TSP Using Genetic Algorithm

20170741 Hwang Seunghyun

We usually face a TSP problem in DP(Dynamic Programming) not GA. So, I don't have any idea about TSP Genetic Algorithm. Thus I image structure of algorithm to construct basic structure.

0. Parser
1. Load Dataset from tsp file.
2. Make initial population
3. Evaluation Fitness Proportional Selection
4. Create Offspring(Using Crossover)
5. Mutate Offspring
6. Make Population(3~5 repeat)
7. Evaluate Fitness Proportional Selection
8. 4-7 Repeat
9. Return result

Now I explain each part more concretely.

0. Parse

   Using argparse

```
parser = argparse.ArgumentParser(description='typing p, f')
parser.add_argument('name', help = "filename")
parser.add_argument('-p', required=False, default="100", help = "population")
parser.add_argument('-f', required=False, default="1000", help = "max Fitness
evaluations")
```

   To meet problem condition I make parser like above.

   You can run like $python3 tsp_solver.py berlin52.tsp -p=number1 -f=number2

1. Load Dataset from tsp file

```
while not("DIMENSION" in a.split(" ")[0]):
```

   Using while for check Dimension

```
while not("1" in a.split(' ')):
```

   Using while for get coordinates data

   Update in DataList

   And create Orderlist to Indexing

```
        for i in range(dimension):
        OrderList.append(i)
```

1.5. Setting

   Evaluate_distance(), distance_setting()

   Evaluate distance and make distance List

2. Make Initial population

   Using random.shuffle(OrderList)

   Append in PopulationList

   PopulationList : List of gene

   Gene : travel route

3. Evaluation Fitness Proportional Selection

   Make calculate_dist(gene) function

   It calculate travel distance in one Gene(travel route)

   Inverse it

   Sum all calculate distances inverse in population = total

   FPS = each travel distance inverse /total

4. Create Offspring(Using Crossover)

   Using Roulette Wheel Sampling

   Using FPS and random.uniform(0,1)

   Choose two parent using above function

   Doing crossover

   Crossover algorithm :

   Number of dimension's = component in gene

   Division = Random number in (0, dimension)

   Take parent1[:division]

   Take parent2 empty space(Erase overlap with parent1[:division] and connect other parent2's component to parent1[:division])

5. Mutate Offspring

   Mutationlate => mutation probability

   Using if condition

```
if random.uniform(0,1) < mutationlate:
```

   then mutating

   I use mutate algorithm which exchange two index.(index choosing is random)

6. Make Population(3~5 repeat)

   Repeat number = Population

   ⇨ To make next generation

   (make_offspring)

7. Evaluate Fitness Proportional Selection

   ⇨ To make next generation FPS

8. 4-7 Repeat

   Repeat number = maxrepeat(max fitness evaluations)

   ⇨ Genetic Algorithm

(big_cycle)

9. Return result

   Save minimum distance => mindist

   Save minimum distance population => optimapopulation

   Using csvwriter we can make csv file like instructions.