CVRP Problem

Michał Stasiak
264 214

1 Introduction

The aim of this project is to compare the effectiveness of selected metaheuristic methods in solving the CVRP (Capacitated Vehicle Routing Problem). This problem involves finding optimal routes for a fleet of vehicles that must serve a set of customers with specified demands, assuming a limited vehicle capacity.

2 Benchmark instances

The experiment uses instances from the classic CVRP Set A benchmark: A-n32-k5, A-n37-k6, A-n39-k5, A-n45-k6, A-n48-k7, A-n54-k7, A-n60-k9.

3 Algorithm Descriptions

3.1 Genetic Algorithm (GA)

The genetic algorithm operates on a population of solutions represented as permutations of customer nodes. In each generation, a new population is created using tournament selection, ordered crossover (OX), and mutation (swap or inversion). The fitness function considers depot returns and vehicle capacity. The algorithm stops after a fixed number of evaluation function calls.

3.2 Simulated Annealing (SA)

SA operates on a single solution, which is modified through mutation. Worse solutions are accepted with a probability depending on the current temperature, which gradually decreases. The process stops after a given number of evaluations.

3.3 Greedy Algorithm

The greedy algorithm builds a route iteratively by selecting the nearest available customer that can be served without violating the capacity constraint. It runs quickly but usually produces suboptimal results.

3.4 Random Search

This algorithm randomly generates permutations of customer nodes and remembers the best solution. It serves as a baseline for comparison.

4 Solution Representation

In all algorithms used in this project (GA, SA, Greedy, Random), a solution is represented as a list of vehicle routes. Each route is defined as a list of customer nodes assigned to a vehicle, without including the depot explicitly. The full solution is therefore a list of such sublists, for example:

```
[[1, 3, 4], [2, 5], [6]]
```

This corresponds to the following interpretation:

- Route 1: $0 \to 1 \to 3 \to 4 \to 0$
- Route 2: $0 \to 2 \to 5 \to 0$
- Route 3: $0 \to 6 \to 0$

Where `0` denotes the depot. The total cost of a solution is calculated as the sum of distances traveled along all such routes.

5 Genetic Algorithm Parameter Tuning

To further analyze the genetic algorithm, we experimented with six different configurations by changing the crossover (P_c) and mutation (P_m) probabilities. Each setup was evaluated on all seven instances with 10 runs each. Below are the average results for each configuration.

P_c	P_m	Best (avg)	Worst (avg)	Avg (mean)	Std (mean)
0.70	0.10	1003.36	1170.57	1061.58	68.75
0.90	0.10	994.38	1168.17	1070.96	67.78
0.50	0.10	975.39	1130.70	1075.17	68.83
0.70	**0.30**	**996.89**	**1126.38**	**1055.51**	**63.93**
0.70	0.05	1013.92	1152.13	1101.26	62.41
0.90	0.30	985.28	1126.65	1066.10	69.77

Table 1: Averaged GA performance across all CVRP instances for each parameter configuration

Conclusion: Based on tuning results, the configuration $P_c = 0.70$, $P_m = 0.30$ was selected for final comparison with other algorithms.

6 Results and Analysis

The table below compares the performance of the final selected GA configuration ($P_c = 0.70$, $P_m = 0.30$) against other metaheuristic and baseline methods across all benchmark instances.

Instance	GA_best	GA_worst	GA_avg	GA_std	SA_best	SA_worst	SA_avg	SA_std	Random	Greedy
A-n32-k5	843.95	982.88	903.74	43.98	832.78	903.51	859.05	22.12	1417.36	1146.40
A-n37-k6	991.45	1161.31	1075.90	57.50	982.20	1032.88	1006.88	12.66	1591.87	1341.88
A-n45-k6	1007.97	1184.40	1114.16	62.16	996.39	1127.42	1071.10	42.62	2027.93	1485.30
A-n39-k5	909.34	1041.46	966.29	45.16	884.40	966.31	924.58	27.66	1506.50	1032.57
A-n48-k7	1147.06	1404.58	1262.93	80.11	1163.83	1287.35	1216.81	45.28	2135.40	1476.57
A-n54-k7	1293.72	1527.87	1385.10	61.49	1214.53	1390.37	1324.88	46.18	2285.40	1433.64
A-n60-k9	1495.44	1726.41	1629.38	80.86	1458.93	1571.05	1512.18	38.01	2761.02	1969.98

Table 2: Final comparison of GA ($P_c = 0.70$, $P_m = 0.30$) with other algorithms

Experimental Parameters:

- Genetic Algorithm:
  - Population size: 100
  - Evaluation limit: 100000
  - Crossover: Ordered Crossover (OX), $P_c = 0.70$
  - Mutation: Swap/Inversion, $P_m = 0.30$
  - Selection: Tournament (size 5)
- Simulated Annealing:
  - Initial temperature: 1000
  - Cooling rate: 0.9999

Observations: Despite extensive parameter tuning, the Genetic Algorithm did not reach the same level of solution quality as Simulated Annealing, especially on larger instances. SA consistently achieved lower average and best-case costs. However, this does not imply that GA is inherently weaker. It remains a flexible and powerful method with strong exploration capabilities, and its performance may improve further with hybridization or problem-specific enhancements.

7 Visualizations

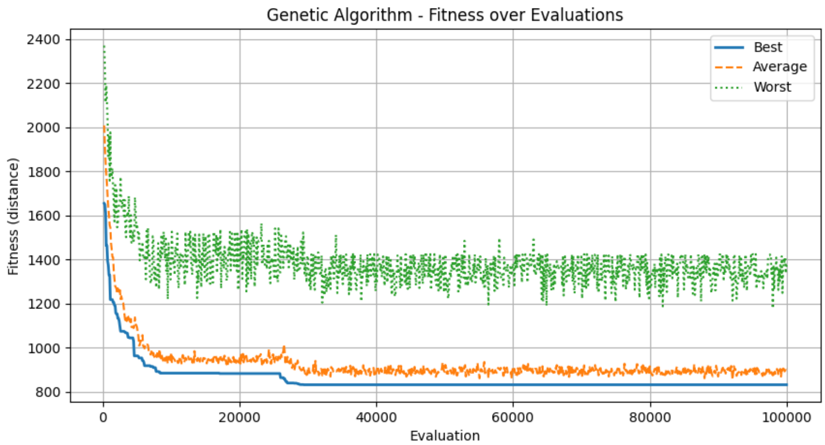

Figure 1: Fitness progression over evaluations for GA

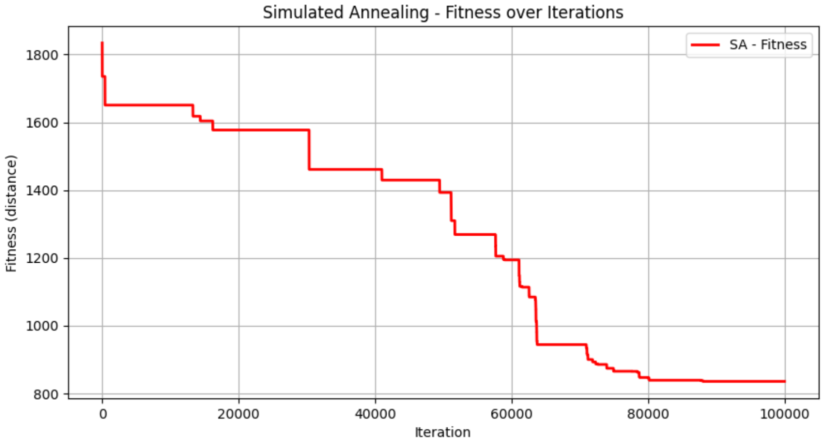

Figure 2: Fitness progression over evaluations for SA

8 Conclusions

This project compared multiple metaheuristic approaches for solving the CVRP problem. After tuning the Genetic Algorithm, the best-performing configuration ($P_c = 0.70$, $P_m = 0.30$) was selected for final evaluation. Although the GA provided competitive and diverse solutions across all instances, Simulated Annealing outperformed it in terms of both average and best-case results, particularly on larger and more complex benchmark sets.

Nevertheless, the Genetic Algorithm remains a promising and flexible approach with high exploratory potential. Its performance could be further improved through hybridization, adaptive parameter control, or integration with local search techniques. Both GA and SA clearly surpass simple baseline methods such as Greedy or Random Search, highlighting the strength of metaheuristic methods in tackling combinatorial optimization problems such as CVRP.