MMDM Lab №1

Mugaddam N, Seliverstov A, Ketsba A

April 3, 2024

1 Problems 1-2

The main steps of algorithm for solving problems 1-2 are as follows.

First of all, we generate set N with n points, with $(x, y) \in [-10, 10]^2$. Then we start to iterate t times with next steps.

1. Select $m < n$ solutions with the best score from our population N and add them to $M = \{a_1, a_2, ..., a_m\}$. In Problem 2 this step is modified with NSGA-II algorithm adding non-dominated sorting and crowding distance sorting.

2. Generate new points by mating two random points and taking their mean $a' = \big((x_1 + x_2)/2, (y_1 + y_2)/2\big)$ for R^2

3. Apply mutate operator by adding a small number s to a', while s decreases as the number of iterations increases:

$$a'' = a' + s, \qquad \lim_{t \to \text{inf}} s = 0$$

   Finally, after all iterations we take points with the best scores.

2 Problem 3

2.1 Problem description

The aim of problem is to find a set of minimum total cost routes for a capacitated vehicles (couriers). Each courier starts and ends the route at the flower market and has to serve a set of cities with following constraints:

1. Each city needs only certain number of flowers

2. Couriers are paid for the distance travelled

3. The total demand of route does not exceed the capacity of the courier

Let's try to formalize our problem.

- N is the set of cities with $N = \{1, 2, 3, ..., n\}$
- M is the set of courier with $M = \{1, 2, 3, ..., m\}$
- q_v is the capacity of the courier v
- d_i is the demand of the city i
- c_{ij} is the distance between city i and city j
- y_{iv} is the amount of cargo delivered to client i by courier v

Our goal is to minimize the total cost (distance):

$$\min_x \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{v=1}^{m} c_{ij} x_{ijv} \tag{1}$$

$$x_{ijv} \in \{0, 1\}, \tag{2}$$

1 - courier v visited city i, 0 - otherwise

$$\sum_{i=1}^{n} \sum_{v=1}^{m} x_{ijv} = 1, \qquad \forall j \in \{1, 2, ..., n\} \tag{3}$$

$$\sum_{i=0}^{n} x_{izv} = \sum_{j=0}^{n} x_{zjv}, \qquad \forall z \in \{0, ..., n\}, \forall v \in \{1, ..., m\} \tag{4}$$

$$y_{iv} \leq d_i \sum_{j=1}^{n} x_{ijv}, \qquad \forall i \in \{1, ..., n\}, \forall v \in \{1, ..., m\} \tag{5}$$

$$\sum_{v=1}^{m} y_{iv} = d_i, \qquad \forall i \in \{1, ..., n\} \tag{6}$$

$$\sum_{i=1}^{n} y_{iv} \leq q_v, \qquad \forall v \in \{1, ..., n\} \tag{7}$$

The function (1) minimizes the total delivery cost (distance). The claim (3) ensures that each city is visited by exactly one courier and the limitation (4) requires that every courier can leave the flower store only once, and the number of couriers visited every city and returning to the store is equal to the number of the couriers leaving. Constraints (5) and (6) guarantee that the amount of cargo delivered by courier is equal to city demand. And the last requirement (7) ensures that the courier can carry only certain number of cargo (flowers).

2.2 Algorithm

1. First of all, we generate initial population $P = \{s_0, s_1, ..., s_k\}$. Here we have k solutions that includes set of the couriers with sequence of the cities they serve. To generate s_i we randomize distribution so that the courier's cargo does not exceed its capacity.

2. Then we select $l < k$ best solutions from our population P and add them to $P' = \{s_0, s_1, ..., s_l\}$. In our case we select 50% solutions.

3. Randomly chose 2 parents s_1 and s_2 from P' and apply our crossover operator: $child = C(s_1, s_2)$. If child is valid and the cargo of each courier does not exceed its capacity we add it to P'. We mate solutions until we reach the size of initial population $P' = P$

4. Then we apply mutation operator in order to expand the search area and preserve diversity in the population: $s'_i = M(s_i)$. Mutation operator randomly swaps cities between $courier_1$ and $courier_2$. It is important that the sum of the demand is equal to the sum of the couriers' capacities. So, we have to change cities among couriers such a way to save each one's total cargo (7). In our algorithm we randomly choose from 2 variants:

   (a) We choose 1 city from each of two couriers: $d_{1|1} = d_{1|2}$, where $d_{i|j}$ is demand of the city i that is served by the courier j

   (b) We choose 1 city from 1st courier and 2 cities from 2nd one: $d_{1|1} = d_{1|2} + d_{2|2}$

5. We repeat step 2 to step 4 t times to reach a sufficient number of iterations to find the best solution.