Chapter 2

Literature Review

[Extracted from C. Hjorring, The Vehicle Routing Problem and Local Search Metaheuristics, PhD thesis, Department of Engineering Science, The University of Auckland, 1995]

2.1 Early work

The VRP was first formulated by Dantzig and Ramser [DR59] who called it the truck dispatching problem. They present a heuristic method that starts with an initial solution where each vehicle services only one customer. In a number of stages the customers are aggregated into groups. However, due to restrictions imposed the method can link together customers that are geographically distant. The method puts more emphasis on filling vehicles to near capacity than on minimising distance.

Clarke and Wright [CW64] present an improved method which is now commonly known as the savings method. As in Dantzig and Ramser's method, the algorithm starts with an initial solution where each vehicle services only one customer. The method then computes the distance "savings" achieved by combining the end of a route with the start of another route, for all feasible route combinations. The routes with the greatest savings are combined, and the savings are updated. This continues until routes can no longer be combined due to the capacity constraints.

The method is simple and easy to implement and has formed the basis of several computerised routing packages. The method produces better results than [DR59] but the results are still far from optimal for most problems. Suppose that we wish to find the customer that maximises the savings when combined with customer i. The method will often choose a customer that is distant from the depot but relatively distant from i, in preference to a customer that is closer to the depot than i but also close to i. Many researchers have tried to overcome these and other problems by modifying the savings measure and other aspects of the method. The savings method and some of its modifications, along with other heuristics approaches are further discussed in Section 2.3.

Balinski and Quandt [BQ64] solved some small VRPs to optimality using a set partitioning formulation. They minimised

$$\sum_{j=1}^{r} c_j x_j \tag{2.1}$$

subject to

$$\sum_{j=1}^{r} a_{ij} x_j \quad = 1 \quad i = 1, \dots, n \tag{2.2}$$

$$\text{Where:}\quad \begin{aligned} a_{ij} &= \begin{cases} 1, \text{ if customer } i \text{ is serviced by route } j \\ 0, \text{ otherwise} \end{cases} \\ x_j &= \begin{cases} 1, \text{ if route } j \text{ is used} \\ 0, \text{ otherwise} \end{cases} \\ c_j &= \text{ is the cost of route } j \end{aligned}$$

Constraint 2.2 ensures that each customer is serviced exactly once. There are r columns in the constraint matrix each of which correspond to a feasible route. Using a dominance criterion (remove column j if there exists a set of columns S such that $\sum_{i \in S} c_i \leq c_j$) to reduce the number of columns, and Gomory cutting planes to force integrality, Balinski and Quandt were able to solve to optimality 9 out of 10 problems from an actual business situation. The tenth problem was not solved within the prescribed limit of 200 simplex iterations. The problems considered were very small, ranging from 5 to 15 customers. A major difficulty with the set partitioning formulation is that the number of feasible routes (columns) for most problems is enormous. Just enumerating all the columns will take a considerable amount of time. To solve the problem to optimality a TSP needs to be solved for each column in order to determine the c_j, and then the set partitioning problem must be solved.

In the following sections we review some of the important exact and heuristic methods for the VRP, and then consider some of the extensions to the VRP. A more extensive survey of the routing and scheduling literature is presented by Bodin et al. [BGAB83]. Exact algorithms for the VRP are surveyed in detail by Laporte and Nobert [LN87]. Recent exact and heuristic methods are briefly reviewed by Laporte [Lap92].

Throughout this chapter a notation consistent with that of the formulation from Section 1.1 is used, i.e. n customers with the depot indexed by 0, K vehicles, order size of customer i denoted by q_i and capacity of a vehicle denoted by Q.

2.2 Exact Methods

As noted in Section 1.2 the VRP is $\mathcal{NP}$-hard. Thus exact methods that will run in a reasonable amount of computing time are difficult to develop. Nevertheless several methods have been developed that will solve problems with up to 50 customers, and one method that has solved a 100 customer problem. Exact methods for the VRP can be classified into three approaches:

1. Direct tree search
2. Dynamic Programming (DP)
3. Integer Linear Programming (ILP)

Most work has concentrated on the ILP approach. This can be further categorised by the formulation used as being based on one of the following formulations:

1. Set Partitioning
2. Vehicle flow
3. Commodity flow

2.2.1 Direct Tree Search Algorithms

Direct tree search methods consist of incrementally building vehicle routes by means of a branch and bound tree. Christofides and Eilon [CE69] present a branch and bound approach in which routes are built up edge by edge. The problem is transformed into a TSP by creating $K-1$ artificial depots and adding them to the problem. A branch in the tree is created by either including or excluding an edge. A branch is fathomed if any of the routes are infeasible or if it is impossible to service the unrouted customers. A minimal spanning 1-tree, a well known relaxation of the TSP, is used to determine bounds for nodes of the search tree. The 1-tree relaxation was found to produce tighter bounds than the assignment relaxation. The algorithm was tested on a 6 customer problem and a 13 customer problem. The problems were solved in 1.5 and 15 minutes respectively on an IBM 7090. The algorithm was not tested on larger problems (starting from 21 customers) because the computation time and memory space requirements became prohibitive.

In [Chr76], Christofides describes a different branching scheme. Instead of branching on edges the algorithm branches on routes. Going down the tree one level corresponds to adding one new route. Thus the tree will have at most K levels. At each node a list of routes passing through customer i is generated to branch on. Obviously the performance of the algorithm is directly dependent on the number of routes generated at each node. Customer i should be chosen so that the number of routes is small, i.e. a customer with a large order or one that is distant from other customers. Christofides fathoms route branches by;

- searching for constraint violations,
- using a lower bound to show that selecting the branch cannot lead to a better solution than the current best feasible solution, and,
- by applying dominance tests.

The largest problem solved to optimality contained 31 customers.

The performance of a branch and bound algorithm is strongly influenced by the quality of the lower bounds computed at the nodes of the search tree. A sharp bound can dramatically reduce the number of nodes that need to be searched. Christofides et al. [CMT81a] propose two bounds for the VRP.

The first bound uses a *k-degree centre tree* (k-DCT) as a relaxation of the VRP. A k-DCT is a spanning tree where the number of edges incident to the depot is constrained to k. Christofides et al. note that the edge set of a VRP solution (denoted as E) can be partitioned into 3 subsets:

1. E_1 : edges forming a k-DCT (where $k = 2K - y$ and $K_1 \leq y \leq K$)
2. E_2 : y edges incident to the depot
3. E_3 : $K - y$ edges not incident to the depot

where K_1 is a lower bound on the maximum number of single customer routes in the optimal VRP solution. Note that if y is set to some value less than K_1 then the k-DCT relaxation is no longer valid as it would preclude VRP solutions with K_1 single customer routes. Christofides et al. described two bounds, based on capacity and distance violations, for determining K_1.

Let ξ_l^t $(t = 1, 2, 3; l \in E)$ be 0-1 variables taking the value 1 if and only if edge l belongs to E_t. Christofides et al. show that a valid lower bound to the VRP is given by

$$\sum_{l \in E} c_l(\xi_l^1 + \xi_l^2 + \xi_l^3) \tag{2.3}$$

For a set of ξ_l^t to be a valid solution to the VRP the continuity constraints

$$\sum_{l \in A_i}(\xi_l^1 + \xi_l^2 + \xi_l^3) = 2 \qquad i = 1, \dots, N \tag{2.4}$$

where A_i is the set of edges incident to vertex i, must be satisfied. Christofides et al. improve the bound of Equation 2.3 by incorporating the continuity constraints into the objective in a Lagrangean fashion.

The second bound is based on *q-routes*. Let W be the set of all possible loads that could exist on any vehicle route and let $q(l)$ be the value of the l'th element of W. Define $\psi_l(i)$ as the value of the least cost path

- passing through customer i
- starting and ending at the depot
- having no loop of the form (i_1, i_2, i_1)
- having a total weight $q(l)$

Christofides et al. term this a q-route. They then prove that

$$\sum_{i=1}^{n} \min_{l=1,\dots,|W|} \left(\frac{\psi_l(i) q(i)}{q(l)} \right) \tag{2.5}$$

is a valid lower bound to the VRP. They also show that the $\psi_l(i)$ can be efficiently calculated through a shortest path type procedure.

The two bounds were embedded in branch and bound algorithms and applied to a set of 10 VRPs ranging from 10 to 25 customers. For the k-DCT bound two branching strategies were used; branching on edges and branching on routes. The q-route bound was only tested with the edge branching strategy. For the k-DCT bound the route branching scheme was generally superior. However the q-route bound with edge branching performed best overall, finding optimal solutions to all 10 problems in the allotted time limit of 250 seconds on a CDC 7600 computer.

Recently Fisher [Fis94] has developed a very successful approach that uses K-trees to provide lower bounds. A K-tree and a k-DCT are similar in that they are both VRP adaptations of the spanning tree relaxation proposed by Held and Karp [HK70] for the TSP. For a VRP with n customers a K-tree is defined as a set of $n + K$ edges that span all vertices. To further strengthen the bound the degree of the depot in the K-tree is forced to $2K$. Note that this prohibits single customer routes, since single customers routes use the same edge for depot departure and arrival. However, Fisher shows that the optimal solution cannot contain single customer routes for 10 of the 12 test problems considered. The remaining two problems are unlikely to contain a single customer route in the optimal solution since the deletion of the largest customer and a vehicle would result in a very tightly constrained problem.

Let x_{ij} be 1 if edge (i, j) is in a solution, 0 otherwise. Because edges are undirected Fisher adopts the convention that x_{ij} and x_{ji} denote the same variable. Define $x = (x_{01}, x_{02}, \dots, x_{0n}, x_{12}, \dots, x_{n-1,n})$ and let $X = \{x\}$ represent a K-tree with degree $2K$ at the depot. Fisher then formulates the VRP as:

$$\min_{x \in X} \sum c_{ij} x_{ij} \tag{2.6}$$

such that

$$\sum_{j=0, j\neq i}^{n} x_{ij} \quad = 2 \qquad i = 1, \dots, n \tag{2.7}$$

$$\sum_{i \in S} \sum_{j \notin S} x_{ij} \quad \geq 2r(S) \quad \text{for all } S \subseteq N, |S| \geq 2 \tag{2.8}$$

where $r(S)$ is a lower bound on the number of vehicles required to service the customers in the set S. Constraints 2.7 and 2.8 represent the continuity and vehicle constraints, respectively. Fisher incorporates these constraints into 2.6 to form a Lagrangean relaxation of the VRP. Because there are $O(2^n)$ constraints in the set 2.8, Fisher generates these constraints dynamically as they are violated.

Fisher solves the Lagrangean relaxation using subgradient optimisation and obtained tight lower bounds, much tighter than the bounds in [CMT81a]. Fisher presented three heuristics that take the K-tree from the end of each subgradient iteration and transform it into a feasible solution to the VRP. As the K-tree lower bound approaches the optimal solution these heuristics are able to find better and better upper bounds.

Fisher tested the approach on a set of 12 test problems. Six from [CMT79] (the first five problems and the clustered "realistic" 100 customer problem), and six problems (ranging from 25 to 134 customers) taken from real world applications. Three of the real world problems (25–36 customers) were solved to optimality using only the Lagrangean relaxation and the three heuristics. Two further real world problems and the clustered 100 customer problem were solved to optimality using a branch and bound scheme with manual intervention that took advantage of the clustering of the customers.

While the approach worked well for clustered problems it was less impressive on problems in which the customers were uniformly distributed. For clustered problems the gap between the lower and upper band was on average 0.8%, while the gap was 9% for the uniform problems. Fisher also describes how the approach can be extended to cope with asymmetric costs, time windows, and a heterogeneous fleet. The computational performance of the approach on these problems is currently an open question.

2.2.2 Dynamic Programming Algorithms

In addition to direct tree search algorithms, Christofides et al. [CMT81b] have published work on a dynamic programming approach to solving the VRP exactly. Dynamic programming algorithms start at some initial state and step through to some destination state in a number of stages. At each stage a number of intermediate states need to be considered. For the approach to be computationally feasible the number of states must be kept to a minimum. Christofides et al. describe feasibility criteria, based on order quantities, that reduce the number of states. These reductions are not sufficient to allow medium sized VRPs to be solved exactly so Christofides et al. introduced a *state-space relaxation* to reduce the number of states. Because a state-space relaxation maps the original state space to a state space with smaller cardinality it can only provide a lower bound on the optimal solution. To obtain optimal VRP solutions Christofides et al. embed the method in a branch and bound scheme. They present three dynamic programming schemes, one using q-routes as the mapping function. Using this scheme, the bound produced for the 10 customer problem from [CMT81a] is in fact the optimal solution. For the other nine problems the ratio of lower bound/optimum varied between 93.1% and 99.6%.

2.2.3 Set Partitioning Formulations

The set partitioning approach to solving VRPs forms a constraint matrix in which columns represent feasible routes and rows correspond to customers. The first published method to solve the VRP using a set partitioning formulation was that of Balinski and Quandt [BQ64]. The method, described on page 11, was only applied to very small problems. Even for those problems the method failed in one case to find the optimal solution within the allotted iteration limit.

Foster and Ryan [FR76] present an optimal approach that uses a set partitioning formulation and column generation. The approach uses the petal method, discussed in Chapter 4, to generate an initial feasible solution. It then uses the shadow prices, or dual variables, of the current solution to generate a new column (route) that will improve the current solution. If none can be found the method has found an optimal solution. Columns are generated by solving a dynamic program in which the stages are given by the distance travelled and the states are given by the customers serviced. Unfortunately the approach converges slowly, and is therefore only suitable for small problems.

Agarwal et al. [AMS89], inspired by the work of Foster and Ryan, present another set partitioning approach that uses column generation. The method uses a different column generator, along with other refinements, in order to increase the convergence rate. Let $\mathbf{y}$ represent the new column to be generated, i.e. $y_i = 1$ if customer i is serviced, 0 otherwise. Agarwal et al. show that the following subproblem can be used to generate a column:

$$\text{minimise } f(\mathbf{y}) - \sum_{i=1}^{n} \pi_i y_i \tag{2.9}$$

subject to

$$\sum_{i=1}^{n} q_i y_i \le Q \tag{2.10}$$

where $f(\mathbf{y})$ is the cost of the optimal TSP tour associated with $\mathbf{y}$ and π_i is the dual variable associated with the i-th constraint. Because $f(\mathbf{y})$ is a very complex function Agarwal et al. replace it with a linear lower bound. The subproblem is then reduced to a knapsack problem, which is relatively easy to solve. Various methods are used estimate the dual variables, significantly reducing computation times. The algorithm was compared with the algorithm of Christofides et al. [CMT81a] on seven problems from [CMT81a], ranging in size from 15 to 25 customers. The algorithm was able to solve all problems to optimality and was roughly 13 times faster than that of [CMT81a].

2.2.4 Vehicle Flow Formulations

The formulation given in Section 1.1 has been used by Golden et al. [GMN77] to solve asymmetrical VRPs. The formulation is known as a three-index formulation because of the three indices associated with the decision variables, x_{ij}^v. For

a homogeneous fleet a more compact formulation, known as a two-index formulation, can be defined by discarding the vehicle index. Two-index formulations have primarily been used for symmetric VRPs. The following formulation for a VRP with capacity and time constraints is adapted from Laporte et al. [LND85]. The cost matrix is assumed to satisfy the triangle inequality.

$$\text{minimise } \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij} \tag{2.11}$$

subject to

$$\sum_{j=1}^{n} x_{0j} = 2K \tag{2.12}$$

$$\sum_{i=0}^{i<k} x_{ik} + \sum_{j=k+1}^{n} x_{kj} = 2 \qquad k = 1, \ldots, n \tag{2.13}$$

$$\sum_{i,j \in S} x_{ij} \le |S| - V(S) \qquad S \subseteq \{1, \ldots, n\}, |S| \ge 3 \tag{2.14}$$

$$x_{ij} = \begin{cases} 0, 1, 2 & i = 0, j \in \{1, \ldots, n\}, c_{ij} \le 1/2 T \\ 0, 1 & i, j \in \{1, \ldots, n\} \end{cases} \tag{2.15}$$

where $V(S)$ is a lower bound on the number of vehicles required to service all the customers in S. For this formulation x_{ij} indicates the number of times (0, 1, or 2) edge (i, j) $(i < j)$ is used in the optimal solution. An edge with $x_{ij} = 2$ represents a single customer route. The x_{ij} are only defined if the following three conditions hold:

1. $i < j$ (x_{ij} is interpreted as x_{ji} whenever $i > j$)
2. $q_i + q_j < Q$
3. $t_{0i} + t_{ij} + t_{j0} + \delta_i + \delta_j \le T$

Constraints 2.12 and 2.13 specify the degree of the vertices while constraints 2.14 prohibit infeasible routes, i.e. routes that

- are disconnected from the depot, or
- violate capacity constraints, or
- violate time constraints.

Laporte et al. [LND85] solve symmetric VRPs using a constraint relaxation algorithm based on the above formulation. The integrality and subtour constraints are first relaxed; integrality is obtained by branching on the x_{ij}; and infeasible routes are removed by dynamically generating constraints of the form of 2.14. With a time limit of 500 seconds on a CYBER 173 the algorithm was able to solve problems with up to 50 customers exactly.

2.2.5 Commodity Flow Formulations

A less common integer programming approach is the commodity flow formulation. In this formulation there are additional variables, y_{ij}, which represent the quantity of goods travelling on arc (i, j). This type of formulation was first proposed by Garvin et al. [GCJS57] for an oil delivery problem. Variants were later developed and analysed by Gavish and Graves [GG79, GG82]. The following formulation for a VRP without constraints on route times is due to [GG79]

$$\text{minimise} \sum_{i=0}^{n}\sum_{j=0}^{n} c_{ij}x_{ij} \tag{2.16}$$

subject to

$$\sum_{i=0}^{n} x_{ij} = 1 \qquad (j = 1, \dots, n) \tag{2.17}$$

$$\sum_{j=0}^{n} x_{ij} = 1 \qquad (i = 1, \dots, n) \tag{2.18}$$

$$\sum_{i=1}^{n} x_{i0} = \sum_{j=1}^{n} x_{0j} = K \tag{2.19}$$

$$\sum_{i=0}^{n} y_{ij} - \sum_{i=0}^{n} y_{ji} = q_j \qquad (j = 1, \dots, n) \tag{2.20}$$

$$q_j x_{ij} \le y_{ij} \le (Q - q_i)x_{ij} \qquad (i, j = 1, \dots, n) \tag{2.21}$$

$$x_{ij} \in \{0, 1\} \qquad (i, j = 1, \dots, n) \tag{2.22}$$

Constraints 2.17, 2.18 and 2.22 ensure that each customer is serviced exactly once. Constraints 2.19 and 2.22 specify that there are K routes. Constraints 2.20 are flow balancing constraints that ensure that each customer's order is satisfied exactly. Infeasible routes are prevented by constraints 2.20 and 2.21. Constraint 2.21 links the y_{ij} with the x_{ij}, ensuring that goods can only be transported through an arc if a vehicle uses the arc. The constraint also eliminates routes that exceed the vehicle capacity.

Gavish and Graves [GG79] present results for various relaxations of the above formulation on problems ranging from 10 to 30 customers. Fairly large gaps were present between the VRP optimum and the LP optimum for $n \ge 15$. No problems were solved to optimality.

2.3 Heuristic Methods

There have been a great number of heuristics published for the VRP. In this section we survey some of these heuristics, grouping them according to the following

categories:

1. Order-Partition Methods
2. Construction Methods
3. Mathematical Programming Based Methods
4. Iterative Improvement Methods
5. Interactive Methods

This categorisation is somewhat approximate as some heuristics use ideas from several of the above schemes. For example, we have classified the repeated matching heuristic of Wark and Holt [WH94] as a construction method, due to its similarity to the parallel savings method of Altinkemer and Gavish [AG91]. However, the repeated matching heuristic occasionally breaks apart the routes in an effort to find better groupings, and could thus be classified as an iterative improvement method.

Order-partition methods sort the customers into some order, and then partition the ordering into contiguous subsets, such that each subset forms a feasible route. These methods are discussed in detail in Section 4.1. The remaining four approaches are now discussed.

2.3.1 Construction Methods

The most popular construction method is the savings method of Clarke and Wright [CW64]. As has already been mentioned, the method computes the "savings" achieved by combining the end of a route with the start of another route, for all feasible route combinations. Clarke and Wright considered four possible route structures that could eventuate from linking customers i and j. They showed that none of the new structures will be profitable unless either i or j (or both) were at the start or end of a route, thus the method restricts attention to customers at the start or end of a route. In this case the savings, s_{ij}, is given by

$$s_{ij} = c_{0i} + c_{0j} - c_{ij} \tag{2.23}$$

Gaskell [Gas67] analysed the savings measure by examining lines of equal savings. For the standard saving measure, s_{ij}, given by Equation 2.23, the lines of equal savings are hyperbole as shown in Figure 2.1a. From Figure 2.1a we see that customers that are distant from A, as measured by the straight line distance, are often preferred to closer customers. For instance, P_1 will have a greater savings value than P_2. Gaskell notes that most planners would prefer P_2 over P_1 and consequently considers a number of other measures. Of these, two measures were found to give good results; they were:

- $\pi_{ij} = s_{ij} - c_{ij}$, and,
- $\lambda_{ij} = s_{ij}(\bar{c} + |c_{0i} - c_{0j}| - c_{ij})$, where $\bar{c}$ = average c_{0i}.

Contours for π_{ij} are shown in Figure 2.1b and for λ_{ij} in Figures 2.1c and 2.1d.

In [CW64], multiple routes are generated at the same time, each customer joins the route which maximises its savings (as long as no constraints are violated). Gaskell termed this the multiple approach, which is now popularly known as the parallel savings method. He compared this with a sequential approach where customers are added onto a single route until the route reaches capacity or time limits. A new customer is then chosen as the seed for a new route and the process continued. Gaskell found that the multiple approach is usually superior to the sequential approach.

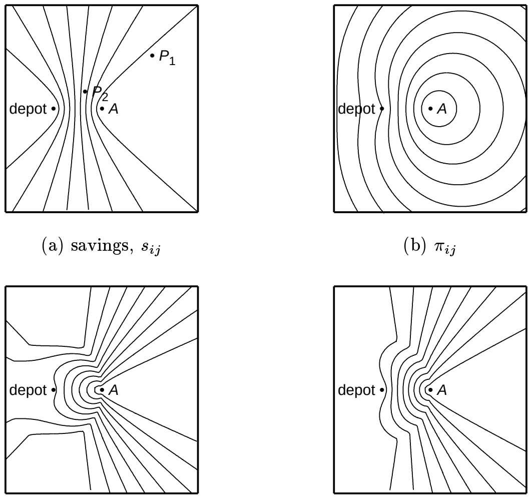
Yellow [Yel70] defines a generalised savings value, s'_{ij}, given by

$$s'_{ij} = c_{i0} + c_{j0} - \gamma c_{ij} \tag{2.24}$$

where γ is the route shape parameter. Special cases for γ are:

- $\gamma = 1$, the standard savings approach, s_{ij}
- $\gamma = 2$, Gaskell's π method, π_{ij}

As γ is increased, greater emphasis is placed on the distance between the customers, rather than their positions relative to the depot. Figure 4.6c on page 63 shows contours for $\gamma = 1.2$. Yellow also shows that the computational efficiency of the method can be improved "by using a simple geometrical search technique on an ordered list of the polar coordinates of the" customers.



(a) savings, s_{ij} (b) π_{ij}

(c) λ_{ij}, $\bar{c} \le c_{0A}$ (d) λ_{ij}, $\bar{c} > c_{0A}$

Figure 2.1: Lines of equal "closeness" from customer A for different distance metrics

Paessens [Pae88] defines another savings measure, termed the *parametrical savings function* which is defined by

$$s^*_{ij} = c_{0i} + c_{j0} - \gamma c_{ij} + \eta |c_{0i} + c_{j0}| \tag{2.25}$$

The $\eta|c_{0i} + c_{j0}|$ component encourages the matching of customers whose inter-depot distances differ markedly. Suppose i is a customer distant from the depot, j is a customer close to the depot, and $\theta_i \approx \theta_j$. This should be a good pairing, but s_{ij} will be small. The $\eta|c_{0i}+c_{j0}|$ component of s^*_{ij} corrects for this. However, when the customer angles are not close, the $\eta|c_{0i} + c_{j0}|$ component will still encourage the pairing. A savings measure incorporating the θ_i may give better results.

Paessens repeats the algorithm several times with different values for γ and η, chosen within the ranges

$$0 < \gamma \le 3, \quad 0 \le \eta \le 1$$

This produces differing solutions, thus increasing the chance of finding high quality solutions. Varying γ and η produced better results than the standard savings measure for a set of standard test problems, reducing the average gap from the best known solutions from 3.2% down to 0.7%.

Altinkemer and Gavish [AG91] use a matching-based procedure to merge multiple routes in each step. The algorithm starts with a solution where each vehicle services one customer and then iterates, at each step solving a maximum cost weighted matching problem. The solution is represented on a weighted graph in which vertices represent routes (clusters), and edges connect vertices whose corresponding clusters can be feasibly merged. The weight of an edge is the savings realised by merging the two clusters. Altinkemer and Gavish present three variants of the algorithm:

PSA1 the savings are calculated exactly by solving a TSP on the newly merged cluster

PSA2 the savings are approximated by joining the routes at their endpoints, no further reordering of the customers is performed

PSA3 as in PSA2 except that for the final clusters a TSP is solved

One problem with the algorithm is that it can reach a situation where each cluster has a load greater than $Q/2$, but much less than Q. In this case no clusters can be merged even though vehicle utilisation is low, and the solution is far from optimal. To avoid this problem, Altinkemer and Gavish introduce dummy clusters to slow the growth of the real clusters. This has the effect of matching only T real clusters in each iteration, where T is a user defined parameter. Values of T from 2 up to 12 were trialled on the 14 test problems of [CMT79]. Altinkemer and Gavish found no relationship between T and solution quality so they suggest trying all values and picking the best. PSA1 and PSA3 find different solutions for most problems, on average the quality of the solutions found by the two variants is similar. Overall the algorithm finds better solutions than earlier heuristics (pre-1990) but is outperformed by more recent heuristics.

Wark and Holt [WH94] describe another application of repeated matching to the VRP. As in Altinkemer and Gavish's method a matching algorithm is used at each iteration to decide which clusters should be combined. Both methods start with an initial solution of n single customer clusters. However, at each iteration Wark and Holt consider two rules, described below, for calculating matching costs. Thus at each iteration up to two new solutions may be formed. Wark and Holt investigate both solutions by forming a search tree in which one of the solutions and its descendants are investigated, and then the second solution is investigated. A second important difference is that when two clusters are matched, the outcome may be one cluster (as in Altinkemer and Gavish's method) or two different clusters.

To admit the possibility of two new clusters being produced when two clusters are matched, Wark and Holt introduce the notion of a *slide*. The customers contained in a cluster are kept in a 2-optimal order. Following Wark and Holt's notation we denote the two clusters being matched as $S_i = [x, \ldots, y]$ and $S_j = [u, \ldots, v]$, and let U be the ordered union $[x, \ldots, y, u, \ldots, v]$. Rewrite U as $[u_1, u_2, \ldots, u_p]$. Then a slide U_k of U $(k = 1, \ldots, p)$ is a partition of U into two clusters $[u_1, \ldots, u_k]$ and $[u_{k+1}, \ldots, u_p]$, which are possibly reordered so that the routes are 2-optimal. Note that when $k = p$ only one cluster will be formed. This corresponds to the merger used by Altinkemer and Gavish.

The two rules used for calculating the matching costs between clusters i and j are:

1. total matching M_{ij}^T
2. load-modified matching M_{ij}^L

When $i \neq j$, M_{ij}^T is the minimum cost slide of the two clusters. Slides in which one of the partitions violate a constraint are ignored, as well as the slide in which the partitions are S_i and S_j. When $i = j$ (self-matching), M_{ij}^T is the cost of the route servicing the customers in S_i. Load-modified costs differ from the total matching costs in that the distances from the depot to the endpoints of clusters i and j are weighted according to how close the clusters are to a vehicle's capacity (except for the self-matching case). The reasoning behind this modification is "that a cluster whose load and route time are significantly less than the vehicle capacity and maximum route time is unlikely to stand alone, and will probably be merged with another cluster in a subsequent matching step. Thus, the smaller the cluster load and timespan, the less important are the distances of the endpoints of the cluster from the depot."

As well as considering nodes of the search tree generated by the two matching costs, Wark and Holt generate additional nodes by randomly splitting some of the clusters in two. Because this can cause the search tree to become unreasonably large they disable this feature when the search reaches a user specified size. The algorithm was tested on the 14 standard problems of [CMT79]. The resulting

solutions were of high quality, producing best known solutions for seven of the problems and near best solutions for the other problems. While computation times were large ($\approx$ 100 minutes on a Sun 630MP for the 199 customer problems) they were of the same order as [GHL91].

2.3.2 Mathematical Programming Based Methods

Fisher and Jaikumar [FJ81] present a heuristic in which an assignment of customers to vehicles is obtained by solving a generalised assignment problem with an objective function that approximates delivery costs. The motivation for this comes from a formulation of the VRP as a nonlinear assignment problem. The objective is nonlinear because it contains the cost of the routes assigned to each of the vehicles. The objective is linearised by using linear approximations to the cost of the TSP tours. A number of linearisations are possible, Fisher and Jaikumar base theirs on a set of K "seed" customers $i_1, \ldots, i_K$, each of which is assigned to a vehicle. Let y_{ik} be 1 if customer i is serviced by vehicle k, 0 otherwise. Fisher and Jaikumar approximate the cost of the route assigned to vehicle k as

$$\sum_{i=1}^{n} d_{ik} y_{ik} \tag{2.26}$$

where

$$d_{ik} = c_{0i} + c_{ii_k} - c_{0i_k} \tag{2.27}$$

represents the cost of inserting customer i into the route in which vehicle k travels from the depot, directly to customer i_k, and back. Fisher and Jaikumar note that they are working on other approaches to approximate route costs. In one, the generalised assignment problem is solved several times, the d_{ik} being iteratively adjusted to better approximate the route costs. Along these lines an exact method can be developed using a Benders decomposition.

Instead of selecting actual seed customers, Fisher and Jaikumar use K points on the plane. These seed points are calculated by dividing the problem into K cones, each of which account for an equal portion of the customer orders. They also suggest that a human scheduler could be used to select the seed customers.

The method solves the assignment problem exactly, so for VRPs that are only capacity constrained the method will always find a feasible solution, if one exists, for a given K. However, Fisher and Jaikumar state that the fifth problem of [CMT79] has a lower bound of 17 vehicles when in fact solutions with only 16 vehicles exist (see [Osm93], Section **??**, and Appendix **??**). The bound is based on capacity arguments, the customer order sizes being summed and compared with the vehicle capacity. Specifically, K^*, a lower bound on K, satisfies

$$(K^* - 1)Q < \sum_{i=1}^{n} q_i \leq K^* Q \tag{2.28}$$

For the fifth problem $\sum_{i=1}^{n} q_i = 3186$ and $Q = 200$. For $K = 16$ the problem is very tightly constrained capacity wise, at least two vehicles must be fully loaded, with the rest being very close to capacity. Because the loading of the vehicles is so important for this problem it would be interesting to see how well the Fisher-Jaikumar assignment heuristic solves this problem with $K = 16$, especially since it makes greater use of capacity constraints in building routes than earlier heuristics.

The direct tree search algorithm described in Section 2.2.1 was truncated by Christofides et al. [CMT79] to form a heuristic procedure. The heuristic is based on the route branching scheme. As before, a customer, i, is chosen which forms the basis of a number of routes. Instead of considering all routes passing through i the heuristic only considers a subset of attractive routes. The subset is generated by using a varying linear combination of the savings criteria, s_{ij}, and an extra-mileage criteria, $m_{ij} = c_{0i} + c_{ij} - c_{0j}$. Denote the i-th generated route as S^i. Then a route is evaluated by:

$$V(S^i) = C(S^i) + E(F^i) \tag{2.29}$$

where $C(s^i)$ is the cost of the route, obtained by a TSP heuristic, and $E(F^i)$ is the cost of the minimum spanning tree of all the unrouted customers. This represents a looser bound than used in the exact algorithm, but it is much easier to compute. The heuristic chooses the best route according to $V(S^i)$, and stops when all customers are routed. No backtracking is performed.

2.3.3 Iterative Improvement Methods

In [CE69], Christofides and Eilon present one of the earliest iterative improvement methods for the VRP, the 3-optimal method. The method is an adaptation of the TSP 2-opt and 3-opt approaches. A 2-opt move consists of breaking two edges, forming two disconnected chains. The chains are the reconnected by using two different edges. A 3-opt move breaks three edges to form three disconnected chains. These can be reconnected in eight different ways. The 3-optimal method starts with a feasible VRP solution, and applies 2-opt moves until the solution is 2-optimal. 3-opt moves are then applied to the solution until it becomes 3-optimal. Exchanges are accepted only if capacity and time constraints are satisfied. For a set of 10 test problems, the 3-optimal method produced superior results to the savings method.

The 3-optimal method is primal in the sense that it always works with feasible solutions. Cheshire et al. [CMN82] present a dual heuristic that allows for a variety of different constraints. The heuristic starts with an initial solution involving K seed customers. A complete but infeasible solution is then created using an adaptation of the savings approach; customers are added to the seed routes based on a cost function that is combination of the standard savings, s_{ij}, and a penalty term. The penalty function uses a set of penalty multipliers which control how important constraint violation is. Initially the multipliers are set at a low value. In the initial construction phase, and the subsequent improvement

phase, all routes are kept "1-optimal", i.e. customers are repositioned one at a time until no further improvements can be found.

Stewart and Golden [SG84] present a dual heuristic that uses a Lagrangean relaxation to transform the VRP into a multiple travelling salesman problem (m-TSP). In general, the routes in a m-TSP solution will not satisfy the additional VRP constraints. To make the m-TSP solution feasible, Stewart and Golden place the capacity constraints into the objective , weighting the capacity violation of route k by the factor λ_k. Stewart and Golden do not consider route length constraints; however, they could easily be added to the model.

The heuristic starts by selecting an initial value for the λ_k and approximately solving the resulting m-TSP by starting with an arbitrary tour and applying standard TSP 3-opt moves until the tour is 3-optimal. If the tour is a feasible solution to the VRP the heuristic stops, otherwise the λ_k are adjusted to further penalise infeasible routes and the tour is made 3-optimal again. The process repeats until the m-TSP tour is a feasible solution to the VRP. To get good quality solutions Stewart and Golden note that initially the λ_k should have low values and be gradually increased in order to make the solution feasible.

Waters [Wat87] extends the primal approach of Christofides and Eilon by considering a number of different exchange procedures. As well as the 2-opt procedure, Waters considers:

single removes a single customer from its present position and then replaces it in the position which gives a minimum total cost

double removes two customers simultaneously and repositions them so as to minimise cost

pair + single repositions a single customer and a pair of adjacent customers (the pair is not split)

double pair reposition two pairs of nodes (again the pairs are not split)

The procedures were each used in turn. A number of different orderings were tested, good results were generally found by applying the procedures in the order; 2-optimal, double pair, pair + single, double, and then single. The heuristic was tested on seven problems with initial solutions generated by the savings method. The heuristic found improvements for six of the seven problems. The heuristic was then tested, on the same set of problems, with solutions generated by an implementation of the sweep algorithm. Although the initial solutions from the sweep algorithm had greater total cost than those generated by the savings method, the solutions after the improvement phase were of better quality when the sweep algorithm was used. Waters attributes this to the petal-shaped structure generated by the sweep algorithm. A similar approach to Waters' exchange procedures was adopted by Dror and Levy [DL86] for an inventory routing problem. This is discussed in Section 4.3.

The procedures of Waters were extended by Fahrion and Wrede [FW90] in their chain-exchange procedure. In this procedure two *chains* (sequences of adjacent customers) exchange positions. The chains vary in length from 1 up to a value related to the number of customers in an average route. The procedure starts with long chains and iteratively reduces the chain lengths down to one. Comparison of the procedure with Waters heuristic is difficult because results were only presented for three common problems.

The above improvement methods all terminate when a local optimum is reached. A number of researchers have applied metaheuristics (Section 1.3) to the VRP to prevent the search from becoming trapped in a local optimum. Not including the current study we are aware of six applications of tabu search, two simulated annealing implementations, and one implementation of the closely related Great Deluge algorithm. Previous to this study there appear to be no be implementations of genetic algorithms to the VRP, although genetic algorithms have been applied to the more complex vehicle routing problem with time windows [TNJ91]. This application is discussed in detail in Section 4.2.2.

Osman [Osm93] applied simulated annealing and tabu search to the VRP. Both methods used the same basic move, the 1-interchange mechanism, to change the current solution. For the VRP a 1-interchange is similar to a combination of Waters' single and double procedures.

Classical simulated annealing schemes randomly choose a move from the list of possible moves which can lead to regions of the problem being unexplored for undesirable lengths of time. Osman takes a more systematic approach, the route indices are randomly shuffled to form a permutation π of the indices $\{1, \ldots, K\}$. He then examines route pairings (R_i, R_j) in the following order:

$$(R_{\pi(1)}, R_{\pi(2)}), \ldots, (R_{\pi(1)}, R_{\pi(K)}), (R_{\pi(2)}, R_{\pi(3)}), \ldots, (R_{\pi(K-1)}, R_{\pi(K)}) \tag{2.30}$$

and for each pairing considers all 1-interchanges in a systematic manner. When the pairings of Equation 2.30 are exhausted the route indices are reshuffled and the search continues.

Osman's tabu search implementation is fairly standard except that a special data structure is used so that information gained about move costs from previous iterations can be used in the current iteration. At each iteration the move with the largest objective improvement is selected, as long it does not reverse a recently accepted move. Details of the method are given in Section 4.1.2.

The two methods were tested on a set of standard test problems. Both methods produced excellent results, of better quality or at least the same quality as earlier heuristics (pre-1990). However, much more computer time was required for Osman's heuristics than for the earlier heuristics. Overall the tabu search implementation outperformed the simulated annealing implementation, producing better quality solutions in less time. The simulated annealing implementation displayed large variance with regard to solution quality and time taken.

The standard simulated annealing scheme has a number of parameters that must be specified in order to define the annealing schedule. For best results these parameters must be tuned for different classes of problems. Unfortunately the best choice of parameter settings for a particular class of problems is not obvious. Hiquebran et al. [HASG94] introduce a *revised simulated annealing* (RSA) scheme

which has parameters that are somewhat easier to understand compared to the parameters of the standard scheme. To apply the revised scheme to the VRP two types of moves are used to change the solution, a swap move and an insert move. These moves are identical to Waters' single and double procedures, and both involve two routes. Instead of just randomly selecting two routes and performing a customer exchange between them, Hiquebran et al. perform a more extensive search. One customer, called the target customer, is randomly selected. Let the route containing the target customer be known as the target route. Then all routes other than the target route are considered for an exchange move with the target customer. If a swap move is being tested a customer is randomly selected from the non-target route. For an insert move the target customer is inserted into the non-target route. The move with the largest objective improvement over all the route pairings is chosen, and then tested using the normal simulated annealing acceptance criterion.

Hiquebran et al. present results for eight of the test problems of [CMT79]. For each problem four different annealing schedules are tried and 100 runs are performed for each schedule. Each run is reasonably short, ranging in time from 16 seconds for the 50 customer problem, up to 170 seconds for 199 customer problem (all times are for a Sun Sparc 2). The best solutions found by RSA are of similar quality to Osman's. The mean objective was 2–4% higher than the best found solutions, indicating some variability between runs.

A detailed description of the tabu search implementations is given in Section 4.1.2, only a brief overview is given here. Of the six known tabu search implementations for the VRP the first is due to Willard [Wil89]. This transforms the problem into a TSP by replicating the depot. The neighbourhood is restricted in order to satisfy the VRP constraints. The results are poor, being outperformed by earlier heuristics. The other five implementations tackle the VRP directly.

Pureza and França [PF91] present a simple implementation of tabu search. The method uses node interchange moves which are identical to Waters' single and double procedures. The edges added and deleted from the interchange are used to prevent reversals of recently accepted moves. The results are of good quality, indicating that tabu search is well suited to solving the VRP. Independently Gendreau et al. [GHL91] and Osman [Osm93] developed tabu search implementations that use the customers involved in a move to prevent reversals. Gendreau et al. restricted the possible moves to just insert moves (a single procedure using Waters' terminology) but allowed solutions to become temporarily infeasible. Osman considered both insert and swap moves. Both approaches used more sophisticated tabu search schemes than Pureza and França's, resulting in better quality solutions.

Taillard [Tai93] drew on ideas from these two implementations to produce one of the best (in terms of objective quality) heuristics for the VRP. For larger problems Taillard partitions the problem into several subproblems, solving each of these using tabu search. Two partitioning schemes are developed, a sector based partitioning suitable for Euclidean problems where the depot is centrally located and customers uniformly distributed, and a shortest path based partitioning suitable for non-Euclidean problems. Taillard also implemented the method on parallel hardware, producing near linear speed ups.

Semet and Taillard [ST93] applied tabu search to an extended version of the VRP which closely models a real life problem facing a Swiss grocery chain. The heuristic was successful, producing better solutions than the manually created solutions and those produced by an earlier developed heuristic. The above implementations show that tabu search performs well for the VRP, finding good quality solutions and being easily adaptable to variants of the VRP.

The metaheuristics tested in this thesis, GRASP, tabu search, and genetic algorithms, can be classified as iterative improvement methods. They are discussed in Chapters 4, 4, and 4.

2.3.4 Interactive Methods

The objective of these methods is not to replace human dispatchers, but to assist them. The methods allow the user to define routes and to examine particular routing possibilities. The cost and effectiveness of a particular route can be examined, possibly graphically, with an interactive package. An example of such a package is given by Waters [Wat84].

In an interesting paper [Dag84], Daganzo presents an idealised analytic model for the VRP that gives optimal route shapes for asymptotically large problems. The resulting guidelines for clustering customers are difficult to implement on a computer. Daganzo gives an example of a newspaper distribution problem where the guidelines, along with some common sense, allowed a human to find a significantly better solution to the incumbent in a reasonable amount of time. While the guidelines become less accurate as problem size decreases, it may be useful to implement the ideas in order to provide starting solutions to heuristics for moderate to large VRPs.

2.4 Extensions to the Vehicle Routing Problem

A number of extensions to the basic VRP have been considered by various researchers. These aim to better model real world applications by imposing extra constraints, including fleet size and mix decisions, allowing different forms of routes, and/or other extensions. For instance, the well researched Vehicle Routing Problem with Time Windows (VRPTW) specifies that customers must be serviced within some time window. For a supermarket distribution application this could be used to model the times when staff are available to unload the delivery vehicles. In this section we briefly survey the following problems:

- The Split Delivery Vehicle Routing Problem (SDVRP)
- The Vehicle Routing Problem with Time Windows (VRPTW)
- The Multi-depot Vehicle Routing Problem

2.4.1 The Split Delivery Vehicle Routing Problem

The VRP does not allow a customer to be serviced by more than one vehicle. Several researchers have studied a relaxation of the VRP, the Split Delivery Vehicle Routing Problem (SDVRP), which allows a customer to be serviced by several vehicles if this reduces overall costs. This is of greatest benefit when the sizes of customer orders are comparable with the capacity of a vehicle.

Burrows [Bur88] modified the generalised savings approach to allow for split deliveries. The modification is simple in that each customer order is split into a number of smaller indivisible orders. The savings method is then applied as normal. Dror and Trudeau [DT90] adapted the improvement algorithm of Dror and Levy [DL86]. Two additional improvement procedures were added:

1. k-split interchange: This splits an order into k portions, each serviced by a different vehicle.
2. Route addition: An extra tour is added which eliminates splitting for a particular customer.

The results presented by Dror and Trudeau illustrate the advantages of allowing split deliveries. When orders ranged from 70–90% of vehicle capacity, an improvement of over 10% in distance travelled was gained over the standard VRP solutions.

Pedder [Ped90] developed a system for routing and scheduling the distribution fleet of a supermarket chain. For this problem, customer orders are often close, or even exceed, the vehicle capacity. Accordingly, Pedder developed a model that allowed split deliveries. The model was similar to the set partitioning approach of Foster and Ryan [FR76], except that a single column could represent several routes, with load-splitting occurring in these routes.

Dror et al. [DLT94] attempted to solve the SDVRP exactly using constraint relaxation with branch and bound. For the problems considered the computed lower bounds were within 9% of the upper bounds. Only one problem was solved to optimality. The problems considered were small, with 10–20 customers. They concluded that the SDVRP is more difficult to solve exactly than the VRP.

2.4.2 The Vehicle Routing Problem with Time Windows

The Vehicle Routing Problem with Time Windows (VRPTW) constrains the basic VRP by specifying that customers must be serviced within some time window. Each time window usually consists of two times, a_i and b_i. We wish the time at which customer i is serviced, t_i, to satisfy $a_i \leq t_i \leq b_i$. Some formulations allow for multiple time windows per customer, while some allow for one sided windows. Two major variants of the VRPTW exist. In the first the time windows are "hard" and must be adhered to. However, the vehicle can arrive early at a customer and delay service until time a_i. In the second variant the windows are "soft" and can be violated, subject to paying some penalty cost.

Solomon and Desrosiers [SD88] provide a survey of the time window constrained routing literature. They discuss relaxations of the VRPTW such as the Travelling Salesman Problem with Time Windows (TSPTW) and the Shortest Path Problem with Time Windows (SPPTW). They then discuss heuristic and exact approaches to the VRPTW, and go on to consider extensions of the VRPTW such as the Dial-A-Ride Problem with Time Windows (DARPTW). Laporte and Nobert [LN87] also describe some exact algorithms for time window constrained problems in their survey. We concentrate on more recent publications, discussing heuristic approaches, then exact approaches.

Solomon [Sol87] was first to generalise a number of VRP heuristics, such as the savings method and the sweep algorithm, to the VRPTW with hard time windows. He developed a suite of test problems, each with 100 customers, which have been used by several authors for comparative purposes. The problems are grouped into six sets based on geographical layout and scheduling horizon. Problem sets R1 and R2 have customers randomly generated so that they are distributed uniformly over a square. Sets C1 and C2 are clustered, and sets RC1 and RC2 have a mix of randomly generated and clustered customers. Problem sets R1, C1, and RC1 have short scheduling horizons. Problem sets R2, C2, and RC2 have long scheduling horizons, allowing many customers to be serviced by a single vehicle. Within each problem set a number of problems were generated, with varying time window widths and varying numbers of customers with time windows. For the suite of test problems Solomon found an insertion heuristic to give best results.

Solomon et al. [SBS88] have developed efficient improvement heuristics based on the 2-opt and 3-opt procedures [Lin65] and the Or-opt procedure [Or76]. The heuristics incorporate the ideas of Savelsbergh [Sav90] to speed up the feasibility checks. For large routes the Or-opt procedure is significantly faster than 3-opt, with little, if any, degradation in solution quality. A competitive algorithm based on a parallel insertion heuristic and the Or-opt procedure has been proposed by Potvin and Rousseau [PR93]. Thangiah et al. [TNJ91] applied a genetic algorithm to the VRPTW with successful results. The GA was able to find significantly better results than [Sol87] and [SBS88]. The method is further described in Section 4.2.2.

There has been less research on the soft time windows problem. Koskosidis et al. [KPS92] extend the Fisher-Jaikumar assignment heuristic [FJ81]. The heuristic is iterative, the results from the previous iteration being used to affect the assignment costs of the current iteration. The test problems solved had loose capacity constraints but restrictive time windows. The assignment/clustering phase tended to assign many customers to one route, thus making a time windows feasible route difficult or impossible to find. Koskosidis et al. avoided this problem by artificially tightening the capacity constraints. The heuristic worked well for clustered problems; however, it was less satisfactory for the uniform problems.

Balakrishnan [Bal93] presents three simple heuristics for the soft time windows problem. Balakrishnan considered a range of time window violation penalties. With large penalties the problem resembles the hard time windows case. For those penalties the heuristics find solutions that are on the average slightly worse than [Sol87]. For smaller penalties the heuristics are able to find solutions requiring fewer vehicles and/or lower total route distance.

Exact methods have only been developed for the hard windows case. Subsequent to the survey by Solomon and Desrosiers [SD88], Fisher et al. [FJM91] presented preliminary results for two exact approaches to the VRPTW, both using Lagrangean relaxation. The first approach uses variable splitting to divide the problem into a semi-assignment problem, and a series of shortest path problems with time windows and capacity constraints. The second approach is an extension of Fisher's K-tree method [Fis94], with the time window constraints being incorporated into the objective in a Lagrangean fashion. Both approaches are capable of solving some of the full Solomon test problems (for some problems they only considered the first 25 or 50 customers). Fisher et al. are critical of the Solomon test problems, especially the clustered sets, and intend to generate more realistic problems.

Desrochers et al. [DDS92] recently presented an exact approach that is able to solve seven 100 customer problems to optimality, two from R1 and five from C1. A set partitioning formulation is used, with column generation to generate new attractive routes. The column generator solves a shortest path problem with time windows and capacity constraints, generating several columns per call in order to reduce the overall computational effort. Since the column generator can generate routes containing cycles, Desrochers et al. solve a set covering problem rather than a set partitioning problem. If necessary, the solution is forced to a set partitioning type solution in the branch and bound scheme. Because the algorithm cannot solve all the 100 customer problems exactly, Desrochers et al. first solve for only the first 25 customers. If this is successful then the first 50 customers, and finally the complete set. In all, 87 problems were attempted.

The LP relaxation provides excellent lower bounds, for 27 of the 87 problems it gives the optimal objective. For the other problems the bound is on average only 1.5% from the optimal value. Through a variety of branching schemes the approach is able to solve many problems optimally. However, there are still problems on which the algorithm fails. On average each problem that was solved to optimality required $\approx$ 100 CPU seconds on a SUN SPARCstation; however, this time varied considerably.

2.4.3 The Multi-depot Vehicle Routing Problem

A company may have several depots from which it can serve its customers. If the customers are clustered around the depots then the distribution problem should be modelled as a set of independent VRPs. However, if the customers and the depots are intermingled then a Multi-depot Vehicle Routing Problem (MDVRP) should be solved.

As well as the ordinary VRP decisions, a MDVRP requires the assignment of customers to depots. A fleet of vehicles is based at each depot. These vehicles originate from that depot, service the customers assigned to that depot, and return to the same depot.

While a large number of papers have been published on the classical VRP, there have been few dealing with the MDVRP. Chao et al. [CGW93] were aware of only six previously published papers that develop solution procedures for the MDVRP. We shall summarise three of these papers, as well as the work of Chao et al..

Wren and Holliday [WH72] describe a method consisting of two parts: constructing an initial solution, followed by a number of improving heuristics. To construct the initial solution a process, similar to the Gillet and Miller [GM74] sweep algorithm, is followed. The customers are temporarily assigned to their closest depot, and the angles from the assigned depot to the customers are calculated. The customers are then sorted into clockwise order, irrespective of depot. Routes are then sequentially constructed, one at each depot. Customers are selected in radial order, and are inserted into the route which minimises the additional cost. If there is no emerging route at a depot, or if the current route is at or near capacity, then a new route involving just the selected customer is considered. The seven improving heuristics — inspect, single, pair, complain, delete, combine, and disentangle — resemble VRP improvement heuristics, and are applied until no improvement to the solution can be found.

Golden et al. [GMN77] present two heuristics for solving the MDVRP. The first is an adaptation of the savings method. The second is designed for larger problems, in which computational efficiency is of extreme importance. The second method consists of two phases; assigning customers to depots, and building of routes at the depots. Customers are divided into border and nonborder points according to the ratio of the distances to the closest and second closest depots. Customers for which the choice of depot is unclear are deemed border points and are assigned to depots using the adaptation of the savings method. The nonborder points are then assigned to their closest depot, and a standard VRP heuristic is applied at each depot. The solution is further improved by a simple post processor.

Chao et al. [CGW93] use a simple initialisation heuristic, combined with an improvement phase that is more powerful than the earlier studies. The improvement phase uses the record-to-record metaheuristic [Due93] to guide the search. Candidate moves consist of repositioning a single customer in another route, which may or may not be based at a different depot. The heuristic produces better results than earlier studies on all of the test problems.

We are aware of only one exact algorithm for the MDVRP, which is due to Laporte et al. [LNA84]. They develop a branch and bound algorithm which uses a LP relaxation, and solve 31 randomly generated test problems that contain up to 25 vertices (including depots and customers).

Chapter 4

The Petal Method

In this chapter we review methods similar to the petal method, and then discuss the petal method in detail. We show that the optimal petal set can be found efficiently by solving several shortest path problems. We numerically compare the shortest path approach with a LP implementation. In later chapters we will be interested in the solution after a small change in the order. In this chapter we will show how the solution can be quickly updated after the repositioning of a single customer.

4.1 The Order-Partition Approach

One approach to solving VRPs is to order the customers in some way, and then to partition the ordering into feasible routes. Gillet and Miller [GM74] devised a heuristic procedure called the *sweep algorithm* based on this approach. We select customer 1 as the starting customer. With the depot as the pivot, we sweep in a counter-clockwise direction a ray originating from the depot, and passing through the starting customer. Customers are added to the current route as they are swept. If adding a customer to the current route would cause a violation of any constraint, then the current route is added to the solution and the customer causing the violation is used to restart the next route. We continue sweeping until all customers have been assigned to a vehicle. Then the costs for the routes are determined by applying a TSP algorithm to each group of customers. At this point we have a feasible solution. This sweep process is repeated with customers $2, 3, 4, \dots, n$ as the starting customer. Gillet and Miller call this process the *forward-sweep algorithm*. The *backward-sweep algorithm* is similar except that the sweep is performed in a clockwise direction. Choosing the best solution from the forward and backward sweep algorithms results in the sweep algorithm.

Foster and Ryan [FR76] present an improvement to the sweep algorithm. They order the customers radially but have a more sophisticated method for partitioning the customers. Every contiguous subset of the radial order is considered as a possible route. Of these routes, those that are feasible become the "petal set", so called because the routes radiate from the depot like the petals of a flower. It can be shown that any solution produced by the Gillet and Miller sweep method is contained within the petal set. A subset of routes from the petal set that contain each customer exactly once will be a feasible solution to the VRP. Foster and Ryan find the optimal subset by formulating the problem as a set partitioning problem. This can be easily solved since the constraint matrix is unimodular under a very mild assumption.

The multiple partition giant tour algorithm introduced by Golden et al. [GALG84] is another example of the order then partition approach to solving VRPs. They apply a 2-opt TSP heuristic to all the customers and use the resulting tour as the order. Denote the TSP tour as $s(1) - s(2) - \ldots - s(n) - s(1)$. The "distance" between customers $s(k)$ and $s(m)$, denoted by $COST(k,m)$, is the cost of servicing customers $s(k), s(k+1), \ldots, s(m-1)$. $COST(k,m)$ is defined as

$$COST(k,m) = c_{0,s(k)} + \sum_{i=k}^{m-2} c_{s(i),s(i+1)} + c_{s(m),0}$$

Golden et al. introduce nodes $s(n+1), s(n+2), \ldots, s(n+M)$ where $s(n+i)$ is a duplicate of $s(i)$ and M is given by

$$\sum_{r=1}^{M-1} q_{s(r)} < Q \leq \sum_{r=1}^{M} q_{s(r)}$$

A directed graph is created with nodes corresponding to customers $s(1), s(2), \ldots,$ $s(n+M)$ and arcs from node k to node m, $k = 1 \ldots 2n-1$, $m = k+1 \ldots 2n$, if the route servicing $s(k), s(k+1), \ldots, s(m-1)$ is feasible. The weight of the arcs is set to $COST(k,m)$. A feasible solution to the VRP consists of a path from node i to node $n+i$. The optimal partitioning is determined by finding the M shortest paths, starting at i and ending at $n+i$ for $i = 1, \ldots, M$, and selecting the shortest of these.

Bibliography

[AG91] K. Altinkemer and B. Gavish. Parallel savings based heuristics for the delivery problem. *Ops. Res.*, 39:456–469, 1991.

[AMS89] Y. Agarwal, K. Mathur, and H. M. Salkin. A set-partitioning-based exact algorithm for the vehicle routing problem. *Networks*, 19:731–749, 1989.

[Bal93] N. Balakrishnan. Simple heuristics for the vehicle routeing problem with soft time windows. *J. of the Opnl. Res. Soc.*, 44(3):279–287, 1993.

[BGAB83] L. Bodin, B. Golden, A. Assad, and M. Ball. Routing and scheduling of vehicles and crews: The state of the art. *Comput. & Ops. Res.*, 10:10, 1983.

[BQ64] M. L. Balinski and R. E. Quandt. On an integer program for a delivery problem. *Ops. Res.*, 12(2):300–304, 1964.

[Bur88] W. Burrows. The vehicle routing problem with loadsplitting: A heuristic approach. In *24th Annual Conference of the Operational Research Society of New Zealand*, pages 33–38, 1988.

[CE69] N. Christofides and S. Eilon. An algorithm for the vehicle dispatching problem. *ORQ*, 20:309–318, 1969.

[CGW93] I.-M. Chao, B. L. Golden, and E. Wasil. A new heuristic for the multi-depot vehicle routing problem that improves upon best-known solutions. *American J. of Math. and Mgmt. Sci.*, 13(3):371–406, 1993.

[Chr76] N. Christofides. The vehicle routing problem. *RAIRO*, 10:55–70, 1976.

[CMN82] I. M. Cheshire, A. M. Malleson, and P. F. Naccache. A dual heuristic for vehicle scheduling. *J. of the Opnl. Res. Soc.*, 33:51–61, 1982.

[CMT79] N. Christofides, A. Mingozzi, and P. Toth. The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, editors, *Combinatorial Optimization*, pages 315–338. John Wiley & Sons, Ltd., 1979.

[CMT81a] N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for the vehicle routing problem based on spanning tree and shortest path relaxations. *Math Prog*, 20:255–282, 1981.

[CMT81b] N. Christofides, A. Mingozzi, and P. Toth. Space state relaxation procedures for the computation of bounds to routing problems. *Networks*, 11:145–164, 1981.

[CW64] G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Ops. Res.*, 12:568–581, 1964.

[Dag84] C. F. Daganzo. The distance traveled to visit n points with a maximum of c stops per vehicle: An analytic model and an application. *Trans. Sci.*, 18:331–350, 1984.

[DDS92] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Ops. Res.*, 40(2):342–354, 1992.

[DL86] M. Dror and L. Levy. A vehicle routing improvement algorithm comparison of a “greedy” and a matching implementation for inventory routing. *Comput. & Ops. Res.*, 13:33–45, 1986.

[DLT94] M. Dror, G. Laporte, and P. Trudeau. Vehicle routing with split deliveries. *Disc. Appl. Math.*, 50:239–254, 1994.

[DR59] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Mgmt Sci*, 6(1):80–91, 1959.

[DT90] M. Dror and P. Trudeau. Split delivery routing. *NRL*, 37:383–402, 1990.

[Due93] G. Dueck. New optimization heuristics: The great deluge algorithm and the record-to-record travel. *J. of Comput. Phys.*, 104(1):86–92, 1993.

[Fis94] M. L. Fisher. Optimal solution of vehicle routing problems using minimum K-trees. *Ops. Res.*, 42(4):626–646, 1994.

[FJ81] M. Fisher and R. Jaikumar. A generalized assignment heuristic for vehicle routing. *Networks*, 11:109–124, 1981.

[FJM91] M. L. Fisher, K. O. Jörnsten, and O. B. G. Madsen. Vehicle routing with time windows. Preliminary results. Technical Report 4/1991, IMSOR, The Technical University of Denmark, June 1991.

[FR76] B. A. Foster and D. M. Ryan. An integer programming approach to the vehicle scheduling problem. *ORQ*, 27:367–384, 1976.

[FW90] R. Fahrion and M. Wrede. On a principle of chain-exchange for vehicle-routeing problems (1-VRP). *J. of the Opnl. Res. Soc.*, 41(9):821–827, 1990.

[GALG84] B. Golden, A. Assad, L. Levy, and F. Gheysens. The fleet size and mix vehicle routing problem. *Comput. & Ops. Res.*, 11:49–66, 1984.

[Gas67] T. J. Gaskell. Bases for vehicle fleet scheduling. *ORQ*, 18:281–295, 1967.

[GCJS57] W. M. Garvin, H. W. Crandall, J. B. John, and R. A. Spellman. Applications of linear programming in the oil industry. *Mgmt Sci*, 3:407–430, 1957.

[GG79] B. Gavish and S. C. Graves. The travelling salesman problem and related problems. Technical Report 7905, Graduate School of Management, University of Rochester, 1979.

[GG82] B. Gavish and S. C. Graves. Scheduling and routing in transportation and distribution systems: Formulations and new relxations. Technical report, Graduate School of Management, University of Rochester, 1982.

[GHL91] M. Gendreau, A. Hertz, and G. Laporte. A tabu search heuristic for the vehicle routing problem. Technical Report CRT-777, Centre de Recherche sur les transports, Montréal, June 1991. To appear in Mgmt Sci.

[GM74] B. E. Gillet and L. R. Miller. A heuristic algorithm for the vehicle-dispatch problem. *Ops. Res.*, 22:340–349, 1974.

[GMN77] B. Golden, T. L. Magnanti, and H. Q. Nguyen. Implementing vehicle routing algorithms. *Networks*, 7:113–148, 1977.

[HASG94] D. T. Hiquebran, A. S. Alfa, J. A. Shapiro, and D. H. Gittoes. A revised simulated annealing and cluster-first route-second algorithm applied to the vehicle routing problem. *Engineering Optimization*, 22:77–107, 1994.

[HK70] M. Held and R. M. Karp. The traveling-salesman problem and minimum spanning trees. *Ops. Res.*, 18:1138–1162, 1970.

[KPS92] Y. A. Koskosidis, W. B. Powell, and M. M. Solomon. An optimization-based heuristic for vehicle routing and scheduling with soft time window constraints. *Trans. Sci.*, 26(2):69–85, 1992.

[Lap92] G. Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *EJOR*, 59(3):345–358, 1992.

[Lin65] S. Lin. Computer solutions of the traveling salesman problem. *Bell Systems Technical Journal*, 44:2245–2269, 1965.

[LN87] G. Laporte and Y. Nobert. Exact algorithms for the vehicle routing problem. *Annals of Discrete Mathematics*, 31:147–184, 1987.

[LNA84] G. Laporte, Y. Nobert, and D. Arpin. Optimal solutions to capacitated multidepot vehicle routing problems. *Congressus Numerantium*, 44:283–292, 1984.

[LND85] G. Laporte, Y. Nobert, and M. Desrochers. Optimal routing under capacity and distance restrictions. *Ops. Res.*, 33:1050–1073, 1985.

[Or76] I. Or. *Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking.* PhD thesis, Northwestern University, 1976.

[Osm93] I. Osman. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of OR*, 41:421–451, 1993.

[Pae88] H. Paessens. The savings algorithm for the vehicle routing problem. *EJOR*, 34:336–344, 1988.

[Ped90] S. J. Pedder. An integer model for scheduling and routeing a groceries distribution vehicle fleet. Master's thesis, University of Auckland, New Zealand, 1990.

[PF91] V. M. Pureza and P. M. França. Vehicle routing problems via tabu search metaheuristic. Technical Report CRT-747, Centre de Recherche sur les transports, Montréal, 1991.

[PR93] J. Y. Potvin and J. M. Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *EJOR*, 66(3):331–340, 1993.

[Sav90] M. W. P. Savelsbergh. An efficient implementation of local search algorithms for constrained routing problems. *EJOR*, 47:75–85, 1990.

[SBS88] M. M. Solomon, E. K. Baker, and J. R. Schaffer. Vehicle routing and scheduling problems with time window costraints: Efficient implementations of solution improvement procedures. In B. L. Golden and A. A. Assad, editors, *Vehicle Routing: Methods and Studies*, volume 16 of *Studies in Management Science and Systems*, pages 85–105. Elsevier Science Publishers B. V., 1988.

[SD88] M. M. Solomon and J. Desrosiers. Time window constrained routing and scheduling problems. *Trans. Sci.*, 22:1–13, 1988.

[SG84] W. R. Stewart Jr. and B. L. Golden. A Lagrangean relaxation heuristic for vehicle routing. *EJOR*, 15:84–88, 1984.

[Sol87] M. M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Ops. Res.*, 35:254–265, 1987.

[ST93] F. Semet and É. Taillard. Solving real-life vehicle routing problems efficiently using taboo search. *Annals of OR*, 41:469–488, 1993.

[Tai93] É. Taillard. Parallel iterative search methods for vehicle routing problems. *Networks*, 23(8):661–674, 1993.

[TNJ91] S. R. Thangiah, K. E. Nygard, and P. L. Juell. GIDEON — A genetic algorithm system for vehicle routing with time windows. In *Proceedings of the Seventh IEEE Conference on Artificial Intelligence Applications*, pages 322–328, Miami Beach, FL, February 1991. IEEE Computer Society Press, Los Alamitos, CA.

[Wat84] C. D. J. Waters. Interactive vehicle routeing. *J. of the Opnl. Res. Soc.*, 35:821–826, 1984.

[Wat87] C. D. J. Waters. A solution procedure for the vehicle scheduling problem based on iterative route improvement. *J. of the Opnl. Res. Soc.*, 38:833–839, 1987.

[WH72] A. Wren and A. Holliday. Computer scheduling of vehicles from one or more depots to a number of delivery points. *ORQ*, 23:333–344, 1972.

[WH94] P. Wark and J. Holt. A repeated matching heuristic for the vehicle routeing problem. *J. of the Opnl. Res. Soc.*, 45(10):1156–1167, 1994.

[Wil89] J. A. G. Willard. Vehicle routing using r-optimal tabu search. Master's thesis, The Management School, Imperial College, London, 1989.

[Yel70] P. C. Yellow. A computational modification to the savings method of vehicle scheduling. *ORQ*, 21(2):281–283, 1970.