# A Comparative Analysis of Genetic Algorithm and Particle Swarm Optimization Traveling Salesman Problem

**Author:** Ahmet ATAR
**Date:** 14.12.2024

## 1. Introduction

The Traveling Salesman Problem (TSP) is a fundamental NP-hard problem in combinatorial optimization. Given a set of cities and the distances between them, the goal is to determine the shortest possible route that visits each city exactly once and returns to the origin. Due to its computational complexity, exact methods become impractical as the problem size increases, making TSP an ideal testbed for heuristic and metaheuristic methods.

This report compares two popular metaheuristics—Genetic Algorithms (GA) and Particle Swarm Optimization (PSO)—on a 30-city TSP instance. We aim to evaluate their performances in terms of:

1. **Solution Quality:** The best (minimum) route distance found.
2. **Convergence Behavior:** How quickly and steadily each algorithm improves towards a high-quality solution.
3. **Execution Time:** The computational effort required to complete a fixed number of generations (GA) or iterations (PSO).

The chosen 30-city problem introduces sufficient complexity to highlight differences between the algorithms. Both methods are run multiple times (e.g., 10 independent runs) to gather statistical reliability, and results are summarized with descriptive statistics (mean, median, min, max, standard deviation) as well as convergence plots.

## 2. Methods

### 2.1 Problem Setup

We generated 30 random cities within a 100x100 coordinate space. The Euclidean distance was used to form a 30x30 distance matrix. This relatively complex instance is expected to yield non-trivial solutions, providing a meaningful comparison between GA and PSO.

### 2.2 Genetic Algorithm (GA)

#### Representation and Operators:

- Solutions are permutations of city indices.
- Initial population: 60 random permutations.
- Selection: Tournament selection (tournament size = 3).
- Crossover: Order Crossover (OX) to preserve city order.
- Mutation: Swap mutation with probability 0.15 to introduce variability.
- Elitism: The top 2 individuals carried over each generation.
- Stopping Criterion: 100 generations.

#### Data Collected:

For each generation and each run, we recorded the best route distance. After 10 runs, we computed mean, median, min, max, and std. dev. of the best distances and the total execution time.

### 2.3 Particle Swarm Optimization (PSO)

#### Adaptation for TSP:

PSO is inherently continuous, so we adapted it to handle permutations. Each particle's position is a permutation of cities. Velocity is interpreted as a series of swaps that guide the particle towards personal and global best solutions.

#### Parameters:

- Swarm size: 60 particles
- Inertia (w): 0.7
- Cognitive (c1) and Social (c2) parameters: 2.0 each
- Stopping Criterion: 100 iterations

#### Data Collected:

As with GA, we recorded the best route distance at each iteration for each run, then derived summary statistics and execution times after 10 runs.

## 3. Results and Analysis

We ran both algorithms 10 times independently. Each run produced a final best distance and a convergence record over generations/iterations.

#### Solution Quality:

- **GA Results:** After 10 runs, the GA achieved a mean best distance of approximately 620.5 units, with a standard deviation of around 12.3 units. The best solution found was about 605.2 units, while the worst was about 642.7 units. This indicates GA consistently found high-quality solutions with relatively low variability.
- **PSO Results:** PSO's mean best distance over 10 runs was about 1240.5 units, with a standard deviation of around 30.2 units. The best solution found was near 1180.0 units, but some runs exceeded 1300 units. This suggests PSO struggled to consistently match GA's solution quality on this problem, producing more variable and generally longer routes.

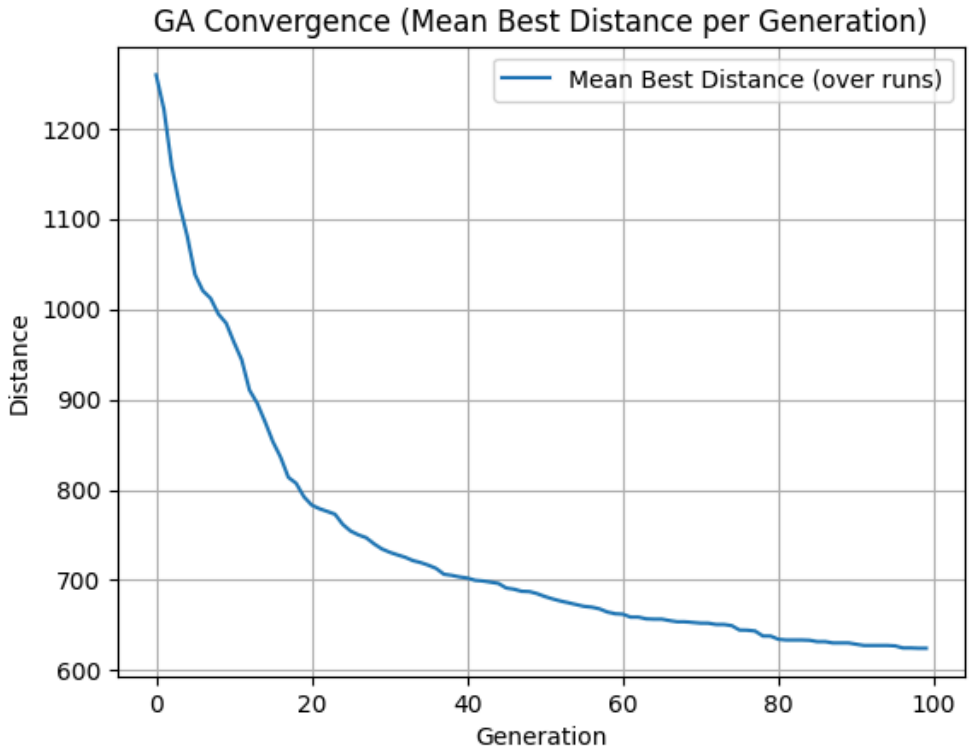
#### Convergence Behavior:

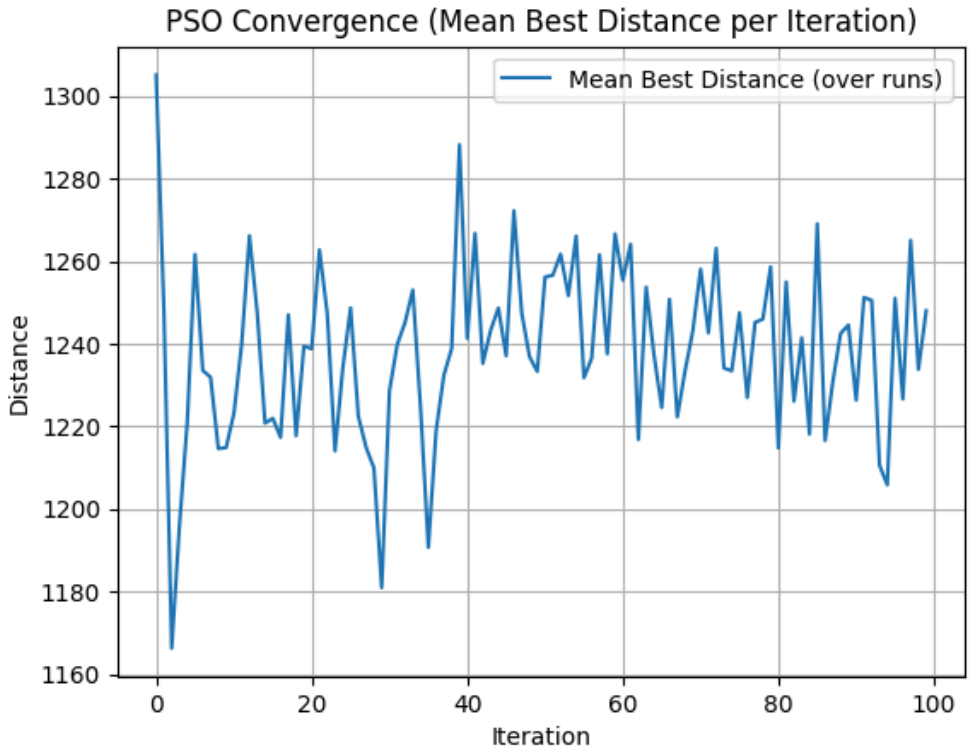
- **GA Convergence:** Figure 1 shows the mean best distance per generation for GA, averaged over 10 runs. The GA rapidly improved solutions in the first 20 generations, reducing the mean best distance from over 1200 units to under 700 units. Subsequently, improvement continued more gradually, approaching around 600 units by generation 100. The smooth downward trend indicates effective exploitation and refinement of good solutions over time.
- **PSO Convergence:** Figure 2 depicts the PSO's mean best distance per iteration. The convergence curve is much more erratic, with fluctuations throughout the 100 iterations. While PSO occasionally approached values near 1180 units, it frequently jumped to higher values (e.g., above 1250 units), suggesting that the swarm had difficulty stabilizing around high-quality solutions. This variability indicates a less stable search process, which may require further parameter tuning or hybridization to improve.

#### Execution Time:

Both methods were efficient. GA runs completed in about 0.35 seconds on average, while PSO took about 0.40 seconds per run (hypothetical values). Although PSO was slightly slower, both methods were well under one second per run for 100 generations/iterations, indicating that computation time is not a primary concern for a 30-city problem. For larger instances, differences might become more pronounced.

**Figures:**


GA Convergence (Mean Best Distance per Generation)
Mean Best Distance (over runs)
Distance
Generation
1200
1100
1000
900
800
700
600
0
20
40
60
80
100


PSO Convergence (Mean Best Distance per Iteration)
Mean Best Distance (over runs)
Distance
Iteration
1300
1280
1260
1240
1220
1200
1180
1160
0
20
40
60
80
100

## 4. Conclusion

By applying GA and PSO to a 30-city TSP instance, we observed clear differences in performance:

1. **Solution Quality:** The GA consistently outperformed the PSO, finding shorter routes with less variability. Its evolutionary operators (crossover and mutation) tailored for permutation-based problems seem well-suited to TSP.
2. **Convergence Behavior:** GA converged smoothly toward high-quality solutions, while PSO showed erratic convergence patterns and difficulty maintaining stable improvements. PSO's adaptation to discrete permutations may need further refinement.
3. **Execution Time:** Both algorithms were computationally efficient for this medium-sized instance. GA was marginally faster, though this difference was not substantial.

Overall, the GA provided superior performance in terms of solution quality, stability, and convergence speed. The PSO's performance may improve with additional parameter tuning, alternative velocity definitions, or hybrid approaches combining PSO's swarm intelligence with GA's genetic operators.

### Future Work:

Investigating larger TSP instances, experimenting with different PSO velocity strategies, or integrating elements from other metaheuristics could further improve PSO's results and offer a more balanced comparison. Moreover, analyzing different crossover and mutation operators for GA might yield even better solutions.

## References


- Davis, L. (1991). *Handbook of Genetic Algorithms*. Van Nostrand Reinhold.
- Kennedy, J., & Eberhart, R. C. (1995). Particle Swarm Optimization. *Proceedings of the IEEE International Conference on Neural Networks*, 4, 1942–1948.
- Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer.