M.O. Ball et al., Eds., *Handbooks in OR & MS, Vol. 8*


# Chapter 1

# Vehicle Routing

*Marshall Fisher*
*Operations and Information Management Department, The Wharton School, University of Pennsylvania, Philadelphia, PA 19104, U.S.A.*

## 1. Introduction

Transportation comprises a significant fraction of the economy of most developed nations. For example, a National Council of Physical Distribution Study [1978] estimates that transportation accounts for 15% of the U.S. gross national product. This economic importance has motivated both private companies and academic researchers to vigorously pursue the use of operations research and management science to improve the efficiency of transportation.

Various modes of transportation exist including air, rail, ship and motor vehicle. The research on transportation has focused on different issues in each mode. In air, the efficient scheduling of airline crews [Lavoi, Minous & Odier, 1988] has received primary attention while in rail, use of large-scale real-time computer control systems has dominated the research agenda (e.g. Harker 1990). The literature on both motor vehicles (trucks, school buses and general passenger buses) and ships has focused on a common problem — the efficient use of a fleet of vehicles that must make a number of stops to pick up and/or deliver passengers or products. The problem requires one to specify which customers should be delivered by each vehicle and in what order so as to minimize total cost subject to a variety of constraints such as vehicle capacity and delivery time restrictions. This chapter provides an introduction to research on this problem.

We'll confine ourselves to the most popular model of the vehicle routing problem, in which a fleet of identical vehicles makes deliveries to customers from a central depot. The model is defined by the following parameters.

$K$ = number of vehicles in the fleet,
$n$ = number of customers to which a delivery must be made. Customers are indexed from 1 to $n$ and index 0 denotes the central depot,
$b$ = capacity (e.g., weight or volume) of each vehicle,
$a_i$ = size of the delivery to customer $i$, measured in the same units as vehicle capacity,
$c_{ij}$ = cost of direct travel between points $i$ and $j$ (we assume $c_{ij} \geq 0$ and $c_{ij} = c_{ji}$ for all $ij$).

The vehicle routing problem is to determine $K$ vehicle routes, where a route is a tour that begins at the depot, traverses a subset of the customers in a specified sequence and returns to the depot. Each customer must be assigned to exactly one of the $K$ vehicle routes and the total size of deliveries for customers assigned to each vehicle must not exceed the vehicle capacity $b$. The routes should be chosen to minimize total travel cost.

This chapter will present the most important algorithms that have been developed for this model. However, before plunging into the mathematical development, it will be helpful to consider some examples of vehicle routing problems and various practical issues that arise in the use of vehicle routing models. Vehicle routing problems are all around us in the sense that many consumer products such as soft drinks, beer, bread, snack foods, gasoline and pharmaceuticals are delivered to retail outlets by fleets of trucks whose operation fits the vehicle routing model. Other examples of vehicle routing problems include the delivery of liquified industrial gases and the collection of milk from farms for transportation to a processing center. Many companies have reported successful implementation of mathematical algorithms to optimize trucking operations including Air Products and Chemicals [Bell, Dalberto, Fisher, Greenfield, Jaikumar, Kedia, Mack & Prutzman, 1983], Chevron [Brown & Graves, 1981], DuPont [Fisher, Greenfield, Jaikumar & Lester, 1982], Edward Don and Company [Walter & Zielinski, 1983; Zielinski, 1985], Exxon [Collins & Clavey, 1986], Kraft, Inc., [Evans & Norback, 1985], North American Van Lines [Powell, Sheffi, Nickerson, Butterbaugh & Atherton, 1986], Southland Corporation [Belardo, Duchessi & Seagle, 1985], and a number of soft drink distributors [Golden & Wasil, 1987].

Real vehicle routing problems usually include complications beyond the basic model considered here. Typical complications include the following.

1. Travel costs can be asymmetric so that $c_{ij} \neq c_{ji}$ in general.

2. The characteristics of the vehicles can introduce a variety of constraints beyond the simple vehicle capacity constraints. The vehicle fleet can be heterogeneous, with each vehicle $k$ having a distinct capacity $b_k$. There can be multiple capacity constraints, arising, for example, if there are both weight and volume restrictions. Sometimes vehicles are divided into compartments for storage of different products (e.g., different blends of gasoline), which further complicates vehicle capacity constraints. Customer/vehicle compatibility constraints may restrict the set of customers that a vehicle can feasibly service. A vehicle may be capable of making more than one trip within a planning horizon. Finally, vehicles may both deliver and pick up product. This complicates vehicle capacity constraints and may also impose precedence constraints on the sequencing of stops on a route if products are picked up at one point for delivery later in the route.

3. The total time duration of a route may be constrained.

4. The time of delivery to customer $i$ may be constrained to fall within a designated 'time window' or windows. Time window constraints arise frequently in practice and usually require fundamental changes in the algorithm used. For this

reason time constrained routing problems are given special attention in Chapter 2 of this Volume.

5. There may be multiple depots with each vehicle in the fleet assigned to a particular depot.

6. Precedence constraints can impose a partial ordering on the customer delivery sequence. A common example of a precedence constraint requires some customers to be the first or last stop on a route.

7. In fleet planning problems, the number of vehicles used is a variable rather than a constant.

8. Delivery to some customers may be optional provided we incur a specified penalty cost for not delivering. This situation can arise when any customer not delivered on the company fleet can be serviced by an outside carrier at a known cost.

9. Period routing problems arise in the distribution of products such as soft drinks, snack foods, beer and bread. In these applications the distributing firm is interested in developing a set of daily routes for some $T$ day period so that each customer receives delivery at a designated frequency. For example, the period might be a 5-day week and the delivery frequency for a specific customer could be twice per week.

10. Inventory routing problems arise in the distribution of liquid products such as industrial gases or gasoline. In these problems, each customer has an inventory of the product, and the distributor must determine the timing and amount of deliveries so that the customer does not run out of product.

11. The objective function in real problems can be quite complex, including terms dependent on the distance travelled, the number of vehicles used, the time duration of routes (as with overtime pay for drivers) and penalties for not delivering to all customers.

The algorithms described in this chapter can usually be extended to accommodate most of these variations, although this is generally more challenging for the exact algorithms.

The data required for an application includes vehicle fleet characteristics (the number of vehicles and their capacity), customer order information (the size of the delivery to each customer), and geographic data (the travel cost and time between any two points). Travel costs and times are by far the most difficult data to obtain. Vehicle characteristics are straightforward to assemble because the amount of information is small, and customer order information can usually be obtained from a company's order entry system. On the other hand, it's rare to find a company that maintains travel costs and distances in computerized form. Usually, when a company is considering introducing an optimization model for vehicle routing, this data exists in the head of a dispatcher responsible for manual routing. The dispatcher knows roughly which customers are close to each other, and will consult a map to augment this informal knowledge as necessary.

Two approaches have been taken to providing geographic information. The first approach is easy to implement and thus preferred if it is accurate enough. It consists of assigning coordinates to each customer and assuming that the travel distance between customers is the Euclidean distance between their coordinate pairs. Cost is taken to be proportional to distance. The distance is often scaled up by a factor to compensate for roads that deviate from a straight-line path between customers. The degree of approximation can be improved further by defining the distance between customers to be the length of a direct path that does not cross various polygons defined to represent rivers, lakes, mountain ranges and other geographic barriers to travel. Finding such a path has been the subject of a number of papers, e.g. Viegas & Hansen [1985].

The second approach obtains travel distances by applying a shortest route algorithm to a computerized network model of the road system. A number of consulting firms and other organizations have compiled road network models. For example, the Swedish Postal Service has developed a network model of all roads in Sweden.

Vehicle routing algorithms can be applied in one of two modes: variable routing or fixed routing. In a variable routing system, an algorithm is used with actual customer delivery requirements to develop routes for the next planning horizon. For example, in a daily variable routing system, customers might be allowed to place orders up to one day prior to delivery time. At the end of a given day, we know precisely which customers require delivery tomorrow and can apply a vehicle routing algorithm to optimize routes for those customers.

Fixed routing is applicable when customer demands are sufficiently stable to allow use of the same routes repeatedly. In a fixed route application, the vehicle routing problem is solved using average demand data. For example, the delivery of bread to grocery stores requires a delivery to most stores every day. While the volume of bread delivered may vary somewhat due to factors such as holidays or days of the week, the set of customers requiring delivery doesn't change so fixed routes are generally used. Companies use fixed routes to avoid the effort required in frequently resolving the vehicle routing problem and to provide stability. For example, it's usually desirable for the same driver to deliver a particular customer repeatedly.

Theory and practice have always been tightly interwoven in vehicle routing research. From the theoretical perspective, the vehicle routing problem contains as special cases some of the most popular models in combinatorial optimization, including the traveling salesman problem and the generalized assignment problem. The many successful implementations of vehicle routing algorithms cited earlier testify to the practical importance of the topic.

The connection between theory and practice dates to the first paper on vehicle routing [Dantzig & Ramser, 1959] which describes both a practical problem concerned with delivering gasoline to gas stations by the Atlantic Refining Company and provides the first formulation of the general vehicle routing problem. Dantzig and Ramser also suggest a solution procedure motivated by their particular example. Soon after, Clarke & Wright [1964] presented a greedy heuristic that improved on the Dantzig–Ramser method. They too were motivated by a practical problem — the delivery of consumer nondurable products by the Cooperative Wholesale Society Limited in the Midlands of England.

This early work introduced what might be called the first generation of vehicle routing research which relied on greedy methods and various local improvement heuristics. A number of commercial vehicle routing systems developed in the 1970's incorporated Clarke and Wright's heuristic, most notably IBM's VSPX system. Literally hundreds of companies tried these commercial systems, but with no reported successes. The generation 1 methodology created in the 60's and 70's simply lacked the sophistication required to solve complex, real problems. Companies that tried these methods frequently reported obtaining solutions inferior to those produced by their existing manual system and noted an inability to handle all of the complex constraints in their real problems.

Success with vehicle routing in the real world had to await the second generation of research which began in the mid to late 70's when a number of researchers began to apply the machinery of mathematical programming to the vehicle routing problem. These efforts employed various mathematical programs that approximated the vehicle routing problem. Although the mathematical programs were solved to optimality, the overall procedures were heuristic since portions of the vehicle routing problem had to be modelled inexactly to achieve tractability. Examples include using a generalized assignment problem to assign customers to vehicles and a set partitioning model to select vehicle routes from a list of candidate routes.

The next major shift in emphasis in vehicle routing research has not yet emerged in a definitive enough form to characterize the third generation of vehicle routing research, but we can try to predict the form this generation will take by looking at the deficiencies of the current technology. What are the major limitations of the generation 2 algorithms? One response is a variation on the old question "If you're so smart, why aren't you rich?" If these algorithms are so effective, why aren't they used by more companies? Certainly the list of implementations given at the beginning of this introduction is impressive, but they barely scratch the surface of the tens of thousands of companies that own truck fleets and make up the 15% of the economy devoted to transportation. What inhibits these other firms from using the algorithms that have been developed?

The answer seems to be a lack of robustness in currently implemented algorithms that makes them hard to transfer from one company to another. In talking with researchers involved in implementing vehicle routing algorithms, one hears a common story. A heuristic algorithm is selected which appears reasonable for a particular situation. Under testing, the algorithm may work well in most cases, but occasionally produces obviously unreasonable results. The heuristic is then 'patched up' to fix the troublesome cases, leading to an algorithm with growing complexity and computational requirements. After considerable effort, a procedure is created that works well for the situation at hand, but one is left with the disquieting feeling that what has been produced is extremely sensitive to nuances in the data and will not perform well when transferred to other environments. It's not uncommon that a heuristic developed for a particular geographic region of a company's operation will perform poorly in another region served by the same company. What's needed are more robust tools for vehicle routing.

As the need for robustness has become more clear during the last decade, fortunately the resources for achieving robustness have also grown. Rapidly decreasing computation costs are pushing the tradeoff between computation time and solution quality in the direction of higher quality solutions. The accuracy of data on the cost of travel between customers has been greatly improved by the creation of road network databases. Finally, the base of fundamental research on which to draw has greatly expanded, including optimization research on related models like the traveling salesman problem and new approaches to heuristic problem solving growing out of the artificial intelligence community.

One could imagine various approaches to achieving robustness. The simplest might be to provide an interactive interface for vehicle routing algorithms based on a graphic display that would allow a human dispatcher to manually correct difficulties with solutions. A more complex approach could draw on past research in artificial intelligence. For example, we could use an expert system to capture the expertise used by an operations research analyst in developing and tuning a vehicle routing algorithm for a particular application. The expert system could have available to it all existing vehicle routing algorithms so as to be able to select an appropriate algorithm for a particular application. Test data could be supplied to the expert system with which to evaluate the performance of candidate algorithms and tune algorithm parameters as appropriate.

My own prediction is that optimization algorithms offer the best promise for achieving robustness. Although optimization has not been considered a practical approach for real problems in the past, rapidly decreasing computation costs and promising new research are causing a reevaluation of this assumption. In a practical application, an optimization algorithm need not be run to full optimality but can be stopped as soon as an acceptable solution has been obtained. As such, these algorithms offer all the computational tractability advantages of heuristics, but they also allow a user to control the tradeoff between solution quality and computational cost.

Finally, significant research has been conducted on the closely related traveling salesman problem. This provides a base of theoretical research on which to draw for vehicle routing optimization. A number of successful vehicle routing optimization algorithms are adaptations of traveling salesman algorithms. The dramatic increase in the size of traveling salesman problems solvable to optimality also suggests what could be accomplished for vehicle routing with a concerted research effort.

The three remaining sections of this paper correspond to the three generations of vehicle routing research and present key examples of simple heuristics, math-programming based heuristics and two recent approaches that are emerging as the focal points of generation 3: optimization algorithms and various heuristic approaches based on new results in artificial intelligence. Other survey's of vehicle routing which one may wish to consult include Bodin, Golden, Assad & Ball [1983], Bodin [1991], Christofides [1985], Golden & Assad [1986], Laporte & Nobert [1987], and Magnanti [1981].

## 2. Generation one — simple heuristics

For convenience in defining vehicle routing heuristics we introduce the graph $G = (N, A)$ with node set $N = \{0, 1, \ldots, n\}$ indexing the depot (node 0) and customers 1 to $n$ and arc set $A = N \times N$ corresponding to all links between points in the problem. The length of arc $ij$ is $c_{ij}$.

The simple heuristics developed during generation 1 can be grouped into three categories: route building heuristics, route improvement heuristics, and two-phase methods. Route building heuristics select arcs sequentially until a feasible solution has been created. Arcs are chosen based on some cost minimization criterion subject to the restriction that the selection does not create a violation of vehicle capacity constraints. Route improvement heuristics begin with a set of arcs $S \subseteq A$ that constitute a feasible schedule and seek an interchange of a set $S_1 \subset S$ with a set $S_2 \subset A - S$ that reduces cost while maintaining feasibility. Two-phase methods first assign customers to vehicles without specifying the sequence in which they are to be visited. In phase 2 sequenced routes are obtained for each vehicle using some traveling salesman problem algorithm. This section describes the most important examples of these three types of heuristics.

The Clarke & Wright [1964] method is the best known route building heuristic. Clarke and Wright begin with an infeasible solution in which every customer is supplied individually by a separate vehicle. By combining any two of these single customer routes, we would use one less vehicle and also reduce cost. Recall that 0 indexes the central depot. The cost of serving customers $i$ and $j$ individually by two vehicles is $c_{0i} + c_{i0} + c_{0j} + c_{j0}$ while the cost of one vehicle serving $i$ and $j$ on the same route is $c_{0i} + c_{ij} + c_{j0}$. Thus, combining $i$ and $j$ results in a cost savings of $s_{ij} = c_{i0} + c_{0j} - c_{ij}$.

Clarke and Wright select the arc $ij$ with maximum $s_{ij}$ subject to the requirement that the combined route is feasible. Customers $i$ and $j$ are now regarded as a single macro customer. A customer $l$ can be linked to the macro customer at cost $\min\{c_{li}, c_{lj}\}$. With this convention, the route combining operation can be applied iteratively. In combining routes, we can simultaneously form partial routes for all vehicles or sequentially add customers to a given route until the vehicle is fully loaded. The latter is called the sequential Clarke and Wright method.

There have been many modifications to the basic Clarke and Wright method. Gaskell [1967] and Yellow [1970] independently introduced the concept of a modified savings given by $s_{ij} - \theta c_{ij}$ where $\theta$ is a scalar parameter. By varying $\theta$, one can place greater or less emphasis on the cost of travel between two nodes relative to their distance from the central depot. This parameter can be altered and different solutions obtained. The best of these is then chosen. Golden, Magnanti & Nguyen [1977] have used computer science techniques to substantially reduce the running time of Clarke and Wright.

Altinkemer & Gavish [1991] have recently achieved significantly improved computational results with an implementation of Clarke-Wright in which several pairs of routes are combined on a single iteration. The choice of which pairs of routes to combine is determined by solving a maximum matching problem in which the nodes correspond to existing partial routes and the weight on an edge joining two nodes is the savings that would result from joining the two partial routes corresponding to the nodes.

Lin [1965] and Lin & Kernighan [1973] demonstrated the effectiveness of local improvement for the traveling salesman problem. They introduced the term $k$-optimal for a traveling salesman solution that can not be improved by deleting $k$ or fewer arcs and replacing them feasibly with different arcs. In extensive computational testing they showed that a 3-optimal solution can be computed quickly and closely approximates the optimal value. Christofides & Eilon [1969] adapted the Lin and Kernighan method for vehicle routing by employing a reformulation of the vehicle routing problem without capacity constraints (called a $K$-traveling salesman problem) into a traveling salesman problem. A solution to the $K$-traveling salesman problem consists of $K$ tours beginning and ending at the depot. This problem can be converted to a traveling salesman problem with $n + K$ cities by constructing $K$ copies of the depot. Each copy of the depot is joined to customer $i$ by an arc of length $c_{0i}$. Arcs joining copies of the depot have infinite length. It's easy to see that a solution to this $n + K$ city traveling salesman problem is also a solution to the $K$ traveling salesman problem. Christofides and Eilon used this mapping to apply the Lin and Kernighan 3-optimal interchange procedure to feasible vehicle routing solutions, with the added requirement that any arc exchange made cannot destroy feasibility of the vehicle capacity constraints. Russell [1977] improved on the Christofides–Eilon procedure by selectively considering some exchanges involving more than 3 arcs. More recent research on local improvement methods is reported in Savelsberg [1985] and Thompson [1988].

Gillett & Miller's [1974] 'Sweep' algorithm is the simplest and best known two phase method. This method is directly applicable only to planar problems in which customers are located at points in the plane and $c_{ij}$ is the Euclidean distance between points $i$ and $j$. In phase 1, customers are represented in a polar coordinate system with the origin at the depot. A customer is chosen at random and the ray from the origin through the customer is 'swept' either clockwise or counterclockwise. Customers are assigned to a given vehicle as they are swept, until further assignment of customers would exceed the capacity of that vehicle. Then a new vehicle is selected and the sweep continues, with assignments now being made to the new vehicle. This process continues until all customers have been assigned to a vehicle. In phase 2, the customers assigned to each route are sequenced using some traveling salesman algorithm. Other two phase heuristics have been suggested by Christofides, Mingozzi & Toth [1979] and by Tyagi [1968].

We note that none of the simple heuristics described here place much emphasis on the vehicle capacity constraints. While these constraints are checked for violation as necessary, they have no other influence on the choices made in forming a solution. For this reason, these simple heuristics can easily terminate with an infeasible or poor solution if capacity constraints are tight.

We conclude this section with a computational comparison of four of the heuristics we have described. Table 1 summarizes problem characteristics and results.

Table 1
Computational comparison of heuristics

| Problem | $n$ | $K$ | $\frac{\sum_{i=1} a_i}{Kb}$ | Clarke & Wright | | Sweep | | Christofides & Eilon | | Russell | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Cost | CPU [a] | Cost | CPU [a] | Cost | CPU [b] | Cost | CPU [c] |
| 1 | 50 | 5 | 0.97 | 585 | 0.8 | 532 | 12.2 | 556 | 120 | 524 | 15 |
| 2 | 75 | 10 | 0.97 | 900 | 1.7 | 874 | 24.3 | 876 | 240 | 854 | 244.8 |
| 3 | 100 | 8 | 0.93 | 886 | 2.4 | 851 | 65.1 | 863 | 600 | 833 | 100 |
| 4 | 150 | 12 | 0.94 | 1204 | 6.6 | 1079 | 142.0 | [d] | [d] | [d] | [d] |
| 5 | 199 | 17 | 0.95 | 1540 | 11.0 | 1389 | 252.2 | [d] | [d] | [d] | [d] |
| 6 | 100 | 10 | 0.91 | 831 | 2.4 | 937 | 50.8 | [d] | [d] | [d] | [d] |

[a] CDC 6600 seconds.
[b] IBM 7090 seconds.
[c] IBM 370/168 seconds.
[d] No results for this problem.

All test problems are planar in the sense that the given data are coordinates of customers and $c_{ij}$ equals the Euclidean distance between customers $i$ and $j$. The customer coordinates for problems 1–3 are randomly generated and are given in Christofides & Eilon [1969]. Problem 4 was produced by adding the customers of problems 1 and 3 with the depot and vehicle capacities as in problem 3. Problem 5 was produced by adding the customers of problem 4 with the first 49 customers of problem 2. The data from problem 6 is given in Christofides, Mingozzi & Toth [1979]. This problem was designed to resemble real problems by grouping customer locations into clusters. These six problems are 'classics' in that virtually every researcher proposing heuristics or optimization algorithms has tested their procedure on these problems. Hence they are useful in providing a common base of comparison across a wide range of algorithms.

The value in the column headed $K$ gives the number of vehicles used. Except for problem 5, $K = \lceil \sum_{i=1}^{n} a_i/b \rceil$, the smallest number of vehicles that will admit a feasible solution. For problem 5, $\lceil \sum_{i=1}^{n} a_i/b \rceil = 16$, but using $K = 16$ results in vehicle capacity utilization of 99.9%. This is such a tightly constrained problem that, with a couple of exceptions that will be noted when we discuss Generation 3 research, all researchers have used $K = 17$. The ratio of total demand to total vehicle capacity is listed in Table 1 to provide a measure of the tightness of vehicle capacity constraints.

Results for the first two heuristics in Table 1 are taken from Christofides, Mingozzi & Toth [1979], for the third from Christofides & Eilon [1969] and for the fourth from Russell [1977]. The solution values for Sweep are the best of $n$ executions with different starting customers and the solution times are the sums of the times for all of these runs. It should also be noted that the parallel implementation of Clarke–Wright based on maximum matchings as developed by Altinkemer & Gavish [1991] achieved a value of 1351 on problem 5, although they do not report the computation time required to find the solution.

One qualification needs to be made for computational results reported throughout this chapter on these six problems. Since $c_{ij}$ is the Euclidean distance between two points, it will in general be a real number. Nonetheless, some researchers have rounded $c_{ij}$ to the nearest integer, or in some cases even truncated to the next smaller integer. Clearly, either of these changes can have a significant effect on the cost of a particular solution. For example, it is known that the optimal cost for problem 1 with all $c_{ij}$ rounded to the nearest integer is 521, while the best feasible solution found to date for this problem with real $c_{ij}$ is 524.61.

This complicates the comparison of heuristics tested on these problems. Because researchers have not been explicit in reporting how they were computing $c_{ij}$, apparent differences between two heuristics can be the result of nothing more than different rounding conventions for the costs. Also, even researchers who used real $c_{ij}$ often report the resulting objective function as an integer either rounding or truncating downwards. For example, the value of 524 reported by Russell for problem 1 is probably the best found real solution of 524.61. In reporting results in the rest of this chapter, I will not try to second guess the rounding assumptions employed by various researchers, but will simply reproduce their results as reported in the published literature. Recently, researchers have become more careful in declaring their rounding assumptions, so for the generation 3 results I shall generally be able to identify the rounding assumption that was used.

## 3. Generation two — mathematical programming based heuristics

Mathematical programming based heuristics are very different in character from the simple heuristics described in the preceding section. They solve to optimality some mathematical programming approximation of the vehicle routing problem. We describe in this section two well known examples of this approach which have produced computational results that are usually superior to the simple heuristics described in the last section. The first uses the generalized assignment problem and the second the set partitioning problem to approximate the vehicle routing problem. Haimovich & Rinnooy Kan [1985], Altinkemer & Gavish [1987] and Bramel & Simchi-Levi [1992] use other advanced concepts from mathematical programming to develop asymptotically optimal heuristics for vehicle routing.

Fisher & Jaikumar [1981] solved a generalized assignment problem approximation of the vehicle routing problem to obtain an assignment of customers to vehicles. The customers assigned to each vehicle can then be sequenced using any traveling salesman algorithm. To motivate the generalized assignment problem approximation, first note that the vehicle routing problem can be represented exactly by the following *nonlinear* generalized assignment problem. Defining

$$y_{ik} = \begin{cases} 1, & \text{if point } i \text{ (customer or depot) is visited by vehicle } k \\ 0, & \text{otherwise} \end{cases}$$

and $y_k = (y_{0k}, \ldots, y_{nk})$, the vehicle routing problem is defined as

$$\min \sum_k f(y_k)$$

such that

$$\sum_i a_i y_{ik} \le b, \qquad k = 1, \ldots, K$$

$$\sum_k y_{ik} = \begin{cases} K, & i = 0 \\ 1, & i = 1, \ldots, n \end{cases}$$

$$y_{ik} = 0 \text{ or } 1, \qquad i = 0, \ldots, n, \; k = 1, \ldots, K$$

where $f(y_k)$ is the cost of an optimal traveling salesman problem tour of the points in $N(y_k) = \{i \,|\, y_{ik} = 1\}$. Defining

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

the function $f(y_k)$ can be defined mathematically by

$$f(y_k) = \min \sum_{ij} c_{ij} x_{ijk}$$

such that

$$\sum_i x_{ijk} = y_{jk}, \qquad j = 0, \ldots, n$$

$$\sum_j x_{ijk} = y_{ik}, \qquad i = 0, \ldots, n$$

$$\sum_{ij \in S x S} x_{ijk} \le |S| - 1, \qquad S \subseteq N(y_k), \; 2 \le |S| \le n$$

$$x_{ijk} = 0 \text{ or } 1, \qquad i = 0, \ldots, n, \; j = 0, \ldots, n$$

Of course, this definition doesn't help us in computations since we lack a closed form expression for $f(y_k)$. The generalized assignment heuristic replaces $f(y_k)$ with a linear approximation $\sum_i d_{ik} y_{ik}$ and solves the resulting linear generalized assignment problem to obtain an assignment of customers to vehicles. To obtain the linear approximation, first specify $K$ 'seed' customers $i_1, \ldots, i_K$ that are assigned one to each vehicle. Without loss of generality, assume customer $i_k$ is assigned to vehicle $k$ for $k = 1, \ldots, K$. Then set the coefficient $d_{ik}$ to the cost of inserting customer $i$ into the route on which vehicle $k$ travels from the depot directly to customer $i_k$ and back. Specifically, $d_{ik} = c_{0i} + c_{ii_k} - c_{0i_k}$. Clearly, the seed customers define the general direction in which each vehicle will travel and the generalized assignment problem completes the assignment of customers to routes given this general framework.

Many algorithms exist for optimal solution of the generalized assignment problem with a linear objective function, such as the Lagrangean relaxation algorithm described in Fisher, Jaikumar & Van Wassenhove [1986]. One can also solve the generalized assignment problem heuristically by assigning customers sequentially to the vehicle with minimum $d_{ik}$ in which they fit. Commonly used sequences for processing customers include decreasing order of distance from the depot or decreasing order of the difference between the assignment cost for best and second best vehicles.

Seed customers can be chosen by a variety of heuristics. Fisher & Jaikumar ([1981] suggest the following procedure for the planar case. They actually determine $K$ seed points $w_1, \dots, w_K$ in the plane rather than $K$ seed customers. These points are used exactly like seed customers in determining $d_{ik}$.

To determine $w_1, \dots, w_K$, the plane is partitioned into $K$ cones with origin at the depot corresponding to the $K$ vehicles. To determine these cones, first partition the plane into $n$ smaller cones, one for each customer. The infinite half ray forming the boundary between two customer cones is positioned to bisect the angle formed by half rays through the two customers. Associate the weight $a_i$ with customer cone $i$ and define $\bar{b} = \sum_{i=1} a_i / K$.

Each vehicle cone is then formed from a contiguous group of customer cones or fractions of customer cones. The weight of the group is required to equal $\bar{b}$. A fraction of a customer cone $i$ contributes the same fraction of $a_i$ to the total group weight. The point $w_k$ is located along the ray bisecting the $k$th cone. The distance of $w_k$ from the origin is fixed so that the weight included inside the arc through $w_k$ is 0.75 $\bar{b}$. This weight is defined to equal the sum of the $a_i$ for all customers inside the arc plus a fraction of $a_i$ for the customer just outside the arc. This fraction is $A/(A+B)$, where $A$ is the distance to the arc from the customer just inside the arc and $B$ is the distance to the arc from the customer just outside the arc.

In the more general case, when planarity is not assumed, seeds are frequently chosen with the following simple rule. Choose the first seed $s_1$ to be a customer farthest from the depot. If $k$ seeds have been chosen, choose $s_{k+1}$ to solve

$$\max_i \min \{c_{i0}, \min_{j=1,\cdots,k} c_{is_j}\}$$

Bramel & Simchi-Levi [1992] have provided an elegant approach to seed selection by expanding the generalized assignment model to involve both choice of which customers will be seeds and assignment of customers to seeds.

Let

$$z_j = \begin{cases} 1, & \text{customer } j \text{ is chosen to be a seed} \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ij} = \begin{cases} 1, & \text{customer } i \text{ is assigned to a route on which customer } j \text{ is a} \\ & \text{seed} \\ 0, & \text{otherwise} \end{cases}$$

Define $d_{ij} = c_{0i} + c_{ij} - c_{0j}$ and $v_j = 2c_{0j}$. Then an assignment of customers to routes is determined by solving

$$\min \sum_{ij} d_{ij} y_{ij} + \sum_j v_j z_j$$

$$\sum_j z_j = K$$

$$\sum_i a_i y_{ij} \le b, \qquad j = 1, \cdots, n$$

$$\sum_j y_{ij} = 1, \qquad i = 1, \cdots, n$$
$$y_{ij} \leq z_j, \qquad \text{all } ij$$
$$y_{ij} = 0 \text{ or } 1, \qquad \text{all } aj$$
$$z_j = 0 \text{ or } 1, \qquad j = 1, \cdots, n$$

The model is called a capacitated concentrator location problem; Bramel & Simchi-Levi [1992] give an optimization algorithm for it based on a Lagrangian relaxation in which the constraints $\sum_j y_{ij} = 1$ are dualized for all $i$.

The generalized assignment heuristic can be adapted to accommodate various side constraints. Examples include time windows [Nygard, Greenberg, Bolkan & Swenson, 1988; Koskosidis, Powell & Solomon, 1989] and split deliveries [Brenninger-Goethe, 1989].

The set partitioning heuristic begins by enumerating a number of candidate vehicle routes. A candidate route is defined by a set $S \subseteq \{1, \ldots, n\}$ of customers to be delivered by a single vehicle and a delivery sequence for these customers. We index the candidate routes by $j$ and define the following parameters.

$$c_j = \text{the cost of candidate route } j$$
$$a_{ij} = \begin{cases} 1, & \text{if customer } i \text{ is included on route } j \\ 0, & \text{otherwise} \end{cases}$$
$$J = \text{number of candidate routes generated}$$

Letting

$$y_j = \begin{cases} 1, & \text{if candidate route } j \text{ is used} \\ 0, & \text{otherwise} \end{cases}$$

the vehicle routing problem can then be approximated by the following set partitioning problem.

$$\min \sum_{j=1}^{J} c_j y_j$$
$$\sum_{j=1}^{J} y_j = K$$
$$\sum_{j=1}^{J} \sum_{k=1}^{K} a_{ij} y_j = 1, \qquad i = 1, \ldots, n$$
$$y_j = 0 \text{ or } 1, \qquad j = 1, \ldots, J$$

There are many effective optimization algorithms for set partitioning. For example, the Lagrangian relaxation algorithm in Fisher & Kedia [1990] has solved problems as large as $n = 200$ and $J = 10{,}000$.

The set partitioning approach will find an optimal solution if the candidate route list contains all feasible routes. In most situations, this would result in a set partitioning problem too large to be solvable, so one instead heuristically generates a candidate route list designed to be short enough to allow solution of the set partitioning problem yet long enough to result in an optimal or near optimal solution to the vehicle routing problem. We can usually do this if the number of feasible routes is small, which can happen if the $a_i$ are large relative to $b$ so that the number of customers delivered by a vehicle is small (on the order of 1–4 deliveries per truck) or if there are additional restrictions (such as time window constraints) that limit the number of feasible routes. The set partitioning approach is guaranteed to find a feasible solution if one initially applies any of the heuristics described in this chapter to obtain a feasible solution and includes the routes of this solution in the candidate list.

The set partitioning approach can easily accommodate certain kinds of complex constraints. For example, any feasibility condition for an individual vehicle route can usually be incorporated in the heuristic for generating candidate routes. Similarly, this approach can handle costs which are a complicated function of route characteristics since all one needs to do is evaluate this function for given routes.

The set partitioning approach was originally proposed by Balinski & Quandt [1964], although not implemented for large scale problems until much later. Successful applications in truck scheduling include Foster & Ryan [1976] and Fleuren [1988] for time window constrained problems and in ship scheduling Fisher & Rosenwein [1989]. Callen, Jarvis & Ratliff [1981] developed an interactive implementation of this approach in which a user (aided by a color graphics interface) specifies additional candidate routes during the solution process. Agarwal, Mathur & Salkin [1989] and Desrochers, Desrosiers & Solomon [1990] (for time window constrained problems) report exact algorithms based on a set partitioning model and column generation. Bramel & Simchi-Levi [1993] provide theoretical results on the effectiveness of the set partitioning/column generation approach.

Table 2 summarizes on the 6 problems introduced in section 2 the computational performance of the Fisher & Jaikumar [1981] generalized assignment heuristic, Bramel & Simchi-Levi [1992] extended generalized assignment heuristic, and the set partitioning heuristic as implemented by Foster & Ryan [1976].

Table 2
Computational comparison of mathematical programming based heuristics

| Problem | $n$ | Fisher & Jaikumar | | Bramel & Simchi-Levi | | Foster & Ryan | |
|---|---|---|---|---|---|---|---|
| | | Cost | CPU [a] | Cost | CPU [b] | Cost | CPU [c] |
| 1 | 50 | 524 | 9.33 | 524.6 | 68 | 523 | 2.6 |
| 2 | 75 | 857 | 11.95 | 848.2 | 406 | 864 | 6.8 |
| 3 | 100 | 833 | 17.7 | 832.9 | 890 | 825 | 16.9 |
| 4 | 150 | 1014 | 33.6 | 1088.6 | 2552 | [d] | [d] |
| 5 | 199 | 1420 | 40.1 | 1461.2 | 4142 | [d] | [d] |
| 6 | 100 | 824 | 6.1 | 826.1 | 400 | [d] | [d] |

[a] DEC 10 seconds.
[b] RS600 Model 550 seconds.
[c] IBM 370/168 seconds.
[d] No results for this problem.

In these computational experiments, the generalized assignment problem was solved using the Lagrangian relaxation algorithm described in Fisher, Jaikumar & Van Wassenhove [1986] and the traveling salesman problem for the customers assigned to each vehicle were solved optimally using an algorithm similar to the one reported in Miliotis [1976].

## 4. Generation three — what are the new frontiers?

### *4.1. Artificial intelligence*

Artificial intelligence techniques have been applied to vehicle routing in two ways. First, expert systems are being developed to assist a user faced with a particular application in choosing an appropriate algorithm and tuning the parameters of that algorithm to the problem at hand. Second, successful new algorithms have recently been developed using artificial intelligence search techniques like simulated annealing and tabu search.

The first approach begins with the observation that there are many types of vehicle routing problems and also many different algorithms have been developed, so that a typical problem faced by a vehicle routing analyst is to match an appropriate algorithm with a particular application. Relevant information about the application includes not only the formal definition of the problem but characteristics of the data such as average number of stops per vehicle or tightness of the vehicle capacity constraints. To illustrate the problem of choosing an appropriate algorithm, consider two of the algorithms presented in the last section — the generalized assignment and set partitioning algorithms. Each of these algorithms works best on problems with particular data characteristics. For example, the set partitioning algorithm would be particularly appropriate for a problem that had an average of 1.5 customers per vehicles because the small number of customers per vehicle would imply that the number of ways customers could be combined to form vehicle routes would also be small. The generalized assignment algorithm would not be a good choice for such a problem since the focus of this method is to efficiently assign remaining customers once a single seed customer has been assigned to each route. With such a small number of customers per route, choosing the seed customers would almost completely specify the routes and the contribution made through solution of the generalized assignment problem would be minimal. On the other hand, the generalized assignment method fits particularly well on problems that have very tight vehicle capacity constraints since the vehicle capacity constraints play a prominent role when the generalized assignment problem is solved.

Each of these algorithms also have parameters whose specification should depend on the data characteristics of the particular problem being solved. For example, the set partitioning algorithm requires rules for generating the candidate route list and the generalized assignment algorithm requires rules for setting seed customers, as well as a choice of whether the generalized assignment problem will be solved optimally or heuristically, and if the latter, a specification of the heuristic to be used.

Several researchers have studied the question of choosing and tuning an appropriate algorithm for a particular vehicle routing application. Desrochers, Lenstra & Savelsbergh [1990] have developed a scheme for classifying vehicle routing problems which is a helpful first step in guiding a user in the selection of a method that is well suited for his specific situation. Kadaba, Nygard & Juell [1991] have applied various artificial intelligence techniques such as neural networks and genetic algorithms to the choice of seed customers in the generalized assignment heuristic.

Finally, Potvin, Lapalme & Rousseau [1989] have developed an interactive graphic computer system that allows an expert user to formulate and test various versions of vehicle routing algorithms within a broad class. The system works within the framework of a generalized assignment heuristic in which the generalized assignment problem is solved heuristically. The graphical interface allows the user to either manually perform or provide rules for performing the functions of choosing seed customers, adding customers to routes in the heuristic solution of the generalized assignment problem and improving the delivery sequence of a set of customers assigned to a particular vehicle.

All of this research constitutes a very promising beginning towards the goal of developing an expert system for automatically choosing and tuning algorithms for a particular application. It's hard to compare the computational performance of this approach with other algorithms, since no researcher has tested a procedure of this type on standard test problems such as the six problems we have been using in this chapter. Indeed, given that the goal of this approach is adaptability to a wide range of applications, testing on a small number of standard problems would not make sense.

Significantly improved results for the six standard test problems have been obtained by adapting artificial intelligence search procedures to vehicle routing. A generic search procedure for the vehicle routing problem begins with a starting solution $S$ and a rule for constructing a neighborhood $N(S)$ of alternative solutions that are near to S in some sense. The search procedure then chooses a new solution from $N(S)$ and iterates until some stopping criterion is reached.

Traditional local improvement methods choose a solution from $N(S)$ with a strictly improved objective function value and stop when $N(S)$ contains no such improving solution. Tabu search and annealing allow the selection of nonimproving solutions under certain conditions to be defined below.

An initial solution for a search algorithm can be obtained by applying any of the heuristics defined in this chapter. Usually the neighborhood $N(S)$ is defined to be all solutions obtained by exchanging $n_1$ customers on a given route with $n_2$ customers on another route. Typically, $n_1 \leq 1$ and $n_2 \leq 1$, so the exchanges permitted consist of moving a single customer from one route to another route or exchanging two customers on different routes.

Different approaches can be used to compute the change in cost resulting from the addition or deletion of a customer on a route. Ideally, one would evaluate this change by optimally solving a traveling salesman problem over the depot plus the customers on the route before and after the addition or deletion of the new customer. However, this approach can be computationally expensive so a less accurate but faster method is generally used. When a customer is deleted, the cost of the new route is computed without changing the sequence of the customers that remain on the route. Similarly, the customer added to a route is inserted between two existing customers without changing the sequence of the original customers on the route. The choice of where to insert the new customer is made to minimize the increase in cost.

With a traditional local improvement method that only accepts solutions that strictly improve the objective function, one can choose whether to accept the first solution in $N(S)$ that improves the objective function or to examine all solutions in $N(S)$ and choose the one producing the greatest reduction in the objective function.

Tabu search chooses the best solution contained in $N(S)$ that does not violate certain restrictions designed to prevent the algorithm from cycling. Typically, these restrictions prevent for the next $t$ iterations a movement of customers that would 'undo' a previous movement. For example, if customer $c$ is moved from route A to some other route on iteration $k$, then on iterations $k+1$ through $k+t$ we prohibit any exchange that would move customer c back to route A. Similarly, if on iteration k customer $c_1$ on route A and $c_2$ on route B are exchanged, then on iterations $k+1$ through $k+t$ we prohibit any movement that would return customer $c_1$ to route A and customer $c_2$ to route B. Tabu search stops after some fixed iteration limit has been reached.

Simulated annealing searches the neighborhood of $N(S)$ in a defined order. Let $\Delta$ denote the increase in object value (over the current incumbent solution $S$) for some $S' \epsilon N(S)$. Then $S'$ is accepted as the new incumbent solution either if $\Delta \leq 0$ or $\Delta > 0$ and $e^{-\Delta/T} \geq \theta$ where $\theta$ is a uniform random parameter $0 < \theta < 1$ and $T$ is a control parameter of the search called the 'temperature.' Typically, $T$ is gradually lowered during the search procedure, steadily reducing the probability that a nonimproving solution will be accepted. If a complete search of $N(S)$ fails to produce a new incumbent, the value of $T$ is raised by some amount and the search repeated. The simulated annealing algorithm stops when a fixed consecutive number of searches of $N(S)$ fails to produce a new incumbent solution.

Three authors have constructed successful implementations of either tabu search or simulated annealing. Gendreau, Hertz & Laporte [1993] applied tabu search using neighborhoods consisting of all solutions that could be constructed by moving a single customer from one route to another. Osman [1993] applied tabu search and simulated annealing using a neighborhood consisting of all solutions that could be constructed by moving one customer from a given route to another or exchanging two customers on different routes. Taillard [1993] applied tabu search with the same neighborhood definition as Osman. Taillard also penalized moves that had been performed frequently by adding a penalty term to the cost improvement for the move. The penalty term was proportional to the number of times the move had been performed thus far. Taillard also designed a parallel implementation of tabu search suitable for large problems. In his parallel

Table 3
Computational comparison of artificial intelligence search algorithms

| Problem | Tabu search | | | | | Simulated annealing | |
|---|---|---|---|---|---|---|---|
| | Gendreau, Hertz & Laporte | | Osman | | Taillard | Osman | |
| | Cost | CPU[a] | Cost | CPU[b] | Cost | Cost | CPU[b] |
| 1 | 524.61 | 6 | 524 | 2 | 524.61 | 528 | 3 |
| 2 | 835.32 | 53.8 | 844 | 3 | 835.26 | 538 | 107 |
| 3 | 826.14 | 18.4 | 835 | 26 | 826.14 | 829 | 156 |
| 4 | 1031.07 | 58.8 | 1044 | 59 | 1028.42 | 1058 | 84 |
| 5 | 1311.35 | 90.9 | 1334 | 54 | 1298.79 | 1378 | 39 |
| 6 | 819.56 | 16 | 819 | 15 | 819.56 | 826 | 11 |

[a] Silicon Graphics 36MHZ workstation minutes.
[b] VAX 8600 minutes.

implementation, the customers are divided into groups and tabu search is initially performed within these groups.

The results achieved by these various implementations of tabu search and simulated annealing for the six standard problems considered in this chapter are shown in Table 3.

All computations are with real $c_{ij}$. Taillard does not report CPU times. The solutions to problem 5 obtained by Osman have $K = 16$ rather than $K = 17$ as used by other researchers.

### 4.2. Optimization algorithms

This section presents four optimization algorithms for vehicle routing based on polyhedral combinatorics, a matching relaxation, a shortest path relaxation and a $K$-tree relaxation. Other research on optimization is reported in Christofides, Mingozzi & Toth [1981b] and Lucena [1986].

The first approach closely parallels the highly successful use of polyhedral combinatorics developed for the traveling salesman problem by Chvatal [1973], Grotschel [1980], Grotschel & Padberg [1979], Padberg & Hong [1980] and Grotschel & Pulleyblank [1986]. The book by Lawler, Lenstra, Rinnooy Kan & Shmoys [1985] contains an excellent review of this research. This research relies on the following formulation of the $n$ city symmetric TSP, where $x_{ij}$ is a 0–1 variable equal to 1 if an only if the salesman travels directly between cities $i$ and $j$.

$$\min \sum_{i<j} c_{ij} x_{ij} \tag{1}$$

$$\sum_{j<i} x_{ij} + \sum_{j>i} x_{ji} = 2, \quad i = 1, \ldots, n \tag{2}$$

$$\sum_{\substack{ij \in S \times S \\ i<j}} x_{ij} \leq |S| - 1, \quad \text{for all } S \subset \{1, \ldots, n\},\ 2 \leq |S| \leq n-1 \tag{3}$$

$$x_{ij} = 0 \text{ or } 1, \qquad 1 \leq i < j \leq n \tag{4}$$

As discussed in Laporte, Nobert & Desrochers [1985], this formulation can be adapted to the vehicle routing problem by adding variables and a constraint like (2) to model the depot and by changing the right-hand side of (3) to impose the vehicle capacity constraints. We present the resulting formulation below. The variable $x_{ij}$ now represents the number of vehicles traveling directly between points $i$ and $j$ and $V(S) = \lceil \sum_{i \in S} a_i / b \rceil$ where $\lceil y \rceil$ denotes the smallest integer not less than $y$. This value for $V(S)$ can be replaced by a tighter lower bound on the number of vehicles needed for delivery to the customers in $S$ if one is available.

$$\min \sum_{i<j} c_{ij} x_{ij} \tag{1'}$$

$$\sum_{j<i} x_{ij} + \sum_{j>i} x_{ji} = 2, \qquad i = 1, \ldots, n \tag{2}$$

$$\sum_{\substack{ij \in S \times S \\ i<j}} x_{ij} \leq |S| - V(S), \qquad \text{for all } S \subset \{1, \ldots, n\},\ 2 \leq |S| \leq n-1 \tag{3'}$$

$$x_{ij} = 0 \text{ or } 1, \qquad 1 \leq i < j \leq n \tag{4}$$

$$\sum_j x_{0j} = 2K \tag{5}$$

$$x_{0j} = 0,\ 1 \text{ or } 2, \qquad j = 1, \ldots, n. \tag{6}$$

The case $x_{0j} = 2$ corresponds to a route containing only customer $j$. If single customer routes cannot occur, we can require $x_{0j} = 0$ or 1.

Laporte, Nobert & Desrochers [1985] applied a general purpose integer programming algorithm to this formulation. They added capacity constraints (3′) to the formulation as they were violated since these constraints are too numerous to specify a priori.

In gauging the computational difficulty of a vehicle routing problem, we must consider the number of customers, number of vehicles and tightness of the vehicle capacity constraints as measured by average vehicle utilization $(\sum_{i=1} a_i)/Kb$. Laporte and coworkers were able to optimally solve randomly generated problems with 50 to 60 customers, 1 or 2 vehicles and average vehicle utilization to 0.74.

The TSP formulation (1)–(4) can be strengthened by the addition of other valid inequalities, such as the comb inequalities developed by Chvatal [1973] and generalized by Grotschel & Padberg [1979]. Laporte & Nobert [1987] and Laporte & Bourjolly [1984] generalized these comb inequalities to the following valid constraints for the vehicle routing problem. Let $W_l \subseteq \{1, \ldots, n\}, l = 0, \ldots, k$ denote sets satisfying

$$|W_l - W_0| \geq 1 \qquad (l = 1, \ldots, k)$$

$$|W_l \cap W_0| \geq 1 \qquad (l = 1, \ldots, k)$$

$$|W_l \cap W_{l'}| = 1 \qquad (1 \leq l, < l' \leq k).$$

Then the following comb inequality holds for every feasible vehicle routing solution, where, as before, $V(S)$ denotes any lower bound on the number of vehicles required to deliver the customers in set $S$.

$$\sum_{l=0}^{k} \sum_{i,j \in W_l} x_{ij} \leq \sum_{l=0}^{k} |W_l| - \left\lceil \frac{1}{2} \sum_{l=1}^{k} [V(W_l) + V(W_l - W_0) + V(W_l \cap W_0)] \right\rceil$$

Using comb inequalities and formulation (1′)–(6), Cornuejols & Harche [1989] solved to optimality the first problem reported in Table 1 with $c_{ij}$ rounded to integer values. The size of this problem (50 customers, 5 vehicles) and the tightness of the vehicle capacity constraints ($\sum_{i=1}^{n} a_i/Kb = 0.97$) makes this a notable accomplishment. Campos, Corberan & Mota [1991] and Araque, Kudva, Morin & Pekny [1994] have specialized the polyhedral approach to problems with $a_i = 1$ for all $i$.

Miller [1993] has developed an optimization algorithm that dualizes constraints (2) in formulation (1′)–(6) to obtain a maximum $b$-matching problem that is solved using the algorithm in Miller & Pekny [1994]. This algorithm has solved to optimality the first problem reported in Table 1 with $c_{ij}$ rounded to integral values.

Christofides, Mingozzi & Toth [1981a] compute bounds on the vehicle routing problem using a shortest path calculation. They define a graph with nodes $(i, q)$ for $i = 0, 1, \ldots, n$ and $q = 0, 1, \ldots, b$ and with an arc of length $c_{ij}$ joining each pair of nodes $(i, q)$ and $(j, q + a_j)$. Letting $l(i, q)$ denote the length of a shortest path from node $(o, o)$ to node $(i, q)$, $F(i, q) = l(i, q) + c_{i0}$ is a lower bound on the cost of delivering $q$ units on a vehicle route in which customer $i$ appears last. This route is feasible except that some customers may receive more than one delivery.

Define

$v(i, q, k) =$ minimum cost of $k$ routes carrying load $q$ with different last customers chosen from $\{1, \ldots, i\}$.

We can compute $v(i, q, k)$ with the following recursion

$$v(i, q, k) = \min\{v(i-1, q, k); \min_{q'}[v(i-1, q-q', k-1) + F(i, q')]\}$$

Then a lower bound on the vehicle routing problem is given by $v(n, \sum_{i=1} a_i, K)$. This bound corresponds to a set of $K$ routes that constitute a feasible solution except that some customers may not be delivered exactly once. The bound can be strengthened by introducing Lagrangian penalties on these violated customer delivery constraints.

Kolen, Rinnooy Kan & Trienekens [1987] generalized this algorithm to accommodate time window constraints. Christofides [1985] reports that an algorithm based on state space relaxation solved to optimality a problem with 53 customers and 8 vehicles. In addition to the vehicle capacity constraints, there were constraints on driving time and a few loose customer time window constraints.

Fisher [1990] describes an optimization algorithm in which lower bounds are obtained from a relaxation based on a generalization of spanning trees called $K$-trees. Given a graph with $n+1$ nodes, a $K$-tree is defined to be a set of $n+K$ edges that span the graph. The vehicle routing problem is modeled as a minimum cost degree constrained $K$-tree problem with side constraints. The side constraints are then dualized to obtain a Lagrangian relaxation.

Let $N = \{1, \ldots, n\}$, $N_0 = N \cup \{0\}$ and $x_{ij}$ denote a $0-1$ variable equal to 1 if edge $(i, j)$ is selected in a solution. Since the edge $(i, j)$ is undirected, it will simplify notation to adopt the convention that the subscripts $ij$ on $x_{ij}$ are an unordered pair, so $x_{ij}$ and $x_{ji}$ denote the same variable. Then $x_{ij}$ is defined for the $[n(n+1)]/2$ unordered pairs in $N_0 \times N_0$. Also define

$$
\begin{aligned}
x &= (x_{01}, x_{02}, \ldots x_{0n}, x_{12}, \ldots, x_{n-1,n})\\
X &= \left\{x \mid x = 0\text{–}1 \text{ and defines a } K\text{-tree satisfying} \sum_{i=1}^{n} x_{0i} = 2K\right\}.
\end{aligned}
$$

For $S \subseteq N$, let $\bar{S} = N_0 - S$, $a(S) = \sum_{i \varepsilon S} a_i$ and $r(S) = \lceil [a(S)]/b \rceil$ where $\lceil y \rceil$ denotes the smallest integer not less than $y$. For $S \subseteq N_0$, let $E(S)$ denote the edge set of a complete, undirected graph on the node set $S$, i.e., $E(S)$ is the set of all unordered pairs $ij$, $i \in S$, $j \in S$, $i \neq j$. Then the VRP can be formulated as

$$
Z^* = \min_{x \in X} \sum_{ij \in E(N_0)} c_{ij} x_{ij} \tag{7}
$$

$$
\sum_{\substack{j \in N_0 \\ j \neq i}} x_{ij} = 2, \qquad \text{for all } i \in N \tag{8}
$$

$$
\sum_{i \in S} \sum_{j \in \bar{S}} x_{ij} \geq 2r(S), \qquad \text{for all } S \subseteq N \text{ with } |S| \geq 2 \tag{9}
$$

In this formulation, routes with single customers are not allowed. Generally this is not a serious restriction for problems with tight vehicle capacity constraints and no $a_i$ that is large relative to $b$. Note that customer $j$ cannot appear alone on a route if $(\sum_{i=1}^{n} a_i) - a_j > (K-1)b$, because using a dedicated vehicle to deliver customer $j$ would leave insufficient vehicle capacity to service the remaining customers. Rearranging terms in this expression, we can see that the prohibition of single customer routes is not constraining if $a_j/b < \sum_{i=1}^{n} a_i/b - (K-1)$ for all $j$, a condition that will be satisfied if vehicle capacity constraints are sufficiently tight and no customer is large relative to vehicle capacity. The $K$-tree approach could be modified to allow for particular customers to be served on single stop routes by including two edges between these customers and the depot in the graph used to compute lower bounds, although, of course, this could affect lower bound strength.

Letting $u_i, i \in N$ and $v_s \geq 0$ for $S \subseteq N, |S| \geq 2$ denote Lagrange multipliers for constraints (2) and (3), we can define the following Lagrangian relaxation of (1)–(3).

$$Z_D(u, v) = \min_{x \in X} \sum_{ij \in E(N_0)} \bar{c}_{ij} x_{ij} + 2 \sum_{i=1}^{n} u_i + 2 \sum_{S \subseteq N} v_S r(S) \tag{10}$$

where $u_0 = 0$, and

$$\bar{c}_{ij} = c_{ij} - u_i - u_j - \sum_{\substack{S \text{ such that } i \in S,\, j \in \bar{S} \\ \text{or } i \in \bar{S},\, j \in S}} v_S \tag{11}$$

It is well known and easy to show that $Z_D(u, v) \leq Z^*$. A polynomial algorithm for (10) is given in Fisher [1991]. To obtain a tight lower bound, the subgradient method is used to approximate an optimal solution to $\max_{u,\, v \geq 0} Z_D(u, v)$. Since there are $0(2^n)$ constraints in the set (9), it is not feasible to tabulate explicitly all of these constraints prior to computation. Rather, a subset of these constraints is generated dynamically as they are violated. All other constraints have $v_s = 0$ and are ignored. Feasible solutions are obtained from a Lagrangian heuristic applied to Lagrangian solutions to remove infeasibilities.

Constraints (9) also lend themselves to tightening in the following way. The rationale underlying (9) is that $br(S)$ defines the minimum vehicle capacity that must enter and leave $S$ to feasibly carry the customers in set $S$. However, if $x_{ij} = 1$ in the left hand side of (9) for some customer $j \in \bar{S}$, then customer $j$ is serviced by the same vehicle as at least one customer in set $S$ and its demand subtracts from capacity available for set $S$. Hence, the total vehicle capacity entering and leaving set $S$ must be at least $a(S)$ plus the customer demand for all customers $j \in \bar{S}$ with $x_{ij} = 1$. This implies

$$\sum_{i \in S} \sum_{j \in \bar{S}} x_{ij} \geq 2 \left\lceil \frac{a(S) + \sum_{\substack{i \in S \\ j \in \bar{S}}} a_j x_{ij}}{b} \right\rceil \tag{11}$$

Constraint (11) is not directly useful, since the right-hand side is a nonlinear function of $x$, but it can be used to derive other constraints. An example is given below.

For any $S \subseteq N$, let $S' = \{j \; \varepsilon \; \bar{S} | j \; \geq \; 1 \text{ and } a_j > b\, r(S) - a(S)\}$

$$e_j = \begin{cases} 0, & j \in S \\ 0, & j \in S' \text{ and } |S'| \leq 2 \\ \dfrac{r(S)}{r(S)+1}, & j \in S' \text{ and } |S'| > 2 \\ 1, & j \in \bar{S} - S' \end{cases}$$

We can now define the tightened vehicle capacity constraints

$$\sum_{j=0}^{n} e_j \sum_{i \in S} x_{ij} \geq 2r(S) \quad \text{for all } S \subseteq N \text{ with } |S| \geq 2 \tag{9'}$$

Computational results using the $K$-tree approach are presented in Table 4. The first 6 problems are the ones introduced in Table 1, Section 2 of this chapter except that for problem 5, $K = \lceil \sum_{i=1}^{n} a_i/b \rceil = 16$ here, rather than the value of 17 used in most previous studies. The data in problems 7–12 are taken from real vehicle routing applications. Problems 7–9 are concerned with the delivery of industrial gases in cylinders and are based on data provided by Air Products and Chemicals, Inc. Problems 10 and 12 represent a day of grocery deliveries from the Peterboro and Bramalea, Ontario terminals, respectively, of National Grocers Limited [see Arizza & Karellas, 1983]. Problem 11 is concerned with delivery of tires, batteries and accessories to gasoline service stations and is based on data provided by Exxon.

All problems except problems 7–9 were planar and $c_{ij}$ is the distance between points $i$ and $j$, computed as a single precision real value. In problem 7–9, integral values for all $c_{ij}$ were defined as part of the input.

The condition $a_j/b < \sum_{i=1}^{n} a_i/b - (K - 1)$, $j = 1, \cdots, n$ was satisfied for 8 out of the 12 test problems used (problems 1, 2, 3, 5, 7, 8, 11, 12) so the prohibition of single customer routes was not constraining since these problems have no feasible solution with single customer route. For two other problems (problems 9 and 10), a few customers violated this condition, but the resulting single route solutions were easily shown to be nonoptimal using the lower bounding procedure presented here and a negligible amount of computation time. While it has not been formally established that the optimal solutions to problems 4 and 6 do not contain single customer routes, it seems unlikely since deleting the largest customer and a single vehicle would leave a very tightly constrained problem.

These computations suggest that real problems are easier for optimization than randomly generated problems. For example, compare real problems 10 and 11 together with the 'realistic' problem 6 with the random problems 1–3 of similar size. For problems 6, 10 and 11, the lower bounds computed with no more than 2000 subgradient iterations were 99.1% of the optimum, on average, while for problems 1–3, the lower bounds were 94% of the optimum, on average and required 3,000 subgradient iterations.

The difference between uniform random and real problems is illustrated in Figures 1 and 2, which show depot and customer locations for problems 2 and 11, respectively. The rather even distribution of customers in Figure 1 is typical of problems 1–5, just as the grouping of customers into clusters is typical of problems 6–12. This clustering seems to play a role akin to sparsity in linear programming in providing a structure that can be exploited in computations. For example, a group of customers clustered together tends to act as one customer for vehicle capacity constraints (9); they are either all in a set $S$ defining a constraint or none of them is.

Table 4
Computational results with the $K$-tree algorithm

| Problem | $n$ | $K$ | $\frac{\sum_{i=1} a_i}{Kb}$ | Upper bounds* | | Lower bound | $\frac{\text{Lower}}{\text{Upper}}$ | Subgradient iterations | Computation time Apollo domain 3000 minutes |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Best previously known feasible solution | Optimal or improved solution found by the $K$-tree algorithm | | | | |
| 1 | 50 | 5 | 0.97 | 524.61[1] | – | 507.09 | 0.97 | 3000 | 95.75 |
| 2 | 75 | 10 | 0.97 | 635.26[1] | – | 755.50 | 0.90 | 3000 | 183.97 |
| 3 | 100 | 8 | 0.91 | 826.14[1] | – | 785.86 | 0.95 | 3000 | 307.95 |
| 4 | 150 | 12 | 0.93 | 1028.42[1] | – | 932.68 | 0.91 | 3000 | 682.41 |
| 5 | 199 | 16 | 0.999 | 1334.55[2] | – | 1096.72 | 0.82 | 3000 | 1186.00 |
| 6 | 100 | 10 | 0.91 | 819.56[1] | 819.56[4] | 817.77 | 0.998 | 2000 | 259.64 |
| 7 | 25 | 3 | 0.92 | 3104[3] | 3070[4] | 3070 | 1.00 | 769 | 11.23 |
| 8 | 29 | 4 | 0.94 | 5830[3] | 5829[4] | 5829 | 1.00 | 2444 | 53.33 |
| 9 | 36 | 4 | 0.76 | 5032[3] | 4961[4] | 4961 | 1.00 | 320 | 4.87 |
| 10 | 44 | 4 | 0.90 | 723.54[3] | 723.54[4] | 720.76 | 0.996 | 2000 | 49.74 |
| 11 | 71 | 4 | 0.96 | 244.92[3] | 241.97[4] | 237.76 | 0.98 | 2000 | 105.03 |
| 12 | 134 | 7 | 0.95 | 1216.66[3] | 1163.60[4] | 1133.73 | 0.97 | 2000 | 253.84 |

* Sources for upper bounds: [1] Taillard [1992].
[2] Osman [1993].
[3] Algorithm in Bramel & Simch-Levi [1992]. These problems were run and the results communicated to me by Julien Bramel and David Simchi-Levi.
[4] First Lagrangian heuristic described in Fisher [1990].

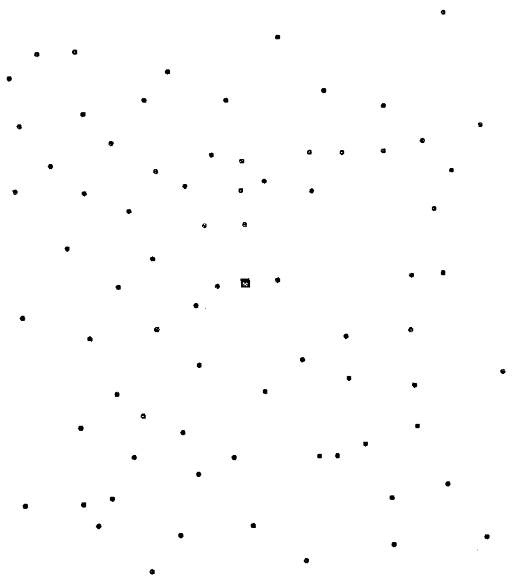

Fig. 1. Customer and depot locations for problem 2 (square is depot and dots are customers).

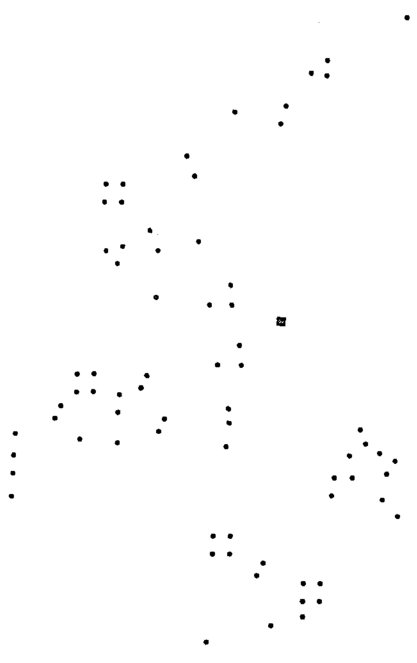

Fig. 2. Customer and depot locations for problem 11 (square is depot and dots are customers).

The $K$-tree algorithm found proven optimal solutions to problems 7–9 without branching and to problems 6, 10 and 11 using a branch and bound algorithm based on two approaches to branching. The first is a traditional approach in which nodes of the search tree correspond to partially-sequenced sets of customers [e.g., see Christofides, Mingozzi & Toth, 1981b]. We can branch on either edges or customers. An edge branch is executed by selecting an edge $(i, j)$ and creating two branches, one with edge $(i, j)$ forced into the solution and one with it forced out.

Customer branching is executed from a node of the branch and bound tree at which a sequence has been established for a set of customers $i_1, \ldots, i_k$ that comprises a portion of a vehicle route. We chose either end of the sequence and branch by enumerating various customers that could be appended to the partial route. Assume that we are branching from customer $i_k$. We execute a customer branch step by identifying a set $T \subseteq N_0 - \{i_1, \ldots, i_k\}$ of unbranched customers satisfying $a\{i_1, \ldots, i_k, j\} \leq b$ for all $j \in T$. We create a node corresponding to the sequence $i_1, \ldots, i_k, j$ for all $j \in T$ and an additional node at which customer $i_k$ cannot link to any customer $j \in T$, i.e., the edge $(i_k, j)$ is excluded for all $j \in T$. Normally, $T$ would be selected so that points in $T$ are close to $i_k$.

A route is completed when the depot has been appended to both ends in the sequence. We do not allow a route to be completed if the unused capacity on the vehicle is so great that the remaining vehicles have insufficient capacity to deliver the remaining customer orders.

This procedure begins with an edge branch on an edge $(i_1, i_2)$. At the node of the search tree where $(i_1, i_2)$ is forced into the solution, a customer branch using the partial sequence $i_1, i_2$ is executed. At the other node, another edge branch is executed. In general, at any node of the tree defining a sequenced subset of customers corresponding to a partial vehicle route, customer branching is used. Otherwise, edge branching is used.

This procedure was applied to several of the test problems, but was unsuccessful in finding a proven optimal solution for any of them. The major problem with this approach is that the decisions resolved when we branch are quite minor. To illustrate the difficulty this can create, consider a problem with a cluster of $k$ customers close to each other. Any solution in which these customers are delivered contiguously on the same route in some sequence will have about the same cost. Hence, when we branch so as to resolve the sequence for these customers, unless the lower bound is exceptionally tight, we will be unable to fathom any of the $0(k!)$ nodes generated. Looking at Figure 2, one can see many clusters of 4 to 5 customers where this problem could and did arise.

As suggested in Christofides, Mingozzi & Toth [1981a], a dominance fathoming test can be formulated that mitigates this problem to some extent. A node of the branch and bound tree corresponding to a sequence $i_1, \ldots, i_k$ for a set of customers cannot lead to an optimal solution, and therefore can be fathomed, if there is a different sequence for the customers that begins with customer $i_1$, ends with customer $i_k$, and has lower cost. This test can be operationalized by applying the Lin & Kernighan [1973] 3-opt rule to the customer sequence specified at a

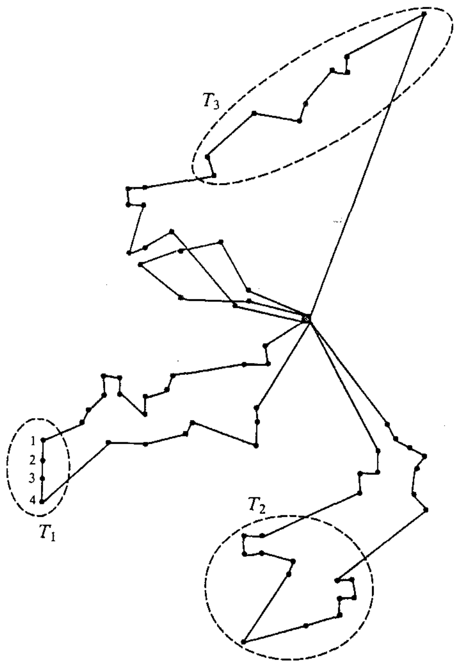


Fig. 3. Examples of branch clusters for problem 11.

node. If the 3-opt rule finds an improved sequence, then the node is fathomed. This dominance test improved performance somewhat, but not enough to allow optimal solution of any of the test cases.

This experience suggests that one might obtain a better branching procedure by identifying macro properties of an optimal solution whose violation would have a sufficiently large impact on cost to allow fathoming. Figure 3 shows the optimal solution for problem 11. One property that stands out in this optimal solution is the presence of clusters of customers delivered contiguously on the same route which are close to each other and far from the remaining customers in the problem. Three examples are encircled in Figure 3 and labelled as $T_1$, $T_2$, and $T_3$. Requiring the customers in any of these three clusters to be delivered on two different routes would appear to have a significant impact on cost.

This observation was used to develop a new branching rule. Let $I(T)$ denote the incidence of edges on the node set $T$ in a solution graph, i.e., $I(T) = \sum_{i \in T} \sum_{j \in \bar{T}} x_{ij}$. In the solution depicted in Figure 3, $I(T_k) = 2, k = 1, 2, 3$. Note that in a feasible solution the incidence on any customer set must be an even integer not less than $2\lceil (a(T))/b \rceil$. Branching occurs by selecting any set $T \subseteq N$ and creating two nodes corresponding to $I(T) = 2\lceil (a(T))/b \rceil$ and $I(T) \geq 2\lceil (a(T))/b \rceil + 2$. The constraint $I(T) \geq 4\lceil (a(T))/b \rceil$ is of the same form as constraints (9′) and hence easy to incorporate within the Lagrangian relaxation. Branching in this way for the sets $T_1, T_2$ and $T_3$ produces a lower bound at the node corresponding to $I(T_k) \geq 4$ greater than the feasible value of 244.92, the best known feasible value prior to applying branch and bound, thus establishing that $I(T_k) = 2, k = 1, 2, 3$ in an optimal solution to problem 11.

The constraint $I(T) = 2\lceil(a(T))/b\rceil$ can be used to derive additional restrictions that tighten the Lagrangian problem. First of all, for any $j \notin T$ such that $a_j + a(T) > b$, we can force out of the solution the edges $(i, j)$ for all $i \in T$. Second, if there is another set $S$ for which $I(S) = 2$, $S \cap T = \phi$ and $a(S \cup T) > b$, we can force out of the solution the edges $(i, j)$ for all $i \in T, j \in S$. Third, if $\lceil(a(T))/b\rceil = 1$ and $b - a(T)$ is sufficiently small, it may be feasible to enumerate all combinations of customers that can fit in the remaining space $b - a(T)$ and branch by generating a node corresponding to each combination. Fourth, if $\lceil(a(T))/b\rceil = 2$, we can select a subset $S \subset T$ of a few large customers for which it is computationally feasible to enumerate all partitions of $S$ into two sets corresponding to the customers assigned to each of the 2 vehicles that must deliver the customers in $T$. We can then branch on the choice of a partition.

Finally, we describe a simple dominance test that can sometimes establish an optimal sequence for the customers in a set $T$ with $I(T) = 2$. By way of illustration, consider set $T_1$ in Figure 3. The customers in this set are indexed from 1 to 4. Because $I(T_1) = 2$, there will be precisely two customers in $T_1$ joined to customers outside of $T_1$. There are $(\frac{4}{2}) = 6$ choices for this pair of customers. Index these pairs $(i_k, j_k), k = 1, \ldots, 6$, and assume $(i_1, j_1) = (1, 4)$. For any pair $(i_k, j_k)$, the path through the remaining two customers must minimize cost and can be determined by enumeration. For example, it is apparent that for $(i_1, j_1)$, the optimal path is $(1, 2, 3, 4)$. Let $C_k$ denote the cost of the optimal path from $i_k$ to $j_k$, e.g., $C_1 = c_{12} + c_{23} + c_{34}$.

We call a pair $(i_k, j_k)$ *dominated* if, for each pair $i, j,\ i \neq j,\ i \in N_0 - T,\ j \in N_0 - T$, there exists $k^* \neq k$ such that

$$C_{k^*} + \min(c_{ii_{k^*}} + c_{jj_{k^*}},\ c_{ji_{k^*}} + c_{ij_{k^*}}) \leq C_k + \min(c_{ii_k} + c_{jj_k},\ c_{ji_k} + c_{ij_k}) \tag{12}$$

It is clear that a path through $T$ joining a dominated pair can be ignored as a sequence for the customers in $T$, since it could be replaced in any feasible solution by a different sequence (namely, the shortest path through $T$ joining $i_{k^*}$ and $j_{k^*}$) without increasing cost. Returning to our example, for the set $T_1$, all pairs $k = 2, \ldots, 6$ can be shown by direct computation of (13) to be dominated by $k^* = 1$. Hence, we can fix the sequence of customers in $T_1$ to 1, 2, 3, 4. For the set $T_3$, there are four sequences that dominate all others. In this event, we can branch by selecting one of the sequences.

In computational work, the dominance test was applied to each $k \in T$ by computing (7) for all possible $ij$ and each possible $k^*$. The step of finding the shortest path through $T$ joining a pair of customers in $T$ can be accomplished by a straightforward modification of the dynamic programming algorithm for the traveling salesman problem given by Held & Karp [1962].

The ideas defined above can be combined in many ways to create a branch and bound algorithm, depending on how branch sets are chosen and the order in which the various methods of branching are combined. We describe here the particular algorithm used in Fisher [1990].

We need to identify sets of customers on which to branch. Two types of branch sets were used. One was the set of customers $S$ on a single route or a pair of crossed routes for which average vehicle utilization exceeded 98%. If $S$ was a pair of crossed routes, add to $S$ any customers within the convex hull of $S \cup \{0\}$ that fit (i.e. $a(S) \leq 2b$ with the customers added), starting with customers farthest from the depot. Sets like these make good branch sets because the branch $I(S) = 2\lceil(a(S))/b\rceil$ removes many edges from the problem, namely those edges $(i, j)$ for which $i \in S$, $j \in \bar{S}$ and $a(S) + a_j > \lceil(a(S))/b\rceil b$.

The second type of branch sets were clusters of customers $S = \{i_1, \cdots, i_k\}$ delivered contiguously in the order $i_1, \cdots, i_k$ on a single route of a starting feasible solution. $S$ was also required to contain the customer on the route farthest from the depot, to be separable from $\bar{S}$ by a straight line and to have a sufficiently small value of

$$D(S) = \frac{\sum_{j=1}^{k-1} c_{i_j i_{j+1}}}{(k-1) \min_{i \in S, j \in \bar{S}} c_{ij}}.$$

The quantity $D(S)$ is used measure the extent to which the customers in $S$ are close to each other and far from the remaining points $\bar{S}$. It's easy to see that the sets $T_1$, $T_2$ and $T_3$ in Figure 3 satisfy the required properties and have relatively small values of $D(S)$.

We describe the specific steps in the branch and bound procedure in the order they are executed. First branch on any route $S$ with $a(S) = b$ and then on $\lfloor n/10 \rfloor$ of the second type of branch sets described above with the smallest values of $D(S)$ for which we can fathom the node corresponding to $I(S) \geq 4$.

Next, the dominance test described previously is applied to any set $S$ with $I(S) = 2$ and $|S| \leq 11$ (the computation time for the dominance test is prohibitive for larger sets). We branch on choice of sequence whenever at most 4 sequences are nondominated. For a set $S$ with $I(S) = 2$ and $b - a(S) > 0$ but sufficiently small that at most one other customer could fit feasibly with $S$ on a single route, we branch by enumerating all feasible completions of the vehicle route containing $S$.

We then branch on all single routes or pairs of crossed routes with average vehicle utilization exceeding 98%. If $S$ is a pair of routes, at the node corresponding to $I(S) = 4$, we branch on all partitions of $R \subseteq S$ into 2 sets corresponding to the set of customers assigned to each of the 2 routes, where $R$ contains the 11 largest customers in $S$. A set $T \subset S$ for which $I(T) = 2$ has been imposed can be treated as a single customer for this purpose. Finally, at any node still unfathomed, the traditional branching method described earlier is applied.

This algorithm has solved to optimality problems 6–11. Results are reported in Table 5. Because of round off error, the problems with real $c_{ij}$ were solved to

Table 5
Optimization results for the $K$-tree algorithm

| Problem | Time to find optimal solution* Apollo domain 3000 minutes | Nodes in branch and bound tree |
|---|---|---|
| 6 | 591 | 89 |
| 7 | 11 | 1 |
| 8 | 64 | 1 |
| 9 | 6 | 1 |
| 10 | 342 | 148 |
| 11 | 948 | 37 |

*Time includes the time (reported in the last column of Table 4) to be bound to the root node.

$\varepsilon$ optimality (the solution obtained may exceed the true optimum by an additive constant $\varepsilon$) with $\varepsilon = 0.0001$.

Optimization now appears on the verge of becoming a practically useful tool. In the last 5 years, the size problems solvable to optimality has been extended to 50–100 customers. Moreover, the methods developed can be or have been extended to incorporate real world complications like time windows.

## References

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

Altinkemer, K., and B. Gavish (1987). Heuristics for unequal weight delivery problems with a fixed error guarantee. *Oper. Res. Lett.* 6, 149–158.

Altinkemer, K., and B. Gavish (1991). Parallel savings based heuristics for the delivery problem. *Oper. Res.* 39, 456–469.

Araque, J.R., G. Kudva, T.L. Morin and J.F. Pekny (1994). A branch-and-cut algorithm for vehicle routing problems. *Ann. Oper. Res.*,

Arizza, A., and P. Karellas (1983). A feasibility study of the ROVER system at National Grocers Co. Ltd., B.A. Thesis of Applied Science, Department of Industrial Engineering, University of Toronto.

Bartholdi, J.J., L.K. Platzman, R.L. Collins and W.H. Warden, (1983) A minimal technology routing system for meals on wheels. *Interfaces* 13, 1–8.

Balinski, M.L., and R.E. Quandt (1964). On an integer program for a delivery program for a delivery problem. *Oper. Res.* 12, 300–304.

Belardo, S., P. Duchessi and J. Seagle (1985). Microcomputer graphics in support of vehicle fleet routing. *Interfaces* 15, 84–92.

Bell, W.J., L. Dalberto, M.L. Fisher, A.J. Greenfield, R. Jaikumar, P. Kedia, R.G. Mack and R.J. Prutzman (1983). Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer, *Interfaces* 13, 4–23.

Bodin, L.D. (1991). Twenty years of routing and scheduling, *Oper. Res.* 38, 571–579.

Bodin, L.D., B.L. Golden, A. Assad and M. Ball (1983). Routing and scheduling of vehicles and crews: the state of the art. *Comput. Oper Res.* 10, 69–211.

Bramel, J., and D. Simchi-Levi (1993). On the effectiveness of set partitioning formulations for the vehicle routing problem, Working Paper, Department of Industrial Engineering and Oper. Res., Columbia University.

Bramel, J., and D. Simchi-Levi (1992). A location based heuristic for general routing problems, Working Paper, Department of Industrial Engineering and Oper. Res., Columbia University.

Brenninger-Goethe, M. (1989). Two vehicle routing problems — Mathematical programming approaches, Linkoping Studies in Science and Technology. Dissertations 200. Department of Mathematics, Linkoping University, Linkoping.

Brown, G., and G. Graves (1981). Real-time dispatch of petroleum tank trucks, *Manage. Sci.* 27, 19–31.

Campos, V., A. Corberan and E. Mota (1991). Polyhedral results for a vehicle routing problem. *Eur. J. Oper. Res.* 52, 75–85.

Christofides, N. (1985). Vehicle routing, in: E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys (eds.), *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, Chichester.

Christofides, N., and S. Eilon (1969). An algorithm for the vehicle dispatching problems, *Oper. Res. Q.* 20, 309–318.

Christofides, N., A. Mingozzi and P. Toth (1979). The vehicle routing problem, in: N. Christofides, A. Mingozzi, P. Toth and C. Sandi (eds.). *Combinatorial Optimization*, Wiley, Chichester, pp. 318–338.

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

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

Chvatal, V. (1973). Edmonds polytopes and weakly Hamiltonian graphs. *Math. Program.* 5, 29–40.

Clarke, G., and J. Wright (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Oper. Res.* 12, 568–581.

Collins, D.H.G., and W.M. Clavey (1986). The optimization of Route scheduling: a case study. *Proc. Int. Trans. Conf.*, Institute of Industrial Engineers.

Cornuejols, G., and F. Harche (1989). Polyhedral study of the capacitated vehicle routing problem, Management Science Research Report No.553, Carnegie Mellon University, Pillsburgh, PA.

Cullen, F., J. Jarvis and D. Ratliff (1981). Set partitioning based heuristics for interactive routing. *Networks* 11, 125–144.

Dantzig, G.B., and J.H. Ramser (1959). The truck dispatching problem, *Manage. Sci.* 6, 80–91.

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

Desrochers, M., J.K. Lenstra and M.W.P. Savelsbergh (1990). A classification scheme for vehicle routing and scheduling problems. *J. Oper. Res.* 46, 322–332.

Evans, S., and J. Norback (1985). The impact of a decision-support system for vehicle routing in a food service supply situation. *J. Oper. Res. Soc.* 36, 467–472.

Fisher, M.L. (1991). A polynomial algorithm for the degree constrained $K$-tree problem, Working Paper 91-07-06, Department of Decision Sciences, University of Pennsylvania, to appaear.

Fisher, M. L. (1990). Optimal solution of vehicle routing problems using minimum $K$-trees, Working Paper, Department of Decision Sciences, University of Pennsylvania, to appear.

Fisher, M.L. (1982). Greenfield, A.J., Jaikumar, R. and Lester, J.T., III, A computerized vehicle routing application. *Interfaces* 12, 42–52.

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

Fisher, M.L. (1986). Jaikumar, R. and Van Wassenhove, L., A multiplier adjustment method for the generalized assignment problem. *Manage. Sci.* 32, 1095–1103.

Fisher, M.L., and P. Kedia (1990). Optimal solution of set covering/partitioning problems using dual heuristics. *Manage. Sci.* 36, 674–688.

Fisher, M.L., and M.B. Rosenwein (1989). An interactive optimization system for bulk cargo ship scheduling. *Nav. Res. Logist.* 36, 27–42.

Foster, B.A., and D.M. Ryan (1976). An integer programming approach to the vehcle scheduling problem. *Oper. Res.* 27, 367–384.

Fleuren, H. (1988). A computational study of the set partitioning approach for vehicle routing and scheduling problems, Ph.D. Dissertation, University of Twente, Enschede.

Gaskell, T.J. (1967). Basis for vehicle fleet scheduling, *Oper. Res. Q.* 18, pp. 281.

Gillett, B. and Miller, L. (1974). A heuristic algorithm for the vehicle dispatch problem. *Oper. Res.* 22, 240–349.

Gendreau, M., A. Hertz and G. Laporte (1993). A tabu search heuristic for the vehicle routing problem, Centre de Recherche sur les Transports, Publication 777, Université de Montreal.

Golden, B.L., and A.A. Assad (1986). Perspectives on vehicle routing: exciting new developments. *Oper. Res.* 14, 803–810.

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

Golden, B., and E. Wasil (1987). Computerized vehicle routing in the soft drink industry. *Oper. Res.* 35, 6–17.

Grotschel, M. (1980). On the symmetric traveling salesman problem: solution of a 120-city problem. *Math. Program. Studies* 12, 61–77.

Grotschel, M., and M.W. Padberg (1979). On the symmetric traveling salesman problem: I–II. *Math. Program.* 16, 265–280 and 281–302.

Grotschel, M., and W. Pulleyblank (1986). Clique tree inequalities and the traveling salesman problem. *Math. Oper. Res.* 11, 537–569.

Harker, P.T. (1990). The use of advanced train control system in scheduling and operating railroads: models, algorithms and applications, in: *Transportation Research Record* 1263, TRB, National Research Council, Washington, D.C., pp. 101–110.

Haimovich, M., and A.H.G. Rinnooy Kan (1985). Bounds and heuristics for capacitated routing problems. *Math. Oper. Res.* 10, 527–542.

Held, M., and R.M. Karp (1962). A dynamic programming approach to sequencing problems. *J. Soc. Industr. Appl. Math.* 10, 196–210.

Kadaba, N., K.E. Nygard and P.L. Juell (1991). Integration of adaptive machine learning and knowledge-based systems for routing and scheduling applications. *Expert Systems Appl.* 2, 15–27.

Kolen, A.W.J., A.H.J. Rinnooy Kan and H.W.J.M. Trienekens, (1987). Vehicle routing with time windows. *Oper. Res.* 35, 266–273.

Koskosidis, Y.A., W.B. Powell and M.M. Solomon (1989). An optimization based heuristic for vehicle routing and scheduling with time window constraints, No. TMS-88-09-1, The Institute for Transportation Systems, The City College of CUNY, New York, NY.

Laporte, G., and J.-M. Bourjolly (1984). Some further results on $k$-star constraints and comb inequalities. *Cahiers du GERAD, G-84, G-84-17, Ecole des Hautes Etudes Commerciales de Montreal.*

Laporte, G., and Y. Nobert (1987). Exact algorithms for the vehicle routing problem. *Ann. Discr. Math.* 31, 147–184.

Laporte, G., Y. Nobert and M. Desrochers (1985). Optimal routing under capacity and distance restrictions. *Oper. Res.* 33, 1050–1073.

Lavoi, S., M. Minoux and E. Odier (1988). A new approach for crew pairing problems by column generation with an application to air transportation. *Eur. J. Oper. Res.* 35, 45–58.

Lawler, E.L., J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys, eds. (1985). *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, Wiley, Chichester.

Lin, S. (1965). Computer solution of the traveling salesman problem. *Bell Systems Tech. J.* 44, 2245–2269.

Lin, S., and B. Kernighan (1973). An effective heuristic algorithm for the traveling salesman problem. *Oper. Res.* 21, 498–516.

Lucena, A.P. (1986). Exact solution approaches for the vehicle routing problem, Ph.D. Thesis, Department of Management Science, Imperial College of Science and Technology, University of London.

Magnanti, T. (1981). Combinatorial optimization and vehicle fleet planning: perspectives and prospects. *Networks* 11, 179–214.

Miliotis, P. (1976). Integer programming approaches to the traveling salesman problem. *Math. Program.* 10, 367–378.

Miller, D. (1993). A matching based exact algorithm for capacitated vehicle routing paper, Working Paper, Central Research & Development Department, E.I. du Pont de Nemours and Company, Wilmington, DE.

Miller, D.L., and J.F. Pekny (1994). A staged primal–dual algorithm for perfect $b$-matching with edge capacities. *ORSA J. Comp.*.

National Council of Physical Distribution Management Study, (1978). Measuring productivity in physical distribution, prepared by A.T. Kearney, Inc., Chicago, IL.

Nygard, K.E., P. Greenberg, W.E. Bolkan and E.J. Swenson, (1988). Generalized assignment methods for the deadline vehicle routing problem, in: B.L. Golden and A.A. Assad (eds.), *Vehicle Routing: Methods and Studies*, Elsevier Science Publishers B.V., North-Holland.

Nygard, K.E., and P.L. Juell (1991). Integration of adaptive machine learning and knowledge-based systems for routing and scheduling applications. *Expert Systems Appl.* 2, 15–27.

Osman, I.H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Ann. Oper. Res.* 41, 421–451.

Padberg, M.W. and Hong, S. (1980). On the symmetric traveling salesman problem: a computational study. *Math. Program. Studies* 12, 78–107.

Potvin, J.-Y., G. Lapalme and J.-M. Rousseau (1989). ALTO: A computer system for the design of vehicle routing algorithms. *Comput. Oper. Res.* 16, 451–470.

Powell, W.B., Y. Sheffi, K.S. Nickerson, K. Butterbaugh and S. Atherton (1988) Maximizing profits for North American Van Lines' Truckload Division: a new framework for pricing and operations. *Interfaces* 18, 21–41.

Russell, R.A. (1977). An effective heuristic for the $M$-tour traveling salesman problem with some side conditions. *Oper. Res.* 25, 5217–524.

Savelsberg, M. (1985). Local search in routing problems with time windows, *Oper. Res.* 4, 285–305.

Taillard, E. (1993). Parallel iterative search methods for vehicle routing problems, *Networks* 23,

Thompson, P.M. (1988). Local search algorithms for vehicle routing and other combinatorial problems. Submitted in partial fulfillment of the requirements of the degree of Doctor of Philosophy in Operations Reserach at the Massachusetts Institute of Technology.

Toth, P. (1984). Heuristic algorithms for the vehicle routing problems, presented at the Workshop on Routing Problems, Hamburg,

Tyagi, M. (1968). A practical method for the truck dispatching problem, *J. Oper. Res. Soc. Jpn.* 10, 76–92.

Viegas, J., and P. Hansen (1985). Finding shortest paths in the plane in the presence of barriers to travel (for any $1_p$-norm). *Eur. J. Oper. Res.* 20, 373–381.

Walter, D.F., and E. Zielinski (1983). Computerized vehicle routing package–management issues. Talk delivered at the 1983 Annual Meeting of the National Council of Physical Distribution Management, New Orleans.

Zielinski, E.L. (1985). Implementing a computerized vehicle routing system: A case study. *Auerbach Rep. 1*, 24, 1–14.

Yellow, P. (1970). A computational modification to the savings method of vehicle scheduling. *Oper. Res. Q.* 21, 281–283.