Vehicle Routing Problem -
Practical Implementations using OSRM

Madhav Mittal[1]
Guide : Prof. N.S. Narayanaswamy[2]

Abstract
The vehicle routing problem is a well-studied problem with many proposed solutions. In this project, we study the different variants of the vehicle routing problem - CVRP, VRPPD, MDVRP. We also look at applying some available solution to get practical implementations for deliveries in the real world. We make use of the Open Source Routing Machine engine for this practical implementation. During this study, a lack of standardised data that is based on the real world was also felt. Thus, we have tried to propose an easy way of generating data for testing the practical implementation.

[1] *CS19B029, Department of Computer Science and Engineering, IIT Madras*
[2] *Department of Computer Science and Engineering, IIT Madras*

Contents

Introduction

Today, the logistics sector represents roughly five percent of India's Gross Domestic Product and is estimated to employ 22 million people. Also, logistics cost are high in India, going up to 14 percent as a share of GDP. Inefficient use of road transportation is a major contributor to this poor performance of logistics in India as it comprises 71 percent of all freight. For example, the trucks here have empty running rates as high as 40 percent. All of this also leads to higher emissions that exacerbate the climate crisis [1].
All of these seem to be contemporary issues of modern India, but these are fundamental problems that all parts of the world have been facing from a long time. Such problems have led to the development of the field of Operations Research. Thus, many problems have been formulated mathematically and have been worked upon to help resolve the issues faced by the logistics sector.
One such problem that is discussed widely is the Vehicle Routing Problem (VRP). It was first described by Dantzig and Ramser as the "Truck Dispatching Problem" published in 1959. They described it as a generalisation of the Travelling Salesman Problem [2]. Clarke and Wright worked on this formulation with some new restrictions to develop a novel solution in their paper "Scheduling of vehicles from a central depot to a number of delivery points." published in 1964 [3]. The "savings algorithm" proposed by them then is considered one of the best approaches to VRP known till date.
In due course of time, as many as 50[1] varieties of the Vehicle Routing Problem have been described with different constraints and objectives. These aim to address different issues of real world logistics. Capacitated Vehicle Routing Problem (CVRP) is VRP with the additional constraint that vehicles have a fixed capacity. VRP with Pickups and Deliveries (VRPPD) has different pickups and deliveries for the consignments instead of a central depot from which all consignments are picked up. Multiple Depot VRP (MDVRP) has multiple depots from

[1] http://www.vrp-rep.org/variants.html

which deliveries can be fulfilled.
Data for testing is another important aspect of the field of research in VRP just like any other. Multiple standardised datasets have been proposed and developed in literature through the years, specially for CVRP [2]. There is also a renewed interest in generating a high volume of datasets to train and test machine learning models [4] to solve various VRPs. However, most of these datasets are for Euclidean models of the problem and not based on the real world. This presents a scope to utilise the existing datasets to generate new datasets based in the real world. This is another field of study that deals with Embeddings. We have looked at a more experimental approach instead of theoretical for achieving this.

1. Domain Knowledge Representation

Following domain knowledge representation can be used for different VRP formulations:

Data Entities :

{ common entities for VRP are given irrespective of variant }

- Vehicle : sometimes called carrier or container.
  - Unique identifier - id
  - Physical properties - capacity (numerical), volume (numerical or set of numerical dimensions)
  - Configuration in some state - free_cap, free_vol
  - Assigned Consignments - list of consignments
  - Assigned Route - route
  - Location at current instant - loc
  - Constraints - fuel left, serviceable area, assigned depot, refrigeration etc.
- Consignment : sometimes called package or order.
  - Unique identifier - id
  - Physical properties - weight, volume
  - Pickup location - pickup (or depot depending on variant)
  - Dropoff location - drop
  - Pickup window - ptime
  - Dropoff window - dtime
  - Other constraints - stackability, temperature sensitivity etc.
- Depot : sometimes called warehouse.
  - Unique identifier - id
  - Location - loc
  - Assigned vehicle(s) - vehicle
- Route : ordered set of stops for a vehicle.
- Location : position coordinates in some system (like geographical or euclidean)
- **Distance function** : dist(a, b) - gives (euclidean/ road/ etc.) distance from point a to point b
- **Time function** : time(a, b) - gives time taken to travel from point a to b

Input :

- C : list of consignments.
- V : list of vehicles.
- D : list of depots (if required in the problem)

Output :

- Mapping of vehicles to consignments.
- Routing of vehicles such that the objectives are satisfied (not necessarily exact).

Objectives :

{ satisfy the constraints of the individual data entities and one or more of the following }

- Minimize distance traveled by each vehicle.
- Minimize total distance to be traveled by the entire fleet.
- Minimize time taken for each vehicle to complete all its deliveries.
- Minimize average time overrun for all deliveries.

Predicates :

{ represent relations }

- $running(v)$ - true when vehicle v is running, false if idle
- $allocated(v,c)$ - consignment c is allocated to vehicle v

[2] http://vrp-rep.org/datasets.html

- $volume\_available(v,c)$ - vehicle v has enough free_vol left for consignment c
- $capacity\_available(v,c)$ - vehicle v has enough free_cap left for consignment c
- $route\_points(r,a,b)$ - point a occurs before point b in route r
- $route\_contains(r,a)$ - route r contains point a

Actions :

{ represent changes in the state of the problem (p_i are parameters) }

- Allocate(vehicle v, consignment c)
  - Precondition : $volume\_available(v,c) \wedge capacity\_available(v,c) \wedge (\neg allocated(v,c))$
  - Effect : $allocated(v,c)$
  - Cost : $p_1 * c.volume + p_2 * c.weight + p_3$
- Deallocate(vehicle v, consignment c)
  - Precondition : $allocated(v,c)$
  - Effect : $\neg allocated(v,c)$
  - Cost : $p_1 * c.volume + p_2 * c.weight + p_3$
- Move(vehicle v, point loc)
  - Precondition : $location(v,loc_1) \wedge running(v)$
  - Effect : $location(v,loc)$
  - Cost : $p_1 * time(loc_1,loc) + p_2 * dist(loc_1,loc) + p_3$
- Add(route r, point x)
  - Precondition : $\neg route\_contains(r,x)$
  - Effect : $route\_contains(r,x)$
  - Cost : $p_1 * (time(a,x) + time(x,b) - time(a,b)) + p_2 * (dist(a,x) + dist(x,b) - dist(a,b)) + p_3$
- Merge(route r1, route r2)
  - Precondition : $\forall_{x_2 \in r_2} \neg route\_contains(r_1,x_2) \wedge \forall_{x_1 \in r_1} \neg route\_contains(r_2,x_1)$
  - Effect : $\forall_{x \in r_1 \cup r_2} route\_contains(r_3,x)$ where r_3 is new generated route
  - Cost : $p_1 * (\text{length of } r_3) + p_2 * (\text{travel time of } r_3) + p_3$

.

Temporal Constraints :

{validity depends on variant of VRP}

- Pickup/depot for a consignment should be visited before the dropoff.
- Consignment should be allocated to a vehicle before points associated with it are added to a route of the vehicle.
- The pickup window of a consignment should be before its dropoff window.
- If a consignment is perishable, its dropoff should be before its perishing time.

2. Implementation

2.1 VRPPD

Vehicle Routing Problem with Pickups and Deliveries VRP where different deliveries are to be picked up from different locations.

Problem Description

Given a vehicle v located at loc and n consignments c_j with pickup points P_j $(j = 1 \ldots n)$ and delivery points D_j $(j = 1 \ldots n)$, route the vehicle to minimize the total distance covered by the vehicle.

Algorithm

As pickups must necessarily be done before delivery, simply using the savings algorithm to get the route over all points won't work without imposing this constraint. The following three methods to deal with the above situation were tested :

1. Use the savings algorithm separately on the pickup points and delivery points. That is, first pickup all the consignments and then deliver all of them.
2. Visit the nearest neighbour from current point in consideration. We start with considering all the pickups as the set of points to chose from. When a pickup point is visited, add its corresponding delivery point to the set of points.
3. Use the savings algorithm on the pickup points to generate a route. Then, starting with the last pickup in this route, insert the corresponding delivery point in the route where it causes least detour after its pickup. Similarly, insert all the corresponding deliveries into the route from upto the first pickup.

2.2 MDVRP

Multiple Depot Vehicle Routing Problem
VRP where there are multiple depots from which the deliveries can be serviced.

Problem Description

Given m depots d_i located at loc_i each having a vehicle v_i with capacity cap_i assigned and n consignments c_j with delivery points P_j $(j = 1 \dots n)$, find routes to minimize the total distance covered by all the vehicles.

Algorithm

We implemented a two phase solution to this problem.

Phase I Allocate the consignments to the depots. We do this based on the proximity of depots to the pickup points.

Algorithm 1: MDVRP : Phase I

Input: list of depots D, list of consignments C
Output: list of depots with assigned consignments

```
for c in C do
    d = depot nearest to c
    while c not allocated do
        if c can be allocated to d (constraints) then
            allocate c to d
        else
            d =next nearest depot to c
```

Phase II Now, the problem has been reduced to CVRP where each vehicle v_i with capacity cap_i has to service all the deliveries allocated to its assigned depot d_i. Hence we use Algorithm 2 to find all the routes to be travelled.

2.3 CVRP

Capacitated Vehicle Routing Problem.
VRP where vehicles have a finite capacity in terms of weight, volume, number of consignments, etc.

Problem Description

Given a vehicle v with capacity cap and n consignments c_i with delivery points P_i $(i = 1 \dots n)$ and a depot D where all the consignments are sourced and where the vehicle starts and must end, find a list of routes to minimize the total distance covered by the vehicle.

Algorithm

The savings algorithm given by Clarke and Wright is used [3]. In this algorithm, a heuristic approach is taken to greedily merge pairs of routes with the highest "savings" i.e. the decrease in distance to be travelled if two routes are merged. While merging the routes, the constraints such as vehicle capacity are checked i.e. two routes are merged only if the vehicle has the capacity to transport all the corresponding consignments. The merging of the routes is done at the ends only.

Algorithm 2: CVRP : savings algorithm

Input: integer cap, list c, list P, location D
Output: list of routes r

```
savings = [ ]
for P_i in P do
    for P_j in P do
        if saving > 0 then
            saving = dist(P_i, D) + dist(D, P_j) − dist(P_i, P_j)
            savings.append([saving, P_i, P_j])
sort savings in decreasing order considering the first element of each value as key
for [saving, P_i, P_j] in savings do
    r_k = route in r containing P_i
    r_m = route in r containing P_j
    if r_k == None and r_m == None and c_i + c_j ≤ cap then
        r.append([P_i, P_j])    // add new route
    else if P_i is last stop in r_k and r_m == None and weight(r_k)+c_j ≤ cap then
        r_k.append(P_j)
    else if r_k == None and P_j is first stop in r_m and weight(r_m)+c_i ≤ cap then
        r_m.append(P_i)
    else if P_i is last stop in r_k and P_j is first stop in r_m and weight(r_k)+weight(r_k)≤ cap then
        r.remove(r_m)
        r.append(r_k.extend(r_m))
for r_i in r do
    add D at start and end of r_i
return r
```

3. Dataset generation

> Water, water, every where,
> Nor any drop to drink.
>
> *The Rime of the Ancient Mariner*
> Samuel Taylor Coleridge

During the course of this project, a lack of standardised data with benchmarks for the real world was felt. On the other hand, there is a large number of datasets available for the 2 dimensional Euclidean system with euclidean distances as the edge costs. Such datasets like those proposed by Augerat in 1995 [5] or those that were proposed by Uchoa et.al. in 2017 [6] are used extensively. Hence, there is scope for novel ways of generating artificial test data for real world VRP problems. That too, deriving from standard datasets like for CVRP that are studied widely in literature.
Our objective is to get a set of points on the Earth's surface with road (or some other means of travel) distances (or time) as edge weights. Generating street address instead of just geographical coordinates can be an added layer of utility to test real world applications.
We start with looking at a certain geographical region like a country or city. In our case, we look at the city of Chennai.

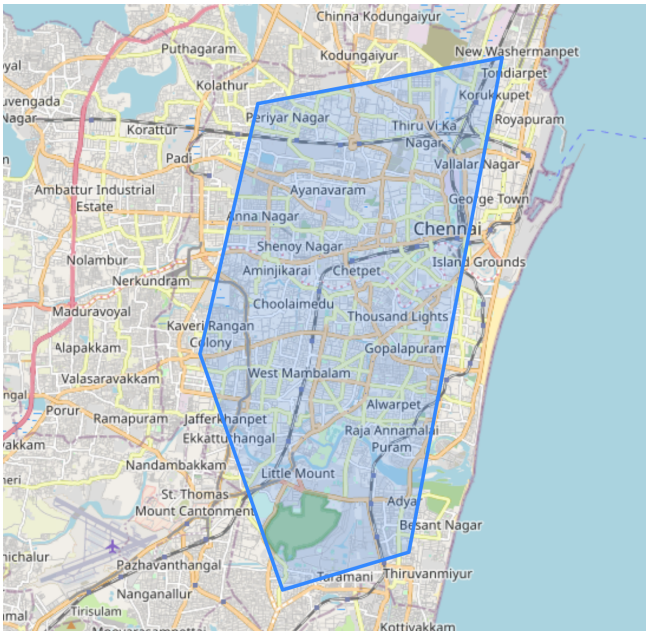

Figure 1. A Polygon enclosed within Chennai boundaries.

The following experimental process was used to embed euclidean dataset into the geographical one:

1. Consider a polygon P contained within the boundaries of the geographical region like fig. 1.
2. Find the centroid of the polygon C. We consider this as the center of the set of points and homologous to the center of the euclidean dataset.
3. Compute the unit "as the crow flies" distance at C along the latitudes as D_{lat} and along longitudes as D_{lon}.
4. Now, considering a linear relationship between the longitude and D_{lat} and between the latitude and D_{lon}, transform the euclidean coordinates to the geographical coordinates such that the set of points lies completely within P.
5. If street addresses are required, use the Reverse API of Nominatim [3] on the coordinates generated in the previous step with suitable setting of *max zoom*.

For example, consider the Set A-n32-k5 by Augerat[4] as visualised in fig. 2. After using the above process, the set of geographical points obtained was as depicted in fig. 3.

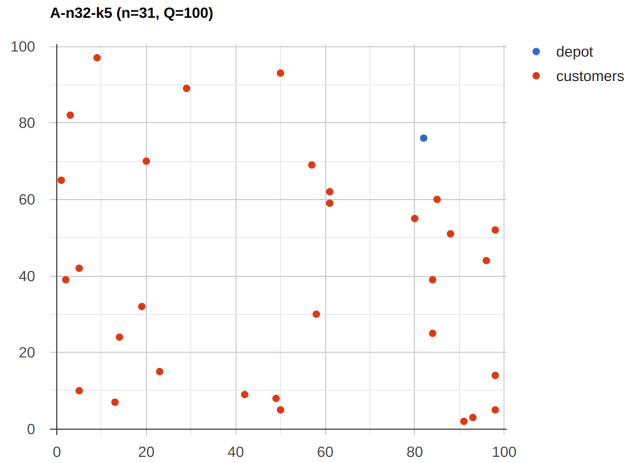

Figure 2. Set A-n32-k5 in euclidean plane

To analyse the quality of this embedding, we first consider that the lines of longitude and latitude are close to straight lines on a 2D plane. This is more accurate

[3] https://nominatim.openstreetmap.org/ui/reverse.html
[4] http://vrp.atd-lab.inf.puc-rio.br/index.php/en/plotted-instances?data=A-n32-k5

Figure 3. Set A-n32-k5 embedded in Chennai

Figure 4. Lines of latitude and longitude over Chennai.

in the regions along the equator, and for smaller areas. Thus, as seen in fig. 4, it is a good approximation for the city of Chennai.

Now, we find the road distances between all the points using OSRM. We scale the original euclidean distances by a factor such that the mean of these scaled distances and the mean of the road distances coincide. Then, we find the root mean squared difference of all the pairwise-distances. Then, we compute it as a percentage of mean of the scaled euclidean distances (same as mean of road distances). Some of the results on datasets by Augerat are presented in Table 1.

Table 1. difference between embedded and original

Dataset	mean of distances	% rms difference
A-n32-k5	3868	12.9
A-n45-k6	3780	12.8
A-n64-k9	3151	15.7
A-n80-k10	3409	13.4

Considering the fact that roads are not well-snapped to the coordinate system in Chennai, this level of variation in the distances seem reasonable. However, more rigorous mathematical analysis is required to find a better metric to judge the quality of this embedding.

4. Analysis

Time complexity

As written by Dantzig and Ramser, the VRP is a generalization of the Travelling Salesman Problem (TSP). That is, TSP can be reduced to a VRP in linear time. For example, TSP can be reduced to CVRP where the vehicle has infinite capacity. Thus, VRP is a NP-hard problem. So, the time to find an optimal solution to a VRP using existing algorithms grows superpolynomially with respect to the number of deliveries [2][3].

On the other hand, heuristic algorithms that give near optimal solutions (as opposed to exact solution) can be run in polynomial time. The savings algorithm is one such heuristic. The bottleneck step in the savings algorithm described in Algorithm 2 is sorting the savings, so it takes $O(|savings| \log |savings|) = O(n^2 \log n^2)$ time where n is number of deliveries.

Similarly, for n consignments in VRPPD, approach 1 takes $O(n^2 \log n^2)$ time. Approach 2 can be computed in $O(n^2 \log n)$ time using a minheap/priority queue. Approach 3 takes $O(n^2 \log n^2)$ time for pickup route generation and $O(n)$ time for each minimum detour insertion of delivery points i.e. total $O(n^2 \log n^2)$ time.

For the MDVRP, considering m depots and n consignments, Phase I i.e. Algorithm 1 takes $O(n.m^2)$ time. Phase II takes time to the order of $\sum_{i \in [1,m]} n_i^2 \log n_i^2 \leq n^2 \log n^2$ i.e. $O(n^2 \log n^2)$ time. Hence, total $O(n.m^2 + n^2 \log n^2)$ time.

Quality of solution

For testing the VRPPD and MDVRP implementations, artificially generated test data based on street addresses in Chennai was used. As this was a randomly generated dataset, we couldn't analyse the quality of the solution. However, relative analysis of the quality of the 3 approaches to VRPPD is possible to some extent.

Table 2. VRPPD : cost of 3 different approaches

case	# cons	1	2	3
1	3	80.64	83.78	80.64
2	7	77.57	114.32	77.01
3	8	71.41	77.89	71.89
4	30	209.72	235.17	193.11

Table 2 contains the total distance for the route generated by each of the approaches in kilometers. We can see that approach 2 generally performs poorly in comparison to the other two approaches. On the other hand, approach 3 generally gives the best performance excpet for one case (3) where it performs slightly worse than approach 1.

5. Further Work

- Comparative studies on other variants of VRP can be undertaken in a similar fashion. Using available solutions to one variant of VRP, new solutions can be devised for other variants also.
- The performance of the method of embedding to generate data can be compared for different geographical entities. Different sets of data such as the CVRP data can also be utilised or combined to generate data for other variants of VRP such as MDVRP or VRPPD.
- Similar ways of generating data can also be explored for the rescheduling and orienteering problems.

Acknowledgments

I am thankful to my guide, Prof. N.S. Narayanaswamy, for all his time and guidance and for always being available to help me. I also thank the Department of Computer Science and Engineering, IIT Madras for allowing undergrad students like me to experience research first-hand through UGRC. Thanks also to Prof. Shweta Agrawal and Prof. Chester Rebeiro for coordinating this offering of UGRC.
I also owe gratitude to all the wonderful people working at the Logistics Innovation Lab @ IITM led by Anuj. Thanks to my friends Hakesh, Hariharan and Ankit who were researching along with me. Special thanks to the members of my team at the lab - Jayakamal, Hemanth, Shashwathy, Sreya and Yogasanthoshi.

References

[1] NITI Aayog RMI India and RMI. Fast Tracking Freight in India: A roadmap for clean and cost-effective goods transport. Technical report, June 2021.

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

[3] G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, 12(4):568–581, 1964.

[4] Eduardo Queiroga, Ruslan Sadykov, Eduardo Uchoa, and Thibaut Vidal. 10,000 optimal CVRP solutions for testing machine learning based heuristics. In *AAAI-22 Workshop on Machine Learning for Operations Research (ML4OR)*, 2022.

[5] Philippe Augerat. Approche polyèdrale du problème de tournées de véhicules. (polyhedral approach of the vehicle routing problem). 1995.

[6] Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, and Anand Subramanian. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858, 2017.