

Capacitated Electric Vehicle Routing

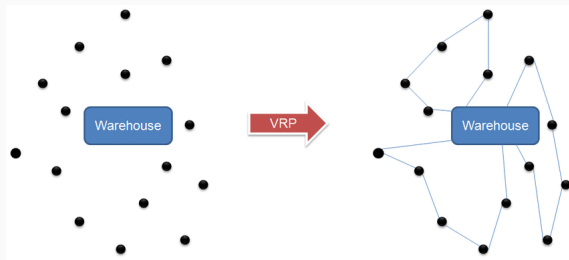
Andrew Struthers

June 1, 2023

Background

The Electric Vehicle Routing Problem (EVRP) is a complex combinatorial optimization problem that builds off of the Vehicle Routing Problem (VRP) and the Capacitated Vehicle Routing Program (CVRP). The EVRP seeks to find the optimal route plan for a fleet of capacitated vehicles that satisfy battery-related restrictions.

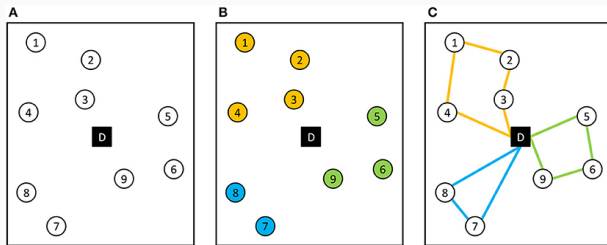
Vehicle Routing Problem



VRP

- Determines the optimal set of routes a fleet of vehicles can traverse to reach every customer
- Generalization of the Traveling Salesman Problem
- NP-Hard problem

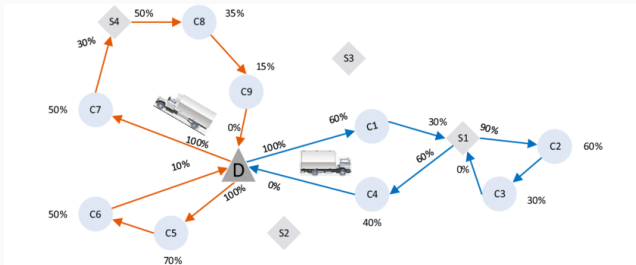
Capacitated Vehicle Routing Problem



CVRP

- Determines the optimal set of routes (subtours) a fleet of vehicles can traverse to reach every customer
- Each customer has a certain demand, and each vehicle has a set capacity
- Generalization of TSP and the Bin Packing Problem

Electric Vehicle Routing Problem



EVRP

- Vehicles are capacitated and have electric engines, instead of diesel
- EV charging stations are not as common as gas stations
- Instead of fuel capacity and nominal fueling time, we care about battery capacity and lengthy recharging times

The primary source of information I used for this project came from “The Electric Vehicle Routing Problem and its Variations: A Literature Review”, published in “Computers & Industrial Engineering” by Ilker Kucukoglu, Reginald Dewil, and Dirk Cattrysse[1]

This source compiles 136 published papers that consider the routing of battery electric vehicles. We will also be using the same dataset as the one discussed. We will be focusing on the most basic version of the EVRP.

Objective Function

$$\text{Min } z = \sum_{i \in V'_0} \sum_{j \in V'_{N+1}} \sum_{k \in K} d_{ij} x_{ij}^k$$

This objective function tells us that we want to minimize the total distance of the electric vehicles. Of course, no good objective function would be complete without some constraints.

Subject To

$$\sum_{j \in V_{N+1}} \sum_{k \in K} x_{ij}^k = 1 \quad \forall i \in V \quad (1)$$

$$\sum_{j \in V_{N+1}} \sum_{k \in K} x_{ij}^k \leq 1 \quad \forall i \in F' \quad (2)$$

$$\sum_{j \in V'} x_{0j}^k \leq 1 \quad \forall k \in K \quad (3)$$

$$\sum_{i \in V'_0} x_{ij}^k = \sum_{i \in V'_{N+1}} x_{ji}^k \quad \forall j \in V', \forall k \in K \quad (4)$$

$$y_j^k \leq y_i^k - (h \cdot d_{ij}) x_{ij}^k + Q(1 - x_{ij}^k) \quad \forall i \in V, \forall j \in V'_{N+1}, \forall k \in K \quad (5)$$

EVRP Mathematical Model

This looks quite complicated in math terms, but in English:

- (1) handles the connectivity of the customer nodes
- (2) ensures that each charging station can be visited at most once
- (3) makes sure that each vehicle can only use one route plan
- (4) ensures the total number of outgoing routes and incoming routes are equal at each customer and charging station node
- (5) is the beginning of the constraints that track the battery level. There are many constraints for this, including battery state after recharging

There are many more possible constraint equations according to the literature review, 22 to be precise, but these constraints are the required equations for the basic EVRP model.

One of the most common ways to solve this problem, from the 136 sources the literature review presented, was using Adaptive Large Neighborhood Search (ALNS), which is an algorithm that comes from the field of combinatorial optimization. This is typically a good way to solve the TSP.

Some other common solutions were using Ant Colony Optimization, Genetic Algorithms and Tabu Search, or some variation of Tabu such as Adaptive Tabu Search or Greedy Tabu Search. Tabu is a metaheuristic algorithm that allows exploring search spaces of solutions efficiently by preventing the revisitation of already explored solutions.

Attempted

- Brute Force
- Ant Colony Optimization (ACO)
- Adaptive Large Neighborhood Search (ALNS)

Attempted

- Brute Force
- Ant Colony Optimization (ACO)
- Adaptive Large Neighborhood Search (ALNS)

Implemented

- Genetic Algorithm, with genome sequence as list of customer nodes visited
 - For example, $\{5, 3, 1, 4, 2\}$
- C++, standard libraries only

Dataset

StringID	Type	x	y	demand
D0	d	40.0	50.0	0.0
S0	f	40.0	50.0	0.0
S1	f	77.0	52.0	0.0
S3	f	57.0	82.0	0.0
S16	f	48.0	8.0	0.0
S20	f	93.0	43.0	0.0
C98	c	58.0	75.0	20.0
C78	c	88.0	35.0	20.0
C4	c	42.0	68.0	10.0
C13	c	22.0	75.0	30.0
C95	c	62.0	80.0	30.0
C100	c	55.0	85.0	20.0
C54	c	42.0	10.0	40.0
C27	c	23.0	52.0	10.0
C89	c	63.0	58.0	10.0
C96	c	60.0	80.0	10.0

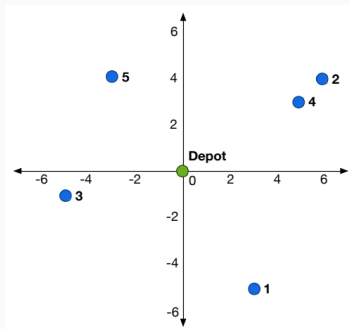
Figure 1: Type *d* represents the depot, aka starting point. Type *f* represents a charging station. Type *c* represents a customer

There are many datasets from a common database of example EVRP situations. The data shown in Figure 1 is data generated a clustered customer distribution method, with the condition that one charging station is always located at the depot, and the remaining charging stations are located randomly with the assumption that each customer can be reached from the depot using at most two charging stations. There are also datasets that include randomly distributed customer locations, as well as a mix of clustered and random distribution. Larger datasets included up to 100 customers.

Proof of Concept - CVRP

Before attempting the EVRP, I first prototyped a GA that solved the CVRP on a small example. In this case, the capacity of the vehicle is 10 units. We want to minimize the number of subtours needed to service each node, and we want to minimize the distance traveled in those subtours.

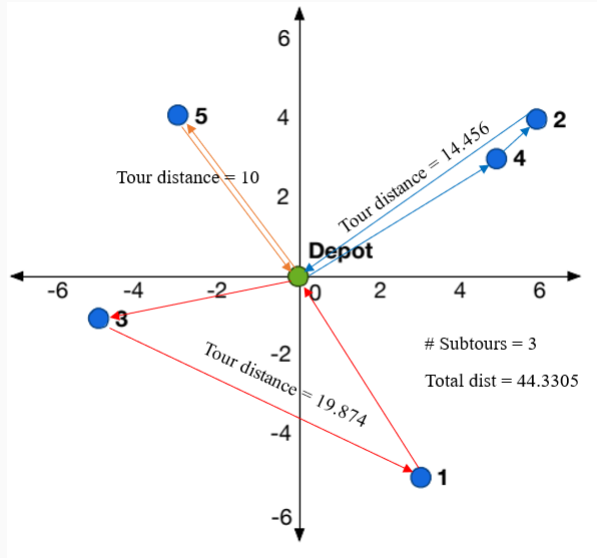
Tour	x	y	Demand
1	3	-5	2
2	6	4	6
3	-5	-1	8
4	5	3	4
5	-3	4	6



Proof of Concept - CVRP

```
Best tour: Tour: 2 4 5 3 1
Fitness calculations:
We have 10 remaining supply, and node 2 has demand 6
    After servicing this node, we now have 4 remaining supply
We have 4 remaining supply, and node 4 has demand 4
    After servicing this node, we now have 0 remaining supply
We have 0 remaining supply, and node 5 has demand 6
    Before servicing this node, we must go back to the depot. We have visited the depot 1 times
    After servicing this node, we now have 4 remaining supply
We have 4 remaining supply, and node 3 has demand 8
    Before servicing this node, we must go back to the depot. We have visited the depot 2 times
    After servicing this node, we now have 2 remaining supply
We have 2 remaining supply, and node 1 has demand 2
    After servicing this node, we now have 0 remaining supply
Returning to the depot at the end of the tour
=====
Tour: 0 2 4 0 5 0 3 1 0
There are 3 subtours in this route, with a total distance of this route is: 44.3305
```

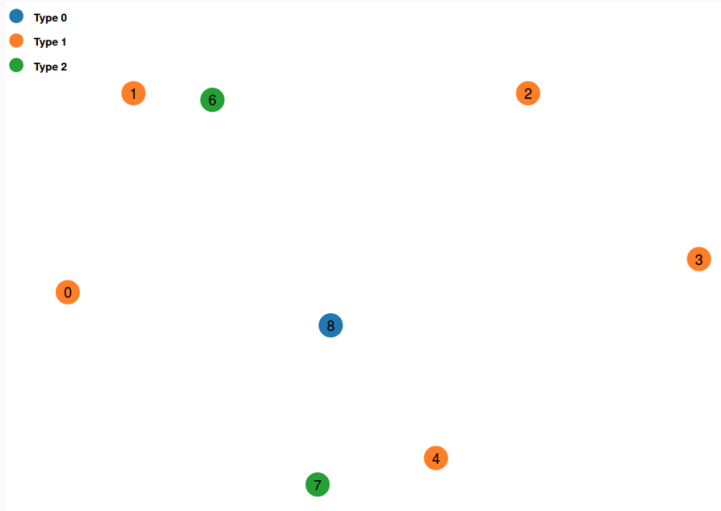

Proof of Concept - CVRP



Now that the proof of concept for the CVRP is complete, we can add more constraints such as battery and charging station nodes to the problem.

The most widely used database of EVRP sample problems contains problems with 5 customers, 10 customers, and 100 customers. I used problem C101_5 for proof of concept. The name of the problem implies there is a clustered distribution of 5 customers.

Proof of Concept - EVRP



I wrote a Genetic Algorithm to find the optimal tour through all customer nodes, charging at the charging station nodes when necessary. Because the charging station nodes are visited in an order that is dependent on the tour through the customer nodes, it doesn't make sense to do crossover or mutation with those nodes. Thus, the genome sequence for the GA was purely a vector of customer node indices.

In order to still capture all of the constraints, the fitness calculation function is heavily modified and contains the “core” of the optimization problem.

EVRP Algorithm

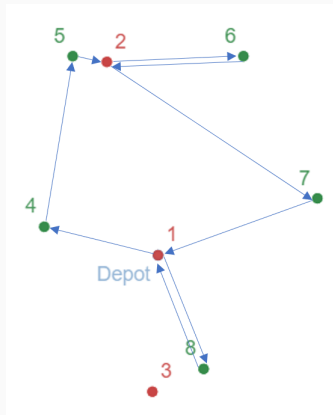
The basic fitness calculation algorithm is as follows:

- 1 Start at the depot (index 0)
- 2 Set the battery capacity and the carry capacity to the max defined in the dataset
- 3 Iterate from the current index to the next index in the genome sequence
- 4 Check if the vehicle can safely travel from current location to the next customer
- 5 If yes, travel to the customer, subtracting the battery cost from the capacity
- 6 If no, find the nearest charging station and travel there
- 7 If no charging station is within range, add a heavy penalty to the fitness
- 8 Repeat steps 3 through 7 until all customers from the genome sequence have been visited
- 9 Return to the depot from the last node visited, going to charging stations as necessary

EVRP Results


Tour: 0 4 5 2 6 2 7 1 8 0

There are 1 subtours in this route, with a total distance of this route is: 222.069



I have implemented the dependency on battery and charging stations, but this is not the complete EVRP yet.

- Need to Implement
 - Capacity
- Could Implement
 - Serviceable time windows
 - Charging time and travel time
 - Weight dependency
 - Full charging vs partial charging policy
 - Heterogeneous fleet vs homogeneous fleet

-  Ilker Kucukoglu, Reginald Dewil, and Dirk Cattrysse.
The electric vehicle routing problem and its variations: A literature review.
Computers & Industrial Engineering, 161:107650, 2021.