SOLWİNG TRAVELING SALESMAN PROBLEM WITH GENETIC ALGORITHYM AND PARTICLE SWARM OPTIMIZATION ALGORITHYM

Serdar Altanay

22290661

tspGA:Genetic Algorithym file

tspPSO:Particle Swarm Optimization Solution file

1) METHODS

1.a) Genetic Algorithym (GA):

Genetic Algorithm are optimization algorithms inspired by the principles of natural selection and genetics.

Pseduocode of GA:

Initialize population with random individuals

Evaluate the fitness of each individual in the population

While termination condition not met

Select parents from the population based on their fitness

Apply crossover to selected parents to create offspring

Apply mutation to offspring for variation

Evaluate the fitness of the offspring

Select individuals for the next generation (elitism can be applied)

Return the best solution found

1.b) Particle Swarm Optimization (PSO):

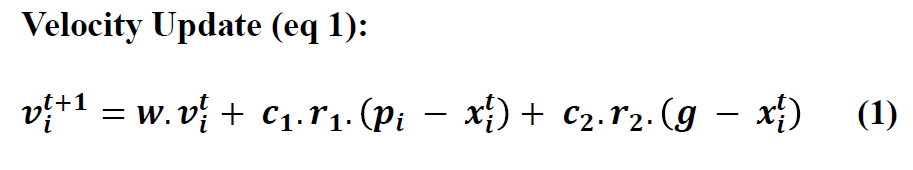
Particle Swarm Optimization is a popular optimization algorithm inspired by the social behavior of birds flocking or fish schooling

Pseduocode of PSO:

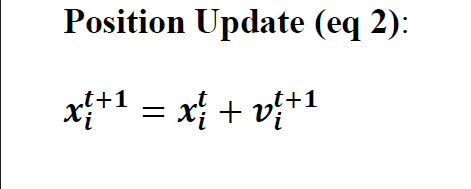
Initialize swarm with random positions and velocities

For each particle, calculate its fitness and set pBest = current position

Set gBest as the position with the best fitness in the swarm

Repeat until termination condition:

For each particle:

 Update velocity using eq 1

Update position using eq 2

Evaluate the fitness of the new position

If the new fitness is better than pBest

Update pBest = new position

If pBest is better than gBest

Update gBest = pBest

Return gBest as the best solution found

1.c) Implementation GA and PSO to Traveller Salesman Problem:

Swarm(PSO) or chromosome(GA)Encoding:

* Each individual in the population represents a possible solution (tour). A chromosome is an array of cities ID’s (0 = A , 1 = B , 2 = C …).
* City 0 (A) is a start and enpoint of all tours(individuals)
* For example, a chromosome [0, 3, 1, 2, 4] represents a tour starting from city 0, then visiting city C, followed by city 1, city 2, and city 4, before returning to city 0.

Calculating Fitness:

* The fitness of an individual is determined by the total distance of the tour. The shorter the tour, the better the fitness score.
* For a tour represented by [0, 3, 1, 2, 4], the fitness function calculates the total distance by summing up the distances between consecutive cities based on a distance matrix.

Termination:

* Each algorithym stops with significant number of iteration.

For GA:

Selection:

* Tournement selection method is applied to select parents. (randomly select a few individuals and choose the best among them.)

Crossover:

* Two-point crossover is applied to each individual.

Mutation:

* Swap mutation is applied. Randomly 2 cities swapped.

For PSO:

Swarm Initialization:

* Particles (Solutions): Each particle represents a tour (a permutation of the cities).
* Velocity: Each particle has a velocity vector that determines how it moves through the solution space.
* Particles and its velocities randomly created in initilize\_swarm function.
* Other pso steps maded directly like pseduecode

2) RESULTS AND ANALYSIS

In this project **comparison.py** created to compare pso and ga algorthyms with using time and matplotlib libraries and creates a graphs that compares both algorithym’s convergancy and execution time in significant number of turns.

To compare my algorithyms with diffrent number of cities ı created **distances.py** . There is 6,10,15,20 citied distance matrixes and you can compare both algorithm with different size of matrixes just with removing or adding comment lines at the begginning.

The graphs that ı used below created by my own **comparison.py** file and you can genereate different graphs with manipulating different datas(number of cities, number of turns, number of generations etc) directly.

2.a) Solution Quality:

Comparison.py can compare both algorithyms best solutions that found in significant run. If we look the outputs that generated by **comparison.py** 500 ıteration:

**In 20 run 10 cities:**

Overall Best GA Distance: 185

Overall Best PSO Distance: 186

**In 40 run,10 cities:**

Overall Best GA Distance: 184

Overall Best PSO Distance: 184

**In 20 run, 20 cities:**

Overall Best GA Distance: 1100

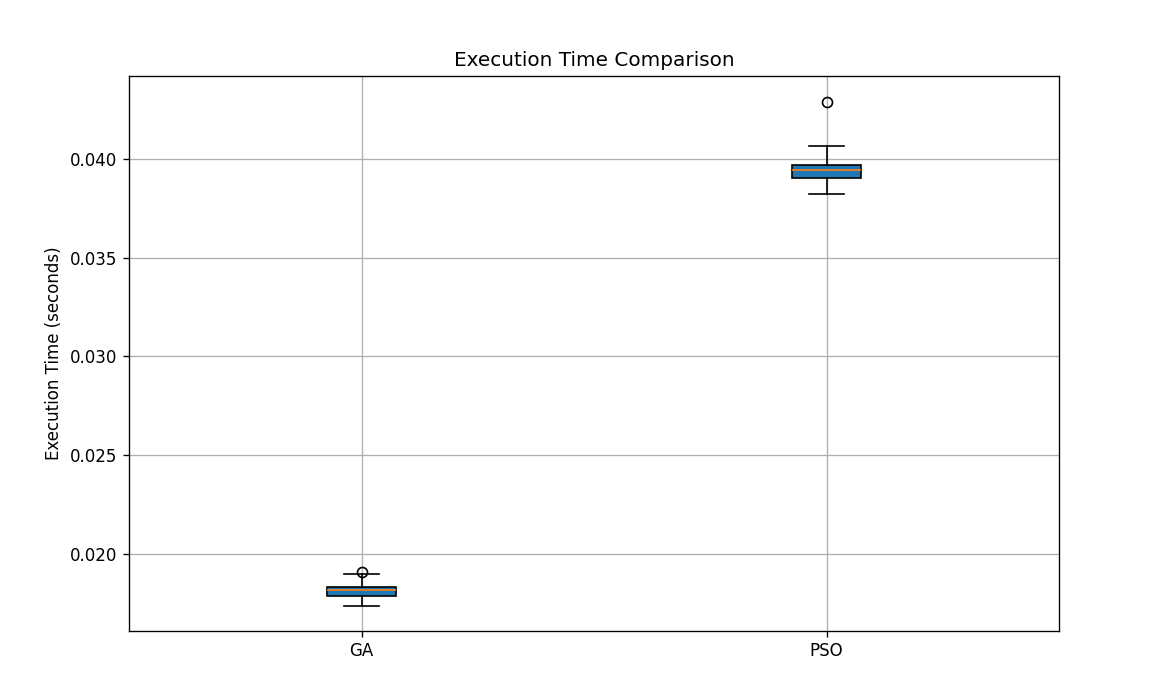
Overall Best PSO Distance: 1185

**In 40 run, 20 cities:**

Overall Best GA Distance: 1091

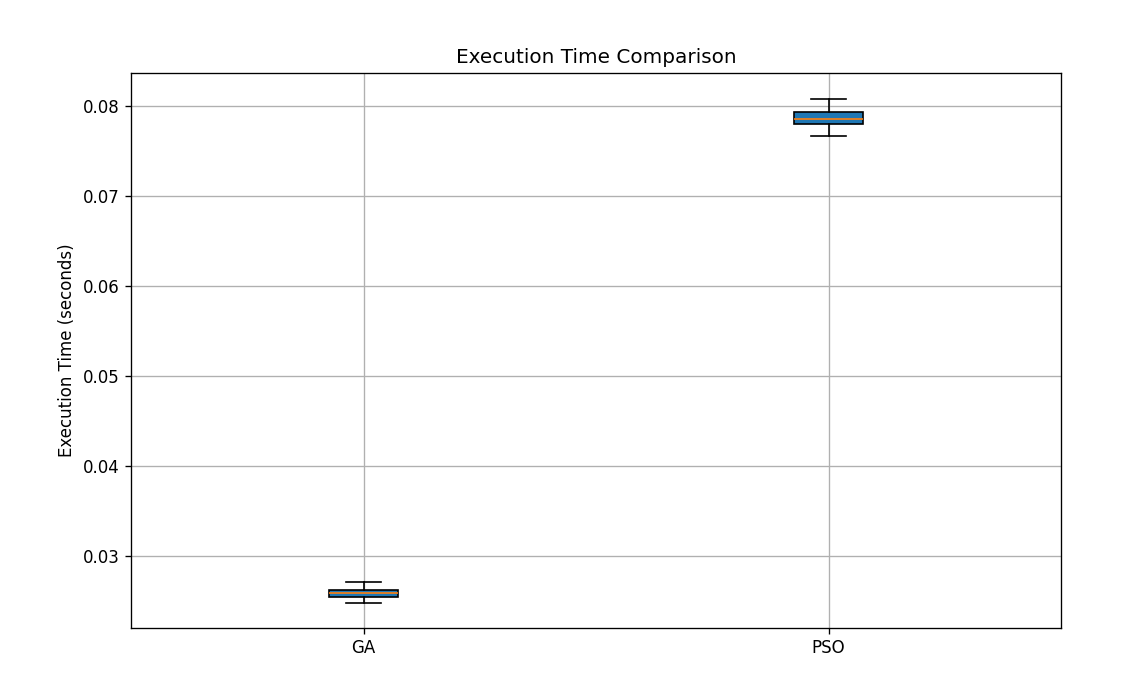
Overall Best PSO Distance: 1211

2.b) Execution Time:

10 cities , 500 iterations:

Average GA Time: 0.0181 seconds

Average PSO Time: 0.0395 seconds

20 cities, 500 iterations:

Average GA Time: 0.0258 seconds

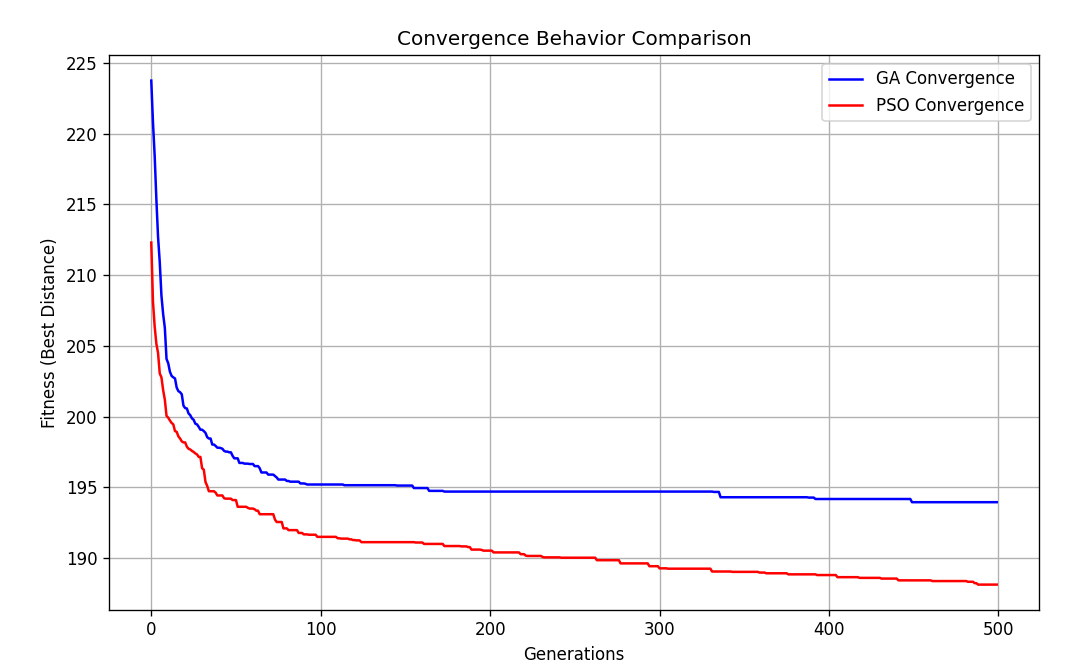
Average PSO Time: 0.0788 seconds

As we can see GA algorithym Works in less time than PSO algorithym with both 10 and 20 city data.

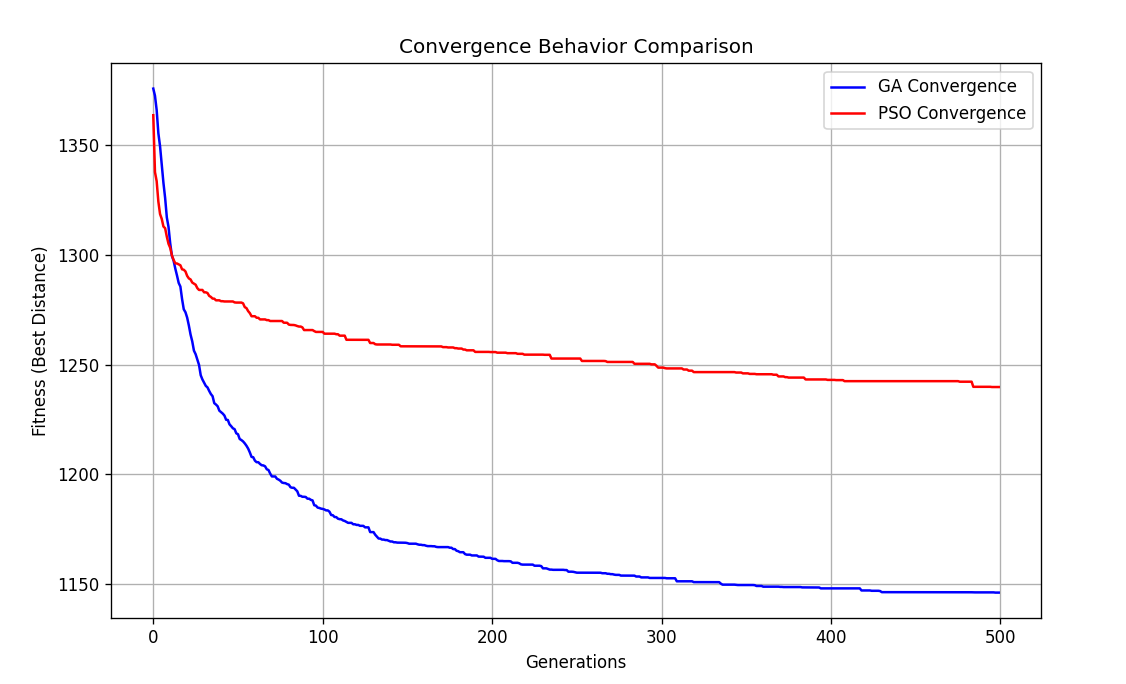
There is almost a 3-fold difference between the two in both cases.

2.c) Convergance Behaviour:

10 cities , 500 iterations:



20 cities ,500 iterations:



As we can see if number of cities increases GA algorithym Works more effective than PSO algorithym. But if we have less number of cities PSO algorithym Works more convergent to solution.

3) CONCLUSİON:

We can solved TSP problem with both GA and PSO and compared them with many ways like time , quality and convergance

In comparing the Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) across varying numbers of cities and iterations, the results reveal key insights. For both 10 and 20 cities, GA consistently outperforms PSO in terms of execution time, with GA running almost three times faster in each case.

Regarding solution quality, the best distances found by GA but it nearly same as that founds by PSO, particularly in larger datasets (20 cities). The convergence behavior shows that GA is more effective for larger problem sizes, while PSO exhibits better convergence with fewer cities. Overall, GA proves to be more efficient and effective as the problem size grows.