Group members:

Abeera Farooq

Marium Rehman

Zain-ul-Haq

Data Structures and Algorithms

Final Project Report

Project Title: Darwinism

Inspiration: Genetic Algorithm

About Genetic Algorithm:

A Genetic Algorithm is a search heuristic algorithm inspired by the Theory of Natural selection which was given by Charles Darwin in which the fittest individuals from a population are selected to produce offspring of next generation.

The algorithm is used to generate high quality solution for optimization problems. Unlike traditional search algorithms GA is based on probabilistic rules rather than fully deterministic rules, which is at times helpful in finding a global optimum solution. GA has its application in the fields of machine learning and AI.

Major phases or operators of GA:

* Initial population ----> creates a set of possible solution to the problem
* fitness calculation ----> calculates the fitness of solution according to the required objective
* parent selection -----> selects the fittest individuals from population
* cross-over -----> breeds the selected solutions to get new better solutions
* mutation -----> causes randomized mutation in some of the solutions

Objective of the project:

Tackling the problem of finding the shortest path from source node to destination node using the approach of Genetic Algorithm

Expectations:

* writing a code of genetic algorithm catering the above stated objective
* the code would be expected to converge to the optimum solution over the course of generations
* Plot the solutions found in each generation to visualize the progress of the genetic algorithm.
* Analyzing the runtimes of the functions and the complete algorithm

Approach:

* The code is implemented using the data structure of graphs in the form of adjacency list.
* The code constitutes of a main function of Genetic algorithm, object functions: Initial population, fitness, parent selection, cross-over, mutation and a few helper functions.
* The main function requires a graph (adjacency list), source node and the destination node initially as parameters.
* The initial population function is used to generate list of possible paths between the source and destination nodes, which is referred as the population.
* Fitness function calculates the distance of each path present in the population.
* For generating the next generation a function of new\_population is made in which the parent selection, crossing-over and mutation functions are called.
* Parent selection picks the pair of best solutions (paths with shortest distances according to our objective) as parents from the existing population.
* Crossing over function takes the two parents as parameters and tries to find a common node in both parents, when a common node is found, it breaks the path at this node, then forms a new path taking one half of path from one parent and second half from the other.
* The mutation function takes a path as parameter and randomly tries to change it by sometime reducing a node in the path, sometimes adding one or changing one node in the existing path (the crossing over and mutation functions ensure that the resulting solution are always valid paths )
* With a combination of paths made through cross over, mutation functions and some repeated members from previous generation, a new generation of population is made.
* The algorithm continues to make new generations until a number of maximum generations is achieved (maximum number of generation are a pre decided value)

Results:

Plots of a sample result of the above explained code showing how the algorithm converges to optimum solution over the course of generations.

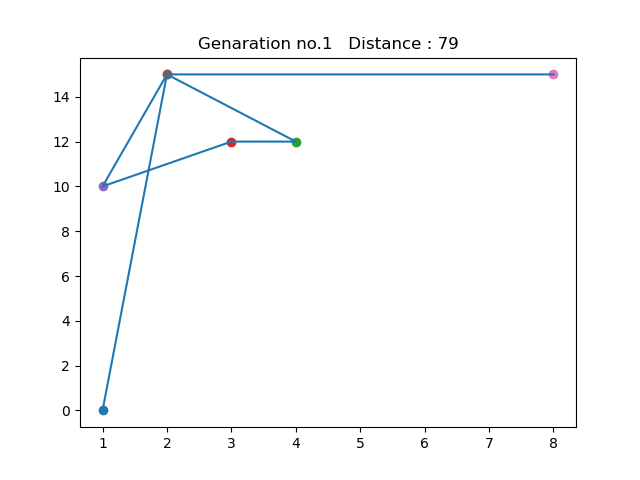
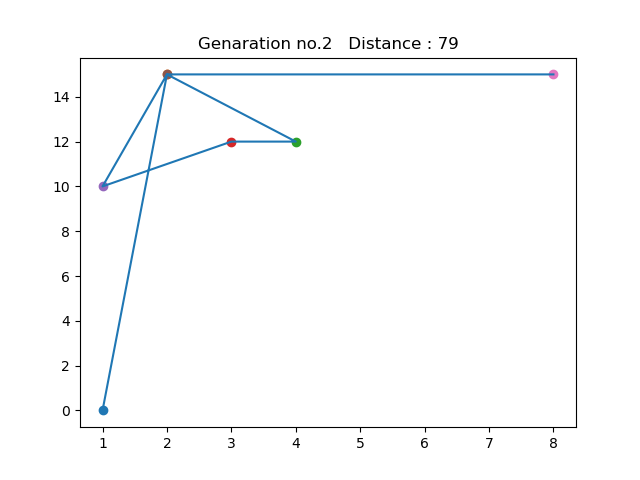
 

Figure: fittest solutions of generation1 & 2 showing paths with corresponding distance values.

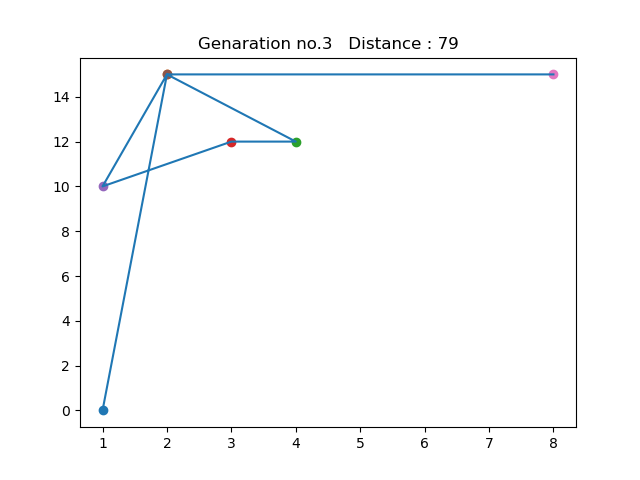
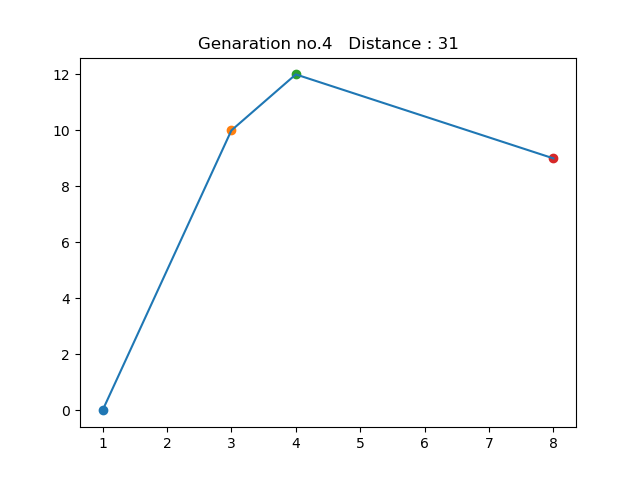
 

Figure: fittest solutions of generation3 & 4 showing paths with corresponding distance values.

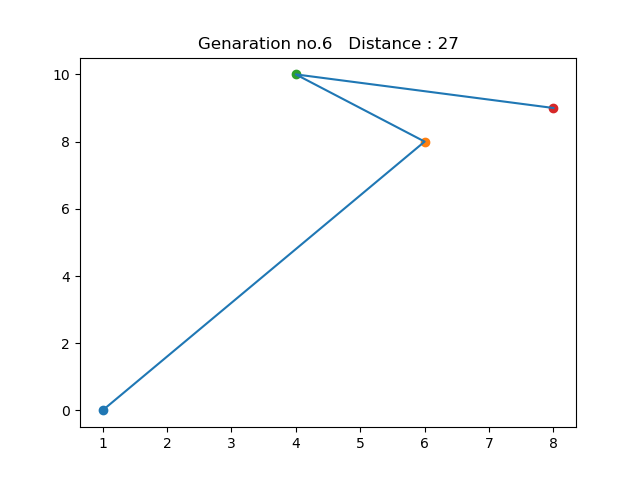
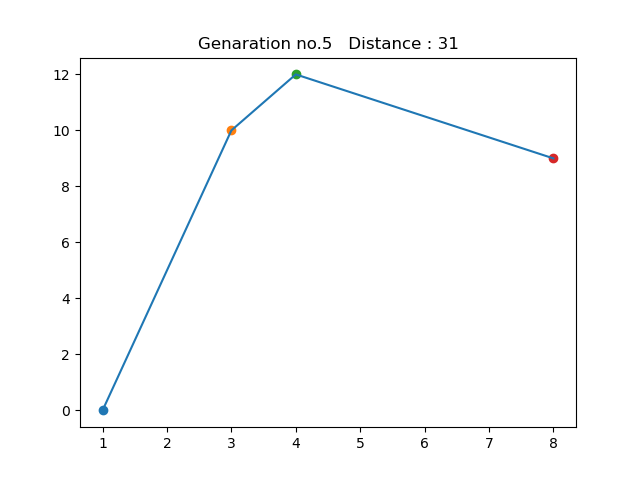


Figure: fittest solutions of generation 5 & 6 showing paths with corresponding distance values.

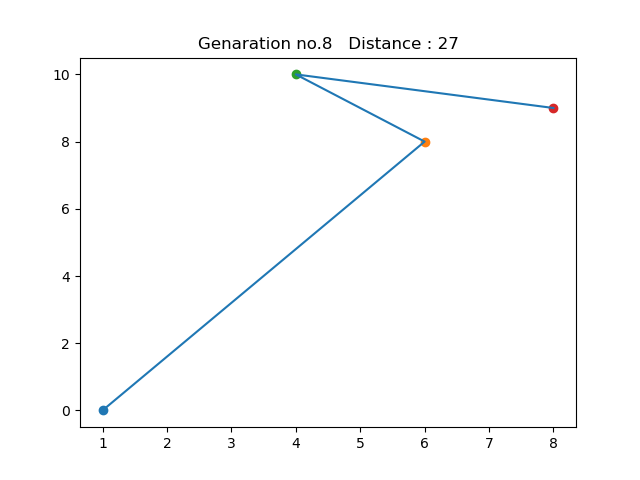
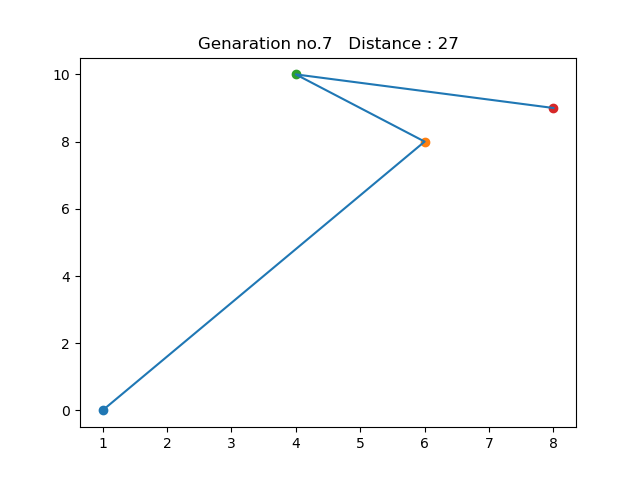


Figure: fittest solutions of generation 7 & 8 showing paths with corresponding distance values.

Validation of results:

We ran the same input for Digikstra Algorithm and received the same output as the Genetic Algorithm i.e. for the input of above shown results dijikstra algorithm also gave the same shortest path with distance value 27.

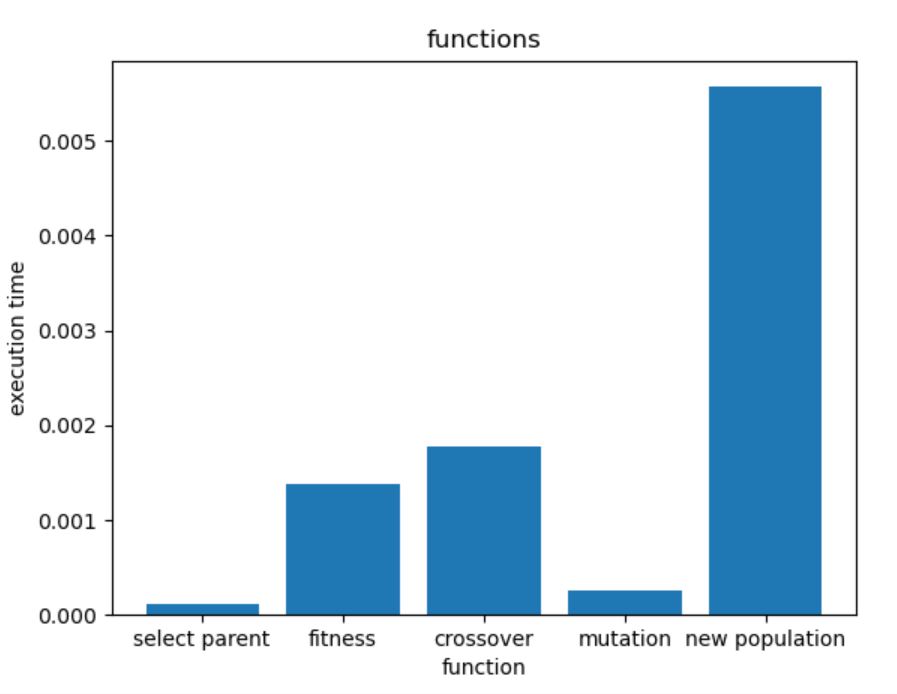
Challenges addressed:

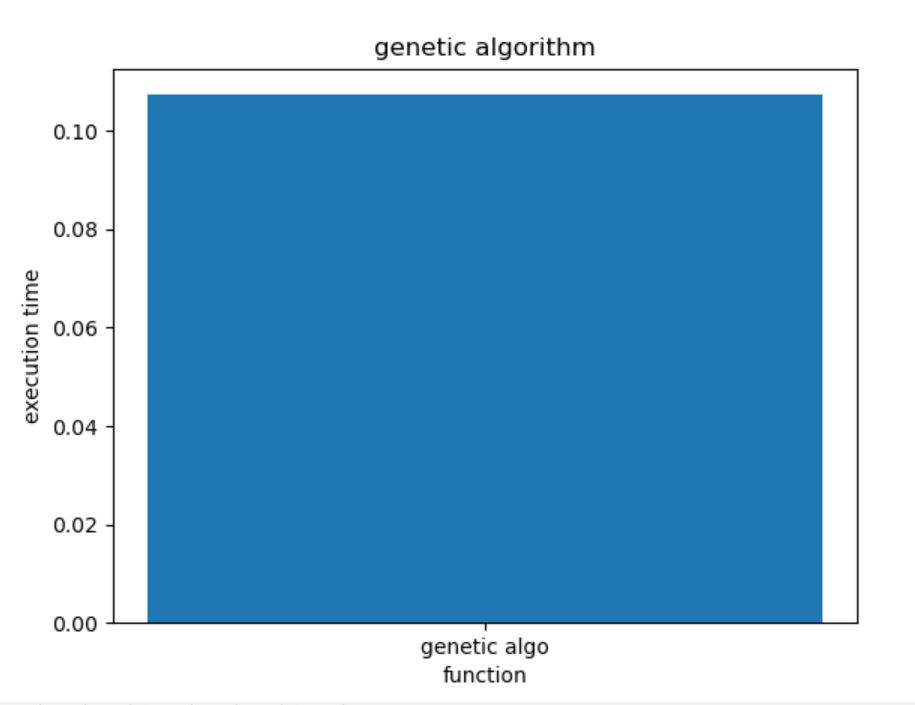
* The GA is simply a probabilistic approach which means it works with probabilities and there is no deterministic rules governing GA. This shows that there is no guaranteed solution of our requirement. Instead, we assume whatever solutions are arrived from the output of GA is the best one.
* The path representation may sometime results in the formation of an empty cell or violate some constraints. This means that GA will find some solutions which may not exist in reality or difficult to implement in practical situations. To counter this, we have used a helper function which checks if the resulting path is valid or not.

Analysis:

.plot showing no. of iterations for reaching optimal solution as population size increases.

Runtime:





Libraries:

* Matplotlib
* random

References:

* <https://www.youtube.com/watch?v=XP8R0yzAbdo&t=451s>
* <https://www.geeksforgeeks.org/genetic-algorithms/>
* <http://als.zim.pcz.pl/files/POSSIBILITIES,-OBSTACLES-AND-CHALLENGES-OF-GENETIC-ALGORITHM-IN-MANUFACTURING-CELL-FORMATION.pdf>
* <https://www.sciencedirect.com/science/article/pii/S111086651000023X#s0050>
* <https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3>