

Assignment 4:
Optimization with Genetic Algorithms
Neuronal and evolutionary computing
2023 - 2024

Pawel Puzdrowski

2024-01-XX

Contents

0.1	Description of the chromosome, and adaptations done to the algorithm	2
0.1.1	Translation of the problem to chromosome	2
0.1.2	Methods for selection/crossover/mutation	3
0.2	2.	5

0.1 Description of the chromosome, and adaptations done to the algorithm

0.1.1 Translation of the problem to chromosome

Firstly the dataset is imported with help of `tsplib95` library. A simple `tsplib95.load()` function is used to import the dataset and get proper structure of the problem so it can be handled by the genetic algorithm. The genetic algorithm has several stages and this stages are inspired from `geeksforgeeks` source [1]

- Initialization
- Fitness score calculation
- Elitism
- Selection
- Crossover
- Mutation
- Population update

At the Initialization stage the original population is created with `create_initial_population()` function. The cities are extracted from the dataset and tours are created randomly.

Fitness score calculation is done with `fitness()` function where following fitness score calculation is done:

$$\frac{1}{\text{calculate_distance()}}$$

Function `calculate_distance()` is basically calculating the total distance of a tour. Then if the distance is high the fitness score will get low and if the distance is low the fitness score will get high. Every tour is assigned one fitness score.

The part where elitism is done, it's taken from this source [2]. Elitism is done with `elitism()` function. The best tour from the population is chosen and added to the new population. This way the best tour is always saved in the new population and the evolution is faster.

Selection is choosing the parents for the mating part. Depending on the method the parents are chosen differently. Crossover section is creating the children from the chosen parents and depending on the method the children's inheritance of parents' genetics is done. Mutation part has the responsibility to perform mutations on the children and as well mutation has different methods to perform mutations so the mutations can vary.

To finish off the algorithm, all the parents create a new child and at the end a new population has arrived and is selected to do the same mating process again depending on how many generations are specified.

The different selection methods were taken from this source [2] and the different selection methods are listed below

0.1.2 Methods for selection/crossover/mutation

Selection methods: [2]

- Fitness proportionate selection - parents are chosen randomly, but the parents with higher fitness score have higher chance of being selected
- Tournament selection - 4 parents are chosen randomly from the set and the parent with highest fitness score and the second highest score is chosen to mate.

Crossover methods: [3]

- One point crossover - a random point is chosen on the tour and the tour is split into two parts. Visualization of one point crossover is shown in figure 1
- Uniform crossover - each gene (city) is treated separately instead of dividing the parent into one or more parts. Basically a flip-coin is done on every city if it is included or not. Visualization of uniform crossover is shown in figure 2

Mutation methods: [4]

- One bit flip mutation - only one bit is mutated which in this case means that one city is swapped with another random city from the same tour. Visualization of one bit mutation is shown in figure 3
- Scramble mutation - a random set of cities is chosen and swapped with another random set of cities. Visualization of scramble mutation is shown in figure 4

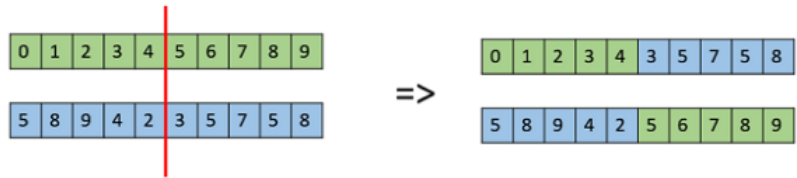


Figure 1: One point cross

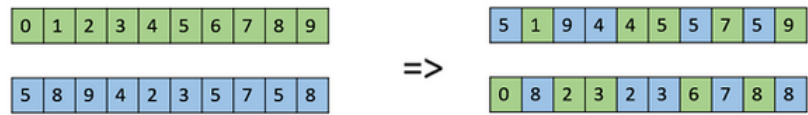


Figure 2: Uniform crossover

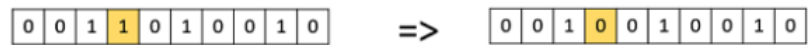


Figure 3: One bit mutation

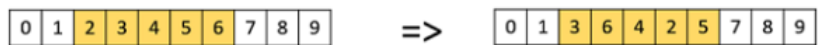


Figure 4: Scramble mutation

0.2 2.

Bibliography

- [1] Geeksforgeeks. Traveling salesman problem using genetic algorithm. Link to source, 2023. Accessed 2024-01-05.
- [2] Eric Stoltz. Evolution of a salesman: A complete genetic algorithm tutorial for python. Link to source, 2018. Accessed 2024-01-05.
- [3] Tutorialspoint. Genetic algorithms - crossover. Link to source. Accessed 2024-01-08.
- [4] Tutorialspoint. Genetic algorithms - mutation. Link to source. Accessed 2024-01-08.