Bi-objective Vehicle routing problem using NSGAII and SPEA2

1. Mathematical Model Formulation:

Dantzig et al. [1] first described the concept of CVRP in their work "The truck dispatching problem". The main objective of classical CVRP model to minimize the distance or cost of traveling to serve the customers. Here in the given challenge another objective is introduced to minimize the number of vehicles during the service of the customers.

In general, the CVRP can be stated as follows: Let $G = (V, E)$ be a graph, $V = \{j_0, j_1, j_2, \dots \dots, j_n\}$ being the vertex set, where j_0 refers to the depot, the customers are indexed $j_0, j_1, j_2, \dots \dots, j_n$, and $E = \{(j_m, j_n) : j_m, j_n \in V\}$ is the edge set. Let n be the number of customers, $N = \{1, 2, \dots, n\}$ be the set of customers, and $\{0, 1, \dots, n\}$ be the set of all customers and the depot, which is identified by 0. Let $K = \{1, 2, \dots, k\}$ be the set of vehicles. Let d_i denote the demand of customer $i \in N, L_h$ indicate the load of vehicle $h \in K$. Furthermore, $max\ d_i \le max\ L_h$. Let c_{ij} be the transport cost or distance between customer i and j.

The ILP model for the capacitated vehicle routing problem is as follows:

Let us define

$$y_{ih} = \begin{cases} 1, & if\ point\ i\ is\ visited\ by\ vehicle\ h \\ 0, & otherwise \end{cases}$$

$$x_{ijh} = \begin{cases} 1, & if\ vehicle\ h\ travels\ directly\ from\ node\ i\ to\ node\ j \\ 0, & otherwise \end{cases}$$

$$Min\ Z1 = \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{h=1}^{k} c_{ij}x_{ijh} \tag{1}$$

$$Min\ Z2 = k \tag{2}$$

Subject to

$$\sum_{i=0}^{n} d_i\, y_{ih} \le L_h, \forall h \tag{3}$$

$$\sum_{h=1}^{k} y_{ih} = 1, i = 1,2, \dots \dots .., n \tag{4}$$

$$\sum_{i=0, i\neq j}^{n} x_{ijh} = y_{jh}, \qquad j = 1,2, \dots \dots .., n\ \ \forall\ h \tag{5}$$

$$\sum_{j=0, j\neq i}^{n} x_{ijh} = y_{ih}, \qquad i = 1,2, \dots \dots .., n\ \ \forall\ h \tag{6}$$

$$x_{ijh} = 0 \; or \; 1, \qquad i,j = 0,1,\ldots,n,\forall h \tag{7}$$
$$y_{ih} = 0 \; or \; 1, \qquad i,j = 0,1,\ldots,n,\forall h \tag{8}$$

In the above model, (1) and (2) represents the objective functions, which minimizes the total distance and number of total vehicle used to serve the customers respectively. Constraint (3) refers to the preservation of vehicle capacity, that is, certain vehicles must deliver goods to customers within vehicle capacity; (4) represents that every customer must be served; (5) and (6) ensure that each customer is served by only one vehicle.

2. Solution Methodology

The above model is bi-objective in nature and can be solved using exact algorithms as well as metaheuristic algorithms. As the problem is combinatorial in nature the computational cost and time will increase exponentially with the increase of the problem size. But, if we apply multi-objective metaheuristic algorithm we can get an approximate front within reasonable computational cost and time.

Here I choose NSGAII and SPEA2, very well-known state-of-art population based multi-objective evolutionary algorithm. As both the algorithms are efficiently applied to solve combinatorial multi-objective optimization problem by several researcher.

To solve the problem I use jMetalPy1.5.3 [2] python package. jMetalPy 1.5.3 is a python package for solving both single and multiple-objective optimization problems using evolutionary algorithms and heuristic algorithms.

A class VRP is designed to fit the mathematical model. At the time of instantiation, the path of the input data is to be passed as the parameter of the constructor. The class inherits the class PermutationProblem of jmetal.core.problem package and PermutationSolution of jmetal.core.solution package.

3. Observation

On execution of both the method to solve the model, NSGAII is taking less execution time than SPEA2. The result obtained from the both the algorithm on an independent run is given below. In most of the cases very less number of solution is present in the approximate front as the given problem size is small.

Run no.	NSGAII	SPEA2
1	Solution 1 no of vehicle= 5 distance= 309 route 1 -> Depot city 3 city 7 city 8 city 11 city 10 Depot route 2 -> Depot city 15 city 16 city 17 city 23 Depot	Solution 1 no of vehicle= 5 distance= 385 route 1 -> Depot city 17 city 13 city 11 Depot route 2 -> Depot city 16 city 15 city 19 city 18 Depot

	route 3 -> Depot city 2 city 1 city 4 city 6 city 9 city 5 Depot route 4 -> Depot city 13 city 18 city 19 city 14 city 12 Depot route 5 -> Depot city 25 city 24 city 22 city 21 city 20 Depot	route 3 -> Depot city 21 city 23 city 7 city 8 city 4 city 5 Depot route 4 -> Depot city 25 city 10 city 9 city 3 city 1 city 2 Depot route 5 -> Depot city 22 city 20 city 24 city 14 city 12 city 6 Depot

4. Future Improvements:

i. Rigorous parameter tuning is needed to get stable and good solution.

ii. If we consider model it can be made more realistic by considering more objectives depending on mode of transportation, vehicle type etc.

References:

1. G. Dantzig & J. Ramser, “The truck dispatching problem.” Management Science 6 (1959), 80-91.

2. Antonio Benítez-Hidalgo and Antonio J. Nebro and José García-Nieto and Izaskun Oregi and Javier Del Ser, “jMetalPy: A Python framework for multi-objective optimization with metaheuristics.” , Swarm and Evolutionary Computation (2019), doi: 10.1016/j.swevo.2019.100598