WP - 54

VEHICLE ROUTING PROBLEM AND SIMULATED ANNEALING

by

Dr. S. Rajagopalan* and Dr. A.K. Rao**

September, 1993

INDIAN INSTITUTE OF MANAGEMENT
BANGALORE

---

*Asst.Professor, Indian Institute of Management Bangalore
**Professor, Indian Institute of Management Bangalore

ABSTRACT

The Vehicle Routing Problem (VRP) is concerned with finding efficient routes for a fleet of vehicles/buses to pick up employees from pre-determined bus stops and bring them to the work place. We believe that the VRP will increase in importance as the fuel prices rise and as the extent to which an organization will subsidize the transportation decreases in the face of increased competition as a result of liberalization. In this paper, two heuristics, both based on iterative improvement of an initial solution, have been developed. One of the routines has been developed using Simulated Annealing. We compare these routines with some existing routines and the results are favourable.

1. Introduction

In India, it is a well known fact that the public transport system is not adequate to meet the demands of the commuters. This usually results in overcrowding of transport vehicles/buses and has an effect on the work efficiency of the workers who depend on the public road transport to commute to their places of work. In view of this, many organizations, both public and private, encourage their work force to use some comfortable private transport services and reimburse their travel expenses. Some organizations provide some alternative modes of transport exclusively for their employees; either they engage a transport contractor or maintain a fleet of buses for this purpose. Serious thought must be given to the economical operation of the fleet especially when the employees have to be picked up and dropped at scattered locations far away from the

place of work. The Vehicle Routing Problem (VRP) will increase in importance as the petroleum prices rise and the extent to which an organization is willing to subsidize the transportation decreases. (It may also be noted that a study ([4]) computed the annual distribution costs in USA to be at approximately 21% of the GNP.) Hence an efficient scheduling and routing of buses (which includes an optimum fleet size) is important.

It has been shown in [7] that the VRP is an NP-complete problem. Therefore researchers have developed heuristics to arrive at schedules which will (locally) minimize the total cost under different constraints.

In this paper we assume that the buses have to start from the work place (depot) to pick up employees from pre-assigned points and bring them to the work place. (Similarily, at the end of the shift, the buses leave the depot, to drop the employees at the points from where they were picked up, and return to the depot.) Simulated Annealing method ([9]), goal programming and heuristics are used to obtain near optimal solutions. The results are compared with Single Point Exchange ([1]), and Two Optimality ([8], [11]) heuristics using the data (obtained from [1]) for a large public sector company located in Bangalore.

2. Notations

We assume that each bus starts from the work place (bus stop 1) and completes the route by returning to the work place. A bus is said to "visit" one stop if it picks up/drops passengers at that stop. A bus visiting a stop immediately

after visiting another stop may have to go via some other stop (if it lies on the minimum cost path). We wish to avoid having variables which keep track of whether a bus visits a stop or merely crosses that stop. Therefore we assume that every pair of stops is connected by a (artificial) road that does not include any other stop and we set the cost of plying on this road equal to the minimum cost on the actual network.

N = Number of pick up/drop points. The pick up/drop points are numbered 1 through N with 1 being the work place.

M = Available number of buses.

B = {1, 2, ..., M}

S = {1, 2, ..., N}

Q_k = capacity of the kth bus, $k \in B$

q_i = Number of passengers (to be picked up or dropped) at bus stop i, $i \in S-\{1\}$. We assume that $q_i \leq Q_k$, $k \in B$.

c_{ij} = Cost associated with the movement of a bus from stop i to stop j, i and $j \in S$. (The cost may be in terms of distance or time or any other relevant cost. This is assumed to be independent of the capacity of the bus.)

D = Upper limit on the total cost of any subroute.

For $i, j \in S$, and $k \in B$, let:

$$x_{ijk} = \begin{cases} 1, & \text{if bus k visits stop j from stop i} \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ik} = \begin{cases} 1, & \text{if bus k visits stop i} \\ 0, & \text{otherwise} \end{cases}$$

$$r_k = \begin{cases} 1, & \text{if bus k picks up at least one passenger} \\ 0, & \text{otherwise} \end{cases}$$

3. Linear Formulation of the basic VRP

Here we consider two objective functions which are to be minimized. The first objective minimizes the number of buses. The second minimizes the total cost. For other formulations, refer [6]. Explanation of the constraints (also refer [2]) are given at the end of the formulation.

Minimize $\{ \sum_k r_k, \sum_{i,j} c_{ij} \sum_k x_{ijk} \}$

Subject to:

$$\sum_k y_{ik} = \begin{cases} 1, & \text{if } i \in S/\{1\} \\ \sum_k r_k, & \text{if } i = 1 \end{cases} \quad \text{..... 1}$$

$$\sum_i q_i y_{ik} \leq Q_k, \quad i \in S-\{1\} \text{ and } k \in B \quad \text{..... 2}$$

$$\sum_{i,j} c_{ij} x_{ijk} \leq D, \quad i, j \in S \text{ and } k \in B \quad \text{..... 3}$$

$$r_k \leq \sum_i y_{ik} \leq N.r_k, \quad i \in S \text{ and } k \in B \quad \text{..... 4}$$

$$\sum_j x_{jik} = \sum_j x_{ijk} = y_{ik} \quad \text{for } i,j \in S,\ k \in B \quad \text{..... 5}$$

$$\sum_{k,j} x_{ijk} = 1, \quad \text{for } i,j \in S,\ k \in B \quad \text{..... 6}$$

Remarks on the constraints

1. *Depot (stop 1) is visited by every bus. Other stops are visited by exactly one bus to pick up passengers.*
2. *The Capacity of the kth bus is* Q_k, $k \in B$.
3. *The running cost of each bus is less than a pre-assigned D.*
4. *A bus is utilized if and only if it travels to at least one stop to pick up or drop passengers.*
5. *If a bus visits a stop, it must also exit the stop.*
6. *Every stop is visited by exactly one bus which does so immediately after visiting some other stop.*

4. Goal Programming Technique

We apply the technique in the following manner: The number of available buses is initially set to k = 1. If a feasible solution is found for k buses, we then optimize the second objective by any one of the heuristics given below. This will be the solution to the VRP. If we cannot find a feasible solution for k = M, then the problem is considered to be not solvable.

5. Heuristics to Minimize the Second Objective

A typical procedure to solve the VRP (for a given number of buses) uses the cluster-first/route-second approach ([3]). The Sweep heuristic is applied to obtain a feasible solution, and then "improvement" heuristics are run to minimize the costs.

Below we briefly describe the Sweep heuristic and three "improvement" heuristics. There are a number of variants in the first two "improvement" heuristics and we have presented one for each. These were the ones we have programmed. The third "improvement" heuristic, developed by us, is a combination of the first two "improvement" heuristics. All these "improvement" heuristics converge to a local minimum. Later we present another "improvement" heuristic based on Simulated Annealing which converges to a global minimum.

For easier explanation of the heuristics, we need the following concept. We regard each subroute as a **sequence** of stops (visited by the bus assigned to this subroute) which starts from the depot and ends at the depot; the order in the sequence is the order in which the stops are visited by the bus

(starting from bus stop 1). A subset of stops in a subroute can be ordered (as above) to form a **subsequence** of this subroute.

(H0) *The Sweep Heuristic* ([1], [3]): Polar coordinates are computed for all the stops. The nodes are then 'swept' into clusters from the smallest angle to the largest. Each cluster contains a set of nodes which satisfy the constraints. The procedure is as follows.

Step 1. Reorder the stops with polar coordinates (r_i, θ_i) such that $\theta_2 \leq \ldots \leq \theta_N$.

Step 2. Choose an unused bus k ($k \in B$).

Step 3. Starting from an unrouted stop i with the smallest angle, include consecutive stops in the route until the inclusion of the next stop would violate the capacity constraint. Insert stop 1 at the beginning and at the end of the route.

4. If all the stops have been "swept", stop. Else return to Step 2.

(Our computer program for (H0) has been coded exactly as above.)

(H1) *Two Optimality Heuristic* ([8], [11]): This heuristic involves cutting and rejoining links which connect stops in a subroute; this essentially permutes the stops in a subroute.

Let a and b, and u and v be pairs of adjacent stops in the present configuration of a subroute with $\{a, b, u, v\}$ being the subsequence of the subroute. The links ab and uv are deleted and the links au and bv are inserted. This forces $\{a, u, b, v\}$ to be the subsequence of the subroute. The cost of the new configuration is computed to see whether to replace the

previous configuration with the new one.

Subroutes are considered one at a time. The pairs of links are selected by some user-defined rule. If there is an improvement, the old subroute is replaced by the new permutation. This procedure is repeated until there is no improvement.

(We programmed an iteration of this heuristic in the following way. Two nodes were selected from the subroute under consideration. Considering the subroute as a sequence, the links which had these nodes as the end nodes were then replaced by the appropriate links as required above.)

(H2) *Single Point exchange Heuristic* ([1]): Pairs of stops, say u and v, with the stops being in different subroutes, say SR1 and SR2, are selected by some user-defined rule. In SR1 u is replaced by v, and in SR2 v is replaced by u. If the exchange is an improvement, then it is taken as a permanent exchange. This process is repeated until there is no further improvement.

(We programmed an iteration of (H2) just as described above.)

(H3) *2-Opt/One-Point-exchange Heuristic*: We randomly generate pairs of stops. If the stops are in the same subroute, then the permutation of (H1) is affected; otherwise the exchange of (H2) is affected. If the exchange is an improvement, then it is taken as the new configuration. This process is repeated until there is no further improvement.

(In our computer code, the pair of stops was generated by the random number generator of the computing system. The rest of the code is just as described in the procedure.)

6. Simulated Annealing with reference to VRP

We briefly describe the method of Simulated Annealing with respect to VRP. References [5] and [9] have the general theory. This will converge to a global minimum but the time to termination is not polynomially bounded; therefore for practical solutions, the maximum number of iterations is fixed.

For some years, interest has been shown in the use of simulated annealing to obtain "good" solutions to a number of combinatorial problems. The central idea of this method is that certain uphill steps may be required to prevent an optimization scheme from being stuck in a "poor" local minimum. In general, improvements are always accepted while uphill moves are accepted with a probability that depends on the size of the increment and a number of **controlling parameters**. For a particular problem, one must specify a **topology** (refer [10]) and a method of moving from one feasible solution to a (topologically) neighbouring one.

Let k be the number of buses that have to be run (refer Section 4). Let $\mathcal{X}$ be the set of all feasible solutions of the VRP with the following properties:

1. The number of subroutes in each solution equals k.
2. For any two solutions: for each subroute in one solution, there is a subroute in the other solution with the same number of stops.

(These conditions may appear stringent, but any series of solutions generated by first applying the Sweep Heuristic followed by iterations of any of the improvement heuristics described above has these properties.)

If $f: \mathcal{X} \to \mathbb{R}$ denotes the second objective function, then

the problem is to minimize f. Let $\mathcal{T}$ be any topology on $\mathcal{X}$ with the following essential condition:

Given any two points, x0 and x in $\mathcal{X}$, we can construct a sequence x0, x1, x2, ..., x, where any two adjacent points in the sequence are neighbours in the topological sense.

Let $\mathcal{M}$ be the maximum number of iterations. Let t_o and t_f be pre-determined control parameters and let $\mathcal{B} = (t_o - t_f)/(\mathcal{M} * t_o * t_f)$. Let s_o be any initial solution. At the ith step, let s_i be the current solution, let t_i be the corresponding control parameter, and let $f_i = f(s_i)$. The heuristic may be stated as follows.

Repeat until $t_i \leq t_f$

1. Generate a neighbour s_p of the current solution. If f_p is an improvement or if $\exp((f_i - f_p)/c_i) > 0.5$,

$$s_{i+1} = s_p \text{ else } s_{i+1} = s_i.$$

2. $t_{i+1} = t_i - dt_i$, where $dt_i = t_i/(1 + \mathcal{B} * t_i)$.

The sequence of solutions is generated by a Markov process with a transition matrix that is irreducible. If, in an iteration, the probability of every neighbour being chosen as a candidate is the same, then it can be shown that the method can be made to terminate arbitrarily close to the global optimum with probability arbitrarily close to 1. Unfortunately, the time to termination is not polynomially bounded.

7. A Heuristic based on Simulated Annealing

As noted in the previous section, to apply Simulated Annealing, we need to define a topology on $\mathcal{X}$ (the solution space as defined earlier), a method to generate a neighbour of any point in $\mathcal{X}$ and a method to generate the initial solution.

We define a **topology** on $\mathcal{X}$ as follows. Two points in $\mathcal{X}$ are said to be neighbours if one can be generated from the other by either (1) permuting stops in some subroute of one as in (H1) or (2) by the single point exchange (as in (H2)). It can be verified that this topology has the essential condition mentioned in Section 6.

Therefore we generate a neighbour of a feasible solution by the method stated above, and an initial solution is generated by the Sweep Heuristic.

(H4) *2-Opt/One-Point-exchangewith Simulated Annealing:*

* Let K ($1 \leq K \leq M$) be the maximum number of possible routes. (Refer Section 4.)

*. Let $\mathcal{N}$, t_0 and t_f be fixed. Calculate $\mathcal{B}$.

* Generate the initial solution, s_0 by the sweep heuristic.

* Let s_i be an intermediate solution, t_i be the corresponding control parameter and let $f_i = f(s_i)$.

* To perturb s_i, we randomly select two stops; if the stops are in different subroutes, they are exchanged as in (H2), else they are exchanged as in (H1). Denote this neighbour by s_p.

* If $f_p = f(s_p)$ is an improvement or if $\exp((f_i - f_p)/c_i) > 0.5$,

$$s_{i+1} = s_p \quad \text{else} \quad s_{i+1} = s_i.$$

* $t_{i+1} = t_i - dt_i$, where $dt_i = t_i/(1 + \mathcal{B} * t_i)$.

The last three steps are iterated $\mathcal{N}$ times.

(As in (H3), our code use the random generator of the computing system to generate the pair of nodes. We then followed the procedure detailed above.)

Since every neighbour has equal chances of being selected. this method can be made to terminate arbitrarily close to the global optimum.

8. Experiment and Results

The VRP of a public sector company has been modelled for solution. The data were obtained from [1]. The number of stops is 327 with the first stop as the factory; the number of available buses was 30 and the capacity of each bus was taken to be 55. The coordinates of each stop were available and we defined the cost of going from one stop to another to be the Euclidean distance between the two.

We wished to compare the four heuristics (H1), (H2), (H3) and (H4). We kept the number of buses that had to be run fixed at 28 for all four heuristics.

In any iteration of any of these heuristics, a pair of stops has to be selected. Since, for (H3) and (H4), the next pair of stops needs to be chosen randomly, we decided to do the same with the other heuristics. We used the random number generator of the computing system to generate the same sequence of pairs of nodes for each heuristic. We generated three sequences of random pairs using the seeds 136, 188 and 1122.

Since we cannot predict when a heuristic will converge to a local or global optimum, we decided to compare the solutions reached by the various heuristics at different stages of the computations. The stages were defined by the expression **C*factor**, where C is the total number of possible pairs that could be selected (for the particular heuristic) in an iteration, and factor was assigned pre-determined numerical values. The solutions were examined when all the heuristics had completed C*factor iterations (for different values of factor). Thus the number of iterations executed at the end of any stage was:

(H1): (n-1)*(n-2)/2*factor (for each subroute), where n is the number of stops in that subroute;

(H2): (N-1)*(N-2)/2*factor;

(H3): (N-1)*(N-2)/2*factor;

(H4): (N-1)*(N-2)/2*factor.

Although we did not keep track of the time taken by each heuristic, it is clear that the time taken for a particular heuristic is (almost) proprtional to the total number of iterations executed. This implies that the time taken for (H1) is significantly lower than the others. In (H2), if the stops generated are from the subroute then it is ignored and the count (of the number of iterations) is increased by one; but in (H3) this pair will be considered. Therefore (H3) will take more time than (H2). (H4) will take longer than (H3) since configurations that are rejected by (H3) may be accepted by (H4) if the increment is small enough. But the difference in the times will not be significantly different.

The code was written in C and the program was run on the VAX 3300.

The value of the second objective function after the execution of the Sweep heuristic was **4888.2 Km.** This is the value of the initial solution for all the heuristics. Table 1 below contains the computational results. The data in the last four columns are the values of the second objective function for the corresponding heuristics at different stages of the computation (given by the first column) and for different sequences of random pairs generated by the seeds shown in the second column.

TABLE 1

Factor	Seed	H1	H2	H3	H4
1	136	3237.6	3067.0	2828.9	3356.4
1	188	3282.4	3019.9	2789.6	3072.9
1	1122	3202.2	3250.5	2737.4	3330.7
4	136	3036.4	2840.4	2613.2	2745.6
4	188	3038.0	2698.9	2554.3	2787.3
4	1122	3036.3	2961.4	2560.3	2986.9
6	136	3035.5	2839.1	2611.1	2787.6
6	188	3038.0	2651.2	2528.9	2827.8
6	1122	3036.2	2929.9	2545.7	2609.9
10	136	3035.2	2836.3	2611.1	2576.5
10	188	3038.0	2650.8	2525.8	2559.4
10	1122	3036.1	2885.3	2545.7	2558.8
14	136	3035.0	2836.3	2611.1	2466.9
14	188	3038.0	2650.8	2525.8	2460.4
14	1122	3036.1	2882.0	2545.7	2532.9
16	136	3035.0	2836.3	2611.1	2402.9
16	188	3038.0	2650.8	2525.8	2432.6
16	1122	3036.1	2882.0	2545.7	2419.0
20	136	3035.0	2836.3	2611.1	2426.5
20	188	3038.0	2650.8	2525.8	2351.9
20	1122	3036.1	2882.0	2545.7	2445.4
22	136	3035.0	2836.3	2611.1	2372.3
22	188	3038.0	2650.8	2525.8	2439.3
22	1122	3036.1	2882.0	2545.7	2390.3
28	136	3035.0	2836.3	2611.1	2335.5
28	188	3038.0	2650.8	2525.8	2396.0
28	1122	3036.1	2882.0	2545.7	2358.9
30	136	3035.0	2836.3	2611.1	2349.2
30	188	3038.0	2650.8	2525.8	2401.0
30	1122	3036.1	2882.2	2545.7	2323.3

10. Conclusions

The investigations are at a preliminary stage. The heuristics must be run on more test data before firmer conclusions can be drawn. At this present juncture, we may say the following:

1. As far as convergence is concerned, H3 converges the fastest, followed by H1 and H2. From the last column of Table 1, it can be seen that H4 had not converged. If an organization has to compute the routes on a daily basis, then H3 appears to be more appropriate.

2. Although H4 did not converge, it gives better results if enough iterations are executed. Therefore, if the computations are to be done for a schedule which will not be changed frequently, H4 gives a better result.

Reference

1. Computer Aided Management Centre, IIM, Bangalore Bus Utilization System Evaluation Software, Project Report No. 24, Indian Institute of Management, Bangalore (1987).

2. M.L. Fisher and R. Jaikumar, "A Generalized Assignment Heuristic for Vehicle Routing", Network 11 (1981), 109-124.

3. B. Gillet and L. Miller, "A heuristic algorithm for the vehicle despatch problem", OR 22 (1974), 340-349.

4. A.T. Kearney, Inc., Measuring and Improving Productivity in Physical Distribution, A report prepared for National Council of Physical Distribution Management, Oakbrook, Illinois, USA (1983).

5. S. Kirkpatrick, C.D. Gelatt, Jr. and M.P. Vecchi, "Optimization by Simulated Annealing", Science 220 (1983), 671-680.

6. E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys, The Travelling Salesman Problem, John Wiley and Sons Ltd. (1985).

7. J.K. Lenstra and A.H.G. Rinnooy Kan, "Complexity of Vehicle Routing and Scheduling Problems", Networks 11 (1981) 221-227.

8. S. Lin, "Computer Solutions to the Travelling Salesman Problem", Bell System Technical Journal 44 No. 10 (1965), 2245-2269.

9. M. Lundy and A. Mees, "Convergence of an Annealing Algorithm", Mathematical Programming 34 (1986) 111-124.

10. J.R. Munkres, Topology: A First Course, Prentice-Hall, Englewood Cliffs, New Jersey, USA (1975).

11. C.H. Papadimitriou and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Englewood Cliffs New Jersey, USA (1982).