epsilon S} f^{is}(x) = 1 \ \forall \ i \ \epsilon \ S, x \ \epsilon \ R \tag{v}$$

$$0 \leq f^{is}(x) \leq 1 \ \forall \ i, s \ \epsilon \ S : \gamma^i \leq \gamma^s, x \ \epsilon \ R \tag{vi}$$

$$f^{is}(x) = 0 \ \forall \ i, s \ \epsilon \ S : \gamma^i > \gamma^s, x \ \epsilon \ R \tag{vii}$$

$$n^d(x) \geq 0 \ \forall \ d \ \epsilon \ D, x \ \epsilon \ R \tag{viii}$$

Constraints (i) ensure that vehicle routes do not exceed capacity. Constraints (ii) define the items collected per stop as the demand density divided by the node density about a point (all accumulated demand between visits is collected). Constraints (iii) and (iv) define the auxiliary decision functions. Constraints (v) ensure that all nodes are assigned to a schedule. Constraints

(vi) and (vii) ensure that no node is served with a lower frequency than the minimum specified. Constraints (viii) are non-negativity constraints on the decision function.

The discrete PVRP-SC can be seen as a special case of the continuous PVRP-SC with results of the continuous model being discretized or use complimentarily with the discrete model. The continuous PVRP-SC is preferred by practitioners because *'the complexity of the discrete PVRP increases significantly when service choice is introduced'* [FS06]. Problem instances in the literature are solved by dividing the service region into subregions such that the node densities and demand rates are approximately uniform within each region. This type of step is not as simple as it sounds as other considerations, such as time at customer, must also be balanced.

Francis, Smilowitz and Tzur [FT06] define a set of metrics to quantify the desirability of routing solutions in a periodic distribution context by examining operational complexity and flexibility. They note that *'operational flexibility can help to avoid under-serving customers with high service requirements and over-serving customers with low requirements'* [FT06].

While introducing operational flexibility in periodic distribution problems can increase efficiency in terms of vehicle routing costs and customer service benefits, it poses challenges in modeling flexibility accurately, addressing the computational effort needed to solve problems with such flexibility and implementing resulting solutions. A Tabu search method developed by the authors allows for the following operational flexibility options to be incorporated: increasing the set of visit schedules, deciding visit frequency, varying the drivers visiting the customers and deciding on the amount to be delivered. Quantitative measures are also established to evaluate the trade-off between flexibility and complexity in distribution problems:

- arrival span - affects customer staffing
- driver coverage - driver familiarity can improve performance
- crew-size - number of different drivers visiting a customer over a period

Managerial observations from their study include the finding that introducing flexibility is more beneficial when high frequency nodes are located close to the depot (as in a traditional city with the possibility of sprawl on the outskirts).

The PVRP is computationally complex to solve and solutions are limited to problems of small to moderate size (examples in the literature use 4 vehicles and 3 service day frequency combinations). Approximate solutions for larger instances (with over 100 delivery points) may be found in reasonable time with continuous approximation models, although their use has been limited. Aggregated data are sometimes used as inputs which smoothes minor dynamic and stochastic variations which are not critical in strategic planning. Other benefits include allowing the system designer to experiment with multiple settings of the input parameters and developing managerial insights.

2.5 Applications of the VRP

2.5.1 Logistics decisions

The Council of Logistics Management defines logistics management as: *'the process of planning, implementing, and controlling the efficient, effective flow and storage of goods, services, and related information from point of origin to point of consumption for the purpose of conforming to customer requirements'* [BSL97]. The objective is to design or reconfigure the logistics network so as to minimize annual system-wide costs including production and purchasing costs, inventory holding costs, facility costs (storage, handling and fixed costs) and transportation costs, subject to a variety of service level requirements. It is the latter costs that shall be minimized by the decisions systems and methods mentioned. Day-to-day operational level decisions include how to schedule, route and load trucks.

The decision support system must have the capability of dealing with issues (customer-specific service level requirements, expansion of existing warehouses and flow patterns from the depot/warehouse to customers) with little or no reduction in its effectiveness. The latter requirement is directly related to the so-called robustness of the system. This stipulates that the relative quality of the solution generated by the system, that is, cost and service level, should be independent of the specific environment, the variability of the data or the particular setting.

Some of the contributing authors in *Logistic Systems* [LC05] state that: *'Logistics decisions may be divided or grouped in several dimensions based on various criteria. The common grouping into strategic, tactical and opera-*

tional levels may be based on one or more of the following criteria associated with the decisions: the time frame, the resource requirements or the level of managerial responsibility'. Routing decisions are usually classified as strategic level decisions, even though *'in reality the range of decisions may be better viewed as a continuum on all dimensions'* [LC05].

It is also noted in *Logistics Systems* [Cor05] that *'strategic decisions have a significant impact on long-term profitability and do not recur frequently, hence justifying the use of sophisticated analytical and simulation models. On the other hand, operational decisions tend to recur on a daily basis, or even more frequently for that matter, so that the main concern is in having algorithms which yield consistently good solutions quickly. This therefore points the way to the development of efficient heuristics, given that most problems in this category are combinatorial. As for tactical models, they lie in-between strategic and operational models in importance and characterizing an ideal algorithm for them depends on specific circumstances, particularly execution frequency'.*

2.5.2 Deadline VRP

The deadline vehicle routing problem (DVRP) has the following characteristics as described by Nygard, Greenberg, Bolkan and Swenson [NS88]. There is a fleet of vehicles, all housed at the depot, and each has the same capacity. There is a set of stops with known demands and time deadlines (latest allowable arrival times). The goal is to find the shortest possible set of tours for which no deadlines are violated. Without the latest possible arrival times defined, the problem is simply the standard VRP.

The DVRP is solved using the generalized assignment approach. Stops are first clustered using the generalized assignment approach and then routes created using the traveling salesman approach. Successive solutions of the generalized assignment problems (GAPs) with altered cost coefficients are used to find routes which meet the time deadlines. Two post-processors are used to polish up the routes in terms of the deadlines.

The DVRP is the same as the VRPTW except that no earliest arrival time is defined, only a latest time. The generalized assignment approach outlined below (where each vehicle can have a different capacity) is a type of 'cluster-first route-second' heuristic, also known as a 'cluster-first vehicle-second' heuristic.

The objective is to minimize:

$\sum_{k\epsilon K} \sum_{j\epsilon J} c_{kj}x_{kj}$

Subject to:

$\sum_{j\epsilon J} r_{kj}x_{kj} \leq b_k \ \forall \ k \ \epsilon \ K$

$\sum_{k\epsilon K} x_{kj} = \mathbf{1} \ \forall \ j \ \epsilon \ J$

$x_{kj} = \mathbf{0 \text{ or } 1} \ \forall \ k \ \epsilon \ K, \ j \ \epsilon \ J$

$$x_{kj} = \begin{cases} 1 & \text{if stop } j \text{ is assigned to vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

Where K is a set of vehicles each of capacity b_k and if stop j (from the set J) is assigned to vehicle k it consumes r_{kj} units of capacity.

The cost coefficient c_{kj}, where

$$c_{kj} = \textbf{Distance (depot, stop } j\textbf{) + distance (stop } j\textbf{, seed } k\textbf{) - distance (seed } k\textbf{, depot)}$$

is used to measure the desirability of assigning stop j to vehicle k. It is only after the TSP step is complete that route quality can be assessed. *'The primary challenge of the approach lies in choosing cost coefficients that translate the linear objective function of the GAP into the highly non-linear TSP tour creation process'* [NS88]. c_{kj} can be regarded as the extra distance traveled by vehicle k due to the addition of stop j. Generally a seed point is chosen for the group of stops for each vehicle.

A suggestion for applying this technique to the ARPNDD case would be as follows: cluster the customer points geographically, with each customer repeated as many times as they are serviced. Within each cluster create up to a maximum of five routes (one for each weekday). For the single drop per week customers, set their latest delivery time to the end of the day on Friday. The customers with two drops per week need two latest arrival times specified. A possible approach would be to set the first time to the end of Tuesday or Wednesday and the second to the end of Friday. This will not

necessarily ensure the stops are spaced over the week but a check can be built in to prevent consecutive visit days for these clients. All other stores have to be serviced on the days specified in their service schedule. The cluster-first route-second approach in Section 3.2 applies a similar methodology.

2.5.3 Probabilistic VRP

In probabilistic VRP's (referred to as the PrVRP in this paper to distinguish it from the periodic VRP) a subset of potential customers is chosen to be served by some probability law. The paper *The probabilistic vehicle routing problem* [JO88], describes the case where it is of interest to find a predefined (minimum length) tour through a set of points. If the tour visits n points but only m ($m < n$) customers need to be visited, then the route will visit the m customers in the same predefined order and simply skip the remaining $n-m$ customers. An example given of a probability law which could be used to select the subset of customers is: let each customer have a probability p_i of requiring a visit independent of any of the other customers (and independent of other days).

The pre-defined tour is actually an expected minimum length tour, where the expectation is computed over all possible instances of the problem. Given an a priori tour t, if problem instance k occurs with probability α_k and will cover a distance $r_{t,k}$ then this problem instance will receive a weight $\alpha_k r_{t,k}$ in the computation of expected length. The problem is to find a tour t^* which minimizes the expected value of the random variable L_t (the tour length), i.e. $E[L_t] = \sum_k \alpha_k r_{t,k}$.

The expected number of customers to be serviced on any given day is $W = \sum_{i=1}^{n} p_i$. A TSP tour may not be the answer because when certain customers are skipped it may not remain 'well behaved'. Also, $E[L] \leq E[L_t]$, i.e. the value calculated above is a tight upper bound on the quantity of interest since once it is known which customers will be served, a new tour can be found if desired. Once $E[L]$ has been found, restrictions imposed by vehicle capacity constraints etc. can be incorporated by solving essentially a route-first cluster-second problem.

A 'giant a priori' tour will be designed and each time a vehicle runs out of capacity and needs to be sent to the depot, this will be regarded as a separate route. The giant tour will be broken up into clusters of customers in this way. The results presented in the paper are of little use to the ARP-

NDD problem at hand even if the probability of being served is calculated from a customer's service level agreement. Although the assumptions (that each customer has demand of 1 with probability p, independent of the other customers, i.e. a binomial probability law) are reasonable, the method allows for only one vehicle size. A flaw in the model occurs if the vehicle capacity is changed. Even if a larger vehicle is used, the tour may increase because of the point along the tour at which the vehicle runs out of capacity.

A heuristic solution, suggested by the authors Jaillet and Odoni [JO88], involves the following approach. A nearest neighbour (greedy) algorithm is used. In order to find the next customer to visit, the customer with the shortest expected distance from the current point must be determined. The computational effort to determine this for the PrVRP is much greater than that of the corresponding deterministic problem.

2.6 Clustering Techniques

Data clustering is a common statistical analysis technique used in many fields and is defined by Wikipedia [Wik07a] as *'the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often proximity according to some defined distance measure. An important step in any clustering is to select a distance measure, which will determine how the similarity of two elements is calculated. This will influence the shape of the clusters, as some elements may be close to one another according to one distance and further away according to another.'*

Cluster analysis enables customer groupings, which are visually fairly easy to find, simpler to find automatically. Different proximity measures have been found to work for different types of customer distribution. Barreto et al [BPS06], conclude that the best techniques are those that produce consistently good results under many different scenarios. They use the proximity measures in Table 2.6 to cluster customers.

For the ARPNDD case, proximity measures based on distance alone are not sufficient; the problem needs to be considered in higher dimensions. For each customer (point to be clustered) in the current problem, there are 12 variables; longitude, latitude and the time at customer and size of order for each week day (for those customers where the delivery day needs to be de-

Measure	Formula
Single linkage (nearest neighbour)	$d(A,B) = min_{I \epsilon A, J \epsilon B} d(I,J)$
Complete linkage (farthest neighbour)	$d(A,B) = max_{I \epsilon A, J \epsilon B} d(I,J)$
Group average	$d(A,B) = [\sum_{I \epsilon A, J \epsilon B} d(I,J)]/[\|A\|\|B\|]$
Centroid	$d(A,B) = d(m_A, m_B)$
Ward	$d(A,B) = SEQ(A+B) - SEQ(A) - SEQ(B)$
Saving	$d(A,B) = min_{I \epsilon A, J \epsilon B}[min(\alpha, \beta)]$ $\alpha = d(I,J) + d(K,L) - d(I,K) - d(J,L)$ $\beta = d(I,J) + d(K,L) - d(I,L) - d(J,K)$

Table 2.6: Comparison of proximity measures where m_A is the centroid (gravity centre) of a group of customers and *SEQ* is the sum of squares error of a group. SEQ(A) = $\sum_{I \epsilon A}[d(I, m_A)]^2$ and $m_A = (\frac{\sum_{I \epsilon A} x_i}{|A|}, \frac{\sum_{I \epsilon A} y_i}{|A|})$.

cided by the model, these are an expected time and mass). Calculating the correlation between customers based on the time and mass variables is not appropriate as it would typically be in a clustering analysis. The location of customers is still the strongest factor to consider when clustering so it may be suitable to apply higher weightings to these variables as suggested in *Cluster Analysis* [Eve74].

Other methods which rather apply a top-down approach to segregating the customers have also been investigated. Daganzo and Newell [ND86], in their paper which discusses routing in a ring-radial network (similar to that in Figure 3.1(c)), carry on the work done by Daganzo in earlier years where he investigated very simple manual tour construction. Not only were these manual methods efficient but they also resulted in tours with lengths only a few percent larger than those obtained by lengthy computer algorithms. Both the authors believe that *'the key to any detailed routing to minimize the cost of delivery (by hand or computer) is first to partition the region into zones in which individual vehicles make deliveries'*. They discuss various shapes and orientations of delivery zones with different road-network design schemes.

Very simply, a fleet of identical vehicles each carrying at most C items per day with items being dropped off at each visit is considered. The local delivery destinations are assumed to be on a fine road-network grid whereas

there is a hierarchy of widely spaced faster roads which are used to gain access to the smaller delivery zones. *'If there is a (nearly) continuous metric in the space, it seems clear that, for any efficient strategy, the C points in each group could be imbedded in ...a delivery zone, so that these zones together form a partition of the whole delivery region. If the location of the delivery points changes from day to day, the optimal zones would also change, maybe not very much for $C \gg 1$, but one may find it more convenient to keep the delivery zones (nearly) fixed if the cost of such a strategy is sufficiently close to the minimum'* [ND86].

In the ring-radial setup considered, the optimal zone shape turned out to be rectangular and elongated towards the source (depot). Nearer the source wedges are preferred. This had the effect of reducing the 'line-haul' cost of actually reaching the delivery zone. The size of the rectangles was also allowed to vary from day to day so that each zone would contain approximately C points or the capacity of the vehicle could be assumed sufficiently flexible to service a fixed size rectangle with an expected value of C points.

Region partitioning heuristics are usually designed to divide the customer set into subregions with no more than q customers each. Two such heuristics are those of Marchetti-Spaccamela [MSS84] and Karp [Kar77]. Marchetti-Spaccamela proposes a polar region partitioning (PRP) methodology similar to Figure 3.1 (c) except that the arcs do not join to form concentric circles. Beginning with a circle centred at the origin, it gets partitioned with circular concentric arcs (substituting for horizontal cuts) and radial lines (substituting for vertical cuts). Once this has been done, the objective is to minimize the maximum route length while ensuring all routes have approximately the same length. If the problem requires delivery days to be assigned to customers, who are visited with different frequencies, then this methodology becomes harder to apply.

Karp describes a rectangular region partitioning (RRP) scheme. The algorithm proceeds by dividing the original rectangle containing all the points into subrectangles with at most q points each. Let Y be a rectangle (with its longer side horizontal) containing m cities. x is then chosen to be the $m/2^{th}$ city from the left edge of Y. A vertical cut through x subdivides Y into a left and right rectangle with x on the common boundary. From then the rectangle is cut further by a 'cutting game' process. Two 'players' each taking turns apply a short and a bisection strategy repeatedly. The short strategy involves choosing a cut parallel to the shorter side and the bisection strategy places a cut so as to divide the rectangle into equal halves. The

technique can be applied practically to large problem sizes. In many cases, the single objective of having the same number of points in a region is not sufficient as each point will have a different delivery duration and drop size.

2.7 Genetic Algorithms

A genetic algorithm (GA), as defined in Wikipedia [Wik07b], is a *'search technique used in computing to find exact or approximate solutions to optimization and search problems'*. Genetic algorithms are categorized as global search heuristics and are a particular class of evolutionary algorithms.

2.7.1 Introduction

Genetic algorithms are finding their way into many areas where traditional optimization methods frequently produce less than satisfactory results. Based on Darwin's evolutionary theory and natural genetics, they are robust search tools. GA's are often used in conjunction with other optimization techniques:-

results from runs using existing optimization methods can be used as the initial population input into a GA, or

once a GA simulation has stopped, solution searching can proceed with other heuristics such as hill climbing for example.

With GA's, as is mentioned in the *Handbook of Genetic Algorithms* [Dav91], we are not optimizing directly but creating conditions, as in the natural world, for optimization to occur.

In order to apply a GA to a routing problem, a coding (usually using bit strings) of possible customer sets must be determined as described in *The Design of Innovation* [Gol02]. In addition to *'guiding the evolution of future generations'*of solutions, an objective function must be used to evaluate the current solutions' quality or 'fitness to purpose'. This objective function will measure the total distance of the routes and the time taken. The initial population would best be derived from the existing routes in operation at a site. Various known types of selection (reproduction), mutation and recombination (crossover) operators would need to be tried to determine which combination will work best 'in concert' to evolve future generations of solutions.

Goldberg [Gol97] shows how selection and mutation combine to provide continual improvement to the search technique whilst selection and recombination work together to produce more innovative results. He also points out that *'continuing to experiment in a local neighbourhood is a powerful means of improvement, although it will have a tendency to be fairly local in scope, unless a means can be found for intelligently jumping elsewhere when a locally optimal solution is found'*. So, without recombination, we really just have a hill climbing technique. Fundamental to the understanding of GA's is the notion of de-composing the problem into building blocks and reassembling them to form powerful solutions at the end. The decision making among different building blocks is very statistical in nature. The strongest building blocks will compete and must achieve a higher 'market share' in order for the 'survival of the fittest' concept of GA's to carry through to future generations of solutions.

2.7.2 GA's in the literature

A recent paper [BP02] describing a problem very close to the ARPNDD uses a GA. This problem is known as the DPSS - distribution planning with stochastic demands and scheduled deliveries. Unfortunately very little detail is given about the GA, making it difficult to test on a set of customers with deterministic demands. The formulation does however allow for sleep-outs to occur but not second loads on a vehicle. These are both very real situations experienced in the trucking and distribution industry and are seldom catered for in the theoretical versions of logistics problems.

'Recent business and technological trends have transformed the structure and performance requirements for distribution channels in many industries. Higher service level expectations of retail customers, distribution outsourcing by manufacturers, and the proliferation of advanced information technologies drive these transformations, presenting new problems in supply chain management' [BP02]. The authors, Balakrishnan, Geunes and Pangburn, attribute this distribution channel transformation, that has *'altered the ground rules for providing competitive distribution services'* to:

- Customers and retailers are raising their suppliers' service expectations. This often has the effect of requiring more frequent deliveries of a wide variety of products.
- Manufacturers are increasingly outsourcing distribution. Third party logistics suppliers are forming strong partnerships with retailers so that

both parties can concentrate on their own areas of expertise. Logistics companies are now increasing their service offering beyond just warehousing and transportation.

- Information technologies are providing more timely and detailed supply chain data. Information can readily be used and exchanged by many users simultaneously to facilitate better decision making.

Supply chain management researchers have identified many modeling opportunities for the challenges associated with these new trends. *'For instance, studies of optimal buyer-supplier contracting schemes specify appropriate service level requirements, revenue-sharing incentives, and delivery and payment terms'* [BP02].

As a result of this research, the authors have developed a model to deal with the DPSS problem which *'focuses on improving distributor delivery operations and capacity planning'. 'In order to consistently meet delivery commitments, distributors create fixed delivery routes that do not change from week to week'* [BP02]. Their model takes into account demand variability and its associated costs, including overflow, to ensure distributors consistently meet delivery obligations. The model is unique in that:

- Store deliveries must follow a strict schedule - periodic deliveries on the same day each week.
- When demand is high, the distributor can use contingency resources to make deliveries.
- Delivery calendaring is used to assign truck routes to different days of the week (since the same truck can serve different routes on different days).

To deal with demand variability inherent in such a situation, two mutually exclusive approaches can be adopted - safety (buffer) and contingent capacity. Safety capacity ensures that the average demand of customers grouped onto a truck is less than that truck's capacity. This results in lower space utilization of the vehicles. Contingent capacity allows for any excess orders above the average demand for a group on a truck to be 'off-loaded' and delivered using hired vehicles which are more expensive.

'Due to demand variability, only the fixed regular costs for a given distribution plan can be determined with certainty; activity-dependent costs and

even the fixed portion of overflow costs will not be known until demands are realized' [BP02]. The model is structured in such a way that regular routing and overflow costs can be determined independently for each group of customers. If overflow is experienced, additional routing decisions will need to be made; which customers are removed from the regular routes and in what sequence the remaining customers are visited?

Many questions begin to arise when only expected demand can be modeled. Should stores with stable demands be grouped with those that are highly variable to allow for a compromise on total quantity? Another interesting question posed by the authors is: *'should stores near the distribution centre be dispersed across routes to serve as convenient stores to remove from routes when overflow occurs?'* [BP02]

The model inputs are:

- Set of customers with
  - Delivery locations
  - Delivery frequencies
  - Demand distributions for each customer

Note: the model assumes that each customer is serviced just once a week but if this is not the case then duplicates of each customer can be created.

The model outputs are:

- Number of vehicles required
- Groups of stores with a delivery day
- A truck associated with each group

The following notation was used in the paper:

Let i=1...n index the n customers to be serviced
Let k=1...K denote the K trucks (over all truck types) needed to serve the n customers *Note:* the trucks can have different capacities and costs
Let F_k denote the fixed cost (per week) for truck k

Let B_k denote the capacity of truck k

Let g=1 $\ldots m$ denote all feasible groups of customers
(A group of stores is feasible if restrictions on maximum tour duration are adhered to, as well as other requirements such as a need for two proximate customers to be serviced together.)

Let $G(i)$ be the set of all groups g that include store i.

Let C_{gk} and D_{gk} denote the expected regular routing cost and expected overflow cost to serve group g with truck k.

Let $t = 1 \ldots T$ index the available delivery days where T denotes the number of days in the week that customers accept deliveries.

The following binary index variables are used:

Group selection:

$$y_g = \begin{cases} 1 & \text{if we select group } g \\ 0 & \text{otherwise} \end{cases}$$

for all $g = 1 \ldots G$

Group-truck-day assignment:

$$x_{gkt} = \begin{cases} 1, & \text{if we assign group } g \text{ to truck } k \text{ on day } t \\ 0 & \text{otherwise} \end{cases}$$

for all $g = 1 \ldots G$, $k = 1 \ldots K, t = 1 \ldots T$

Truck selection:

$$z_k = \begin{cases} 1 & \text{if we acquire truck } k \\ 0 & \text{otherwise} \end{cases}$$

for all $k = 1 \ldots K$

The DPSS problem is stated as:

Min $\sum_{k=1}^{K} F_k z_k + \sum_{g=1}^{m} \sum_{k=1}^{K} (C_{gk} + D_{gk}) \sum_{t=1}^{T} x_{gkt}$

Subject to:

Customer assignment: $\sum_{g \epsilon G(i)} y_g = 1,\ i = 1 \ldots n$
Group assignment: $\sum_{k=1}^{K} \sum_{t=1}^{T} x_{gkt} = y_g,\ g = 1 \ldots m$

Truck assignment:	$\sum_{g=1}^{m} x_{gkt} \leq z_k$, $k = 1 \ldots K$
Integrality:	$x_{gkt}, y_g, z_k \in \{0, 1\}$, $\forall\ g, k$ **and** t.

The objective function minimizes the total fixed, activity dependent, regular and overflow costs. The customer assignment constraints specify that each customer must be assigned to exactly one group. The group assignment constraints relate the group selection decisions to the assignment of groups to truck-day combinations. The truck assignment constraints serve both as truck-scheduling restrictions and truck-selection forcing constraints.

Each truck can only be assigned to a single group on any one day. Customer delivery day preferences can be incorporated by including only the relevant assignment variables. If a customer in group g cannot accept delivery on day t then we omit variable x_{gkt} for all truck indices k, which reduces the size of the formulation.

The authors developed a *'genetic algorithm that iteratively generates distribution plans (store groupings and group-truck-day assignments)'* [BP02]. The algorithm takes as input the desired number of trucks (i.e. no fleet sizing is done). *'By changing the fleet size and reapplying the GA, we can determine the number of trucks needed to determine the lowest-cost solution'* [BP02]. Truck-day combinations are considered instead of group-truck combinations. Truck-day combinations are indexed from $j = 1$ to KT; starting with the first day for the first truck. Thus the index j corresponds to using truck $k = \lfloor j/T \rfloor$ on day $t = j \bmod (T)$ (KT also serves as an upper bound on the number of groups). The genes (candidate solutions) are represented as integer vectors V of length n. The ith position of this vector (ith allele) can take any integer value $v(i)$ between 1 and KT. This value represents the index of the truck-day combination assigned to store i. All stores assigned to the same truck-day combination constitute a group. For routes covering multiple days, day t represents the starting day of the route. The fitness value of a gene represents the total costs - including penalty costs if constraints are broken (e.g. assigning a truck to a route on Wednesday when the truck is still busy completing a 2-day route initiated on Tuesday). In the DPSS context the typical GA operations are defined as follows:

> Mutation - randomly changing the value of $v(i)$ to another in the allowable range.
> Crossover - interchanging a range of allele values between pairs of randomly chosen genes.

In their implementation, the authors assumed that customer demands were independently and normally distributed (mean and variance calculated from historical data). The GA applied was fairly standard and not fine-tuned to the problem case because the goal of the project was to accurately estimate delivery costs for the 67 customers with variable demand. Heuristic rules were used to select visitation and offloading sequences once the customers had been grouped into clusters geographically. The model is useful to quantify the cost that demand variability incurs. It was found that costs rose fairly steeply as the coefficient of variation of demand increased. It was also found that master routes determined using static demands strive for high truck utilization and thus perform poorly under stochastic conditions when overflow occurs.

A few researchers who have applied GA's to the VRPTW have managed to find solutions which have proven to be the best known answers so far on test cases in the literature. Some of these techniques mentioned could possibly be developed further to adapt them to the ARPNDD problem. One such paper is that by Thangiah [Tha95] which describes a method known as GIDEON for solving the VRPTW using GA's. GIDEON consists of two main steps working in synergy:

1. a global clustering method using GA's adaptive search strategy
2. a local post optimization method to improve the best solution found in step 1

'GIDEON is a cluster-first route-second method that assigns customers to vehicles by a process we call Genetic Sectoring. This process uses a GA to adaptively search for sector rays that partition the customers into sectors or clusters served by each vehicle. The solution quality is based on minimizing the number of routes followed by the distance and route time. That is, a solution with M number of routes is better than $M+1$ routes, even if the distance and route time for the M routes is greater than the $M+1$ routes' [Tha95]. Customers are moved between clusters until all the routes are at least feasible using the post optimization procedure.

The post optimization procedure uses the $\lambda-$interchange local method. Given a solution to the problem represented by a set of routes $S = \{R_1, ..., R_p, ..., R_q, ...R_k\}$. The interchange exchanges or shifts a subset of customers $S_1 \subseteq R_p$ with another $S_2 \subseteq R_q$ of size $|S_i| \leq \lambda$ between the pair of routes. Two new routes $R'_p = (R_p - S_1) \cup S_2$ and $R'_q = (R_q - S_2) \cup S_1$ and

a neighbouring solution, $S' = \{R_1, ..., R'_p, ..., R'_q, ...R_k\}$ are thus formed.

The mixed integer formulation uses the indices $i, j = 1, ..., N$ and $k = 1, ..., K$. The following indices and variables are used:

K = number of vehicles

N = number of customers (0 denotes the central depot)

T = maximum travel time permitted for a vehicle

C_i = customer i, C_0 = the central depot

V_k = vehicle route k

O_k = total overload for vehicle route k

T_k = total tardiness for vehicle route k

D_k = total distance for a vehicle route k

R_k = total route time for a vehicle route k

Q_k = total over-route time for a vehicle route k

t_{ij} = travel time between customer i and j (proportional to the Euclidean distance)

v_k = maximum capacity of vehicle k

t_i = arrival time at customer i

f_i = service time at customer i

w_i = waiting time before servicing customer i

e_i = earliest time allowed for delivery to customer i

l_i = latest time allowed for delivery to customer i

q_{ik} = total demand of vehicle k until customer i

r_{ik} = travel time of vehicle k until customer i (including service time and waiting time)

p_i = polar coordinate angle of customer i

s_i = pseudo polar coordinate angle of customer i (see paragraph below)

F = fixed angle for Genetic Sectoring, $Max[p_i, ..., p_n]/2K$, where $n = 1, ..., N$

B = length of the bit string in a chromosome representing an offset

P = population size of the Genetic Algorithm

G = number of generations the Genetic Algorithm is simulated

E_k = offset of the kth sector, i.e. decimal value of the kth bit string of size B

I = a constant value used to increase the range of E_i

S_k = seed angle for sector k

S_0 = initial seed angle for Genetic Sectoring, $S_0 = 0$

α = weight factor for the distance

β = weight factor for the route time

η = penalty weight factor for an overloaded vehicle

γ = penalty weight factor for exceeding maximum route time in a vehicle route

κ = penalty weight factor for the total tardy time in a vehicle route (getting to a customer late)

Variables:

$$y_{ik} = \begin{cases} 1 & \text{if } i \text{ is serviced by vehicle } k \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{if the vehicle } k \text{ travels directly from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

The objective function used by GIDEON is:

Min $\sum_{ijk} c_{ijk} x_{ijk}$

where $c_{ijk} = \alpha t_{ij} + \beta(t_i + f_i + t_{ij}) + \eta.max\{0, (q_{ik} - v_k)\} + \kappa.max\{0, (r_{ik} - l_i)\} + \gamma.max\{0, (t_i + f_i + t_{ij} - T)\}$

By subdividing a chromosome (represented by a bit string) into K sectors (clusters) of B bits the sector size can be determined. The customer angles

p_i are replaced by pseudo polar coordinate angles s_i by normalization so that the angular difference between any two adjacent customers is equal. Seed angles S_k are placed in the search space with $S_0 = 0°$. Rays are then drawn to each seed angle from the origin and the customers are thus separated into clusters with boundaries falling freely between them. The seed angles are calculated using a fixed angle and an offset. Each chromosome represents a set of offsets. If a fixed angle plus its offset exceed 360° then the seed angle is set at 360° and fewer than K vehicles are thus used for that set of routes.

The customers within each sector are obtained from the chromosomes and routed using the cheapest insertion method. The fitness value for the chromosome is calculated using the objective function above. Those chromosomes with the lowest cost stand the highest chance of being included in successive generations. The genetic sectoring (a meta-search strategy) forms the sectors by exploring the search space and the post optimization procedure gives adjacency information about the clusters back so that information can be exploited.

The technique whereby customer groupings are created using radial lines from a centre point is fairly common in fixed routing exercises. For a problem like the ARPNDD, with multiple objectives and many factors for each customer this method is not appropriate. The authors in the three papers which have been discussed have laid a good foundation for future work in using GA's to solve logistics problems. Like many solution techniques, the parameterization of the GA is the key to the solution quality but also the stumbling block in terms of progress.

GA's in their traditional form do not make use of all the problem information which makes them simpler to use but may put them at a disadvantage in terms of power and appropriateness. Advanced methods have been devised to include problem-specific information. With a routing problem like the ARPNDD, it is usually desired to reduce costs by minimizing total distance traveled and time on the road while maximizing vehicle utilization and minimizing the number of vehicles used. It is not always possible or wise to reduce an optimization problem to a single criterion or objective function. Multi-objective or multi-criteria optimization problems have also been solved using GA's. Essentially this involves the use of the concept of Pareto optimality where all non-dominated points are candidate solutions. These more advanced techniques would be more difficult to test and adapt to the ARPNDD.

The seminal papers discussed in this chapter do not use techniques which are robust or complex enough to handle many different service combinations or the requirement to keep customers in the same delivery group over the period. The problem case to be examined here requires some customer visit days to be decided on by the model and others not. Also, for customers serviced less than five times per week, rules governing the spacing of these visits apply. These types of complexities make the balancing of the workload over the week even more challenging. The difficulty of the problem is directly related to the number of customers with fixed delivery days and the number serviced more frequently than once per week. Most of the remaining methods in this literature review would require a fair amount of modification to make them suitable for the ARPNDD. It is felt that the test case discussed in this dissertation encapsulates many of the requirements of companies seeking solutions to ARPNDD type problems. So by seeking a solution method for this case, most other variations of the problem could be handled.

Chapter 3

Method

The main contribution of this research is to investigate means to find practical solutions to the ARPNDD problem. Four approaches have been be explored:

1. Using a semi-manual iterative 'route-first cluster-second' method with the help of commercially available routing software.

2. Using a semi-manual iterative 'cluster-first route-second' method with the help of commercially available routing software.

3. Running the problem through a mixed integer programming tool with the objective function and constraints as defined.

4. Applying a clustering technique that assigns customers to delivery groups (clusters) as well as delivery days (where necessary). An objective function balances the tradeoff between minimizing distance traveled and smoothing the workload (time and load size) per cluster per day. Finally, the results of the allocation algorithm will be input into routing software to sequence the stops on the most appropriate vehicle.

Each method will be described in Section 3.1 to 3.4 with detailed steps of each heuristic supplied in Appendices 2 to 5 respectively. The features and results of each model will be compared in Chapter 4.

A similar exercise has already been performed for the client using only the route-first cluster-second method. The data used for this exercise (and reported in Section 3.1) was from November 2005. The sales data used in the comparison reported in Section 4 is from 30 January to 11 March 2006. The main difference between the original and the new simulation which will be

run is: there will be a daily weighting for different drops, i.e. if a customer orders 900kg's of product per week (M,W,F) these will not each be 300kg stops, but rather weighted in such a way that the Monday and Friday orders are larger for example.

3.1 Route-first Cluster-second heuristic

The first method involves using existing commercial routing and scheduling software, *FLO* (Fleet Logistics Optimizer) [Sys06b]. *FLO* has a powerful algorithm which handles a number of exceptional constraints and gives the user control over how the algorithm operates (for more detailed information about how *FLO* works see Appendix 1). *FLO's Delivery Groups* concept enables it to be used for fixed/master routing (see Section 1.1). If a customer is not placed in a delivery group then that customer can be placed on any route with any other customers (i.e. it belongs to all delivery groups). There is also the option of inputting multiple delivery groups for a customer. The order of stops on a route will not be fixed and will change depending on what subset of customers in a delivery group have jobs on a particular day. If all points in a delivery group are serviced, this is termed a regular route. If only some are visited, this is known as an invoice-driven route. Most of the routes created for this research project will not be regular routes in this sense because customers in the same delivery group need not be serviced the same number of times per week/planning period.

The algorithm, described in detail in Appendix 2, creates routes on a day-to-day basis for a period of one week but can naturally be extended for longer periods. The main concepts are summarized here:

Firstly, for all customers with two or more drops per week, multiple copies of each customer are made corresponding to the number of times which they are serviced. Each copy of each customer is given a unique code; the customer number with the day code appended (Monday = 1 through to Saturday = 6). Each copy of the customer represents a job (with an average job size in kg) on the corresponding day.

Those customers who must be visited twice a week pose the greatest complication to the model. In Table 1.1 six types of day-combinations exist for these customers: Monday & Thursday, Tuesday & Thursday, Tuesday & Friday, Wednesday & Friday, Saturday & Tuesday or Saturday & Wednes-

day. Part of the model output is the assignment of these customers to one of the day-combinations. It must be noted that the 2-day per week customers cannot necessarily be allocated any of the 2-day per week schedule combinations. For some, the model can only choose from three or four of the options. The customer codes work as follows in this case: if the customer can be on schedule type Tu, Sa or W, Sa for example, the customer codes will be customer number-23 and customer number-6. This indicates that a choice must be made between Tuesday (day 2) and Wednesday (day 3) for this customer's first drop in the week. The second drop on Saturday (day 6) is certain. Single drop customers have code: customer number-12345. These are catered for after all five day's worth of routing of other jobs is complete and they go a long way in ensuring even fleet utilization over the week.

Those customers with the most stops per week (and hence the least flexibility in terms of which days to service them) are allocated to days first. Starting with the first day on which the schedule is run, these customers form delivery groups and act to attract customers with two drops per week. If the two drop per week customers fit well on a route with those more frequently visited customers they get absorbed onto a route. If they do not, they re-enter the mix of unassigned customers. Gradually more and more customers get assigned to a delivery group so that when the last day has been run all customers have been scheduled.

Very little manual intervention in the form of changes to the routes should be required at any stage in this process. After the initial set of routes had been run, a representative of the client examined them in considerable detail. He mainly looked at the qualitative aspects of the routes and his input could be assessed in monetary terms through the change in route costs once the routes were manually overridden in *FLO*. Certain 'constraints' that had not been mentioned before also came to light e.g. restricting the number of chain stores (known for their long queue time) on any route/delivery group. The concept of an anchor store is very prevalent amongst people in the logistics industry but they are also usually wary of too many such stores. Experience has shown that long delays are often experienced which result in the vehicle being tied up for many hours. This causes a high standard deviation for the waiting times for these customers. To try and compensate for such idle time, average waiting times at such stores were increased by 10%.

3.1.1 Other complexities

After satisfactorily solving the problem using this route-first cluster-second heuristic it was realized that additional issues which hadn't been addressed early on needed to be looked at carefully in subsequent model attempts:

- How to include customers with a visit frequency of less than once per week. To be conservative, they can all be included to cater for the possibility that they all order in one week. Otherwise a representative subset of these customers must be chosen and included.
- How to deal with jobs which are larger than the largest vehicle. These jobs can either be split into two equal parts if their total mass is greater than one vehicle but less than two, or split into three if their mass is larger than two vehicles but less than three etc. Alternatively, large jobs can be split in such a way that vehicles are first filled with part of a single order and the balance gets carried over to another vehicle, e.g. if an order for 10 tons is received and the largest vehicle has a capacity of 8 tons, 8 tons will be put on one vehicle and 2 tons on another. Jobs that have been split because the load was too large for a single vehicle must be grouped together to ensure that they go on routes with the same code.

What distinguishes acceptable routes from others often comes down to a comfort factor; routes that 'look nice' to a planner are 'do-able'. Often, what people don't realize is a route that crosses itself for example is not necessarily a bad route. To minimize travel distance and still meet the delivery windows while making optimal use of a truck's working hours may require a route to cross itself. This change of mindset is often very difficult to convey and hampers the progress of many routing studies. Something else to be careful of is allowing the user to change the model constraints and requirements so much that they really allow it only to mimic current practise and revert to existing routing solutions. It is usually worthwhile to get feedback from staff while setting up the model so that exceptional constraints are catered for early on. The purpose of the algorithm is to determine fixed routes that would be used repeatedly, every week, and vary only slightly depending on demand.

3.2 Cluster-first Route-second heuristic

While the first heuristic method described in Section 3.1 allowed for the clustering of customers into delivery groups once they had been routed, the

method which will be described here first groups the customers geographically and then routes them within these predefined groups. Different methods of dividing the customer set into geographic sections or clusters have been tried before (see Section 3.4). Other possible methods include routing in triangular/pie shaped segments (Figure 3.1(a)) (also known as sectorial region partitioning) and in rings (Figure 3.1(b)). Combining these two ideas results in a type of checkered pattern as shown in Figure 3.1(c). Also, since the earth's surface is already divided into squares by intersecting lines of longitude and latitude, it seems natural to consider delivery groups segregated by these lines. The method described here divides the customers in this way.

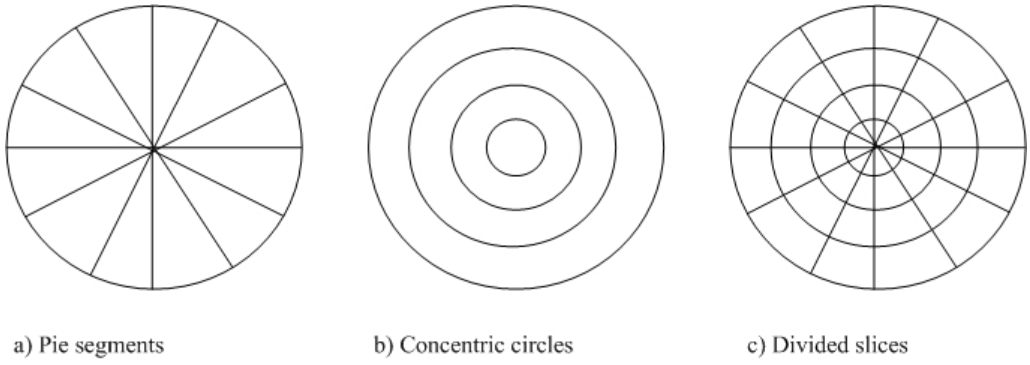


Figure 3.1: Methods of grouping customers geographically based on their dispersion patterns.

The process begins at the depot. If the stores to be visited over the period are not the most dense around the depot, then starting at the centroid of the full delivery zone may be more appropriate. Decide on a fraction of a degree which will be the width and length of each delivery group. If the customers become more sparse as the distance from the depot increases then it is more appropriate to have delivery groups with dimensions increasing accordingly. If tight, circular shaped routes are more suitable to the delivery operation in question, then blocks radiating out from the centre (either the depot or the centroid of the delivery points) of growing size should be used as shown in Figure 3.2. Merging delivery groups of different sizes is more difficult in this situation. Alternatively, petal shaped routing can be achieved with a pattern like that in Figure 3.3 where lines of latitude and longitude form rectangular areas which are more suited to merging. Such a pattern can be achieved using the method described in Appendix 3.

It must be noted that the segmenting of customers using this grid method

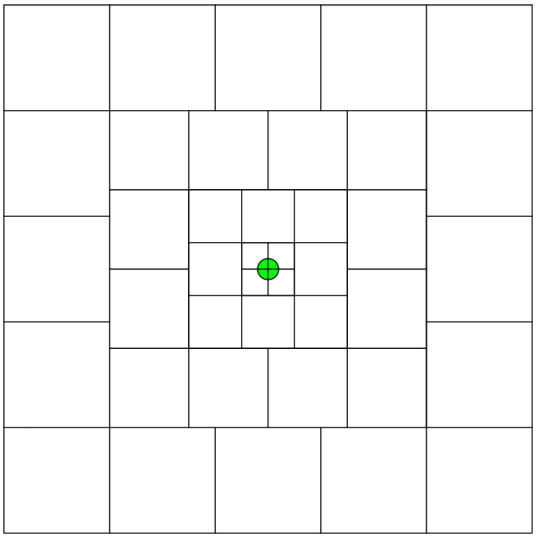


Figure 3.2: An alternative way of splitting the delivery points into routing areas or zones in a grid like pattern.

does not take into account order sizes so it may occur that more than one vehicle will be used to service a delivery group (customer in the same block in the grid). Care must be taken not to make the delivery groups too small to avoid:

- poor vehicle utilization
- more delivery groups than vehicles
- the need for excessive merging

When each delivery group in the newly formed grid is examined, it is not enough that a minimum number of customers in each block be met. The customers in each block must be grouped by the service schedule which they are on. For example, if there are 20 customers in a block and 19 are 1-time a week and 1 is a 6-times a week customer then two alternative decisions need to be made: either the entire block is merged with an adjoining one or just the 6-time a week customer is moved across. It may also be required to merge three or four blocks in some instances to make a delivery group viable. If the average delivery size for each customer is considered, it may be alright to have very few customers with frequent service schedules in a delivery group because their orders may be large enough to fill a truck.

The idea of using a grid is like laying a mesh over the delivery region to segment the area into intuitive delivery zones. A more formal approach to

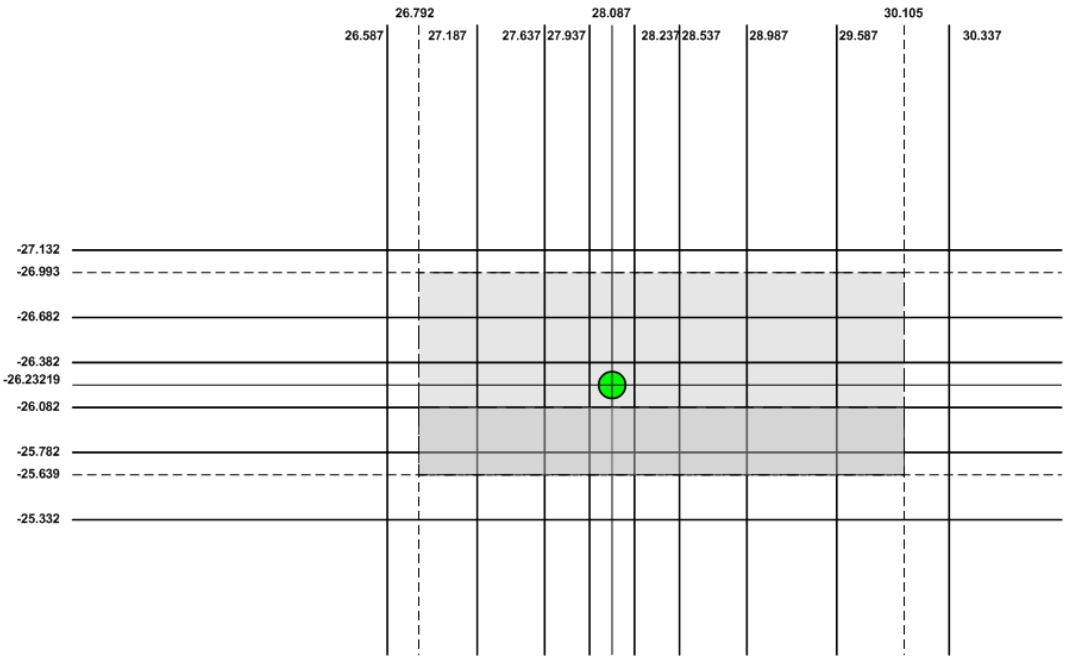


Figure 3.3: The graduation method of dividing the customers.

expand on this involves using established clustering techniques and this is described in Section 3.4.

3.3 Formulation using mixed integer programming

A mathematical formulation of the ARPNDD model is given below. A mixed integer programming tool is used to try and solve the problem. Many such tools are available on the market but *LINGO* [Sys06a] has been chosen to be used here. *LINGO* is a comprehensive tool for building and solving optimization models and performed well in a test done to compare complete global optimization solvers [NV05]. The software selects and invokes the most appropriate solver once the data has been read in. The ARPNDD problem is an example of an integer non-linear program which is one of the hardest types of formulation to solve.

The objective function and constraints in Table 3.1 were derived specifically for the ARPNDD problem.

Input variables:

R_k: cost/km for vehicle k

F_k: fixed cost per day for vehicle k

p_k: payload (in kg) of vehicle k

m_{it}: mass (in kg) of total delivery to customer i on day t

w_i: waiting time (in hrs) at customer i (queue and offload time)

S_i: delivery schedule for customer i

$$y_{it} = \begin{cases} 1 & \text{if customer } i \text{ is serviced on day } t \\ 0 & \text{otherwise} \end{cases}$$

Note: the y_{it} variables will be decision/output variables of the model only for customers who have not yet been assigned a set of delivery days. For the schedules defined in Table 1.1 (with Monday = day 1 through to Saturday = day 6), customers on schedules 4 to 8 have their y_{it}'s predefined.

If $S_i = 4$ then $y_{it} = 1$ for $t = 1, 3, 5$ and $y_{it} = 0$ for $t = 2, 4, 6$

If $S_i = 5$ then $y_{it} = 1$ for $t = 2, 4, 6$ and $y_{it} = 0$ for $t = 1, 3, 5$

If $S_i = 6$ then $y_{it} = 1$ for $t = 1, 2, 4, 5$ and $y_{it} = 0$ for $t = 3, 6$

If $S_i = 7$ then $y_{it} = 1$ for $t = 1...5$ and $y_{i6} = 0$

If $S_i = 8$ then $y_{it} = 1$ for $t = 1...6$

T_i: number of days customer i is serviced, where $\sum_t y_{it} = T_i, 1 \leq T_i \leq 6$

Output variables:

n_{kt}: number of customers/stops on vehicle k on day t, where $n_{kt} = \sum_i v_{ikt}$

$$z_{kt} = \begin{cases} 1 & \text{if vehicle } k \text{ is used on day } t \\ 0 & \text{otherwise} \end{cases}$$

$$v_{ikt} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to vehicle } k \text{ on day } t \\ 0 & \text{otherwise} \end{cases}$$

x_i: longitude of customer i
y_i: latitude of customer i

Co-ordinates of the centroid of each route (vehicle) on each day:
$\bar{x}_{kt} = [\sum_i v_{ikt}.x_i]/n_{kt}$

$\bar{y}_{kt} = [\sum_i v_{ikt}.y_i]/n_{kt}$

where $z_{kt} = 1$

Average distance from each customer location on a route to the centroid of that route (for $n_{kt} > 0$):
$\bar{d}_{kt} = [\sum_i v_{ikt}.DegToKm.|\sqrt{(x_i - \bar{x}_{kt})^2 + (y_i - \bar{y}_{kt})^2}|]/n_{kt}$

where $DegToKm$ converts degrees to kms.

Average distance of each route on each day:
$r_{kt} = 2(d_{0kt} - \bar{d}_{kt}) + (n_{kt} - 1).\bar{d}_{kt}$

where d_{0kt} is the distance from the depot to the centroid of the route for vehicle k on day t.
(see Figure 3.4)

Total offload time: $o_{kt} = \sum_i v_{ikt} w_i$

If a constant speed s is assumed then the average time of each route is:

$l_{kt} = [r_{kt}/s] + o_{kt}$

Objective function

Min $\sum_k \{R_k. \sum_t r_{kt} + F_k. \sum_t z_{kt}\}$

Subject to the following constraints:

Customer-vehicle-day assignment: $\sum_k v_{ikt} \leq 1$

Total service days: $\sum_k \sum_t v_{ikt} = T_i$

Vehicle-day assignment: $z_{kt} = \prod_i v_{ikt}$

Customer-day assignment: $y_{it} = \sum_k v_{ikt}$

Capacity constraint: $\sum_{i=1}^{n_k} \sum_t m_{it} v_{ikt} \leq p_k$

$\sum_k v_{ikt} = y_{it}$

$\delta\{\sum_i v_{ikt}\} = z_{kt}$

$l_{kt} \leq l$; **where** l **is the maximum length of a route in hours.**

Schedule-day constraints:
If $S_i = 1$ **then** $y_{i6} = 0$ **and** $y_{i1} + y_{i2} + y_{i3} + y_{i4} + y_{i5} = 1$

If $S_i = 2$ **then** $y_{i1} + y_{i2} + y_{i3} = 1, y_{i4} + y_{i5} = 1$ **with** $y_{i1} + y_{i5} \leq 1, y_{i3} + y_{i4} \leq 1$ **and** $y_{i6} = 0$

If $S_i = 3$ **then** $y_{it} = 1$ **for** $t = 6$ **and** $y_{it} = 0$ **for** $t = 1, 4, 5$ **and** $y_{i2} + y_{i3} = 1$

Similar constraints linking the $S_i's$ and $y_{it}'s$ can be derived for schedule day combinations which are particular to other problems.

The following constraints, particular to the ARPNDD, ensure that customers in a particular delivery group on one day must remain in that delivery group for all other days on which they are serviced.

If for $i < l$**,**
$\sum_j \sum_k (v_{ijk}.v_{ljk}) \geq 1$
then for $i < l$
$\sum_t y_{it}.y_{lt} = \sum_j \sum_k (v_{ijk}.v_{ljk})$

Table 3.1: Notation and constraints used in the formulation of the ARPNDD.

Once this process has been executed, an assignment of customers to vehicles on each day of the week is available. The algorithm above takes no account of the delivery windows of the customers and no sequencing of stops has been done. In order to ensure that each route is feasible, a TSP with time windows must be solved for each vehicle on each day. Manual changes to routes may need to be done once this step is complete if infeasible routes are detected. If too many routes are infeasible after the TSP step has been run then the model may need to be extended to allow for delivery windows to be

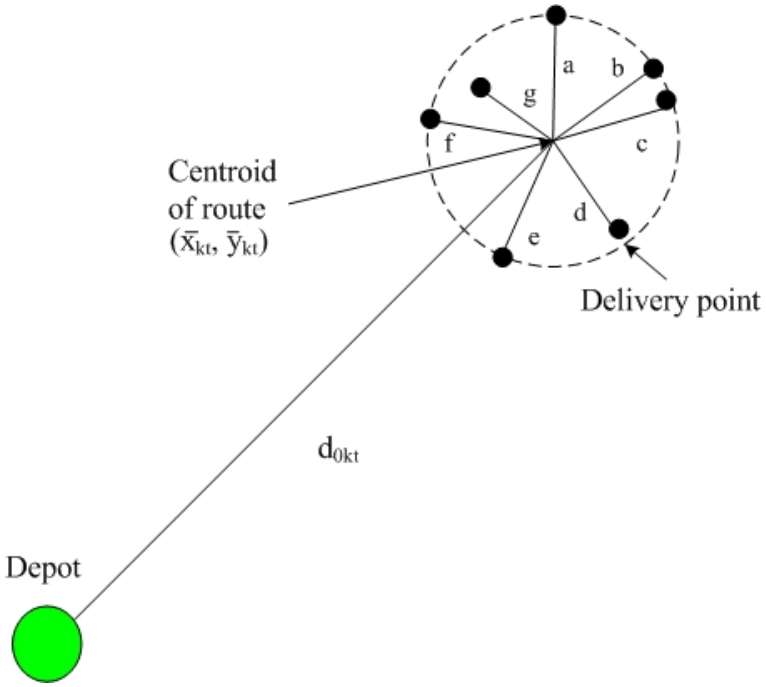


Figure 3.4: Graphical representation of the calculation of the average distance of a route, r_{kt}. $\bar{d}_{kt}$ is the average of the lengths a-g and d_{0kt} is the distance from the depot to the centroid of the route. (See the equations in Table 3.1)

part of the decision whether to add a customer to a particular delivery group.

Two different models (one non-linear and one linear version of the above formulation) are given in Appendix 4. The differences between the models and their respective features are described in Section 4.2. The following parameter settings were made in the models for all runs:

s = 57.5 km/h
DegToKm conversion = 110.5
DegToHrs=57.5/110.5 = 1.92174
l = 10 hrs

3.4 Cluster based approach

Since an exact solution (like the one attempted in the previous section) seems a long way off with current programming and processing power, heuristic

approaches will have to be pursued in order to find solutions to the ARPNDD problem. Heuristics have the advantage in that:

- good solutions can be found (even for large problems) in acceptable time
- several solutions can be compared and the best chosen based on the current scenario
- implementation is easy and modifications can be made because the system is easy to understand

After trying both a route first (tour partitioning) heuristic and a cluster first (region partitioning) heuristic (first two methods described in this chapter) it seems as though the region partitioning heuristics are more powerful for the fixed route ARPNDD problem.

A function has been written in *R* [Tea05] which assigns each customer to a cluster (delivery group) and allocates customers to delivery days where necessary. The details of the procedure are given below and the outline of the program structure is given in Appendix 5.

3.4.1 Method details

The algorithm is designed to build up clusters of customers for the period under consideration. A built-in clustering function is called and the customer with the highest average drop size in each cluster is chosen as a seed/anchor store and the remaining customers are then removed from the clusters. These customers are then added to these seed points (for each day the customer has a job), one at a time, based on different customer rankings. Each customer is hypothetically added to every cluster and those customers who do not have fixed delivery days are assigned to the most appropriate days (which may differ for each cluster tried). The most appropriate cluster for each customer is chosen with the objective of filling up routes (in terms of work time) and ensuring that no route has more product to deliver than the largest truck. Emphasis is usually placed on choosing the closest seed for each new customer which is added. Once different iterations (as a result of using the automatic clustering function with a random element, different customer rankings and different numbers of clusters) have been run, these are evaluated and the one with the lowest total travel distance and best utilization is returned.

Fleet information is taken into account in the postprocessing phase. It did not seem viable to include it in the model as a bin packing type of step would have to be incorporated to cater for this. It is important to note that the program automatically adheres to the rule of placing each customer in the same delivery group over the planning period.

Note that any customers who are 'single trip' deliveries are excluded from the simulation. Also, any customer who orders product which has a total mass larger than the largest vehicle has its job size reduced. Saturdays are excluded from the calculation (no customer receives orders on a Saturday only). Sleep outs and second loads are not explicitly allowed for but these may come out in the postprocessing performed in *FLO*. One disadvantage of using a clustering method like this to build up routes is that it does not consider the underlying road-network linking the customer points. This means that the distances calculated would not account for obstructions that cannot be driven through, such as large bodies of water.

Diagrammatically the process can be simply explained as follows:

Begin by clustering the customer delivery points using the k-means clustering algorithm. Figure 3.5 shows the clusters thus formed. The blue dots represent the seed points of the clusters which were chosen to be the customer with the highest average drop size.

Retain only the seed points and destroy the clusters thus formed.

Consider customer X which must be allocated to one of the clusters (seed points) represented by blue dots in Figure 3.6. To evaluate the suitability of each customer to a cluster consider cluster Y. By hypothetically allocating this customer to the cluster three measures must be considered:

- The distance from X to Y
- The amount X has added to the total route time of cluster Y on each day which X is serviced. This includes both travel time from X to Y and time at store X. As a result, the fourth or Thursday 'beaker' has exceeded the benchmark time (represented by the dotted line). The travel time form the depot (green dot) to the cluster (seed point) is already included in the beaker levels.

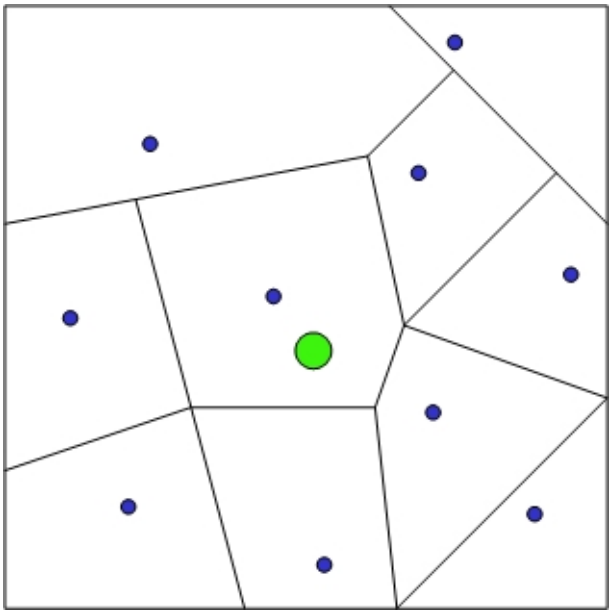


Figure 3.5: An example of the results of this clustering which is geographically based.

- The mass of product for customer X has added to each day's current load in cluster Y. Monday and Thursday trucks are now full but Monday's truck will return back to the depot early.

Once the allocation of customer X to all clusters has been evaluated in this way the best cluster is selected and the total distance, time and mass values updated for the selected cluster. The other clusters are cleared of the temporary data relating to customer X. The next customer in the list is chosen and the same process repeated.

Figure 3.7 shows the clusters which are ultimately formed using this process which may not always be intuitive when viewed on a map.

Notable features of this method include:

- The user stipulates as an input parameter how many clusters should be formed. This is an estimated figure and a typical value would be the number of vehicles in the current fleet. The algorithm then creates a range of different numbers of clusters to be evaluated by considering 20% of the input figure either way. A reasonable number of clusters to consider can be found by adding the total expected waiting time per day and estimated travel time and dividing this by the value the user

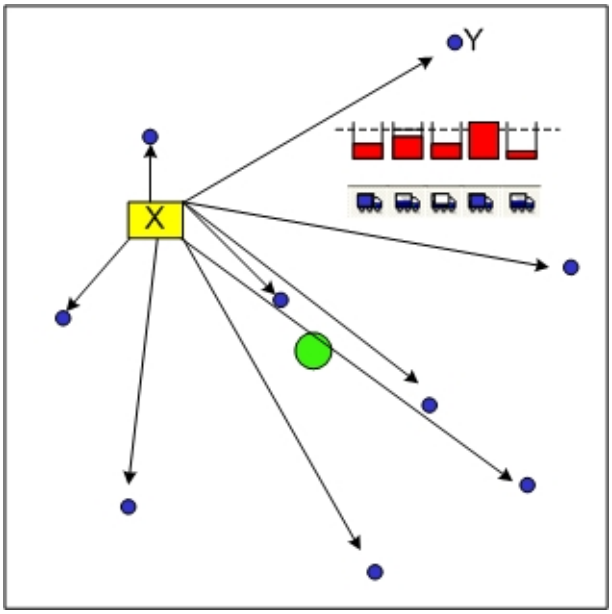


Figure 3.6: Every customer is allocated to each cluster to determine which is the best fit. By adding a customer to a cluster the time and mass carried by the truck servicing that cluster are 'topped up'.

input as a typical length of working day. The largest value over the week will be incorporated into the range of cluster values if it does not fall within the present range.

- The k-means clustering algorithm is used to cluster the customers. This clustering takes nothing more than the geographic location of the customers into account and is not the final clustering result mentioned before. This step simply finds one anchor store per cluster which will act as a seed point for the clustering step to take place later on. Instead of choosing random seed points for this comprehensive clustering step, the customers with the largest average job size per initial cluster are chosen to act as the anchors (seeds). The same clusters are unlikely to be re-formed. The reason for this is that the clustering procedure to come takes into account time at customer and job size as well. This k-means[1] algorithm (which has a random element) is called as many times as the number of iterations input by the user.

- The model will not always favour the iterations with the largest number

[1]This method finds a partition of the customer delivery points by minimizing the total within-group sum of squares over all variables.

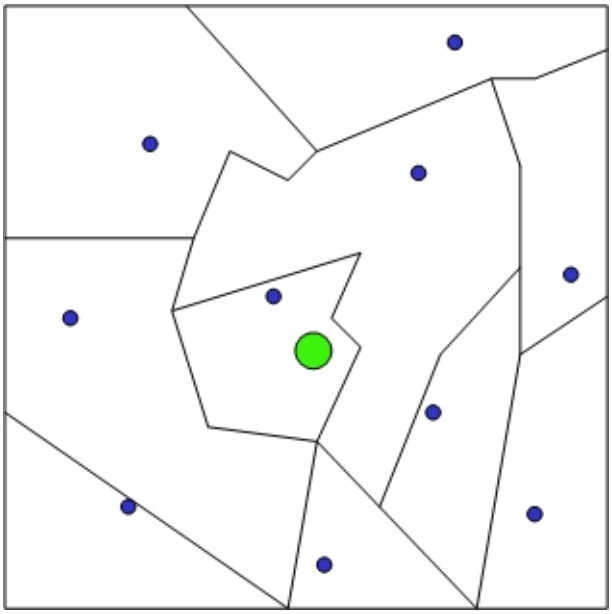


Figure 3.7: The results now take into account other factors besides the pure distance from each customer to each cluster.

of clusters because this will cause the time utilization to be poor. Also, although the travel time (distance) from each customer to the centre of each cluster is lower if there are more clusters, the total travel time (distance) from the depot to all the cluster centres will rise to counteract this.

- It is possible that a seed (anchor) customer is on a schedule type which requires delivery days to be selected by the model. Since the seed customers are 'bedded down' first, there is no initial preference for any particular delivery days. The algorithm currently assigns days randomly to cope with this.

- Since the customers are added to clusters one at a time, the order in which the customers are introduced into the algorithm has a heavy bearing on the results. Five different orderings were found to be useful:

    1. Total number of visits (drops), then by total job size for the period (to break ties)
    2. Total number of visits, then by average job size
    3. Total mass of all jobs delivered over the week

4. Average job size
5. Distance of customer from the depot

Since the quantity of vehicles or vehicle sizes (besides the largest vehicle) is not an input into the model, it is possible to order the customers by characteristics other than job sizes. Many existing models fail if the customers are not introduced into the problem in decreasing order of job size as feasibility problems with the fleet may arise.

- The most appropriate day(s) to service customers is chosen per cluster based on the days within the cluster which have the lowest total time (the total time for each cluster includes the travel time from the depot to the cluster, with the average travel speed as an input). If the addition of a customer causes the total time for a cluster to go above the benchmark, a penalty is incurred.
- The total load for a cluster on any day will be restricted to being at most the size of the largest vehicle. This value is input by the user.

- The first phase of optimization comes with the allocation of each customer to a cluster (for a given number of clusters, seeds (anchors) and ordering of customers). The three factors which form the objective function for this phase are:

  1. The average deviation from the benchmark length of day. Deviations above the benchmark are magnified so that positive and negative deviations do not cancel each other out. Emptier clusters with respect to time would preferably be filled.
  2. Distance from non-seed customer point to seed points (cluster centres).
  3. Coefficient of variation (standard deviation / mean) of the total job loads per cluster before and after the addition of a customer to a cluster. If the variation has been reduced, the customer-cluster match is appropriate because the cluster has been 'filled' or smoothed over the week. This is good as it means that roughly the same size vehicle can be sent to a cluster each day.

- It was decided to incorporate the allocation of customers to days for those customers who do not have fixed delivery days into the model for various reasons:

  - The total load cannot exceed the maximum vehicle size on any day. Without the allocation functionality, too many customers in a particular cluster may have a fixed/certain delivery on a day and then two or more vehicles have to be assigned to that cluster.
  - Similarly, it is unlikely that any delivery groups/clusters will be empty on certain days. (Note that the workload for a cluster may not be the same each day of the week.)
  - The spacing between delivery days for customers with two deliveries per week needed to be obeyed.

- The second phase of optimization involves selecting the best iteration from all those run. These iterations differ in the number of clusters, anchor customers and the ordering of customers. The best iteration will be the one with the lowest total travel distance, best time utilization and smoothest delivery volumes. There is an obvious tradeoff which needs to be captured in the weighting of the time and mass/volume terms relative to the distance term (see Section 4.3.2).

Using the allocation of customers to days and clusters (delivery groups) a weeks worth of jobs can be routed in *FLO*. *FLO* will essentially perform the allocation of vehicle to cluster step and order the customers on each route. In doing this, delivery windows and delivery priorities can be taken into account. Since no fleet information is provided when the function runs, the postprocessing can make use of the existing fleet or a wide variety of vehicles for a fleet sizing. The most appropriate vehicle for each route on each day is selected by *FLO* and customer-vehicle restrictions can be applied at this stage. It may occur than two or more clusters can be merged manually after the scheduling is complete. The delivery group constraint prohibits clusters from being merged on some days only (since the delivery groups are fixed over the week). If it is permitted to service a delivery group with more than one vehicle, merging can take place if it makes sense to do so. This solution does not directly try and reduce the number of vehicles required.

The benefit of using the models from methods 3 and 4 is that when the customer base, fleet or any other constraint changes, a new set of customer to delivery group and delivery day assignments can be made fairly quickly. An exercise like this provides results which are valid only as long as the customer base does not change too much. New customers can usually be added to delivery groups by manually examining routes operating in the neighbourhood

of the new customer. Customers may also leave the database which frees up space on routes for new customers. Similar effects are experienced if a customer relocates to new premises, changes delivery frequency or delivery volumes. New business rules and policies may also be implemented which affect routing decisions.

Re-running the model very frequently is very disruptive to both drivers, customers and staff (who must renegotiate service contracts) and should thus be kept to a minimum. In the event of re-doing the routing exercise, customers who have remained static in the customer base can have their delivery group and delivery days fixed as input to the model so that only changes will be captured in an optimal way in the route plan. That way long term decisions on both a strategic and tactical level can be maintained and customer's requests timeously met in a manner which maximizes profits.

Chapter 4

Results

Currently no optimal solutions exist for most variations of the VRP as they are in general NP-hard. The large number of customers and different service day options makes this ARPNDD problem an enormous combinatorial optimization exercise for which existing heuristics are not appropriate. Four new heuristics were thus created and applied and the results are discussed next. The different techniques are also criticized with respect to their runtime, ease of use and modification, and quality of results.

The input data for the test problem are briefly discussed:

The proportion of each customer type in each schedule option as well as their contribution to the total workload is summarized in Table 4.1.

Sched.	Schedule option	Cust.	Prop.	Drops	Prop.	Tot. Kg's	Prop.
1	Any day	1222	38%	1222	15%	136178	5%
2	1 of 5 2-day combin.	527	16%	1054	13%	235481	8.5%
3	1 of 2 drops T/W	78	2%	156	2%	12758	0.5%
4-8	No flexibility	1392	44%	5755	70%	1324885	86%
Total		3219		8187		2713634	

Table 4.1: Contribution of each customer type to the number of drops over the week. (See Table 1.1 for more detail.)

The order quantities varied fairly widely from customer-to-customer; the

Route-first Cluster-second

Day	Drops	Cost	Penalty cost	Time (min)	Dist.	Time Util.	Space Util.	Wait	3T	4T	7T	10T	16T	20T	No. Veh.
WEEK	7477	365185	898066	260138	49309	85	64	305	5	20	430	45	5	5	510
SAT	710	62037	91093	28380	7062	51	42	49	1	4	84	4	0	0	93
Total	8187	427222	989159	288518	56372	80	61	354	6	24	514	49	5	5	603

Cluster-first Route-second

Day	Drops	Cost	Penalty cost	Time (min)	Dist.	Time Util.	Space Util.	Wait	3T	4T	7T	10T	16T	20T	No. Veh.
WEEK	7477	360699	718282	260113	50185	86	73	150	5	20	430	45	5	1	506
SAT	710	40406	66340	26337	5375	73	65	30	1	4	52	3	0	0	60
Total	8187	401104	784622	286450	55561	84	73	180	6	24	482	48	5	1	566

Difference

Diff.		26118	204537	2068	811	-4	-12	174	0	0	32	1	0	4	37
%Diff.		6.51	26.07	0.72	1.46	-4.8	-16	97							6.54

Table 4.2: Comparison of the route schedules for the two methods under consideration.

coefficient of variation of demand was: 723.35/329.11 = 2.2
The ratio of maximum to mean order quantity ranged as high as 3.77 for a single store and was 23.66 (7788/329.11) across all stores.

The average distance from warehouse to store was 31.54 kms with the distances ranging from 0.45 to 202 kms.

4.1 Comparison of the Route-first and Cluster-first heuristics

The first two methods which were employed each took about two weeks to produce results and were very labour intensive. The steps followed in each method were intuitive but could not easily be formalized. Table 4.2 compares the results based on common routing statistics and it is clear that the cluster-first approach is a stronger methodology for the ARPNDD problem.

- The vehicle usage has dropped significantly. On Saturday's the difference is over 35%. This justifies the point made earlier that too many delivery groups may result when using the route-first method.

- The reduced waiting times between stops waiting for the next store to open, also indicates that the routes are fuller.

- More important than the actual cost of delivery being lower is the substantial drop in the penalty cost. This shows that the routes are of better quality with fewer missed windows and more balanced length.

The most substantial differences between the two methods are:

- Possibly the biggest drawback with routing first, is the dependence of the week's routes on the first day which is chosen for the initial scheduling. A large portion of the customers to be visited will appear on this day and be assigned to a delivery group which will remain fixed for all other days of the period. By clustering first, the full week's routes can be run together, therefore not biasing the results in favour of the peak delivery day. One of the reasons for constructing fixed routes is to balance the workload amongst the delivery days.

- If on the first day of running a schedule with the route-first algorithm, all the vehicles are used (which is very possible since the heuristic begins with the largest day) then the number of delivery groups is equal to the number of vehicles. A problem then arises in that if two or more routes are not grouped together to reduce the number of delivery groups, there may be days later in the week where there are more delivery groups than vehicles. The merging of routes into one delivery group is a subjective step. A possible way around this is combining routes which have few stops and large drops so that the number of customers in a delivery group does not become too large. An alternative would be to allow one vehicle to service two delivery groups on a day with each load being for a separate delivery group.

- The anchor store concept is easier to apply, if it is required, when stores are grouped prior to routing. Before routing begins, an anchor store is chosen for each delivery group. This customer is usually one who orders large amounts and has daily deliveries, thereby justifying a trip to a particular area. In dense areas where more than one vehicle may be used per day, multiple anchor stores can be chosen and a constraint applied to ensure these stores are not serviced together.

- Customers who are serviced less than once per week are difficult to incorporate when routes are first created. They can easily be added to routes if they are part of the initial clustering though. The most

suitable route, on any day which that customer's 'neighbours' in the same delivery group are being serviced, can be chosen.

- When a single customer's order on a particular day has to be split because it is larger than the largest vehicle, the two orders will still be filled together with other customers orders within the same delivery group with the cluster-first algorithm. With the route-first technique, these have to be assigned to the same delivery group manually. If the two routes with orders for the same customer are not similar, the delivery group may cover a larger area than desired.
- From a practical point of view, if stores are more tightly clustered, redeliveries are less likely. Often drivers have to re-order stops or drop off assistants in queues at receiving bays with stock and collect them later to ensure all deliveries are made.
- The clustering first heuristic is quicker to implement

While allowing routes to span larger areas may provide cost benefits, clustering clients geographically is more aesthetically appealing and ensures that driver familiarity benefits are realized. In addition to this, when new customers are recruited these can be added more easily to existing fixed routes with the cluster-first technique.

4.2 Mathematical model formulation

Given the examples of VRP problems reported in the literature, it was anticipated that an exact solution to the ARPNDD problem would not be found. Such a method was nevertheless attempted as it was thought that by relaxing constraints and reducing the number of customers and vehicles an output could be possible. Even though the results described below are disappointing, it was felt that other packages similar to *LINGO* would not have provided significantly better output.

4.2.1 Model runs

The initial model formulation (see Section 3.3) was not ideally suited to a direct translation into programming code. The model needed to be made more parsimonious and detail omitted in favour of a more robust formulation. The process of re-expressing and reconfiguring the model is described

below.

As a result of running the original model in *LINGO*, the following changes were made to eliminate the use of non-linear functions as much as possible (see Table 3.1 for original equations):

1. The Manhattan distance was used so that a square root did not have to be calculated.
2. The *distance* calculations were converted to *time* calculations by changing the DegToKm factor, thus avoiding the need to divide by the speed.
3. The cost/km for each vehicle was converted to a cost/hr.
4. The absolute value calculations for the distances were made linear by defining non-negative variables as follows:
   $c = |a - b|$ replaced with:
   $c >= a - b$ and $c >= b - a$
5. The average distance from each customer location on a route to the centroid of that route was changed to the total distance, thus eliminating the need for dividing by $\bar{n}_k t$.
6. The average distance of each route on each day was changed to the following: $r_{kt} = 2d_{0kt} + d_{kt}$. (This was in line with the change in point 5 where $\bar{d}_k t$ was no longer calculated.)

The code for the first model which ran and produced feasible results in *LINGO* is shown in Appendix 4.1. The notable feature of this model is that the centroid value is a free variable rather than having a lengthy calculation attached to it. This of course also ensures that the centroid is in fact the point which minimizes the sum of the distances from the centroid to all other points in a group. The model, whose characteristics and output details are shown in Table 4.3 and Table 4.4 respectively, is non-linear and the solution time grows dramatically with an increase in the number of customers and/or vehicles.

Due to the fact that a cutting plane algorithm is used to solve problems in which decision variables must be binary, a rounding error is often encountered. This resulted in the need for constraints to be slightly altered e.g.

	Model 1	Model 2
Variables:		
Total	1260	1050
Nonlinear	900	0
Integers	390	900
Constraints:		
Total	1745	425
Nonlinear	60	0
Nonzero coefficients:		
Total	6131	4986
Nonlinear	1200	0

Table 4.3: *LINGO's* model analysis for the ARPNDD models considered with 10 customers.

... ≥ 1 was altered to > 0.999. The paper by S.J. Miller on linear programming [Mil06] gives useful methods of re-expressing IF, AND, OR etc. statements in a linear fashion. These were not incorporated into the model as it was assumed that the solver automatically converted/translated these statements into such forms.

In contrast, the second version of the model is linear (see Table 4.3 and Appendix 4.2). This formulation has reduced runtimes (see Table 4.4) and is far easier for the solver to interpret. Here, a set of fixed potential centroids (at least as many as the number of vehicles) is provided and the solver chooses the best for each route from this set. The candidate centroid set is really a subset of the full customer set (the selection of which can be generalized to be more uniformly distributed by ensuring that the customers appear in a random order in the list and by selecting every 4th customer rather than just the fist 25% for example). The fewer customers that are chosen to be in the centroid set, the faster the runtime as the number of options for the solver to consider when assigning a centroid to each customer and to each vehicle is reduced.

The delivery group constraint (the final constraint in the ARPNDD formulation) was omitted from this version of the model. This constraint is complex and adversely affects the solution time of the model. It was desired, at this stage, to try and achieve an efficient allocation of customers-to-vehicles-to-days even if this restrictive condition was not met.

Model	No. Cust.	No. Fix. Centres	No. Veh.	Runtime	Solution	Obj. Val.
1	6		5	60:14:11	Local Opt	58.82
2	6	6	5	00:00:04	Global Opt	1636.92
1	10		5	Encount error 165:12:34	Local Opt	181.33
2	10	10	5	00:03:21	Global Opt	1876.34
2	15	5	6	00:00:02	Global Opt	2197.78
2	15	8	6	00:02:39	Global Opt	2172.28
2	15	15	6	02:56:09	Global Opt	2171.24
2	20	10	10	00:23:58	Global Opt	2484.13
2	20	20	10	Interrupted at 12:30:00	Feasible soln	2489.30

Table 4.4: Comparison of solution times for the two model versions with different numbers of variables.

It was further suggested by the *LINGO* technical department to make the cost/km for local travel the same for all vehicles. It is computationally expensive to base the decision of vehicle allocation by cost on both the fixed and variable costs. The transport cost from the depot to the chosen centroid still depends on the vehicle chosen for the trip. This ensures that the correct size vehicle is still chosen because the fixed costs are proportional to the payload.

Model outputs

Table 4.4 below shows the various problem sizes considered and their runtimes. All runs were done on a Pentium M, 2 GHZ machine with 1 GIG RAM.

Note that the run for Model 1 with 10 customers did not have the delivery group constraint. This was omitted for comparative purposes (see Section 4.2.1) and because of the long runtimes in the 6 customer model which did include that constraint.

The run with 20 customers and 10 centroids ran surprisingly faster than the equivalent case with 20 centroids. The smaller run was then done again (with exactly the same input data and no adjustments to the solver settings) to verify the solution time and it was just under 23 mins. The objective value was identical to the first but the number of solver steps and iterations

was however different. This suggested that there was a random element to the way in which the solver went about searching the solution space. A third run was done to confirm the suspicion and this time the runtime was 32 mins, again with the same objective value as found before but in a different number of attempts. The variability in solution time on such a small model highlights the effect that this could have on larger model sizes. Also, if the solver is interrupted while it is on a local but not yet global optimum, the same solution cannot be guaranteed after the same amount of time on the same problem.

As expected, for Model 2, the objective value improved as the number of fixed centroids was increased. This improvement, relative to the extra solution time incurred, decreased as the number of centroids approached the maximum possible.

By examining the solution times of the problems with 6 and 10 customers, it is clear that Model 2 runs much faster than Model 1. The objective values cannot be compared because the route costs are not calculated in the same way. To compare the generated routes, each weekly set of jobs was put into *FLO* and the daily routes optimized.

Comparison of the routes

The results from the 10-customer runs of Model 1 and Model 2 were entered into *FLO* so that route quality could be compared. Neither of these result sets adhered to the delivery group constraint. Since no sequencing of the routes had been taken into account in the ARPNDD formulation, it was not appropriate to use customer delivery windows in *FLO*. If this detail had been used, certain routes produced by one or the other model may have seemed infeasible because of missed windows and FLO would place a penalty on the route cost because of this.

Even though the first model took days to run and still no global optimum had been found, the second model was still about 1% cheaper (see Table 4.5). The cost of a route is proportional to the distance and time of each route. The second model, although not as exact as the first, performed well and made a better allocation of customers to days. Had the models been run on different sets of data, Model 1 may have given better results but the speed would always remain an issue.

Day	Drops Model 1	Cost Model 1	Drops Model 2	Cost Model 2
MON	5	304	4	302
TUE	5	324	7	301
WED	3	303	4	321
THU	7	305	6	298
FRI	5	327	4	321
SAT	2	298	2	298
Total	27	1861	27	1841

Table 4.5: Comparison of the final routes for the two model versions with 10 customers. *Drops* refers to the total number of customer visits.

4.3 Cluster based approach

As mentioned in Section 4.1, in pursuing an efficient and reliable means to solve the ARPNDD problem, the concept of clustering the delivery points to be visited seemed most appropriate. A heuristic has been developed which is powerful in the areas of clustering stores, assigning stores to days and balancing workload across routes. Sequencing of stores has not been incorporated and is handled outside of this program. The routes planned by this heuristic are sometimes different from those planned by a regular routing/scheduling program. An example of this occurs where a routing/scheduling program will assign for example six stores to a route - two close to the depot and four far away. It may occur that the route is sequenced such that a close store is visited, followed by the four distant stores and the final stop is near to the depot again. The clustering based algorithm which was written would not naturally plan such a trip and if it did result, it would be from the objective of workload balance.

Although a standard clustering algorithm has not directly been applied to solve the ARPNDD problem, one such procedure is used to determine the starting points for the delivery groups which are successively built up. Prior to applying any type of clustering algorithm to a data set, the authors of *Algorithms for Clustering Data* [JD88] suggest checking for clustering tendency. Such tests have been said to test for spatial randomness as the data is characterized as samples from spatial point processes. Once these requirements for applying a clustering algorithm have been satisfied, the results need to somehow be evaluated. Since the tests of cluster validity devised so far are not robust and the results here need to be compared to the previous two heuristics, it was decided to program the clusters into *FLO* to assess the

Method	Cost	Penalty Cost	Total hrs	Total kms	Time Util.	Space Util.
Prev. best	360699	718282	260113	50185	85.7	73.4
R latest	366004	640366	257539	49993	84.2	45.5
Diff	5305	-77916	-2574	-192	-1.5	-27.9
% Diff	1.47	-10.85	-0.99	-0.38	-1.73	-38

Table 4.6: Comparison of the weekly schedule statistics for the cluster-first heuristic and the R function.

generated routes.

After running the data through the function with the benchmark number of clusters set at 90, 5 random iterations, a typical 8 hr day and 60km/h travel speed, maximum vehicle size of 10 tons and parameters $\alpha_1 = 0.01$, $\alpha_2 = 0.005$, $\beta_1 = 0.01$, $\beta_2 = 0.005$ and subsequently running these results in FLO the result returned is given in Table 4.6.

The previous best method (clustering first heuristic) took approximately two week's worth of work to complete. The result produced with this function was available after 3 days (algorithm ran for 24 hours and FLO for 12). These times are obviously proportional to the problem size and this was an unusually large case to solve. It was found that leaving the single drop per week customers out of the clustering/allocation algorithm and just adding them to the schedules in FLO produced better results. These jobs are the most flexible in the set (with respect to the delivery group constraint) and act as route 'fillers'. This also helps if the existing fleet is used so that different size loads can be built to meet the vehicle sizes. Feasibility problems in term of vehicle capacity may occur for very large jobs though.

The results here are difficult to compare for two reasons:

1. The function does not take into account delivery windows. The delivery windows in FLO were thus opened up so as not to penalize the statistics above.

2. Only a maximum vehicle size is given which may or may not be reached depending on the location of the stores, time at store and travel time. As above, all vehicles were set to have the maximum capacity in FLO. If the existing fleet is used then certain trips need to be second loads.

4.3.1 Alternative use of the cluster based approach

Issue 2 above makes this algorithm ideally suited to a fleet sizing exercise. Such an exercise was carried out for a leading logistics company in South Africa who have a division serving chain stores (known for their long offload times in excess of an hour and strict delivery windows). Approximately 1300 customers (serviced 1, 2 or 3 times a week) had to be organized into fixed weekly routes while minimizing the number of vehicles used. Stores with two visits per week had to have at least one day between visits and those with three visits per week could not have more than two consecutive day visits. No stores had fixed delivery days. As the delivery windows were expressed as *morning*, *afternoon* and *all day* the algorithm could ensure that a limit was placed on the number of morning or afternoon stores on a route. The company was previously using 85 vehicles and the algorithm returned 68.

Since the delivery volumes are erratic in the freight and courier industry, no job sizes were used in the optimization (and hence no fleet was specified). These values were added once the routes had been input into *FLO* and the most appropriate size vehicle was then selected for each route. In the low season many of the stores do not receive the same number of drops as their service frequency indicates so ad-hoc jobs are added to fill up the vehicles without much disturbance to the fixed routes which the drivers are used to. In peak season these jobs are dealt with on additional vehicles if necessary. The number of hours a driver should work in a typical day was slightly reduced in the algorithm since chain stores have a high variance in their offload times and delays often result in many re-deliveries if the routes are filled to capacity with respect to time.

The formulation has also proven to be robust under various test problems. These are discussed in detail next.

4.3.2 Test problem cases

Various test problems were created to verify the robustness of the heuristic as none were available in the literature since the problem is newly defined. Three of these are described below. The first figures in each set show the customer points with the customer number, service frequency, wait time and average drop size. The colours indicate the correct grouping of customers into clusters. The second figures show *R's* clusters as well as the total time (per day for each cluster) and mass (per cluster for the week). The horizontal

lines on the time bar plots reflect the benchmark value.

A. Day selection
Two obvious clusters of points are present in Figure 4.1. One customer in the cluster nearer the depot should rather be assigned to the other cluster as it will not only improve the balance of the job sizes in that cluster across the week but also prevents the nearer cluster from exceeding the maximum vehicle size (12000 kg). The different orderings of the customer input data are critical to the most optimal outcome.

If it was not for the maximum vehicle size constraint, the two clusters which would have been initially created by the k-means algorithm to determine the seed points would have been recreated. The clustering algorithm is not able to reason that in order to get to the more remote cluster, it is necessary to drive right past customers in the nearer cluster and as such no extra kilometers are incurred by not grouping the delivery points into the two natural sets. In order to achieve this, more emphasis (larger alpha values in the objective function) needs to be placed on balancing the workload between the clusters so that kilometers traveled are not the main driver in cluster formation. This would improve the imbalance in total time in Figure 4.2.

It must be noted that it is sometimes impossible to split the customers between clusters in such a way that the benchmark time is not exceeded. As a simple example: the benchmark time is 600 minutes and there are three customers (with time at customer 500, 500 and 200 minutes) which need to be split into two clusters. If the user stipulates the number of clusters required and does not allow the algorithm to try different numbers of clusters, the best result which can be expected is to exceed the time by 100 minutes. A similar scenario can occur with the maximum vehicle size. So although in total the time at customer (ignoring travel time) does not exceed the 1200 minutes total time available, the time constraint must be broken.

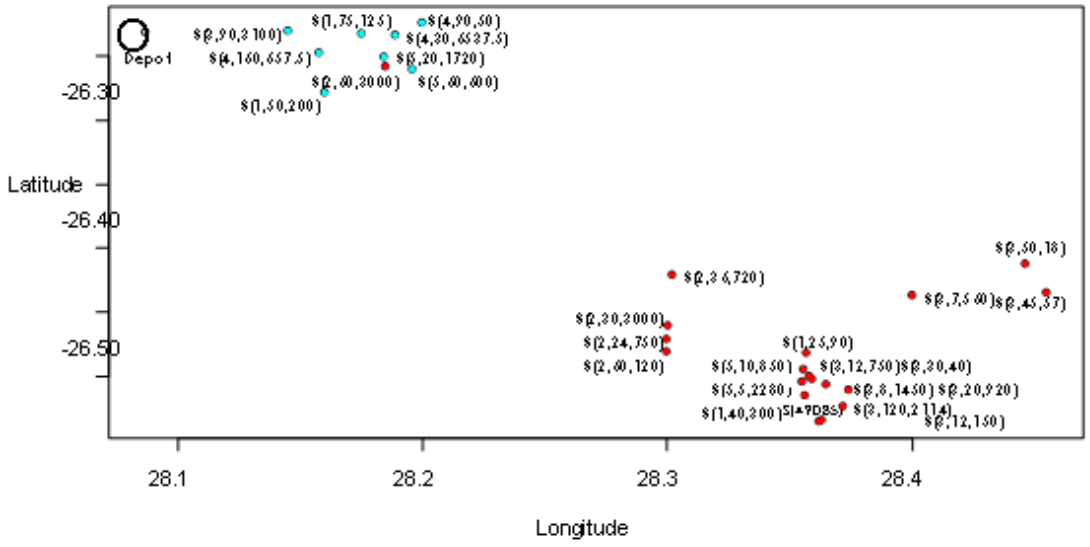


Figure 4.1: Plot 3.1 (a)

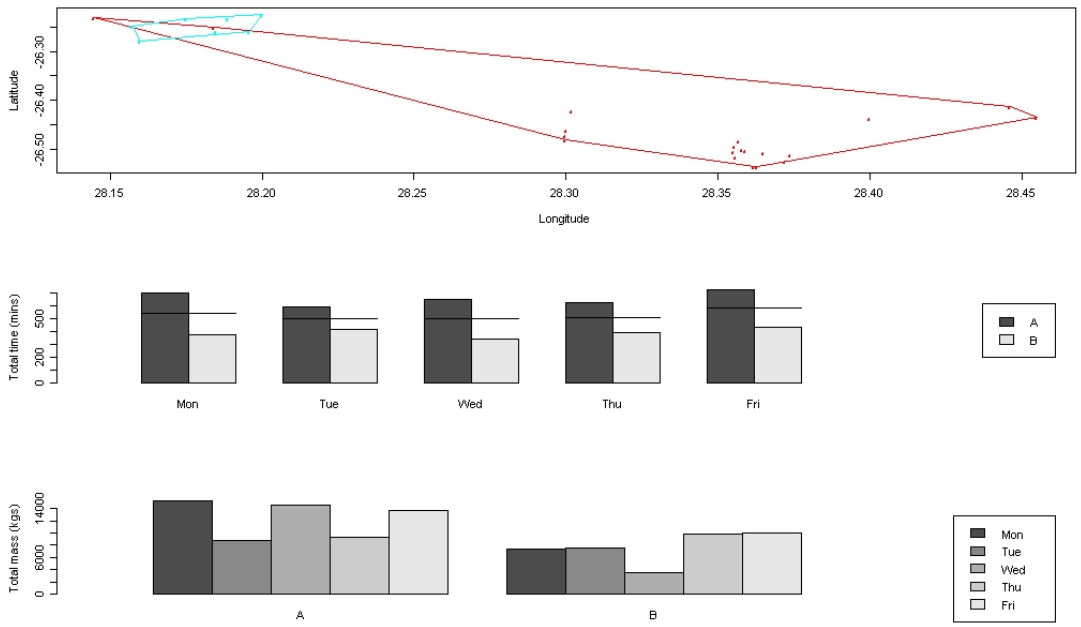


Figure 4.2: Plot 3.1 (b)

B. Maximum vehicle size

Three clusters are evident in Figure 4.3. The cluster south of the depot should remain unchanged. Even if the time and mass distribution could be better balanced by adding some customers from the other two clusters, the additional distance traveled and the time attributed to that travel would not make it viable. The other cluster with five customers, exceeds the maximum vehicle size (12000 kg) and must be split. The lone point must then be grouped with one of the newly formed clusters rather than be serviced alone as an extra vehicle will be incurred with low utilization. The resulting distribution of time and mass in Figure 4.4 is very good.

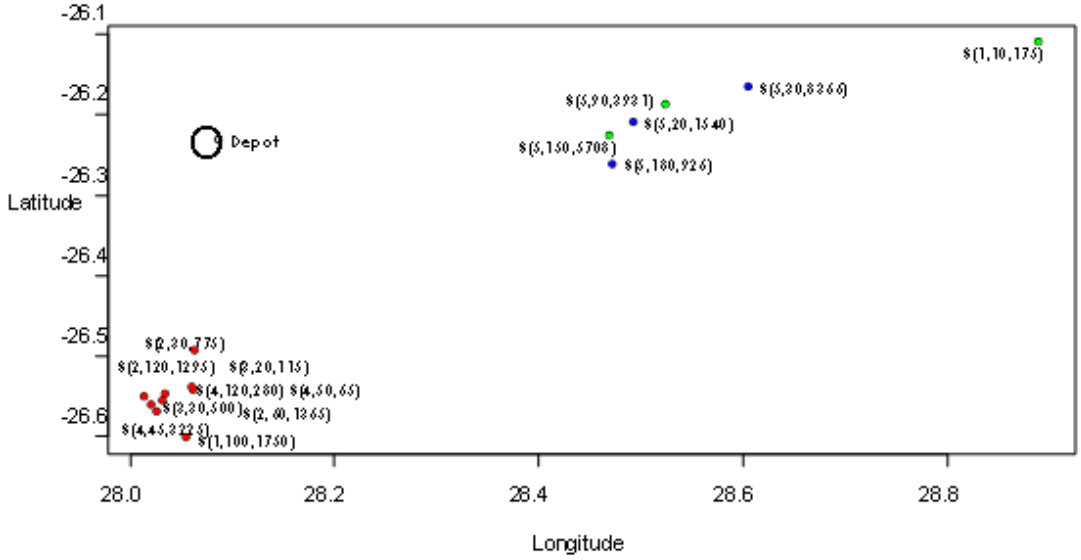


Figure 4.3: Plot 3.1 (c)

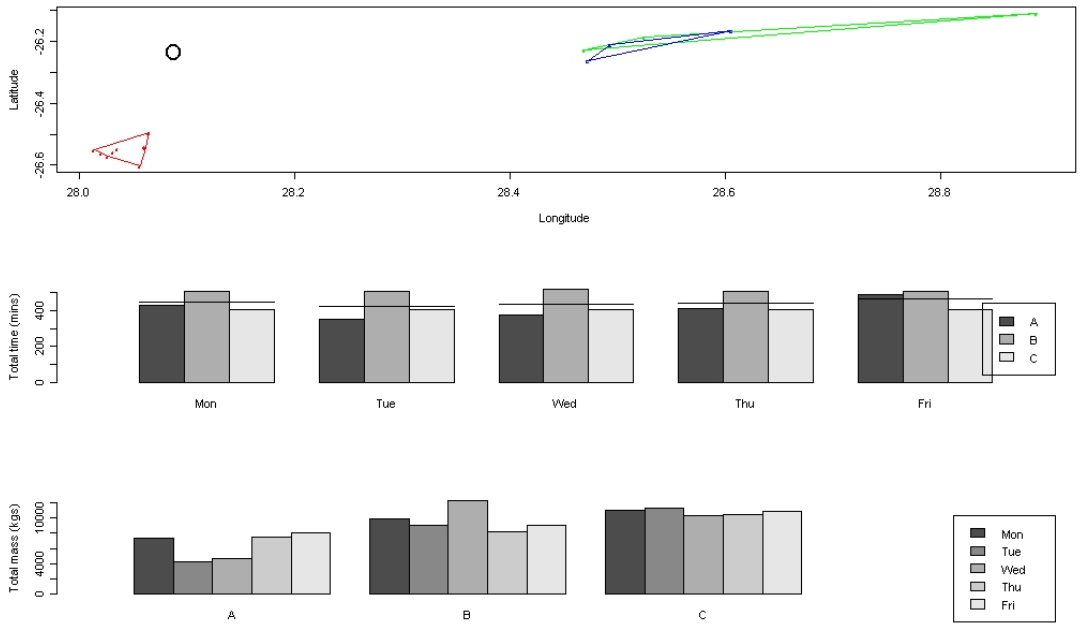


Figure 4.4: Plot 3.1 (d)

C. Number of clusters

The cluster south-east of the depot in Figure 4.5 does not have a high total time at customer but with travel time included is closer to the benchmark (8 hrs). The variance in the job sizes is high but because of the long distance to reach this cluster it must remain like this. In general, trips to far out areas are usually longer than the typical time spent on the road by drivers servicing local territories. It is usually desired to make the most of the time spent far from the depot to avoid frequent trips to such areas. To optimize the time and load characteristics, the remaining customers must be divided into two clusters. Since there are many 3-day per week customers, the allocation of the 1- and 2-day a week customers is critical for even work distribution. Slight overlap in areas is likely to occur but this crossing of routes is optimal in many instances. It is also clear from Figure 4.6 that two smaller and one larger vehicle are required.

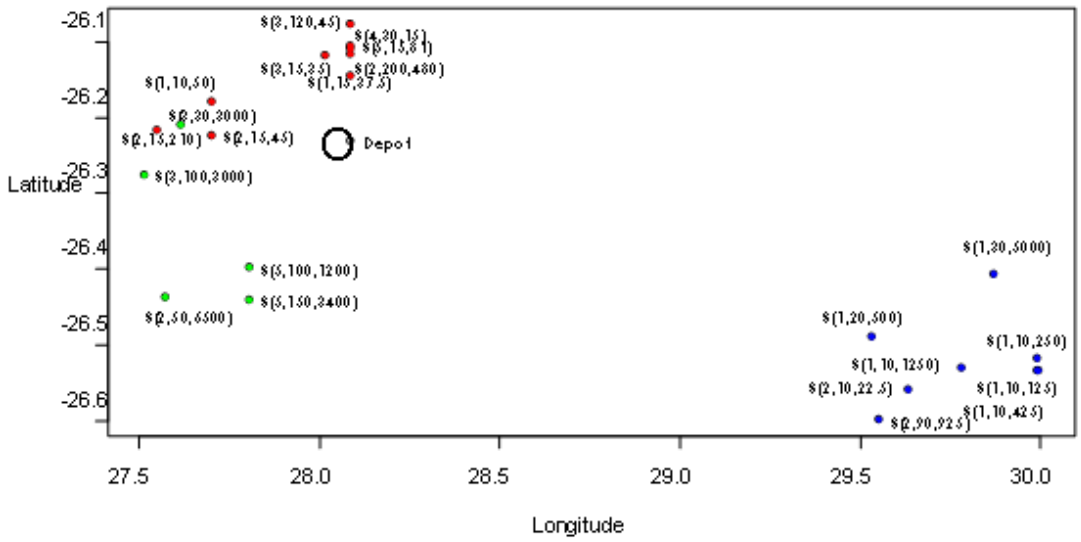


Figure 4.5: Plot 3.1 (e)

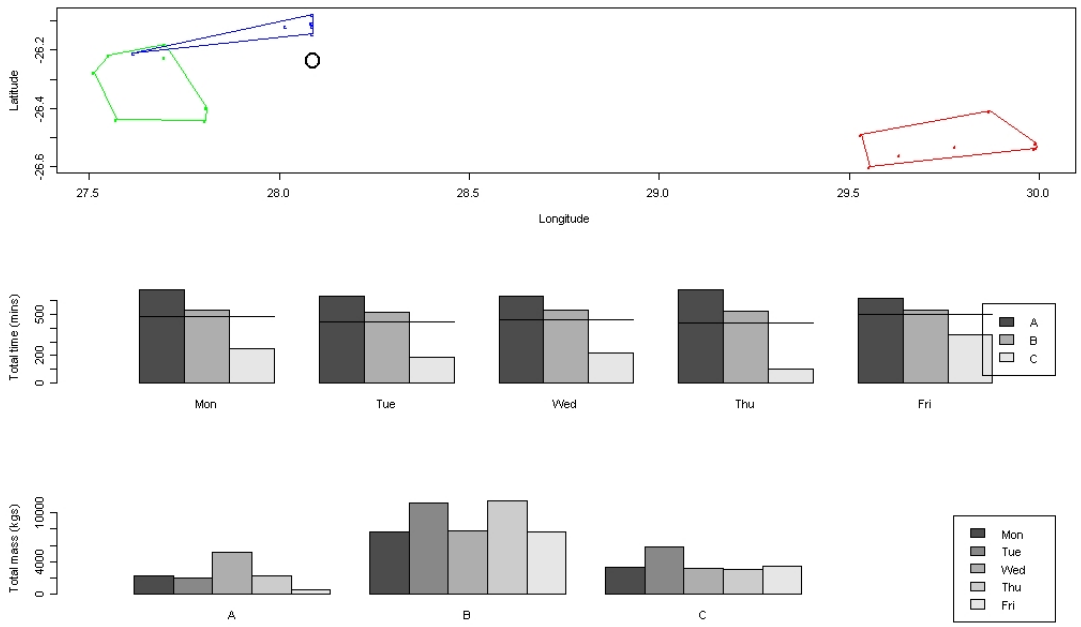


Figure 4.6: Plot 3.1 (f)

A larger problem (where the optimal solution was not known) also produced good results.

The objective functions mentioned in Section 3.4.1 needed to be calibrated. This was done empirically by considering the pairwise relationship between the scaled distance, time and mass terms in the examples above. The aim was determine a set of alphas (objective 1) and betas (objective 2) for the two linear objective functions with the following form:

Objective function 1 (used to determine which cluster (and if relevant, which day/s) a customer should be assigned to):
$Min(y) = distance + \alpha 1.time + \alpha 2.mass$

The distance, time and mass terms are each divided by the absolute value of the maximum of all such terms (for each cluster) for the customer. This ensures that the values are scaled (ranges given below) as the three terms are in different units; kms [0,1], hrs [-1,1] and kgs [-1,1] respectively.

Objective function 2 (used to determine which set of customer to cluster allocations produces the best overall result):
$Min(y) = distance + \beta 1.time + \beta 2.mass$

The three terms in this objective function are scaled by dividing by the minimum of all such terms for each larger iteration (outer loop). Since different measures/statistics are bing used to capture the effect of the different terms in the objective function, it is easier to scale these values rather than try and adjust the parameters to cater for this.

Objective function parameters are affected by the problem size which makes them even harder to determine. No distinct relationship between the alphas and betas could be found, only that better results were produced when the values were similar. Since the time value in the second objective function also includes travel time, less emphasis can be placed on this term as the distance function captures this. The mass value can also have a lower weighting as the variability has been limited by the maximum vehicle size constraint. The main difference between the two objective functions, besides the purpose they serve, is that the first calculates deviations in total time and mass carried per cluster as customers are being added, i.e. not all information is available at the time of calculation as some customers are unallocated.

4.3.3 Extensions

The fourth method described in this dissertation has enabled the solution of various nominated delivery day problems to be found. Improvements and enhancements can still be made and suggested further extensions to the model include:

1. Consider alternative ways of calibrating the model (i.e. finding the $\alpha's$ and $\beta's$ described above) and investigate the relationship between these parameters. A user's preference for the objective can also be captured by prompting for which term should have the greatest weighting (similar to a utility function). An example would be asking whether minimizing total distance traveled is preferred over taking the fastest route.

2. For the smaller problems examined above, it became clear that the natural clustering of the data made the random clustering iteration loop of the algorithm redundant. More importantly, the seed points in each cluster were always the same (the customer with the highest average drop size in each cluster). This pointed out that a further loop, which considered alternative criteria for defining the seed points, would be useful.

3. Allowing for a minimum vehicle size to be stipulated to prevent poor vehicle space utilization. This will currently only occur if the vehicle 'times out' as it has many small drops to make. Alternatively, it may be traveling to a remote area where the time to reach the area is great and combining the stops on the route with any others would mean covering a wide service area which may be impractical.

4. The model currently calculates straight line Euclidean travel distances and multiplies these by a constant 'wiggle' factor so that the travel time is realistic with respect to the time at customer. The travel distance within each cluster is estimated by adding the radial distances from each customer to the anchor customer. An extension would be to include a function which tries to more accurately determine the travel distance on a route by ordering the stops and using a road-network distance from each customer to the next.

5. An anchor customer for each route is used as the centre of the route. Not all anchor customers are serviced each day so an improved method may be to set the centre of the route to the average of the co-ordinates

of all customers on the route. As customers are added to the route during the assignment of customer-to-cluster phase, this value would be recalculated.

6. A postprocessing phase which could add much value would be one which re-looks at all those customers whose schedule days were determined by the model. In the example used here, these would be the 1 and 2-day a week customers. Keeping the customers within the same delivery group (cluster) but just reassigning them to different delivery days may improve the results. This is because the customers are added to delivery groups one at a time and so at any point it is not known what customers are still to come and how they might affect the present routes.

Chapter 5

Conclusions

An unpublished work by Hermann Schichl [Sch03] notes that *'many applications in optimization nowadays are driven by economical interest. It is not a coincidence that another term for the objective function is cost function. In most of the economical applications, like in logistics or cutting stock applications, it is necessary to find a good solution, i.e., a feasible point with sufficiently low objective function value. Usually, an improvement over the solution currently used in the company by 10% is more than enough to satisfy the expectation of the company's CEO. Finding the globally best point usually is not required.'* Since the ARPNDD problem, which attempts to create fixed routes by assigning customers to delivery days, is such a complex problem, satisfactory results for most users fall within the definition above.

The cluster-first, route-first and cluster based heuristics which have been applied have all proven to be able to do this. The application of the first two methods is lengthy and they need to be adapted to handle different service day frequency constraints, unlike the cluster based approach which handles the allocations of customers to days much more efficiently even for large problems. An exact solution using a mixed integer programming tool was also attempted. Even after simplifications to the formulation were made to make the model more linear, only solutions for very small problems (20 customers) were achievable in a practical amount of time.

Some limitations and possible extensions of the current best method (Method 4 - cluster based approach) are addressed in Section 4.3.3. An area worth investigating to enhance the optimization power of this function is Multi-criteria Decision Analysis or Decision Making. The tradeoff between minimizing total route distance while trying to 'fill' routes in terms of work time and total mass transported, results in conflicting evaluation criteria.

The two objective functions in the clustering model (which choose the best customer-to-cluster allocation and then the best result set over all iterations) could be enhanced by techniques from this discipline. To make the solution algorithm more user friendly, a front-end could also be developed, to allow for changes to be made without having to hard code the function for each individual problem.

This research was based on a case study for Clover S.A. at the City Deep distribution centre which serves a store universe of over 3000 outlets with 16 different visit day frequency combinations. Since the original operational setup at Clover involved two separate depots (and no attempt had been made by them to combine the operations at City Deep prior to this study) it is not possible to assess the improvements made by the newly developed heuristic empirically. The new cluster based heuristic provided the best results (in terms of quality and speed) over all methods which were tried and is flexible enough to adapt to other problems with slightly different constraints. Since the clustering method does not explicitly try and create routes which fit perfectly into the supplied fleet of vehicles, it is also useful for fleet sizing exercises.

The clustering model has also been tested on TSP-like problems in the marketing industry where sales representatives (reps) make visits to stores but do not deliver product in capacitated vehicles. Very good solutions have been obtained on problem sizes with over 10 000 outlets and 300 reps. Future work in this area would address a growing need for rep and delivery vehicle routes to be aligned. This need to plan the routes together arises because deliveries usually need to occur within a day of a store's nominated order (rep visit) day. This problem would need to consider times at store for both a rep as well as a delivery vehicle. Further complications arise because the delivery vehicles depart from a central depot and the reps from their homes. Also, not all stores visited by a rep get product delivered because smaller outlets are often required to collect their own stock from a warehouse.

The resulting approach is one which places emphasis on relevant constraints found in operational environments and was developed in order to solve a real problem currently faced by many companies distributing product. The distribution area is operationally 'at the end of the line' in a production or courier type business. Many companies have found that by correcting inefficiencies and instituting discipline in distribution, many other sectors of their business quickly align and improve their ways of working. With time, these repercussions from optimal distribution planning cause operations as a

whole to tend towards optimality.

Appendix

A.1 Basics of the FLO optimization algorithm

The *FLO* algorithm is a 'hill climber' that tries to find the minimal cost solution to the routing problem. This means that it starts with an initial reasonable solution and tries to improve on this by making small steps at time, where each small step improves on the current solution by decreasing the total cost.

In order to include delivery windows and other constraints into this paradigm, a penalized cost is used. If any constraint is broken, a high cost is added to the current solution. For example, if a delivery window is broken, a cost equivalent to two times the cost of that trip could be added. This means that the cost *FLO* minimizes is the sum of the actual costs plus the penalties for broken constraints. This is called the penalized cost.

FLO adds penalties for the following types of constraints:

- Delivery out of delivery window
- Driver hours too long
- Delivery by excluded vehicle type

In addition to these types of constraints FLO also allows specific requirements to be specified for particular jobs. Some examples of these are:

- First on route (all first-on-route jobs are scheduled before non-first-on-route jobs but it is possible for two first on route jobs to be on the same trip if they are close together)
- Single trip (only job on the trip)
- Leave empty

The optimizing algorithm implements these in a different way. It will not even consider routes which break these constraints so that these will always be observed even if they add a huge cost to the solution.

When *FLO* creates a schedule, the available resources (trucks and staff) are always respected. A schedule will never be created with more than the allocated resources. However if fewer resources are needed than are allocated, usually the lowest cost solution will also minimize the number of resources used so the *FLO* solution will often use fewer vehicles than the number that were allocated.

When the optimizer is running it keeps on looking for a better solution to reduce the penalized cost. Once no further improvements on the current solution can be found, it is compared to the best one found and if it is better it becomes the best solution and *FLO* starts with a new starting solution and tries to improve on it. Usually the user sets a timeout to say how long to spend on this process. Also, if *FLO* does not improve on the best solution for three big iterations it returns the best solution found. Often a good enough solution may be found after the first big iteration, so if the user is running a trial schedule it is enough to wait until the end of the first big iteration and then end the process manually. A trial schedule may be useful if the planner wants to see how many routes will be planned or to decide which deliveries not to schedule if there are insufficient vehicles available.

A.2 Application of the Route-first Cluster-second algorithm to the case study data

Based on the existing service level delivery rules (see Table 1.1), the following steps are followed:

Thursdays are the busiest days both with respect to total mass and stops. Thursday is thus treated as a benchmark day as it has the most certainty.

Begin by running a schedule for Thursday. Include the customers with the number of drops specified with the given customer codes and delivery groups; where delpnt-4 indicates a confirmed Thursday delivery and delpnt-45 indicates a Thursday or Friday delivery (to be determined by the results).

# Drops/customer type	Customer code	Delivery group
2 (any 2-day customer without a Sa drop in the historical data)	delpnt-45	THU, FRI
3 (Tu, Th, Sa)	delpnt-4	THU
4	delpnt-4	THU
5	delpnt-4	THU
6	delpnt-4	THU

Note: For the 2-day per week customers, once a customer has been allocated a Thursday or Friday delivery as a result of this run, that customer will further be classified into schedule M, Th; Tu, Th or Tu, F; W, F.

- Look at THU routes.
  - As long as a route has at least one customer with customer code delpnt-4 (delivery group THU) it is regarded as a fixed Th route. All customers on such a route with customer code delpnt-45 must have their customer code changed to delpnt-4 and delivery group changed to THU (i.e. they have been assigned to a Thursday or equivalently to schedule M, Th or Tu, Th).
  - All routes with only delpnt-45 customers can be deleted. These customers will re-enter the pool of unassigned customers with respect to their second delivery day.
- All the Th routes are then 'locked' and renamed: AA-THU, AB-THU etc. All customers on these routes must also have their delivery groups renamed accordingly because otherwise if two customers are on route

AA-THU and they both have Friday deliveries as well, they must appear together on the AA-FRI route.

Run a set of Friday routes with the following customers.

# Drops	Customer code	Delivery group
2 (any 2-day customer without a Sa drop in the historical data)	delpnt-45	THU,FRI
3 (M, W, F)	delpnt-5	FRI
4	delpnt-5	@@-FRI
5	delpnt-5	@@-FRI
6	delpnt-5	@@-FRI

Note: The 2-day per week customers are all those initially eligible to be in schedule M, Th; Tu, Th or Tu, F; W, F except those that as a result of the previous run have been assigned schedule M, Th or Tu, Th. The '@@' is a wildcard indicating that all those customers who are serviced four or more times per week definitely receive both a Thursday and Friday delivery and have been assigned to a delivery group already as a result of Thursday's run. From here on in, these customer groupings will be fixed.

- Look at FRI routes.
  - As long as a route has at least one customer with customer code delpnt-5 (delivery group FRI) it is regarded as a fixed F route. All customers on such a route with customer code delpnt-45 must have their customer code changed to delpnt-5 and delivery group changed to FRI (i.e. They have been assigned to a Friday or equivalently to schedule Tu, F or W, F).
  - All the F routes are then 'locked' and renamed: AA-FRI, AB-FRI etc. The only routes that will have a prefix (@@) not seen before are those routes with a customer from schedule M, W, F on, the rest will have the same names as the equivalent Thursday routes.
- The number of routes on Thursday and Friday as well as the total number of stops and mass need to be assessed. What remains to be allocated to these two days are the routes with only delpnt-45 customers. These customers will form all new routes, exclusively made up of delpnt-45 customers. A 'balancing act' needs to be performed. By determining which vehicles are unused on Thursday and on Friday a separate run can be done with those vehicles (some may appear twice

if they haven't been used on either day) and those customers who still have delpnt-45's. These routes are then split between Thursday and Friday with the aim of smoothing the fleet in mind.

All customers in the Table 1.1 should now have a customer code with suffix -4 or -5 only. Thursday essentially gets first choice on the delpnt-45 customers this way. Besides grouping customers, what this process achieves is really choosing which days to service certain types of two drop per week customers (as they are the only delpnt-45 customers).

Next schedule Monday as it has the next most fixed customers.

# Drops	Customer code	Delivery group
2 (M, Th)	delpnt-12	@@-MON,@@-TUE
3 (M, W, F)	delpnt-1	@@-MON
4	delpnt-1	@@-MON
5	delpnt-1	@@-MON
6	delpnt-1	@@-MON

- Look at Monday's routes.
  - As long as a route has at least one customer with customer code delpnt-1 it is regarded as a fixed M route. All customers on such a route with customer code delpnt-12 must have their customer code changed to delpnt-1.
    Note that all the customers with delpnt-12 who are not now on a fixed Monday route cannot be assumed to be on a Tuesday route. If after Tuesday has been run and there are routes that are exclusively made up of delpnt-12 customers, these must be run again with the remaining fleet as was done for Thursday and Friday. There is thus still a chance that they can be Monday customers.
  - All routes with only customers with code delpnt-12 can be deleted. Lock the routes and rename them as above.

Tuesday's routes are now scheduled.

# Drops	Customer code	Delivery group
2 (Tu, Sa; W, Sa)	delpnt-23	TUE,WED
2 (M, Th; Tu, Th)	delpnt-12	@@-MON,@@-TUE
2 (Tu, F; W, F)	delpnt-23	@@-TUE,@@-WED
3 (Tu, Th, Sa)	delpnt-2	@@-TUE
4	delpnt-2	@@-TUE
5	delpnt-2	@@-TUE
6	delpnt-2	@@-TUE

- Look at Tuesday's routes.
  - As long as a route has at least one customer with customer code delpnt-2 it is regarded as a fixed Tu route. All customers on such a route with customer code delpnt-12 or delpnt-23 must have their customer code changed to delpnt-2. As before, since these customers fit well with customers who must be served on a Tuesday it is natural for them to become Tuesday customers.
  - All routes with only customers with code delpnt-12 or only delpnt-23 can be deleted.
  - Any route with both delpnt-12 and delpnt-23 customers (and no delpnt-2 customers) can become a fixed Tuesday route and the customer codes changed accordingly. Lock the routes and rename them as above.
- All the delpnt-12 customers need to be run separately on the remaining vehicles and split between Monday and Tuesday as done before for Thursday and Friday. Since there are still delpnt-23 customers who could land up on a Tuesday, slightly more of these routes can be placed on a Monday.

Finally Wednesday is done.

# Drops	Customer code	Delivery group
2 (Tu, F; W, F)	delpnt-23	TUE,WED
2 (Tu, Sa; W, Sa)	delpnt-23	@@-TUE,@@-WED
3 (M, W, F)	delpnt-1, delpnt-3	@@-MON,@@-WED
5	delpnt-3	@@-WED
6	delpnt-3	@@-WED

Note: The two drop customers are those that during the previous run were not fixed to Tuesday.

- Look at Wednesday's routes.
  - As long as a route has at least one customer with customer code delpnt-3 it is regarded as a fixed W route. All customers on such a route with customer code delpnt-23 must have their customer code changed to delpnt-3.
  - All routes with only customers with code delpnt-23 can be deleted.
- Lock the routes and rename them as above.
- Now run the delpnt-23 customers as previously for the delpnt-12 customers. Note that this run will not provide the same routes deleted above since not only Wednesday's remaining vehicles but also Tuesday's can be chosen by *FLO* to group these customers into the same delivery group.

Saturday

# Drops	Customer code	Delivery group
2 (Tu, Sa; W, Sa)	delpnt-6	@@-SAT
3 (Tu, Th, Sa)	delpnt-6	@@-SAT
6	delpnt-6	@@-SAT

Nothing new is determined from this run besides which vehicles will be used.

The only customers who have not been assigned to a delivery group/day at this stage are the single drop customers. These are entered into *FLO* without a delivery group. Single drops never form part of Saturday's routing.

FLO has a feature where unscheduled jobs (the one day a week customers at this point) can be added to existing routes. *FLO* selects which route is the most appropriate for such a job as well as which order in the route it should be slotted in to. All the weekday routes determined so far are entered in and those unscheduled jobs are either added to those routes or new routes consisting of just these single drop customers are created for the vehicles that are still available. The exclusively single drop customer routes then need to be assigned a day depending on vehicle availability.

A3. Steps followed in the cluster-first route-second heuristic

Figure 3.3 shows the initial steps taken in setting up the delivery groups for the test data depicted in the map in Figure A.3.1. A graduation of 0.15 of a degree (approx. 16km) was chosen after experimentation. Starting with the lines of longitude and latitude passing through the centre of the depot, successive grid lines were formed by adding multiples of 0.15 of a degree to these start lines. The dotted lines in the diagram show the maximum and minimum longitude and latitude (i.e. those of the most far out points). The shaded area depicts the full delivery region with the above mentioned lines forming the new borders. The last two rows of blocks south of the depot were combined because the second grid line was very close to the farthest latitudinal point.

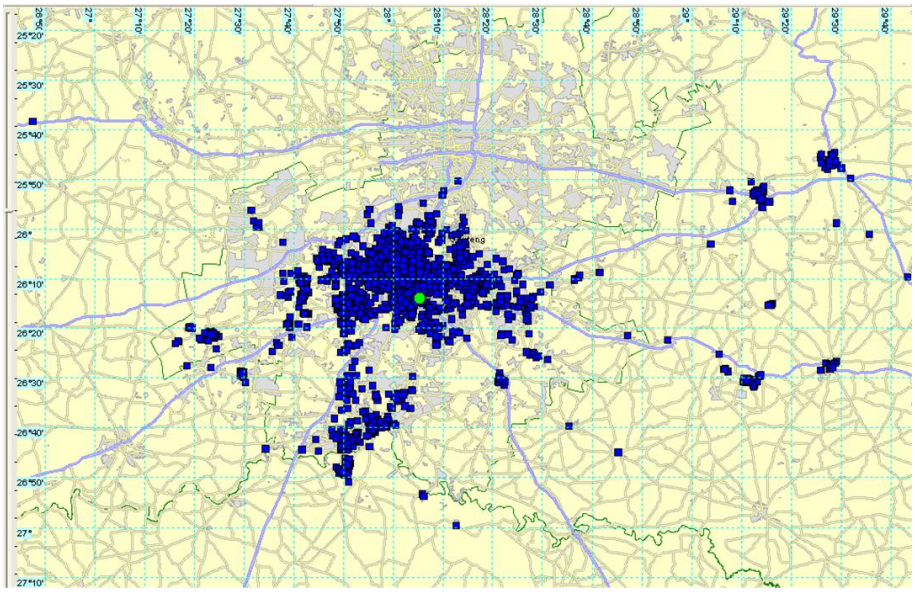


Figure A.3.1 Distribution of the customer locations (indicated by squares) from the test data with respect to the depot (indicated by the dot).

All customers on schedules 4-8 (see Table 1.1) were then routed daily in *FLO* within these predefined groups. The Saturday run included the customers from schedule 3. A summary of the weekly routes thus far is shown in Table A.3.1. The number of remaining vehicles of each type over the week (except Saturday) was then calculated.

Day	Cost	Time (min)	Dist.	Tot. Weight	Ave Time Util.	Ave Space Util.	Wait	3T	4T	7T	10T	16T	No. Veh.
MON	60104	40609	7362	402270	77	69	39	1	4	76	7	0	88
TUE	60703	38573	7730	431001	72	74	15	1	4	79	4	1	89
WED	54564	38775	7247	366532	81	69	24	1	4	71	4	0	80
THU	56437	37428	6882	386068	75	72	20	1	4	72	6	0	83
FRI	62992	41221	7565	507348	75	80	52	1	4	79	8	0	92
SAT	40406	26337	5376	248759	73	65	30	1	4	52	3	0	60
Total	335207	222943	42161	2341978	75	72	180	6	24	429	32	1	492

Table A.3.1 Daily summary of the schedules for all jobs with fixed delivery days.

At this point the following stores remained to be scheduled:

- Schedule 1 and 2 stores from both those delivery groups that have been routed already and those that have not (those squares on the grid that have only schedule 1 and 2 stores).
- Schedule 3 stores that have had their Saturday delivery scheduled and still require their second delivery on either Tuesday or Wednesday to be routed.

The latter stores were routed on the remaining vehicle types and assigned to Wednesday since at this stage Wednesday had the fewest number of used vehicles and this was the only way to increase the number of vehicles without disrupting the balance on other days. All the schedule 2 stores were routed next on the remaining vehicles. These routes were then assigned to different day of the week combinations with the aim of keeping the number of vehicles on each day roughly the same. Finally all single drop customers were added to the routes using *FLO's* 'add jobs to existing routes' feature. No additional vehicles were incurred by doing this and only the existing routes were filled up.

A.4.1 ARPNDD LINGO Model 1

This non-linear model includes the constraint which ensures that stores to be visited remain in the same cluster/delivery group for the duration of the planning period. The notable feature is that the centroid of a route is a free variable and is in fact the point which minimizes the sum of the distances from the centroid to all other delivery points in a group.

```
MODEL:
TITLE ARPNDD;

SETS:

VEHICLES: costkm, fixcost, payload;
CUSTOMERS: wait, schedule, totday, long, lat;
DAYS / 1..6/;
SERVICE(CUSTOMERS, DAYS): visit1, visit2;
AMOUNT(CUSTOMERS, DAYS): weight;
STOPS(VEHICLES, DAYS): drops;
USE(VEHICLES,DAYS): fleet;
CENTROID(VEHICLES,DAYS): x, y;
DISTANCE(VEHICLES,DAYS): dist;
DOKT(VEHICLES,DAYS): depdis;
ROUTEL(VEHICLES,DAYS): leng;
ROUTE_OL_DUR(VEHICLES,DAYS): offload, duration;
ALLOCATION(CUSTOMERS, VEHICLES, DAYS): cvd;
ABS1(CUSTOMERS,VEHICLES,DAYS): abs1x, abs1y;
ABS2(VEHICLES,DAYS): abs2x, abs2y;

ENDSETS

DATA:

xd= 28.087125778;  !depot co-ords;
yd= -26.23219299;
s= 57.5;   !average speed;
l= 10;   !max length route;
DegToKm= 110.5;
DegToHrs= 1.92174;
```

```
VEHICLES, costkm, fixcost, payload =
  @OLE('\LINGO10\VEHICLES15.XLS', 'NAME', 'VARCOST', 'FIXEDCOST', 'LOAD');
CUSTOMERS, wait, schedule, totday, long, lat =
  @OLE('\LINGO10\CUSTOMERS15.XLS', 'NUMBER', 'WTIME', 'SCHED',
   'T', 'LON', 'LATI');
visit1=   @OLE('\LINGO10\YIT15.XLS', 'YITS');
weight=   @OLE('\LINGO10\MIT15.XLS', 'MITS');

ENDDATA


CALC:

@FOR(CUSTOMERS(I):@FOR(DAYS(K)|visit1(I,K) #NE# -1:
visit2(I,K)=visit1(I,K)));
@FOR(CUSTOMERS(I): wait(I) = wait(I)/60);
@FOR(VEHICLES(J): costkm(J) = costkm(J)*57.5);

ENDCALC


[OBJECTIVE] MIN= @SUM(VEHICLES(J):((@SUM(DAYS(K): leng(J,K))*costkm(J)) +
          (@SUM(DAYS(K): fleet(J,K))*fixcost(J))));


@FOR(CUSTOMERS(I): @FOR(DAYS(K):[y_it] @BIN(visit2)));
@FOR(VEHICLES(J): @FOR(DAYS(K): [z_kt] @BIN(fleet)));
@FOR(CUSTOMERS(I): @FOR(VEHICLES(J): @FOR(DAYS(K): [v_ikt] @BIN(cvd))));
@FOR(VEHICLES(J): @FOR(DAYS(K): [v_stops] @GIN(drops)));
@FOR(VEHICLES(J): @FOR(DAYS(K): [centriody_ve] @FREE(y)));
!@FOR(CUSTOMERS(I):@FOR(VEHICLES(J): @FOR(DAYS(K): @FREE(abs1y))));
!@FOR(VEHICLES(J): @FOR(DAYS(K): @FREE(abs2y)));

@FOR(CUSTOMERS(I): @FOR(VEHICLES(J): @FOR(DAYS(K):
   abs1x(I,J,K)>=long(I)-x(J,K))));
@FOR(CUSTOMERS(I): @FOR(VEHICLES(J): @FOR(DAYS(K):
   abs1x(I,J,K)>=x(J,K)-long(I))));
@FOR(CUSTOMERS(I): @FOR(VEHICLES(J): @FOR(DAYS(K):
   abs1y(I,J,K)>=lat(I)-y(J,K))));
@FOR(CUSTOMERS(I): @FOR(VEHICLES(J): @FOR(DAYS(K):
```

```
   abs1y(I,J,K)>=y(J,K)-lat(I)));
@FOR(VEHICLES(J): @FOR(DAYS(K):abs2x(J,K)>=xd-x(J,K)));
@FOR(VEHICLES(J): @FOR(DAYS(K):abs2x(J,K)>=x(J,K)-xd));
@FOR(VEHICLES(J): @FOR(DAYS(K):abs2y(J,K)>=yd-y(J,K)));
@FOR(VEHICLES(J): @FOR(DAYS(K):abs2y(J,K)>=y(J,K)-yd));

@FOR(VEHICLES(J): @FOR(DAYS(K): [n_kt] drops(J,K) =
  @SUM(CUSTOMERS(I): cvd(I,J,K))));
@FOR(VEHICLES(J): @FOR(DAYS(K): [d_bar_kt] dist(J,K) =
  @SUM(CUSTOMERS(I): cvd(I,J,K)*DegToHrs*
   (abs1x(I,J,K) + abs1y(I,J,K)))));
@FOR(VEHICLES(J): @FOR(DAYS(K): [d0kt] depdis(J,K) =
DegToHrs*(abs2x(J,K) + (abs2y(J,K)))));
@FOR(VEHICLES(J): @FOR(DAYS(K): [r_kt] leng(J,K) =
(2*(depdis(J,K)))+dist(J,K)));
@FOR(VEHICLES(J): @FOR(DAYS(K): [o_kt] offload(J,K) =
  @SUM(CUSTOMERS(I): cvd(I,J,K)*wait(I))));
@FOR(VEHICLES(J): @FOR(DAYS(K): [l_kt] duration(J,K) =
leng(J,K) + offload(J,K)));

@FOR(CUSTOMERS(I): [cust_veh] @SUM(VEHICLES(J): @SUM(DAYS(K):
      cvd(I,J,K))) = totday(I));
@FOR(CUSTOMERS(I): @FOR(DAYS(K): [Cus_ve_day]
   @SUM(VEHICLES(J): cvd(I,J,K)) <= 1));
@FOR(VEHICLES(J): @FOR(DAYS(K): [Capacity] @SUM(CUSTOMERS(I):
@SUM(DAYS(K): weight(I,K)*cvd(I,J,K))) <= payload(J)));
@FOR(CUSTOMERS(I): @FOR(DAYS(K): [vy_s] visit2(I,K) =
   @SUM(VEHICLES(J): cvd(I,J,K))));
@FOR(VEHICLES(J): @FOR(DAYS(K): [vz_s] fleet(J,K) =
  @PROD(CUSTOMERS(I): cvd(I,J,K))));
@FOR(VEHICLES(J): @FOR(DAYS(K): [lkt_l] duration(J,K) <= l));


@FOR(CUSTOMERS(I)| schedule #EQ# 1 : visit2(I,1) + visit2(I,2) +
visit2(I,3) + visit2(I,4) + visit2(I,5) = 1); ![Si_yits1a];
@FOR(CUSTOMERS(I)| schedule #EQ# 1 : visit2(I,6) = 0); ![Si_yits1b];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,1) + visit2(I,2) +
                  visit2(I,3) = 1);   ![Si_yits2a];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,4) + visit2(I,5) = 1);
![Si_yits2b];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,1) + visit2(I,5) <= 1);
```

```
![Si_yits2c];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,3) + visit2(I,4) <= 1);
![Si_yits2d];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,6) = 0); ![Si_yits2e];
@FOR(CUSTOMERS(I)| schedule #EQ# 3 : visit2(I,6) = 1);  ![Si_yits3a];
@FOR(CUSTOMERS(I)| schedule #EQ# 3 : visit2(I,1) + visit2(I,4) +
  visit2(1,5) = 0); ![Si_yits3b];
@FOR(CUSTOMERS(I)| schedule #EQ# 3 : visit2(I,2) + visit2(I,3) = 1);
![Si_yits3c];

!Del grp constriants;
@FOR(CUSTOMERS(I): @FOR(CUSTOMERS(L):
@SUM(DAYS(K): visit2(I,K)*visit2(L,K)) = @IF(I #LT# L #AND#
      @SUM(VEHICLES(J): @SUM(DAYS(K): cvd(I,J,K)*cvd(L,J,K))) #GE# 1,
      @SUM(VEHICLES(J): @SUM(DAYS(K): cvd(I,J,K)*cvd(L,J,K))),
      @SUM(DAYS(K): visit2(I,K)*visit2(L,K)))));

END !model;
```

A.4.2 ARPNDD LINGO Model 2

The linear version of the model does not contain the delivery group constraint. Also, the centroid of a route is chosen from a candidate set of points (all existing store locations). The model ran significantly faster than the non-linear version in Appendix 4.1.

```
MODEL:
TITLE ARPNDD 2;

SETS:
 VEHICLES: costkm, fixcost, payload;
 CUSTOMERS: wait, schedule, totday, long, lat;
 DAYS / 1..6/;
 SERVICE(CUSTOMERS, DAYS): visit1, visit2;
 AMOUNT(CUSTOMERS, DAYS): weight;
 USE(VEHICLES, DAYS): fleet;
 DOKT(VEHICLES, DAYS): depdis;
 FIXCENT: flong, flat;
 FXD( FIXCENT, DAYS): fdist;
 CXFXD( CUSTOMERS, FIXCENT, DAYS): cfd;
 VXFXD( VEHICLES, FIXCENT, DAYS): vfd;
ENDSETS

DATA:
 xd= 28.087125778;  !depot co-ords;
 yd= -26.23219299;
 s= 57.5;   !average speed;
 L= 10;   !max length route;
 DegToKm= 110.5;
 DegToHrs= 1.92174;
 locostkm= 1.1;

FIXCENT = 1..5;

VEHICLES, costkm, fixcost, payload =
  @OLE('\LINGO10\VEHICLES15.XLS', 'NAME', 'VARCOST', 'FIXEDCOST', 'LOAD');
CUSTOMERS, wait, schedule, totday, long, lat =
  @OLE('\LINGO10\CUSTOMERS15.XLS', 'NUMBER', 'WTIME', 'SCHED',
```

```
  'T', 'LON', 'LATI');
visit1=   @OLE('\LINGO10\YIT15.XLS', 'YITS');
weight=   @OLE('\LINGO10\MIT15.XLS', 'MITS');
ENDDATA

CALC:
 i = 1;
 @FOR( FIXCENT(f):
   flong(f) = long(i);
   flat(f) = lat(i);
   i = i + 1;
     );

 @FOR(SERVICE(I,K) | visit1(I,K) #NE# -1: visit2(I,K) = visit1(I,K));

 @FOR(CUSTOMERS(I): wait(I) = wait(I)/60);
 @FOR(VEHICLES(J): costkm(J) = costkm(J) *57.5);
 locostkm = locostkm *57.5;
ENDCALC


@FOR( CXFXD( i,f,k): @BIN(cfd(i,f,k)));
@FOR( VXFXD( j,f,k): @BIN(vfd(j,f,k)));

[OBJECTIVE] MIN=
   @SUM( USE(j,k): costkm(J)*2*depdis(j,k)) ! Out and back cost;
   + locostkm* @SUM(FXD(f,k): fdist(f,k))      ! Local travel cost;
   + @SUM( USE(j,k): fixcost(J)* fleet(J,K));  ! fixed cost;

! Can assign at most one vehicle to a FIXCENT each day;
  @FOR(FXD(f,k):
       @SUM(VEHICLES(j):vfd(j,f,k)) <= 1
      );
! Can assign vehicle j to at most one FIXCENT f each day k;
  @FOR(USE(j,k):
       @SUM(FIXCENT(f): vfd(j,f,k)) <= 1
      );
! Force customer i to be visited correct number of times;
  @FOR(CUSTOMERS(i):
       @SUM(FXD(f,k): cfd(i,f,k)) = totday(i)
      );
```

```
! Set fleet(j,k) = 1 if vehicle j used on day k;
  @FOR(USE(j,k):
      fleet(j,k) = @SUM( FIXCENT(f): vfd(j,f,k))
      );
! Compute local travel time from centroid f on day k;
  @FOR(FXD(f,k):
       fdist(f,k) =
       @SUM(CUSTOMERS(i): cfd(i,f,k)*DegToHrs*(@ABS(long(i)-flong(f))
       +@ABS(lat(i)-flat(f))))
      );
! Compute depot to centroid time for vehicle j on day k;
  @FOR(USE(j,k):
       depdis(j,k) = @SUM( FIXCENT(f):
       DegToHrs*(@ABS(xd - flong(f))+@ABS(yd - flat(f)))*vfd(j,f,k));
      );
! Weight of customers assigned to FIXCENT(f) must not exceed payload
  capacity of vehicle assigned to FIXCENT f;
  @FOR(FXD(f,k):
       @SUM(CUSTOMERS(i): weight(i,k)*cfd(i,f,k)) <= @SUM(VEHICLES(j):
       payload(j)* vfd(j,f,k))
      );
! Compute visits (0 or 1) to customer i on day k;
  @FOR(AMOUNT(i,k):
       visit2(I,K) = @SUM(FIXCENT(f): cfd(I,f,K))
      );
! Trip length constraint on day k at center f:
      out and back time to f
      + local delivery time assigned to f
      + wait time assigned to f <= delivery time provided to f;
  @FOR(FXD(f,k):
      @SUM( VEHICLES(j):vfd(j,f,k)*2*DegToHrs*(@ABS(xd - flong(f))+
      @ABS(yd - flat(f))))
      + fdist(f,k)
      + @SUM(CUSTOMERS(i): cfd(i,f,k)*wait(i)) <=
      L*@SUM(VEHICLES(j): vfd(j,f,k)));

! For customers on special schedules to get visited on correct days;
@FOR(CUSTOMERS(I)| schedule #EQ# 1 : visit2(I,1) + visit2(I,2) +
visit2(I,3) + visit2(I,4) + visit2(I,5)>= 1);  ![Si_yits1a];
@FOR(CUSTOMERS(I)| schedule #EQ# 1 : visit2(I,6) = 0);  ![Si_yits1b];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,1) + visit2(I,2) +
```

```
   visit2(I,3) >= 1);  ![Si_yits2a];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,4) + visit2(I,5) >= 1);
![Si_yits2b];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,1) + visit2(I,5) <= 1);
![Si_yits2c];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,3) + visit2(I,4) <= 1);
![Si_yits2d];
@FOR(CUSTOMERS(I)| schedule #EQ# 2 : visit2(I,6) = 0);  ![Si_yits2e];
@FOR(CUSTOMERS(I)| schedule #EQ# 3 : visit2(I,6) = 1);   ![Si_yits3a];
@FOR(CUSTOMERS(I)| schedule #EQ# 3 : visit2(I,1) + visit2(I,4) +
   visit2(1,5) = 0);  ![Si_yits3b];
@FOR(CUSTOMERS(I)| schedule #EQ# 3 : visit2(I,2) + visit2(I,3) >= 1);
![Si_yits3c];


END !model;
```

A.5 Pseudo-code for clustering function

The function takes as input the following information per customer:

1. Customer number
2. Longitude and latitude
3. Expected waiting times per day
4. Expected job sizes per day
5. Number of times a customer is serviced per week (b/w 1 & 5)
6. Schedule type (see Table 1.1)
   (The following fields are then calculated:)
7. Average job size
8. Travel time to the depot
9. Total mass delivered over the week

and returns as output per customer:

1. Customer number
2. Delivery group (cluster)
3. Delivery days where applicable

Separate files are also produced to import the job sets and customer delivery groups into *FLO* for detailed routing in a postprocessing phase.

```
    Begin Function
(parameters:- input data, number of random iterations, number of clusters,
length of day, average travel speed, maximum vehicle size, objective
function parameters)

Extend number of clusters value to allow for range of numbers of clusters
For (lower to upper limit of the number of clusters)
```

```
     Determine average work time per day per cluster

     For (1 to iterations)
          Call R's built in k-means clustering algorithm
          Seed points are customers with highest average drop size in
            each cluster
          Separate customer data for seed and non-seed customers
          Populate matrix of distances from each non-seed point to seeds
          Set initial job load and time (wait time + travel time) per cluster
            per day based on seeds
          Calculate mean and standard deviation of job load and deviation
            of wait time from the average for each cluster

          For (1 to 5 different orderings of customer input data)
               For (1 to number of non-seed customers)
                    For (1 to number of clusters)
                         Add customer to cluster and appropriate schedule
                           day(s)
                         Temporarily modify cluster's job load and wait
                           time
                         Re-calculate coefficient of variation of job
                           loads & average deviation of wait time
                         Evaluate first objective function
                    End For

                    Find cluster which minimizes objective function
                    Update chosen cluster's job load & wait time
                      characteristics
               End For

               Evaluate second objective function
               If objective function is better then the best so far
                    Overwrite stored data corresponding to best solution
               End If
          End For

     Plot convex hulls around each cluster and bar charts for job
       load & wait time
     End For

End For
Write output data to file
End Function
```

Bibliography

[AG88] A.A. Assad and B.L. Golden. *Vehicle Routing Methods and Studies*. North Holland, Netherlands, 1988.

[Bal88] M.O. Ball. *Vehicle Routing Methods and Studies*, chapter Allocation/Routing: Models and algorithms, pages 199–221. North Holland, Netherlands, 1988.

[BB74] L. Bodin and E. Beltrami. Networks and vehicle routing for municipal waste collection. *Networks*, 4:65–94, 1974.

[BC84] J.E. Beasley and N. Christofides. The period routing problem. *Networks*, 14(2):237–256, 1984.

[Bea84] J.E. Beasley. Fixed routes. *The Journal of the Operational Research Society*, 35(1):49–55, 1984.

[Bel98] A.S. Belenky. *Operations Research in Transportation Systems*. Kluwer Academic Publishers, Netherlands, 1998.

[BP02] Geunes J. Balakrishnan, A. and M.S. Pangburn. *Supply Chain Management: Models, Applications and Research Directions*, chapter Coordinating the distribution chain: New models for new challenges. Kluwer Academic Publishers, Dordrecht, 2002.

[BPS06] Ferreira C. Barreto, S., J. Paixao, and B.S. Santos. Using clustering analysis in a capacitated location-routing problem. European Journal of Operations Research, 2006.

[BSL97] J. Bramel and D. Simchi-Levi. *The Logic of Logistics*. Springer, United States of America, 1997.

[Cor05] G. Cormier. *Logistics Systems Design and Optimization*, chapter Operational research methods for efficient warehousing, pages 93–122. 2. Springer, United States of America, 2005.

[CS05] Gendreau M. Hertz A. Laporte G. Cordeau, J.F. and J.S. Sormany. *Logistics Systems Design and Optimization*, chapter New heuristics for the vehicle routing problem, pages 279–297. 2. Springer, United States of America, 2005.

[Dav91] L. Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

[Eve74] B. Everitt. *Cluster Analysis*. Heinemann, London, 1974. Pg. 48-51.

[FS06] P. Francis and K. Smilowitz. Modeling techniques for periodic vehicle routing problems. Transportation Research, 2006.

[FT05] Smilowitz K. Francis, P. and M. Tzur. The period vehicle routing problem with service choice. 2005.

[FT06] Smilowitz K. Francis, P. and M. Tzur. Flexibility and complexity in periodic distribution problems. 2006.

[Gol97] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, United States of America, 1997.

[Gol02] D.E. Goldberg. *The Design of Innovation*. Kluwer Academic Publishers, United States of America, 2002.

[HS88] Rinnooy Kan A.H.G. Haimovich, M. and L. Stougie. *Vehicle Routing Methods and Studies*, chapter Analysis of heuristics for vehicle routing problems. North Holland, Netherlands, 1988.

[JD88] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[JO88] P. Jaillet and A.R. Odoni. *Vehicle Routing Methods and Studies*, chapter The probabilistic vehicle routing problem, pages 293–317. North Holland, Netherlands, 1988.

[Kar77] R.M. Karp. Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane. *Mathematics of Operations Research*, 2(3):209–224, 1977.

[LC05] Riopel D. Langevin, A. and J.F. Campbell. *Logistics Systems Design and Optimization*, chapter The network of logistics decisions, pages 1–38. 2. Springer, United States of America, 2005.

[Mil06] S.J. Miller. An introduction to linear programming. www.math.brown.edu/ sjmiller/54/index.htm, August 2006.

[MSS84] Rinnooy Kan A.H.G. Marchetti-Spaccamela, A. and L. Stougie. Hierarchical vehicle routing problems. *Networks*, 14:571–586, 1984.

[ND86] G.F. Newell and C.F. Daganzo. Design of multiple-vehicle delivery tours - i: A ring-radial network. *Transportation Research B*, 20(345-363), 1986.

[NS88] Greenberg P. Bolkan W.E. Nygard, K.E. and E.J. Swenson. *Vehicle Routing Methods and Studies*, chapter Generalized assignment methods for the deadline vehicle routing problem. North Holland, Netherlands, 1988.

[NS91] S.J. Noronha and V.S.S. Sarma. Knowledge-based approaches for scheduling problems:a survey. *IEEE Transactions on Knowledge and Data Engineering*, 3(2):160–171, 1991.

[NV05] Shcherbina O. Huyer W. Neumaier, A. and T. Vinko. A comparison of complete global optimization solvers. *Mathematical programming*, 103(2):335–356, June 2005.

[RI79] R. Russell and W. Igo. An assignment routing problem. *Networks*, 9:1–17, 1979.

[S.A06] Clover S.A. www.clover.co.za, October 2006.

[Sch03] H. Schichl. *Mathematical Modeling and Global Optimization*. Habilitation thesis, 2003.

[Sys06a] Lindo Systems. www.lindo.com, December 2006.

[Sys06b] OPSI Systems. www.opsisystems.com, October 2006.

[TB84] C.R.R. Tan and J.E. Beasley. A heuristic algorithm for the period vehicle routing problem. *Omega International Journal of Management Science*, 12(5):497–504, 1984.

[Tea05] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. ISBN 3-900051-07-0.

[Tha95] S.R. Thangiah. *Applications handbook of genetic algorithms:New frontiers*, volume 2, chapter Vehicle routing with time windows using genetic algorithms, pages 253–277. CRC Press, Florida, 1995.

[Wik07a] Wikipedia. Cluster analysis — wikipedia, the free encyclopedia, 2007. [Online; accessed 9-October-2007].

[Wik07b] Wikipedia. Genetic algorithm — wikipedia, the free encyclopedia, 2007. [Online; accessed 18-November-2007].