Genetic Algorithm Report

Burak Berk Yilmaz

Group: 4

Album Number: 97342

Part 1: Description of the Algorithm

The implemented algorithm is a Genetic Algorithm (GA) for solving the Traveling Salesman Problem (TSP). Below are the key components:

- Representation: The solution is represented as a sequence of city IDs, with a round-trip format (returning to the starting city).

- Initial Population: A mix of randomly generated and greedy paths, where a greedy path starts from a random city and selects the nearest unvisited city.

- Selection Mechanism: Tournament selection is used, where a subset of the population competes, and the fittest individual is chosen.

- Crossover: Partially Mapped Crossover (PMX) is used to preserve positional and order relationships between parent solutions.

- Mutation: A simple inversion mutation is applied with a probability, where a segment of the path is reversed.

- Local Search Optimization: Two-opt and Simulated Annealing are used for refining solutions.

- Elitism: The top 20% of solutions are preserved in each generation to maintain good solutions.

- Reinitialization: To avoid stagnation, a fraction of the population is periodically replaced with new random paths.

Part 2: Parameter Testing

To evaluate the impact of different parameters, the following experiments were conducted:

Tested Parameters:

1. Population Size: Values tested: 100, 200, 300.

2. Mutation Probability: Values tested:0.1, 0.3, 0.5.

3. Greedy Initialization Ratio: Values tested: 0.5, 0.8, 1.

4. For each test other parameters are set to minimum values from above

5. All results are from the Berlin52 file tests

PARAMETERS	TEST1-	TEST2 –	TEST3
POPULATION	7862.67	7919.03	7874.94
MUTATION	8616.43	8196.03	7674.15
GREEDY RATIO	8090.43	7829.42	8051.94

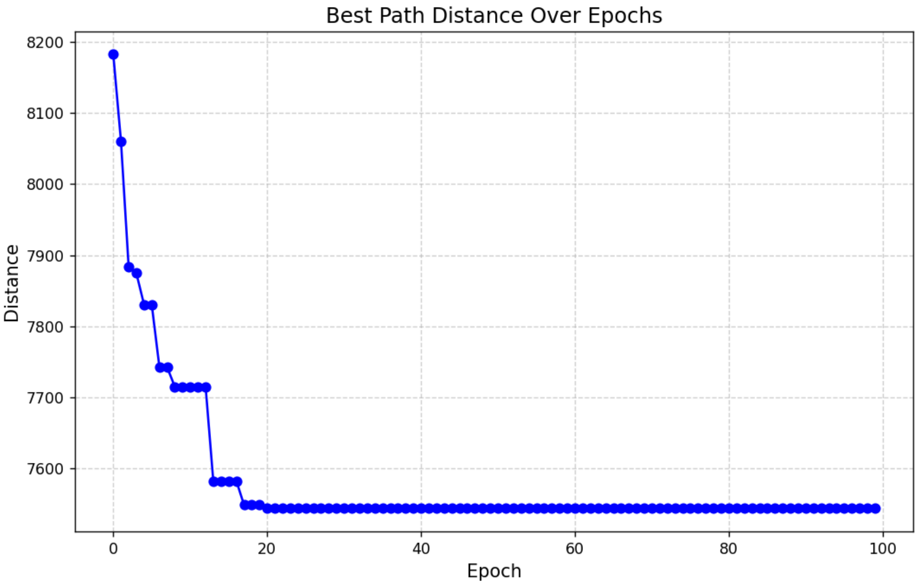
Part 3: Comparison Between Random Solution and Greedy Algorithm

5 Best Runs for Greedy Algorithm(Berlin52):

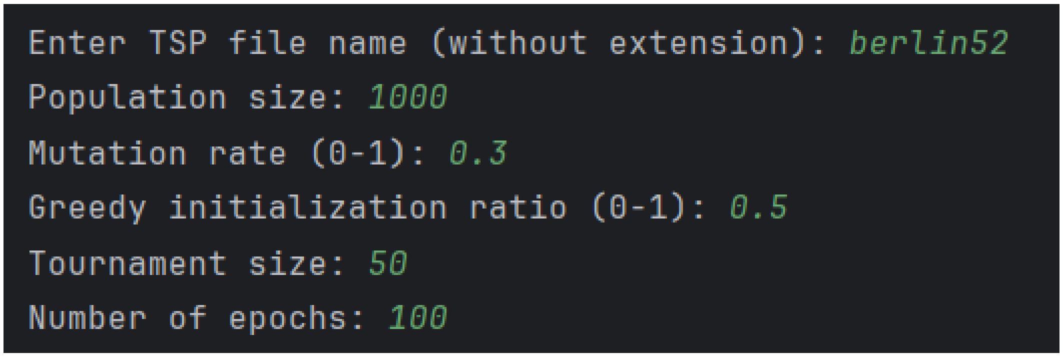
Algorithm	Run Results	Best Result	Average	Standard Deviation	Varience
Random	27357.01,31506.72, 26804.54, 29334.89, 30358.30, 30542.15, 32040.67, 32461.15, 32039.19, 30814.67	26804,54	30,325.93	1,952.56	3,812,502.45
Greedy	7544.37, 7713.03, 7902.94, 8051.94, 7829.42, 7598.44, 7702.45, 7754.51, 7781.76, 7792.63	7544.37	7,767.15	145.16	21,072.59
Greedy Best 5	**7544.37**	**7713.03**	**7598.44**	**8051.94**	**7829.42**

Part 4: Best Solutions

Berlin52 Matplotlib Chart:

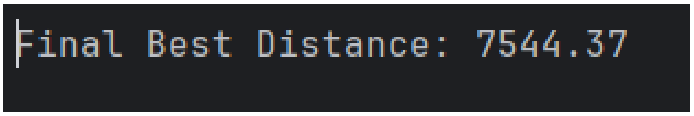


Parameters:



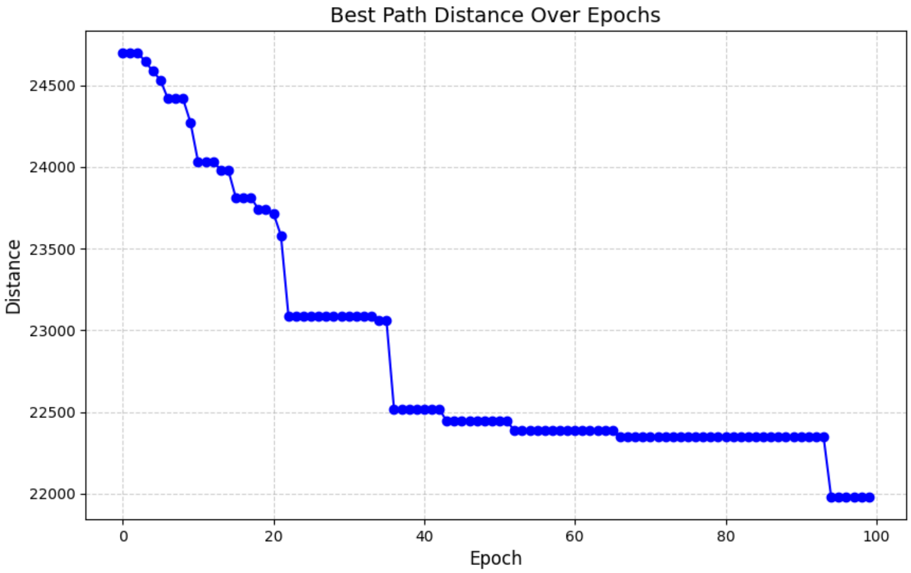
```
Enter TSP file name (without extension): berlin52
Population size: 1000
Mutation rate (0-1): 0.3
Greedy initialization ratio (0-1): 0.5
Tournament size: 50
Number of epochs: 100
```

Solution:

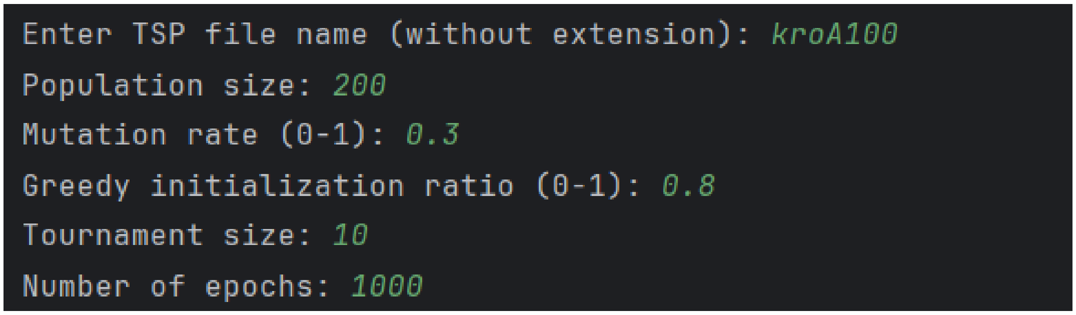


```
Final Best Distance: 7544.37
```

kroA100 Matplotlib Chart:

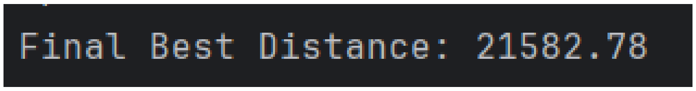


Parameters:



```
Enter TSP file name (without extension): kroA100
Population size: 200
Mutation rate (0-1): 0.3
Greedy initialization ratio (0-1): 0.8
Tournament size: 10
Number of epochs: 1000
```

Solution:



```
Final Best Distance: 21582.78
```

Part 5: Conclusion

- Increasing population size improved results but increased computation time.
- A moderate mutation probability (~0.3) was optimal for solution diversity.
- A high greedy initialization ratio (0.8) led to better initial solutions.
- The GA significantly outperformed random search and was competitive with greedy solutions.
- Local search (2-opt and Simulated Annealing) further improved the final results.
- Final Recommendation: A well-tuned Genetic Algorithm with local search is an effective approach for solving TSP efficiently.