Travelling Salesman Problem - Using Genetic Algorithm

AI For Search Methods - CS6380
Ankit Thawal - CS21M004

October 16, 2021

Abstract

1 Introduction to TSP Problem

The travelling salesman problem (also called the traveling salesperson problem[1] or TSP) asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?" It is an NP-hard problem in combinatorial optimization. Many exact and heuristic algorithms have been developed in the field of operations research (OR) to solve this problem.

2 Some examples to get started

2.1 The role of AI

The technology was created to help and facilitate human activities and work. One technology that is being intensively created to create sophisticated devices is Artificial Intelligence. Many artificial intelligence developers who create implementations of artificial intelligence that are super sophisticated and applied to the real world. Artificial intelligence has the opportunity and potential to be developed further with qualified resources.
AI continues to evolve to benefit many different industries. Machines are transferred using an interdisciplinary approach based on mathematics, computer science, linguistics, psychology, and more.

2.2 Genetic Algorithm

Genetic algorithm (GA) as a computational intelligence method is a search technique used in computer science to find approximate solutions to combinatorial optimization problems.
The genetic algorithms are more appropriately said to be an optimization technique based on natural evolution. They include the survival of the fittest idea algorithm. The idea is to first "guess" the solutions and then combining the fittest solution to create a new generation of solutions which should be better than the previous generation. We also include a random mutation element to account for the occasional mishap.
There are three basic steps in Genetic Algorithm(Goldberg) :
a. Selection
The random tours are generated, which will constitute as a population. Then perfect fitness function is used to calculate the best options from the population. In our example of TSP, the tour length is taken as a fitness function. Using the minimum tour length as a fitness function, the best tours are selected.
b. Recombination
The recombination operator takes output of selection operator and randomly generates offspring by combining two parents. The elitism is followed, to generate best copies from the best/fittest parents.
c. Mutation
GA incorporates mutation to allow for random moves.

2.3 Methodology

Steps of Algorithm

1. Generate Random population

The random tours are generated using rand() function. Here only rule is not to repeat any cities visited earlier. Every cities are visited only once.

Priority queue is maintained which uses min heap. Hence, the tours with minimum tour length will be pushed at the front of queue.

2. Assign Fitness function to every tour

In TSP, fitness function is "minimize the tour length". After every tour is generated, another array is maintained to record the tour lengths.

3.Create new offspring population from two existing chromosomes in the parent population by applying crossover operator.

Crossover is applied to generate offsprings from parents. There are many crossover operators to choose from. I have used the **partially mapped crossover** operator, to generate offsprings.

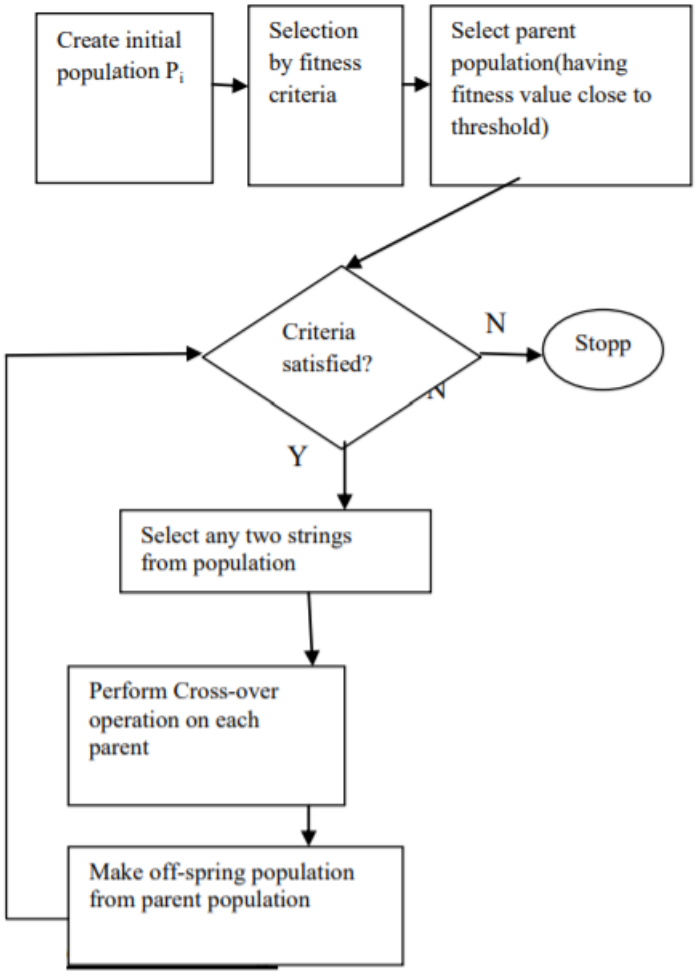
PMX :

The partially mapped crossover (PMX) was proposed by Goldberg and Lingle. After choosing two random cut points on parents to build offspring, the portion between cut points, one parent's string is mapped onto the other parent's string and the remaining information is exchanged.

4.Introducing Mutations.

Mutation is implemented to introduce some random moves. As with crossover, only the fittest parents makes the offspring, the mutation is required so algorithm also explore the new possibilities.

5.Repeat Steps 2 to 4. The steps are repeated until the optimized solution is printed.



References

1. First course on Artificial Intelligence - Prof. Deepak Khemani