Comput. & Ops Res. Vol. 11, No. 1, pp. 49–66, 1984
Printed in Great Britain.

0305-0548/84 $3.00 + .00
Pergamon Press Ltd.

THE FLEET SIZE AND MIX VEHICLE ROUTING PROBLEM

BRUCE GOLDEN†, ARJANG ASSAD‡, LARRY LEVY§ and FILIP GHEYSENS¶
University of Maryland at College Park, College Park, MD 20742, U.S.A.

Scope and Purpose—The effective routing of vehicles has saved and will continue to save government and industry many millions of dollars a year. Sophisticated multi-vehicle routing procedures have increased productivity, improved operations, aided in long-range planning, assisted in contract negotiations, and helped to control the financial impact of adverse weather conditions on vehicle utilization. In this paper, we consider the problem of determining an optimal fleet size and mix with respect to both acquisition and routing costs. Several efficient computerized procedures are proposed and evaluated.

Abstract—In this paper, we address the problem of routing a fleet of vehicles from a central depot to customers with known demand. Routes originate and terminate at the central depot and obey vehicle capacity restrictions. Typically, researchers assume that all vehicles are identical. In this work, we relax the homogeneous fleet assumption. The objective is to determine optimal fleet size and mix by minimizing a total cost function which includes fixed cost and variable cost components. We describe several efficient heuristic solution procedures as well as techniques for generating a lower bound and an underestimate of the optimal solution. Finally, we present some encouraging computational results and suggestions for further study.

INTRODUCTION

In many organizations, the management of distribution activities constitutes a major decision-making problem. This problem has assumed increased urgency due to the significant contribution of distribution cost to total costs, especially in view of the recent dramatic escalation of fuel costs.

Most firms require vehicles for pick-up and/or delivery purposes to service a network of supply or demand locations. The efficient utilization and routing of this fleet of vehicles lies at the heart of almost all distribution-routing problems. In particular, a natural question facing a distribution manager is: How many and what size vehicles are needed in order to accommodate demand at minimal cost? This question, which is difficult to answer due to the enormous number of potential combinations of fleet mix and routing patterns, gives rise to the problem we formulate and address in this paper.

Variants of the vehicle routing problem generally share the following characteristics. A set of routes must be designed for the vehicles, originating from and terminating at a central depot. Each route visits a set of customer locations with known demands, the sum of which must not exceed the capacity of the vehicle assigned to that route. The routing costs associated with the

†Bruce L. Golden is Chairman of the Department of Management Science and Statistics at the University of Maryland. His research interests include network optimization, mathematical programming, and applied statistics, and he has published numerous articles in these fields. He is currently an Associate Editor of *Networks*, a member of ORSA's Long Range Planning Committee, and Chairman of ORSA's Transportation Science Dissertation Prize Committee. Dr. Golden received his B.A. from the University of Pennsylvania in Mathematics. Later he earned his M.S. and Ph.D. in Operations Research from M.I.T.

‡Arjang A. Assad is an Assistant Professor in the Department of Management Science and Statistics at the University of Maryland. His interests include operations management, mathematical programming, and network modeling of transportation/distribution systems and he has conducted applied research projects with a number of firms, and the U.S. Departments of Transportation and Energy. He received his Ph.D. in Management Science from M.I.T.

§Larry J. Levy is a telecommunications analyst with Contel Information Systems (formerly Network Analysis Corporation) in Vienna, Virginia. His work includes the design and analysis of computer and communications configurations. Mr. Levy received his Masters degree in Applied Mathematics at the University of Maryland and his Bachelors degree in Mathematics at Stevens Institute of Technology. His research at the University of Maryland included the study of the traveling salesman and vehicle routing problems.

¶Filip Gheysens is pursuing a doctoral degree in management science at the University of Gent in Belgium. In 1982, Filip won a C.I.M. Fellowship from Belgium to spend the year as a visiting graduate student at the University of Maryland. Filip's major research interests include network location and vehicle routing. He received his Masters degree in Engineering from the University of Gent.

vehicles form one component of total distribution cost. The other essential component consists of fleet acquisition and maintenance costs. It is possible to distinguish among three vehicle routing cases that frequently arise in practice:

(1) *The vehicle routing problem (VRP)*. This case involves a predetermined number of vehicles, all of the *same capacity*. This number may be viewed as an upper bound on the size of the vehicle fleet that will actually be utilized. A key assumption is that the fixed (or acquisition) costs have already been incurred and that only variable (or routing) costs need to be considered explicitly. The objective is to minimize total routing cost which is a function of the total distance traveled by the fleet of vehicles. This problem has received extensive research attention and arises in newspaper delivery, school bus routing, municipal waste collection, fuel oil delivery, and truck dispatching in a variety of industries (see Bodin and Golden[3]).

(2) *The fleet size problem*. The routing decision is frequently preceded and dominated by the question of fleet size, that is, the number of vehicles to purchase or lease in order to satisfy demand. In this case, fixed vehicle costs and variable routing costs both need to be considered. The vehicles in the fleet are assumed to have identical operating characteristics (e.g. cost and capacity). The goal is to determine an economical fleet size. In some cases, complicated tradeoffs between purchasing and leasing need to be addressed. Ball *et al.*[1] have recently conducted preliminary research in this direction. Their application involves a large pharmaceutical company subject to union regulations that impose maximum route-time restrictions on the vehicles. The solution methodology developed is of wide applicability and can be used in the context of barges, railcars, and aircraft as well as trucks.

(3) *The fleet size and mix problem*. This case is a generalization of the previous one in that the assumption that all vehicles have the same cost and capacity is relaxed. Thus, in addition to fleet size, the mix of vehicles in the fleet must be optimized.

The goal of this paper is to tackle the *third* problem listed above. Without loss of generality, we consider a situation in which the distributor decides, for the purposes of convenience and reliability, to *lease*, rather than *acquire*, vehicles for a certain period of time. The number of vehicles of each type needed to handle the system demand effectively must be determined together with a coherent routing policy for such vehicles. We assume that each vehicle type has a fixed leasing cost and a variable routing cost that is proportional to the distance traveled. The variable cost component encompasses the costs of fuel, maintenance, and manpower. While the mathematical formulation of this problem is readily available (see the next section), no efficient technique for obtaining near-optimal solutions to this problem is currently known. Computer simulation is sometimes used to evaluate various specific options, but more analytic techniques still need to be explored. We have begun moving along these lines and in this paper we propose several analytic solution procedures that integrate the fleet composition and routing sub-problems in a reasonable manner.

FORMULATION

In order to clarify and make more precise the problem that we address in this paper, we present a mathematical programming formulation below. The problem as stated is a delivery problem. In the case where pick-ups are made instead, an equivalent problem results.

Minimize

$$\sum_{k=1}^{T} f_k \sum_{j=1}^{n} x_{0j}^k + \sum_{k=1}^{T} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}^k$$

subject to

$$\sum_{k=1}^{T} \sum_{i=0}^{n} x_{ij}^k = 1 \qquad (j = 1, \ldots, n)$$

$$\sum_{i=0}^{n} x_{ip}^k - \sum_{j=0}^{n} x_{pj}^k = 0 \qquad (k = 1, \ldots, T;\ p = 1, \ldots, n)$$

$$r_0 = 0$$

$$r_j - r_i \geq (d_j + a_T) \sum_{k=1}^{T} x_{ij}^k - a_T \qquad (i = 0, \ldots, n; \quad j = 1, \ldots, n) \qquad (*)$$

$$r_j \leq \sum_{k=1}^{T} \sum_{i=0}^{n} a_k x_{ij}^k \qquad (**)$$

$$x_{ij}^k \in \{0, 1\} \text{ for all } i, j, k \qquad (j = 1, \ldots, n)$$

where n = number of customers, T = number of vehicle types, a_k = capacity of vehicle type k $(a_1 < a_2 < \ldots < a_T)$, f_k = fixed acquisition cost of vehicle type k $(f_1 < f_2 < \ldots < f_T)$, d_j = demand of customer j, c_{ij} = cost of travel from customer i to customer j (we assume symmetry and that cost is independent of the vehicle type used), r_i = commodity flow variable associated with customer i, $x_{ij}^k = 1$ if vehicle type k travels from customer i to customer j and $= 0$ otherwise.

It is assumed that an infinite supply of each vehicle type is available. The central depot is denoted by 0 and the term $\sum_{j=1}^{n} x_{0j}^k$ represents the number of vehicles of type k used. Therefore, the first double-sum in the objective function gives the total fixed or acquisition cost; the next triple-sum gives the total variable or routing cost.

The first two sets of constraints ensure that each customer is visited exactly once and that a vehicle arriving at a customer location also leaves that location.

The next three sets of constraints guarantee that vehicle capacities are not exceeded. The variable r_i gives the total demand that a vehicle has serviced on its route after it reaches customer i (the demand of customer i is included). Thus, (**) states that the cumulative demand at any customer location is bounded by the capacity of the vehicle serving that customer. The constraints $r_0 = 0$ and (*) properly define the variables r_i $(i = 1, \ldots, n)$. This is easily seen after observing that $\sum_{k=1}^{T} x_{ij}^k$ equals either 0 or 1. Moreover, these constraints serve as subtour-breaking constraints.

This vehicle routing formulation is rather compact and it permits the incorporation of unlimited choice among vehicle types while not greatly increasing the number of decision variables.

SAVINGS HEURISTICS

The most obvious approach for solving the fleet size and mix problem is to adapt the Clarke and Wright savings technique[5] which was originally designed for the VRP. The Clarke and Wright procedure is extremely simple and flexible and has been shown to solve VRP's quite effectively. Let the index 0 denote the central depot and let n be the number of customers. For the VRP, where all vehicles have the *same* capacity, the Clarke and Wright algorithm proceeds as follows. Recall that a subtour is a closed loop (trip) based at the depot visiting a subset of the n customers. The endpoints of a subtour are the two customers adjacent to the depot. Initially, each of the n customers is serviced by a single vehicle. This results in n subtours with a total routing cost of $2 \sum_{i=1}^{n} c_{0i}$. (At this point, each subtour has only one endpoint.) The Clarke and Wright procedure then proceeds to combine two subtours into one at each step, thereby eliminating one vehicle. Two distinct subtours containing customers i and j can be combined if

(a) both i and j are endpoints of their respective subtours, and

(b) the combined subtour containing both i and j has a total demand not greater than the vehicle capacity.

The savings due to such a combination is $s_{ij} = c_{0i} + c_{0j} - c_{ij}$ and is computed for all $1 \leq i, j \leq n$ $(i \neq j)$. The Clarke and Wright procedure makes combinations in the order indicated by the savings (from largest to smallest), until no further feasible combinations exist.

This procedure is deficient for the fleet size and mix problem due to the fact that it ignores fixed vehicle costs. Since an infinite number of vehicles is assumed to be available, feasibility is certainly no problem. As a result, the savings method will tend to combine subtours until the capacity of the largest vehicle is reached. Unfortunately, a fleet composed almost entirely of largest vehicles may not be the least expensive. This naive application of the Clarke and Wright approach to the fleet size and mix problem will be denoted by CW.

We now seek to extend the concept of savings to include fixed vehicle costs. Every subtour has a total cost equal to the sum of its routing costs and the cost of the vehicle servicing its customers. When two subtours combine to form a larger subtour, the total cost of this larger subtour can be calculated in a similar fashion. To determine which two subtours should combine at any point in the procedure, we need to find the subtours which generate the greatest total savings when combined. These savings must be calculated anew at every step, but all the other attractive features of the CW approach remain. This method, which will be referred to as the Combined Savings (CS) approach, is the basis for the remaining savings algorithms that we consider in this paper.

In order to give a precise representation of this algorithm, some additional notation is necessary. Let $F(z)$ be the fixed cost of the smallest vehicle that can service a demand of size z for a subtour. This defines a step function F on $[0, \infty]$ with $F(0) = 0$. Now consider a subtour I which has node i as one of its endpoints and a subtour J which has customer j as one of its endpoints. If the total demands on these two subtours are z_i and z_j, then the combined savings of joining customers i and j is

$$\bar{s}_{ij} = s_{ij} + F(z_i) + F(z_j) - F(z_i + z_j).$$

The algorithm terminates when $\bar{s}_{ij}$ is negative for all eligible i, j pairs.

The CS approach is a logical extension of the CW approach to the fleet size and mix problem. There is, however, an element of imprecision to the algorithm since $\bar{s}_{ij}$ represents only the *immediate* savings gained by joining customers i and j. Consider the following example. Suppose that subtours I, J and K each have total demand of 100 and are each serviced by a vehicle with a capacity of 100. Suppose further that the next largest vehicle has a capacity of 300 and is therefore considerably more expensive than the 100-capacity vehicle. It might be that having one 300-capacity vehicle service the combination of the I, J and K subtours is cheaper than having these routes individually serviced by three 100-capacity vehicles. But this combined route will not necessarily form under the CS approach. Since subtours can only combine two at

Table 1. Summary of the savings algorithms

Algorithm	Savings	Savings Formula
CW	s_{ij}	$c_{0i} + c_{0j} - c_{ij}$
CS	$\bar{s}_{ij}$	$s_{ij} + F(z_i) + F(z_j) - F(z_i + z_j)$
OOS	s^*_{ij}	$\bar{s}_{ij} + F(P(z_i + z_j) - z_i - z_j)$
ROS	s'_{ij}	$\bar{s}_{ij} + \delta(w)\, F'(P(z_i + z_j) - z_i - z_j)$
ROS-γ	s''_{ij}	$s'_{ij} + (1-\gamma)\, c_{ij}$

where $F(z)$ = the fixed cost of the smallest vehicle that can service a demand of z

$P(z)$ = the capacity of the smallest vehicle that can service a demand of z

$F'(z)$ = the fixed cost of the largest vehicle that has a capacity less than or equal to z

$$w = P(z_i + z_j) - P(\max\{z_i, z_j\})$$

$$\delta(w) = \begin{cases} 0 & \text{if } w = 0 \\ 1 & \text{if } w > 0 \end{cases}$$

a time, it is quite possible that the combined savings for any pair of customers, say i and j (endpoints of subtours I and J), will be negative due to the large fixed cost of the 300-capacity vehicle. In this situation, the CS approach ignores the fact that the 300-capacity vehicle would have an unused capacity of 100 units which could be used to absorb additional subtours later on.

Situations such as the one above motivated us to develop variants of the CS approach, referred to as *opportunity* savings algorithms. Basically, these algorithms consider a total savings based on savings in routing costs, savings in fixed vehicle costs, and opportunity savings. The first two savings are determined in the same manner as in the CS algorithm Opportunity savings is a function of the unused capacity of the vehicle servicing the combined subtours.

The first opportunity savings algorithm that we devised is labeled Optimistic Opportunity Savings (OOS). In this algorithm, opportunity savings is somewhat arbitrarily defined to be the cost of the smallest vehicle that can service the entire unused capacity of the new vehicle. For example, if the available vehicles have capacities of 50, 100 and 200 units, the total savings incurred by combining two subtours with total demands of 40 and 80 units into one with a total demand of 120 units would be $\bar{s}_{ij}$ plus the opportunity savings (the cost of a 100-capacity vehicle—the smallest vehicle that can service the unused capacity of 80 units). The OOS algorithm optimistically assumes that a subtour S with a total demand equal to the unused capacity on subtour R always exists and that at some future time subtour S will combine with subtour R and be absorbed by the vehicle servicing subtour R. The opportunity savings represents the fixed vehicle cost savings that would occur if this future combination materialized. The savings in routing cost from this future combination cannot be estimated as easily and is therefore not included in the OOS algorithm. Clearly, one expects the OOS algorithm to combine subtours more readily than the CS algorithm.

We now provide a formal description of the OOS algorithm. Let $P(z)$ be the capacity of the smallest vehicle that can service a subtour with a total demand of z. Thus, in the previous example, $P(40) = 50$, $P(80) = 100$ and $P(120) = 200$. The OOS algorithm is identical to the CS approach, except that the savings upon linking customers i and j becomes

$$s^*_{ij} = s_{ij} + F(z_i) + F(z_j) - F(z_i + z_j) + F(P(z_i + z_j) - z_i - z_j).$$

Optimism regarding the OOS algorithm does not seem warranted. Indeed, the computational results of Table 2 show that, overall, the CS approach outperforms the OOS algorithm. Actually, both methods are severely flawed. Essentially, the CS approach under-combines and the OOS algorithm over-combines subtours.

A remedial algorithm that comes to mind is the method of Realistic Opportunity Savings (ROS). This method is motivated by two basic assumptions. *First*, the sole purpose of opportunity savings should be to encourage the use of larger vehicles when it seems profitable to do so. Thus, opportunity savings should not be included in the savings formula unless the combining of two subtours requires the use of a larger vehicle. If the vehicle which would service the combination of the subtours is larger than each of the vehicles presently used, we say that a vehicle threshold has been crossed. Opportunity savings should be included only if a vehicle threshold is crossed. *Second*, based on our computational experiments, the total demand of a subtour tends to be close to a vehicle capacity. Therefore, instead of assuming that the vehicle which might be absorbed in some future combination is the smallest vehicle that can service the unused capacity, it is perhaps more appropriate to use the largest vehicle that can be squeezed into the unused capacity. Note that this is the vehicle just smaller than the OOS vehicle, except when the unused capacity equals a vehicle capacity, in which case both algorithms use the same vehicle.

A full description of the ROS algorithm follows. Let $F'(z)$ be a function analogous to $F(z)$, except that $F'(z)$ represents the fixed cost of the largest vehicle whose capacity is less than or equal to z. If $z < a_1$, $F'(z) = 0$. Let I and J denote two subtours with total demands of z_i and z_j and let i and j be endpoints of subtours I and J. Assume, without loss of generality, that $z_i \geq z_j$. If $F(z_i + z_j) = F(z_i)$, then $\bar{s}_{ij}$ is the total savings from linking customers i and j as in the CS approach. If, however, $F(z_i + z_j) > F(z_i)$, then a vehicle threshold has been crossed and

Table 2. Fleet mix costs (fixed + routing)

Problem	1	2	3	4	5	6	7	8	9	1
No. of Nodes	12	12	20	20	20	20	30	30	30	
VC/TC	.49	.66	.61	.93	.62	.92	.84	.68	.65	.
Best Known	602	722	965	6446	1013	6522	7298	2349	2220	23
CW	640	796	1119	7822	1061	9343$^+$	7988	2430	2228	24
CS	636	834	1044	7911$^+$	1060	7016	7839	2789	2278	25
OOS	618*	792	1024	7306	1101	9401$^+$	8227	2590	2412	25
ROS	636	768	1042	7369$^+$	1052	7016	7475	2672	2396	25
ROS-γ	618*	768	1009	6937	1048	6522*	7452	2468	2266	24
SGT	620	722*	965*	6918*	1027	7391$^+$	7527	2368	2248	23
MGT	642	736	989	7345$^+$	1056	7356	7550	2385	2262	24
MGT + 2-opt	634	722*	967	7304$^+$	1031	7338	7506	2383	2253	24
MGT + OROPT	634	722*	966	7300$^+$	1025	7334	7391	2370	2226	23
MGT^5	632	734	981	6933	1037	7019$^+$	7414	2374	2245	23
(MGT + 2-opt)^5	622	722*	967	6930$^+$	1020	6984	7406	2370	2226	23
(MGT + OROPT)^5	622	722*	966	6930$^+$	1013*	6974	7389*	2367*	2220*	23
Lower Bound	554	672	790	6260	866	6375	7060	2204	2110	22
Underestimate	556	678	936	6388	950	6390	7165	2284	2124	22
Best-LB (%)	8.66	7.44	22.15	2.97	16.97	2.31	3.37	6.58	5.21	5.
Best-Underestimate (%)	8.27	6.49	3.10	0.91	6.63	2.07	1.86	2.85	4.52	7.
Mix--Best Known	B^2C^2	A^3C	AB^2CE^2	A^6	$ABDE^2$	A^6	$B^2C^2D^3$	C^2D^2	DE^3	C^2
Mix--LB	B^2C^2	B^2C	C^6E	A^6	C^3E^2	A^6	$A^2B^7C^2$	B^2CD^2	DE^3	BD
Mix--Underestimate	B^2C^2	B^2C	C^6E	A^6	E^3	A^6	B^3C^4D	C^2D^2	DE^3	C^2

Problem	11	12	13	14	15	16	17	18	19
No. of Nodes	30	30	50	50	50	50	75	75	100
VC/TC	.78	.73	.74	.93	.70	.74	.57	.60	.86
Best Known	4763	4112	2438	9132	2640	2822	1783	2432	8721
CW	5428	4276	2550	12000	2885	3026	1968	3447	11319
CS	4959	4243	2650	9689	2763	2978	2043	2677	8741
OOS	5283	4315	2629	10154	2949	2982	2182	2587	10233
ROS	4979	4243	2616	9689	2763	2949	2000	2612	8741
ROS-γ	4953	4221	2566	9178	2763	2894	1958$^+$	2520	8741
SGT	4799	4173	2449	9637	2722	2855	1815	2479	9283
MGT	4965	4268	2494	9174	2742	2912	1837	2520	9411
MGT + 2-opt	4816	4248	2455	9137	2678	2879	1788	2467	9239
MGT + OROPT	4813	4248	2451	9132*	2640*	2861	1783*	2432*	8721*
MGT^5	4828	4268	2471	9174	2706	2845	1837	2510	9290
(MGT + 2-opt)^5	4763*	4136*	2442	9137	2648	2825	1788	2456	9220
(MGT + OROPT)^5	4763*	4136*	2438*	9132*	2640*	2822*	1783*	2432*	8721*
Lower Bound	4636	3878	2119	8874	2264	2504	1380	2002	8290
Underestimate	4632	3882	2384	9054	2501	2631	1703	2237	8537
Best-LB (%)	2.74	6.03	15.05	2.91	16.61	12.70	29.20	21.48	5.20
Best-Underestimate (%)	2.83	5.92	2.27	0.86	5.56	7.26	4.70	8.72	2.16
Mix--Best Known	BC^5	E^6	$A^3B^3CF^4$	A^7B	A^6B^5	AB^6C^2	AB^3C^5	$AB^3C^8D^3$	A^{15}
Mix--LB	A^2C^5	CE^4F	AC^9F^3	A^7B	A^9B^2C	B^{10}	ABC^6	$AB^6C^6D^3$	A^{15}
Mix--Underestimate	BC^5	CE^4F	$C^6D^2F^3$	A^7B	A^{16}	B^8C	C^7	ABC^{13}	A^{15}

opportunity savings need to be considered. In this case, the total savings would be

$$s'_{ij} = \bar{s}_{ij} + F'(P(z_i + z_j) - z_i - z_j).$$

After total savings has been computed, the ROS algorithm proceeds in the same way as the CS approach.

The computational results of Table 2 support the claim that the ROS algorithm is superior to the OOS and CS algorithms. In thirteen of the twenty problems tested, the ROS method performed at least as well as the other two algorithms. Its average and worst case behavior on the twenty problems is also superior to all other savings methods tested.

Still, none of the above methods are powerful enough that single application will consistently generate good solutions. It is therefore necessary to vary these algorithms so that they produce a number of different solutions for each problem, with the least costly of these being selected as a final solution. A method used which introduces variety into the ROS algorithm is referred to as ROS-γ.

The ROS-γ algorithm uses a route shape parameter that changes the Clarke–Wright savings expression to $s_{ij} = c_{0i} + c_{0j} - \gamma \cdot c_{ij}$. In our tests, γ was allowed to vary from 0.0 to 3.0, by tenths, for a total of 31 trials.

The logic behind the ROS-γ approach is fairly straightforward. A major flaw of the savings algorithms is that once a node is assigned to a tour, it cannot be reassigned to another at a later point. Two linkings may seem almost equivalent at an early stage of the algorithm and yet the choice between these can have enormous impact upon the final solution. Varying γ allows the approach to incorporate the variety which is essential to finding a good solution. The Realistic Opportunity Savings algorithm with route shape parameter yields the best results among all savings algorithms that we have developed and tested. Although it is outperformed by another approach, the ROS-γ provides a valid and successful approach to the problem and an interesting extension to the concept of savings.

The savings algorithms described in this section are conveniently summarized in Table 1.

GIANT TOUR ALGORITHMS

These algorithms represent an approach to solving the fleet size and mix problem significantly different from the savings methods. Giant tour algorithms are examples of "route first—cluster second" heuristics. Similar approaches can be found in papers by Newton and Thomas[12], Bodin and Berman[2], and Stern and Dror[16].

Giant tour (GT) algorithms are two-step procedures. First, some method is used to generate a tour that visits all customers. This tour is then partitioned into subtours, each satisfying the problem constraints. The subtours are composed of contiguous segments from the original tour with both ends connected to the depot.

A key to the success of the GT approach is the partitioning procedure. As we shall see, the problem of partitioning the tour can be transformed into a number of shortest path problems. Therefore, the various giant tour algorithms differ mainly in the way in which the initial tour is formed. In the algorithms we tested, the initial tour is formed by solving a traveling salesman problem (TSP). A 2-opt procedure (see [8]) was chosen to solve the TSP, although other intelligent heuristics would have been equally acceptable.

This two-step procedure is easily motivated. Solving the TSP guarantees that adjacent customers will be reasonably close as far as routing cost is concerned, but the formation of the tour in no way considers the effect of clustering customer demands on the fixed vehicle costs (i.e. the "packing" of routes). The partitioning component enables us to include the packing factor in the analysis and compare a huge number of feasible solutions in an effective manner.

THE SINGLE PARTITION GT ALGORITHM

This algorithm (SGT) starts with a tour that begins at the central depot, visits each customer exactly once, and then returns to the depot. Suppose the TSP tour can be written as $0 - s(1) - s(2) - \cdots - s(n) - 0$. The "distance" between customers $s(k)$ and $s(m)$, denoted by COST (k, m), is the total cost of having a vehicle service customers $s(k), s(k+1), \ldots, s(m-1)$ for $k < m$, in that order (let COST $(k, n+1)$ denote the distance between a customer and the depot). The vehicle used is, of course, the smallest that can accommodate such a subtour. That is,

$$\text{COST}(k, m) = c_{0,s(k)} + \sum_{r=k}^{m-2} c_{s(r),s(r+1)} + c_{s(m-1),0} + F\left(\sum_{r=k}^{m-1} d_{s(r)}\right).$$

If the total demand of customers $s(k), s(k+1), \ldots, s(m-1)$ is greater than the largest vehicle's capacity, the distance between these two customers in the shortest path problem is set to infinity. Now that the "distances" have been defined, we seek the shortest path from customer $s(1)$ (that is, the first customer in the tour) to the central depot. Since all arcs point forward, customers are labeled in the order in which they appear in the tour. If the solution to the shortest path problem requires p arcs, then the SGT solution will have p subtours, derived from the arcs in the manner explained above. Figure 1 should help clarify these points.

Like the savings algorithms, single application of the GT approach will sometimes produce unsatisfactory results depending on the sequence in which customers appear in the giant tour. Variety is introduced by starting from several good traveling salesman tours, applying the shortest path procedure to each tour, and then selecting the best solution. In particular, we started from 25 random tours and generated 25 traveling salesman tours that were two-optimal. For the problems tested, a sample size of 25 proved to be large enough to provide the variety necessary to give good results and small enough to ensure the algorithm's efficiency.

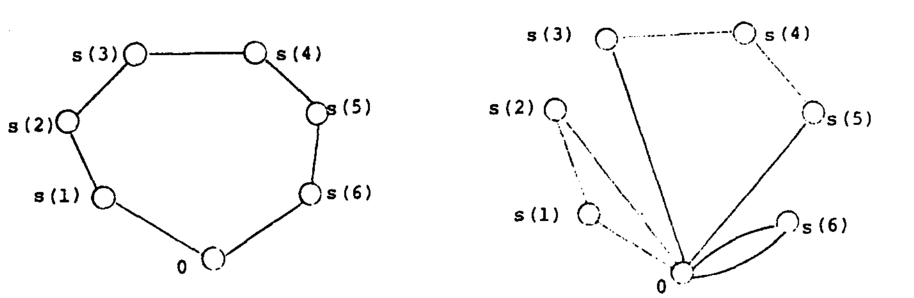
THE MULTIPLE PARTITION GT ALGORITHM

The starting tour in this algorithm (MGT) visits each customer but not the depot. Since, in the previous algorithm, the starting tour visits the depot only once and the final solution visits the depot as many times as there are vehicles, we don't lose much by this alteration. In fact, by excluding the depot we are able to consider a much greater number of partitions, all on the same TSP tour.

Suppose the TSP tour can be written as $s(1) - s(2) - \cdots - s(n)$. Consider then the string $s(1) - s(2) - \cdots - s(n) - s(n+1) - \cdots - s(n+M)$ where $s(n+i)$ is a duplication of node $s(i)$ for $i = 1, \ldots, M$ with M defined by

$$\sum_{r=1}^{M-1} d_{s(r)} < \text{largest vehicle capacity} \leq \sum_{r=1}^{M} d_{s(r)}.$$

The "distance" between customers $s(k)$ and $s(m)$ is defined as before. We can define $S(i)$ to be the length of the shortest path between $s(i)$ and $s(n+i)$ for $i = 1, \ldots, M$. Then, the path corresponding to $\min_{i=1,\ldots,M} S(i)$ gives us the optimal partition of the initial tour. We only need to



TSP Tour

Fleet Size and Mix Solution when shortest path is s(1)-s(3)-s(6)-0

$$\text{COST}(3,5) = c_{0,s(3)} + c_{s(3),s(4)} + c_{s(4),0} + F(d_3 + d_4)$$

Fig. 1. SGT algorithm.

minimize over $i = 1, \ldots, M$ since each path *must* pass through at least one node in the set $\{s(1), s(2), \ldots, s(M)\}$.

As with the SGT algorithm, a single application of MGT may produce unsatisfactory results. Again variety is introduced by starting from a number of TSP tours. Because of the more powerful partitioning, a sample size of 5 has proved to be effective. We refer to this variant of the multiple partition GT algorithm as MGT^5.

IMPROVEMENT ALGORITHMS

Although MGT^5 gives quite reasonable results, improvement algorithms are always worth considering. Indeed, the TSP tour guarantees that adjacent customers are reasonably close, but does not take into account the fact that each vehicle has to start and end its trip at the depot. Moreover, although the partitioning procedure compares a large number of feasible solutions, this number is still only a small fraction of all feasible solutions. As a result, the solution obtained by a partitioning algorithm could easily benefit from an improvement post-processor.

The two improvement algorithms presented here are adaptations of TSP improvement heuristics (see Golden *et al.*[8] for background). We first take the starting solution (as produced, say, by the MGT algorithm) and construct from it an equivalent TSP tour by including as many copies of the depot as there are vehicles in the starting solution. To illustrate, if we take the three-subtour solution from Fig. 1 as a starting solution and refer to the copies of the depot as $c1$, $c2$ and $c3$, an equivalent TSP tour becomes

$$c1 - s(1) - s(2) - c2 - s(3) - s(4) - s(5) - c3 - s(6).$$

To allow the improvement algorithm to use more vehicles than used at present, we insert two additional depot nodes in the tour. Using Fig. 1 once again, the TSP tour might become

$$c1 - s(1) - s(2) - c2 - s(3) - s(4) - s(5) - c3 - c4 - c5 - s(6).$$

Implicit in such a configuration is the fact that the "distance" between two depot copies is zero units. The improvement algorithms that we appeal to are *adaptations* of TSP improvement heuristics in that they consider fixed costs as well as routing costs and they check each time for feasibility. As we shall see, in our computational experience, the MGT algorithm gives the best "one run" results. Consequently, we use this solution as our starting solution.

The first improvement algorithm (MGT + 2-opt) is an adaptation of the 2-opt algorithm. All exchanges of two arcs are tested until there is no feasible exchange that improves upon the current solution.

The second (MGT + OROPT) is an adaptation of the modified 3-opt procedure proposed by Or[13]. This procedure, which we will refer to as OROPT, considers all 2-exchanges and those 3-exchanges that would result in a string of one, two, or three currently adjacent nodes being inserted between two other nodes. By thus limiting the number of exchanges that need to be considered, OROPT requires significantly fewer calculations than does the 3-opt procedure.

The 2-opt and OROPT post-processors reduce the gap between the MGT solution and the best known solution for the test problems considered, as we shall see. Since MGT^5 gives results comparable to (MGT + 2-opt), variety is again achieved by starting from several (in this case, five) TSP tours. This leads to $(MGT + 2\text{-opt})^5$ and $(MGT + OROPT)^5$.

PATHOLOGICAL EXAMPLES

Any time a heuristic algorithm is proposed for a problem, a natural question is how poorly the algorithm can perform in the worst case. For certain heuristics, such as the first-fit decreasing algorithm for bin packing (see Johnson *et al.*[10]), researchers have been able to prove that worst case behavior is not bad at all. For many other heuristics, however, pathological examples do exist and they are important in that they indicate particular weaknesses inherent in the various heuristics (for example, see Papadimitriou and Steiglitz[14]). In this section, we consider the performance of the ROS and GT algorithms on some perverse problems.

In Example 1, we pick up the ROS algorithm at a point where three subtours remain, each of which has total demand of 100 (actually, any three numbers between 90 and 100 will work just as well). Each of these subtours is serviced by a vehicle with a capacity of 120 and a fixed cost of \$100. The only other vehicle type has a capacity of 300 and a cost of \$300. Only the six endpoints of the three subtours are available for combining. The cost of traveling between the depot and each of these customers is \$50. The travel cost between each of the endpoints is negligible compared to the other costs and can, thus, be considered to be \$$\epsilon$. In addition, the costs between all adjacent customers in the present subtours can also be assumed to be negligible—not too unrealistic a supposition (see Fig. 2 for clarification).

The dollar savings for any pair of endpoints is $\{2(50)-\epsilon\}+\{2(100)-300\}+0=-\epsilon$. The opportunity savings is zero because $300-2(100) = 100) = 100 < 120$. Since all savings are negative, the ROS procedure stops with a solution that has an approximate cost of $3(\$50+\$50+\$100) = \$600$. Suppose, however, that the three subtours were combined and serviced by a single vehicle of capacity 300. The cost of this solution would be approx. $\$50+\$50+\$300=\$400$. Thus, the ROS solution is off by a factor of approx. 50% (on another pathological example that we have constructed, the ROS solution is off by 100%). Note that the GT approach, irrespective of the tour formed in the first phase of the algorithm, will easily find the optimal solution. It is probably possible to construct examples for which the ROS procedure performs far worse than it does here. The preceding example, however, is one that could easily occur in practice. The example illustrates a basic flaw of the ROS and other savings algorithms—the inability to consider the combining of more than two subtours at a time. This fundamental weakness in the ROS algorithm is one that the GT approaches seek to avoid.

Example 2 is somewhat more contrived. Suppose there are $4n$ customers in addition to the depot, where n is any positive integer. Let $2n$ of these have demands of 49 and each be denoted by S and let $2n$ have demands of 51 and be denoted by L. The one vehicle type available has a capacity of 100 and a cost of \$$C$. All depot-customer trips cost \$$X$. All trips between two customers with equal demands cost \$$\epsilon$, where ϵ is much less than C and X. Finally, all trips between customers with unequal demands cost \$$2\epsilon$.

The least expensive traveling salesman tour first visits all customers with demands of 49 and then visits all customers with demands of 51. The best partitioning of this tour has 2 subtours of the form depot $-S-L-$ depot, $(n-1)$ subtours of the form depot $-S-S-$ depot, and $2(n-1)$ subtours of the form depot $-L-$ depot. The approximate cost of this solution is $\$(6n-2)X+\$(3n-1)C$. The optimal solution, of course, has $2n$ subtours of the form depot $-S-L-$ depot and costs approximately $\$4nX+\$2nC$. So MGT is about 50% above optimality in performance as n gets large. See Fig. 3 for details.

In this case, the ROS approach fares no better. Since a $S-S$ combination has an associated savings ϵ dollars better than a $S-L$ combination, the ROS approach will pair type S customers together until only type L customers remain. Thus, the solution illustrated in Fig. 3(b) which is about 50% above the optimal solution with an approximate cost of $\$6nX+\$3nC$ will result from applying the ROS algorithm.

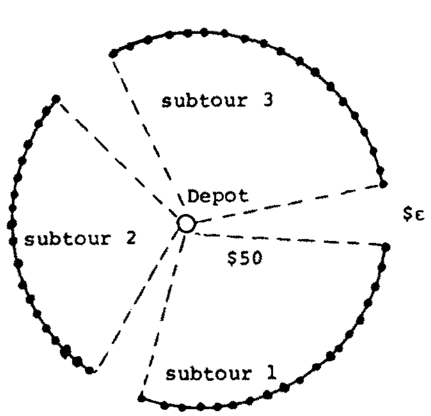


Fig. 2. ROS pathological example.

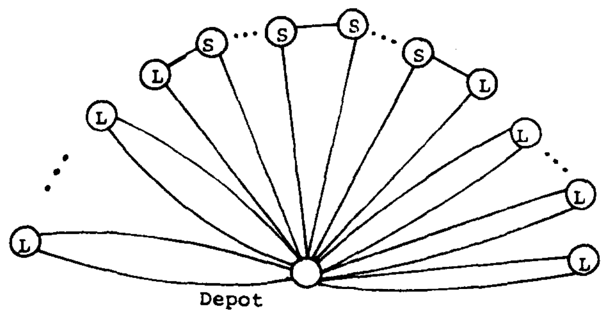


Fig. 3(a). MGT solution to example 2.

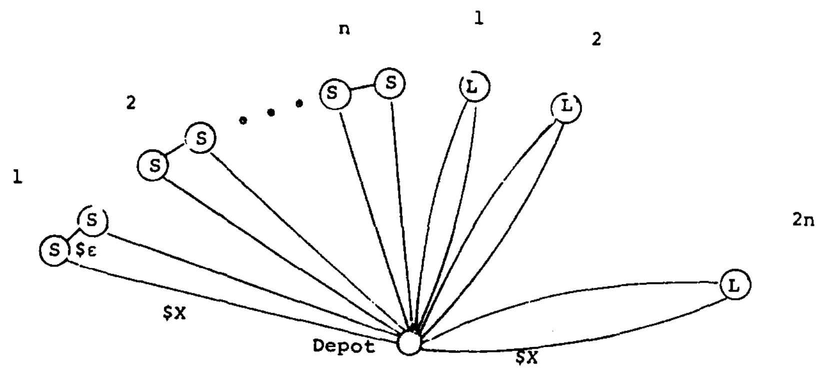


Fig. 3(b). ROS solution to example 2.

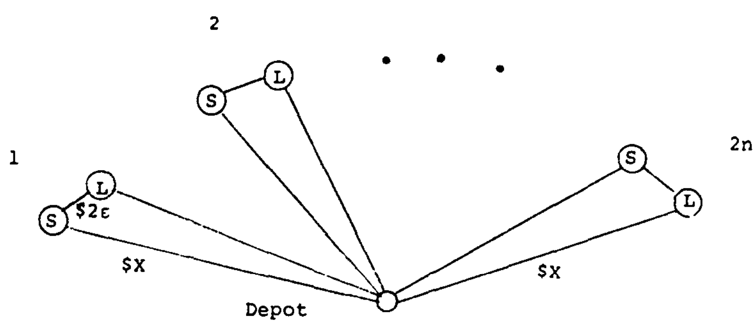


Fig. 3(c). Optimal solution to example 2.

This example exploits the fundamental flaw of the GT approach, the lack of consideration of customer demands in the formation of the giant tour, and carries it to an extreme. The example is certainly not a problem one would expect to encounter in the real world.

There are also problems for which the ROS approach performs well whereas the GT approach performs poorly. Consider Example 3 which is Example 2 but with an additional vehicle type. The vehicle has a capacity of 50 and a cost of $\$C/2$. The best partitioning is the same as in the previous example. So is the optimal solution. For this example, though, the ROS algorithm computes a savings of $-\epsilon$ for $S-S$ combinations and a savings of $C/2-2\epsilon$ for $S-L$ combinations. It therefore obtains the optimal solution without difficulty.

The significance of all this is somewhat diminished by the fact that the variability introduced into each of the algorithms and the use of post-processors are likely to reduce or even eliminate the errors in the examples cited. Creating pathological examples for the ROS-γ algorithm or for the GT approach followed by one of the improvement algorithms is, clearly, a far more complex problem and one that we do not address in this paper. The value of the examples presented is that they illustrate the sort of configurations that might present difficulties for the various algorithms, they help to motivate new algorithms, and they underscore the necessity of introducing variability and/or post-processors into the heuristic procedures.

LOWER BOUND PROCEDURE

In a later section, we will present the results of testing the various algorithms on twenty sample problems. Since, to our knowledge, very little work has been reported on the fleet size and mix vehicle routing problem, we have devised our own set of test problems. In the absence of optimal solutions for these problems, there is no yardstick with which we can readily measure the accuracy of our heuristics. With this in mind, the following important question arises: How can the quality of the solutions generated be assessed?

One method frequently used is to obtain lower bounds on the objective value of the original problem and compare these with the heuristic solutions. There are numerous ways of finding lower bounds for fleet size and mix vehicle routing problems. Unfortunately, most of the easily-derived bounds are virtually worthless in that they fall far below the optimal objective value. The basic difficulty is that the two components that contribute to total cost, namely, the routing cost and the fixed vehicle cost, are highly interdependent. Any method that analyzes these two components separately should be expected to yield inferior results.

Indeed, a minimum routing cost solution would most likely be based on a totally different vehicle mix than would a minimum fixed cost solution. The sum of these two minimum values would provide a lower bound for the problem, but it would probably not be very informative.

The lower bound procedure we present here, trades off fixed costs against routing costs in the following way. First, we order the customers in decreasing distance from the depot. Then we search for a "lower bound" fleet mix using the following ideas. The total cost for each vehicle is greater than or equal to the sum of its fixed cost and the cost of moving from the depot to the farthest customer not yet fully served and back to the depot. Each vehicle serves as much as possible of the demand not yet satisfied (note that splitting of a customer's demand between two vehicles is allowed here). An additional assumption is that the total capacity of available vehicles larger than a certain type must be at least as large as the total demand of customers who must be served by these vehicles. As we shall see, this "lower bound" fleet mix can be determined by solving a shortest path problem.

The procedure we present provides a lower bound by simplifying the original problem in two ways. As far as routing costs are concerned, it takes into account only the radial component. As for the fleet mix, since splitting of demand is allowed, there is no guarantee that the vehicle mix is feasible for the original problem.

Assume without loss of generality, that the customers are ordered in decreasing distance from the depot. Thus, customer 1 is farthest from the depot, customer 2 is second farthest, and so on. Let $P = \sum_{l=1}^{n} d_l$ be the total demand and define $d_0 = 0$ and $C(i) = 2c_{0j}$ where

$$\sum_{l=0}^{j-1} d_l \le i < \sum_{l=0}^{j} d_l \text{ for } i = 0, \ldots, P-1.$$

Also, let s_k be the sum of demands of customers for whom vehicle type k is the smallest one that can service the demand $(s_{T+1} = 0)$. For the sake of convenience we assume that $s_1, s_2, \ldots, s_T$ are all positive. Now, we are ready to construct the following network. The first node is denoted by 0 and is the origin. The other nodes are labeled from 1 to P, which is the destination. For each vehicle type k, there is an arc from node i to node $\min\{P, i + a_k\}$ with a length of $f_k + C(i)$ if and only if $i \ge \sum_{l=k+1}^{T+1} s_l$. In other words, the first s_T nodes in the network represent demands that require the largest vehicle type, the next s_{T-1} nodes represent demands that can be served by either the largest or the second largest vehicle type, and so on. With regard to routing costs, the first d_1 nodes represent the demand of the farthest customer, the next d_2 nodes represent the demand of the second farthest, and so on. The shortest path from the origin to the destination gives us the required lower bound. Each arc used in the shortest path corresponds to a vehicle in the lower bound mix. The example below should help to clarify the procedure.

In Fig. 4, we construct the appropriate shortest path network from a fleet size and mix vehicle routing problem where there are four customers and three vehicle types and the data are as follows:

$$d_1 = 3,\ d_2 = 1,\ d_3 = 2,\ d_4 = 5;\ a_1 = 1,\ a_2 = 4,\ a_3 = 6.$$

This gives

$$s_1 = 1,\ s_2 = 5,\ s_3 = 5 \text{ and } P = 11.$$

Also,

$$C(i) = \begin{cases} 2c_{01} & 0 \le i < 3 \\ 2c_{02} & 3 \le i < 4 \\ 2c_{03} & 4 \le i < 6 \\ 2c_{04} & 6 \le i < 11. \end{cases}$$

Only arcs starting from reachable nodes are drawn in Fig. 4. In general, the node set can be greatly consolidated; however, we do not pursue this matter here.

As might be expected, the lower bound procedure gives excellent estimates of total fixed costs. Due to the fact that it only takes into account the radial component of routing costs, it may greatly underestimate the routing costs. Therefore, we have developed another procedure which we refer to as an underestimate procedure.

UNDERESTIMATE PROCEDURE

Another way to make routing costs and fixed costs "dependent" on each other is first to fix the number of vehicles, m, to be used and then to find the minimum total cost for that many vehicles. Since it is not known *a priori* how many vehicles are needed, we iterate on m and then choose the result which gives the smallest total cost.

Once the number of vehicles is fixed, well-known methods come to mind for estimating the minimum routing and fixed vehicle costs. A lower bound on minimum routing costs could be found by solving the m-traveling salesman problem over all the customers, where m is the number of vehicles. Unfortunately, since the m-TSP is NP-hard, heuristics must be used to solve the problem for a near-optimal answer. Since capacity constraints are ignored in this operation, the answer will remain a lower bound on routing costs in the vast majority of cases. However, to be precise, we use the term "underestimate" to describe this answer.

To estimate the minimum fixed vehicle costs we construct the same network as for our lower bound procedure, however, we totally ignore routing costs. Next we compute the shortest path from node 0 to node P with exactly m arcs. A variant of the Ford–Bellman–Moore shortest path algorithm (see Dreyfus[6]) due to Saigal[15] easily handles this problem.

Once routing and vehicle costs are determined for all possible values of m, the smallest sum of these two components is used as an underestimate for the optimal objective value. If this

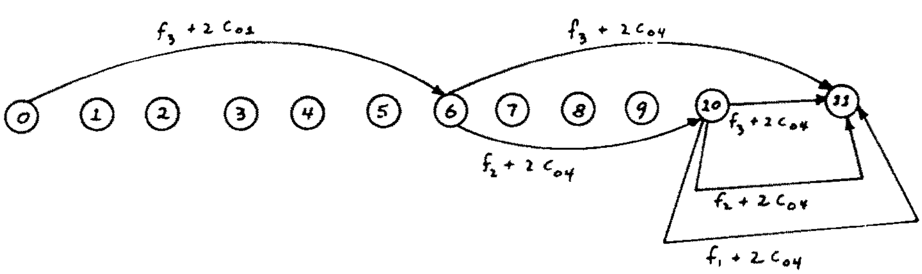


Fig. 4. Lower bound network.

underestimate is a good approximation to the optimal objective value, then its vehicle mix is expected to be quite close to the vehicle mix of the optimal solution. This is not always the case due to the specifics of the routing cost estimation procedure. In many cases, the m-TSP solution includes $m-1$ single customer subtours (usually, consisting of the $m-1$ customers closest to the depot) and one subtour serving the rest of the customers. The depot is, thus, adjacent to only $m+1$ customers, rather than the $2m$ that we mentioned earlier. This phenomenon leads to recommended vehicle mixes with more vehicles than are absolutely necessary. A simple way to overcome this difficulty is to require that all m-TSP subtours have at least two customers (this is easily handled within the context of the 2-opt procedure). This is another reason why the numbers produced are not lower bounds, since a solution with single customer subtours could conceivably be least expensive. However, we again emphasize that the numbers derived *are* underestimates, that is, conservative estimates of the optimal objective value along with highly accurate vehicle mixes but infeasible routings. As such, they are valuable indicators of the quality of the solutions generated by the heuristic algorithms developed in this paper. These underestimates figure to be inaccurate only for those problems where the optimal solution has many single customer subtours.

In addition, the underestimate procedure can be used in other ways. Once a vehicle mix is suggested, the fleet size and mix vehicle routing problem reduces to a vehicle routing problem of the type described by Fisher and Jaikumar[6]. The underestimate procedure can therefore be utilized to supply this mix. Of course, it is entirely possible that this mix is infeasible or is too tight to produce a good solution. Therefore, it would probably be necessary to obtain several vehicle mixes and solve a VRP for each one. The idea obviously needs more work, but it seems worth pursuing.

It should be mentioned that one can calculate a true lower bound using a variation of the underestimate procedure. Iterate on the number of vehicles as before and determine a lower bound on fixed vehicle cost in the same manner as previously. To determine a lower bound on routing cost, distinguish between depot-customer travel and customer–customer travel. Add together the costs of the m cheapest depot-customer trips and multiply the sum by two. This represents the cost of visiting the depot exactly m times. Next, find the cost of the first $n-m$ insertions into a minimal spanning tree over the set of customers $\{i | i = 1, 2, \ldots, n\}$. Adding these two routing costs gives a valid lower bound on the total routing cost. Now, if we sum the lower bounds on routing and vehicle costs, we have a reasonable lower bound on the optimal objective value, but it will not be nearly as informative as the underestimate in that the resulting lower bound will tend to be quite loose.

COMPUTATIONAL RESULTS

The data associated with twenty sample problems are given in the Appendix. Computational results on these problems are summarized in Tables 2–4. The third row of Table 2 gives the ratio of vehicle cost to total cost in the best known solution—this is a compact parameter for classifying problems, which indicates the diversity of the sample problems studied. The fourth line records the cost of the best known solution for each problem. The rows that follow correspond to the various algorithms, the lower bound, and the underestimate that we have discussed in this paper. The next two lines display the difference (in %) between the best known solution value and, respectively, the lower bound and the underestimate. Finally, the vehicle mixes corresponding to the best known solution, the lower bound, and the underestimate are presented. An asterisk indicates a best generated solution for a particular problem. A cross indicates that the algorithm performed poorest on that problem. The percentage above the best known solution value in that case is listed under the column headed "Worst" in Table 3. The column labeled "Average" in Table 3 contains the average percentage above the best known solution value for the twenty problems. In Table 4, the percentages above the best known solution values are given for all problems and all algorithms.

Clearly, the $(\mathrm{MGT}+\mathrm{OROPT})^5$ algorithm is superior to the other algorithms in terms of accuracy. In most cases, however, it performs only marginally better than $(\mathrm{MGT}+\text{2-opt})^5$. In sixteen of the twenty problems tested, the $(\mathrm{MGT}+\mathrm{OROPT})^5$ algorithm produced the best generated solution. Only on problems 1, 4 and 6 did the algorithm perform poorly. For problems

Table 3. Worst and average performance of algorithms

	Worst (%)	Average (%)
CW	43.25	14.10
CS	22.73	8.11
OOS	44.14	12.03
ROS	14.32	6.39
ROS-γ	9.81	3.59
SGT	13.32	2.80
MGT	13.95	4.50
MGT + 2-opt	13.31	3.04
MGT + OROPT	13.25	2.13
MGT^5	7.62	2.77
$(MGT + 2\text{-}opt)^5$	7.51	1.53
$(MGT + OROPT)^5$	7.51	1.03

Table 4. Percentage above best known solution values

Problem	1	2	3	4	5	6	7	8	9	10
CW	6.31	10.25	15.96	21.35	4.74	43.25	9.45	3.45	0.36	2.79
CS	5.65	15.51	8.19	22.73	4.64	7.57	7.41	18.73	2.61	8.99
OOS	2.66	9.70	6.11	13.34	8.69	44.14	12.73	10.26	8.65	8.66
ROS	5.65	6.37	7.98	14.32	3.85	7.57	2.43	13.75	7.93	5.57
ROS-γ	2.66	6.37	4.56	7.62	3.46	0.00	2.11	5.07	2.07	2.36
SGT	2.99	0.00	0.00	7.32	1.38	13.32	3.14	0.81	1.26	1.31
MGT	6.64	1.94	2.49	13.95	4.24	12.79	3.45	1.53	1.89	5.41
MGT + 2-opt	5.32	0.00	0.21	13.31	1.78	12.51	2.85	1.45	1.49	3.76
MGT + OROPT	5.32	0.00	0.10	13.25	1.18	12.45	1.27	0.89	0.27	0.13
MGT^5	4.98	1.66	1.66	7.56	2.37	7.62	1.59	1.06	1.13	0.68
$(MGT + 2\text{-}opt)^5$	3.32	0.00	0.21	7.51	0.69	7.08	1.48	0.89	0.27	0.51
$(MGT + OROPT)^5$	3.32	0.00	0.10	7.51	0.00	6.93	1.25	0.77	0.00	0.08

Problem	11	12	13	14	15	16	17	18	19	20
CW	13.96	3.99	4.59	31.41	9.28	7.23	10.38	41.74	29.79	11.78
CS	4.12	3.19	8.70	6.10	4.66	5.53	14.58	10.07	0.23	2.93
OOS	10.92	4.94	7.83	11.19	11.70	5.67	22.38	6.37	17.34	17.31
ROS	4.53	3.19	7.30	6.10	4.66	4.50	12.17	7.40	0.23	2.34
ROS-γ	3.99	2.65	5.25	0.50	4.66	2.55	9.81	3.62	0.23	2.34
SGT	0.76	1.48	0.45	5.53	3.11	1.17	1.79	1.93	6.44	1.86
MGT	4.24	3.79	2.30	0.46	3.86	3.19	3.03	3.62	7.91	3.27
MGT + 2-opt	1.11	3.31	0.70	0.05	1.44	2.02	0.28	1.44	5.94	1.84
MGT + OROPT	1.05	3.31	0.53	0.00	0.00	1.38	0.00	0.00	0.00	1.43
MGT^5	1.36	3.79	1.35	0.46	2.50	0.82	3.03	3.21	6.52	2.00
$(MGT + 2\text{-}opt)^5$	0.00	0.58	0.16	0.05	0.30	0.11	0.28	0.99	5.72	0.50
$(MGT + OROPT)^5$	0.00	0.58	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

4 and 6 the reason is easily understood. Since vehicle costs are so important and capacity constraints for the optimal mix are so tight, the problem reduces to a packing problem. The GT procedures, however, begin with a TSP tour which is the result of looking at only the routing component. The best of the savings algorithms, ROS-γ, is outperformed by the GT with post-processor approach. Compared with the SGT and MGT approaches without post-processors, ROS-γ performs reasonably well.

The computational results for the lower bound are as expected. The higher the VC/TC ratio, the better the bound. In other words, the procedure gives excellent estimates for the vehicle costs, but due to the fact that only radial distances are taken into account, it does not estimate routing costs as effectively. This observation is confirmed if we compare the vehicle mixes associated with the best known solution and the lower bound. In six cases, they are exactly the same and in most other cases they are very similar; the main reason for differences is probably that splitting of demand is allowed in the lower bound procedure.

The accuracy of the underestimate procedure is quite satisfying. In no problem is the best known solution as much as 10% above the underestimate. Eleven of the solutions are within 5% of this value. Again the vehicle mixes are remarkably close to those for the best known solutions, especially when the ratio VC/TC is high. In nine cases, the mixes are identical and in most other cases they are very similar. Differences here are due to the fact that splitting is allowed in the underestimate procedure and that single customer subtours are forbidden. This is particularly evident in Problem 13, where we have six single customer subtours in the best known solution. In general, our computational work leads us to believe that the underestimates are, in most cases, lower bounds that are capable of indicating optimal or nearly optimal vehicle mixes.

CONCLUSIONS

The fleet size and mix vehicle routing problem is an important and practical problem which has not yet received much attention in the literature. In this paper, several general solution techniques have been described. One of these, the giant tour with post-processor approach, has yielded excellent results on a wide variety of test problems. Moreover, this procedure would have no difficulty in dealing with more realistic situations where, for example, routing costs are dependent upon vehicle type. Indeed, both the partitioning procedures and the exchange procedures can be easily modified to apply to this variation of the vehicle size and mix problem. In addition, the lower bounding and underestimate procedures developed in the paper are of value not only in assessing the quality of heuristic solutions, but also in providing a basis for the determination of a nearly optimal fleet mix.

As for future research, it would be interesting to see if the "generalized assignment" VRP heuristic of Fisher and Jaikumar[7] can be modified to incorporate the more flexible fleet constraints of the fleet size and mix vehicle routing problem. Finally, a realistic variation of the problem that might be investigated is the problem in which there are limits on the number of vehicles of each type.

It is hoped that this paper has succeeded in providing a first set of answers to the many questions it has raised and that this effort will inspire further investigation of the fleet size and mix vehicle routing problem.

REFERENCES

1. M. Ball, B. Golden, A. Assad and L. Bodin, Planning for truck fleet size in the presence of a common carrier option. (To be published).
2. L. Bodin and L. Berman, Routing and scheduling of school buses by computer. *Transpn Sci.* **13**, 113–129 (1979).
3. L. Bodin and B. Golden, Classification in vehicle routing and scheduling. *Networks* **11**, 97–108 (1981).
4. N. Christofides and S. Eilon, An algorithm for the vehicle dispatching problem. *Oper. Res. Quart.* **20**, 309–318 (1969).
5. G. Clarke and J. Wright, Scheduling of vehicles from a central depot to a number of delivery points. *Ops Res.* **12**, 568–581 (1964).
6. S. Dreyfus, An appraisal of some shortest-path algorithms. *Ops Res.* **17**, 395–412 (1969).
7. M. Fisher and R. Jaikumar, A generalized assignment heuristic for vehicle routing. *Networks* **11**, 109–124 (1981).
8. B. Golden, L. Bodin, T. Doyle and W. Stewart, Approximate traveling salesman algorithms. *Opers Res.* **28**, 694–711 (1980).
9. B. Golden, T. Magnanti and H. Nguyen, Implementing vehicle routing algorithms. *Networks* **7**, 113–148 (1977).

10. D. Johnson, A. Demers, J. Ullman, M. Garey and R. Graham, Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM J. Computing* 3, 297–325 (1974).
11. L. Levy, The fleet mix vehicle routing problem. Masters Thesis, Applied Mathematics, University of Maryland at College Park (1980).
12. R. Newton and W. Thomas, Bus routing in a multi-school system. *Comput. Ops Res.* 1, 213–222 (1974).
13. I. Or, Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking. Doctoral dissertation, Northwestern University (1976).
14. C. Papadimitriou and K. Steiglitz, Some examples of difficult traveling salesman problems. *Ops Res.* 26, 434–443 (1978).
15. R. Saigal, A constrained shortest route problem. *Ops Res.* 16, 205–209 (1968).
16. H. Stern and M. Dror, Routing electric meter readers. *Comput. Ops Res.* 6, 209–223 (1979).

APPENDIX: PROBLEM DATA

Problems #1 and 2—12 node problem from [5]

	#1			#2	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	15	20	A	30	60
B	35	50	B	40	90
C	60	100	C	110	300

Problems #3 and 4—depot and first 20 nodes of 50 node problem from [4]

	#3			#4	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	20	20	A	60	1000
B	30	35	B	80	1500
C	40	50	C	150	3000
D	70	120			
E	120	225			

Problems #5 and 6—same as #'s 3 and 4 except depot is now at 20,20

Problems #7-12—30 node problem from [5]

	#7			#8			#9	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	40	150	A	10	15	A	40	30
B	100	500	B	50	50	B	100	100
C	140	800	C	150	200	C	140	160
D	200	1200	D	400	600	D	200	240
E	300	2000				E	300	400

	#10			#11			#12	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	40	30	A	30	60	A	30	40
B	100	100	B	80	200	B	50	80
C	140	160	C	200	700	C	75	150
D	200	240	D	350	1500	D	120	300
						E	180	500
						F	250	800

Problems #13 and 14—depot and first 50 nodes of 75 node problem from [4]

	#13			#14	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	20	20	A	120	1000
B	30	35	B	160	1500
C	40	50	C	300	3500
D	70	120			
E	120	225			
F	200	400			

Problems #15 and 16—50 node problem from [4]

	#15			#16	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	50	100	A	40	100
B	100	250	B	80	200
C	160	450	C	140	400

Problems #17 and 18—75 node problem from [4]

	#17			#18	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	50	25	A	20	10
B	120	80	B	50	35
C	200	150	C	100	100
D	350	320	D	150	180
			E	250	400
			F	400	800

Problems #19 and 20—100 node problem from [4]

	#19			#20	
Vehicle	Capacity	Cost	Vehicle	Capacity	Cost
A	100	500	A	60	100
B	200	1200	B	140	300
C	300	2100	C	200	500