Faculty of Engineering & Technology

Electrical & Computer Engineering Department

ENCS3340, ARTIFICIAL INTELLIGENCE

Project 1

Package Delivery Optimization Using Simulated Annealing and Genetic Algorithm

---

Prepared by:

Name: Yahya Sarhan **Number:**1221858

Instructor: Yazan Abu Farha

Section: 1

Date: 2-5-2025

Introduction

The aim of this project is to find the best way to deliver packages using one or more cars, which is a typical delivery difficulty. Every vehicle may only carry a certain amount of weight and begins at a shop at (0, 0). The packages' priorities, weights, and locations vary.

The issue lies in assigning the packages to the vehicles and determining the best route for every vehicle in order to:

- No car can carry more than it can.
- All vehicles travel the shortest possible total distance.

Additionally, packages with a higher priority are (optionally) regarded as more important along the route.

I applied the Genetic Algorithm and Simulated Annealing as two creative search strategies to solve this. These methods are often used in AI to solve complex issues effectively, particularly when exploring every alternative would be too time-consuming. I have created a basic application with a graphical user interface that allows the user to view the delivery routes on a map, select an algorithm, and load the input file. This enables testing and comparing the outcomes of the two methods.

Table of contents

Contents

Problem Description

In this project, I'm trying to figure out the most intelligent and effective way to move packages using a number of vehicles. The store, which is always at location (0, 0), is where every vehicle begins. Every car has a weight restriction; it can only support a specific amount of weight.

Each package is different from the others. All of them possess:

- A specific location (x and y coordinates) on the map
- A weight
- A priority (with 1 being the highest on a scale of 1 to 5)

Assigning these packages to the cars and determining the order in which each vehicle should deliver its packages are the objectives. However, the solution has to conform to some important rules when achieving this:

- A car shouldn't be loaded with more than it can manage.
- Every vehicle should travel a minimum distance as possible overall.
- High-priority packages (like priority 1 or 2) can affect the route by making them more "urgent" to deliver.

In summary, the challenge is to efficiently pack and route trucks while following to weight and priority regulations in order to save time and distance.

Algorithms Used.

3.1 Simulated Annealing

An intelligent method for finding quality solutions is Simulated Annealing (SA). It takes inspiration from the cooling of hot metal, where atoms gradually shift into more advantageous places as the temperature drops.

The algorithm in our case begins with a random delivery plan. It then gradually makes small changes (such as switching two items or relocating a package to a different car) and determines whether the new approach is better.

It accepts the new plan if it is better. If it's worse, there's little likelihood that it will accept it. The "temperature" value that determines this probability decreases with time. The algorithm freely explores in the beginning but then focuses more on getting better.

how I implemented it:

- I start with developing a random solution in which packages are distributed among trucks at random while staying within capacity.
- After that, I try to make a neighbor (a small change) in order to improve the solution.
- I developed three different types of neighbor operations:
- In the same car, switch two packages.
- Transfer a package from one car to another, if there is space in the other car.
- Transfer a package between two distinct cars.
- For every option, I determine the objective cost (total distance + priority penalty).
- I'll accept the new solution if it's better. If it's worse, I sometimes accept it, however the probability varies with the temperature.
- The program stops when the temperature gradually drops to a minimum limit.

The final output is the best solution found during the process, and I also show the route visually on a map using Matplotlib.

3.2 Genetic Algorithm.

By starting with a collection of possible answers and improving them generation by generation, the Genetic Algorithm simulates the concept of evolution. **how I implemented it:**

- I begin by creating a population of solutions at random.
- A fitness function which includes priority penalty and total distance is used to assess each solution.
- For each new generation:
- Using tournament selection, I choose two parent solutions.
- Using a balanced crossover, I generate a kid solution by mixing package assignments from both parents at random.
- I give the child a mutation in which I transfer certain packages at random to different cars (if space allowed).
- I keep note of the best solution after going through this process several times.
- Similar to SA, a plotted route map is used to visually represent the outcome.

Though more complicated, the GA often results in better outcomes, especially for complex or large-scale applications.

Test Case Results

Test Case 1: Basic Feasibility Test

Vehicles: 2 vehicles with 100 kg capacity each

Total packages = 4
Packages:

X=10, Y= 10, weight = 30, Priority = 1

20 20 40 2

30 30 50 3

40 40 60 4

Algorithms Applied:

- ☑ **Simulated Annealing (SA)**
- ☑ **Genetic Algorithm (GA**

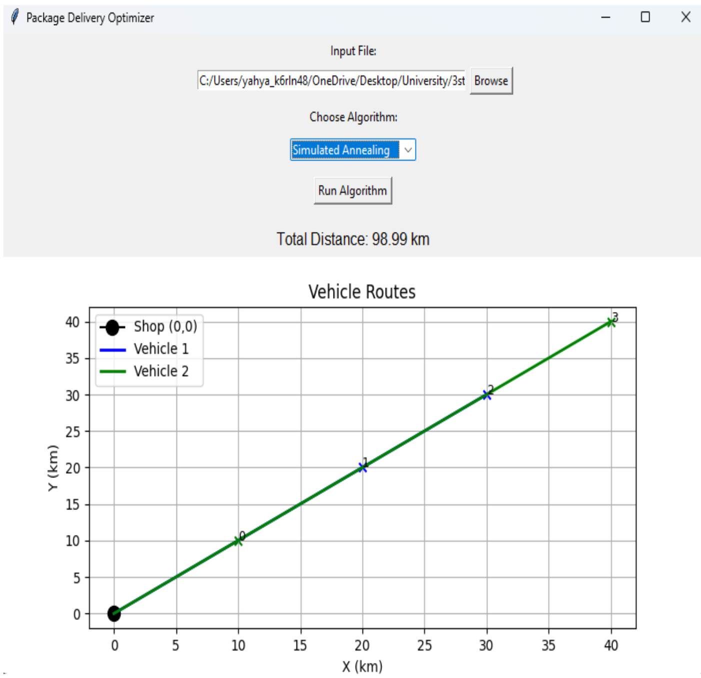
Metric	**Simulated Annealing**	**Genetic Algorithm**
Total Distance	**98.99 km**	**56.57 km**
Packages Assigned	**All**	**All**
Capacity Violations	**None**	**None**

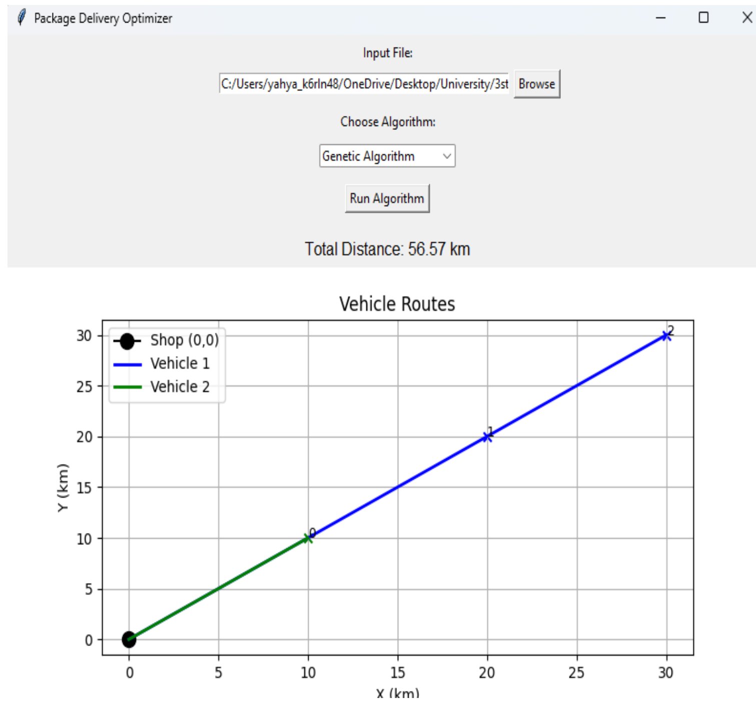
Observations:

- Within limits on capacity, both algorithms provided reliable results.
- In this particular case, the Genetic Algorithm performed significantly better than Simulated Annealing, reducing the overall distance by more than 40 km.

Based on the route plots:

- SA took longer routes but utilized both cars about equally.
- GA placed packages which were closer together and improved route optimization.





Test Case 2: Priority Handling Test

Vehicles: 1 vehicle with 100 kg capacity

Total packages = 3
Packages:

10 10 50 1

20 20 50 2

30 30 50 3

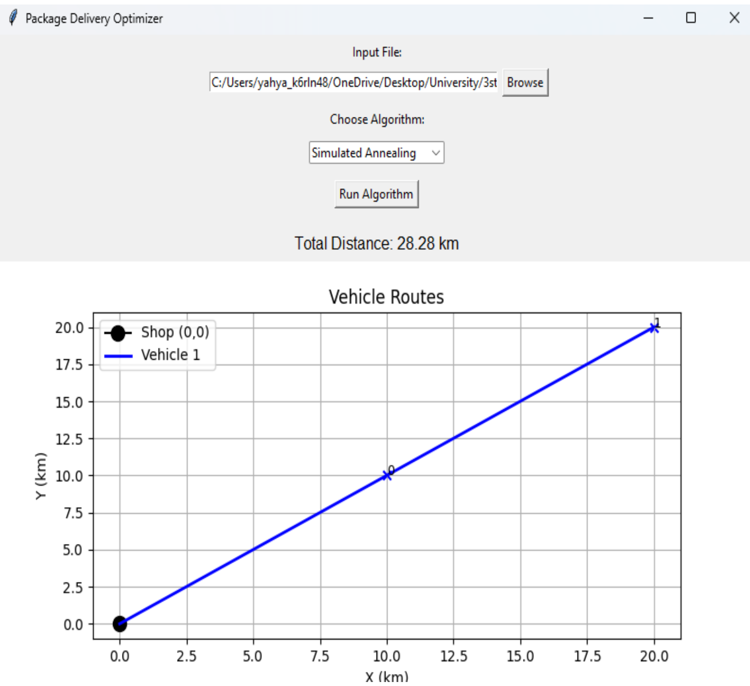
Algorithms Applied:

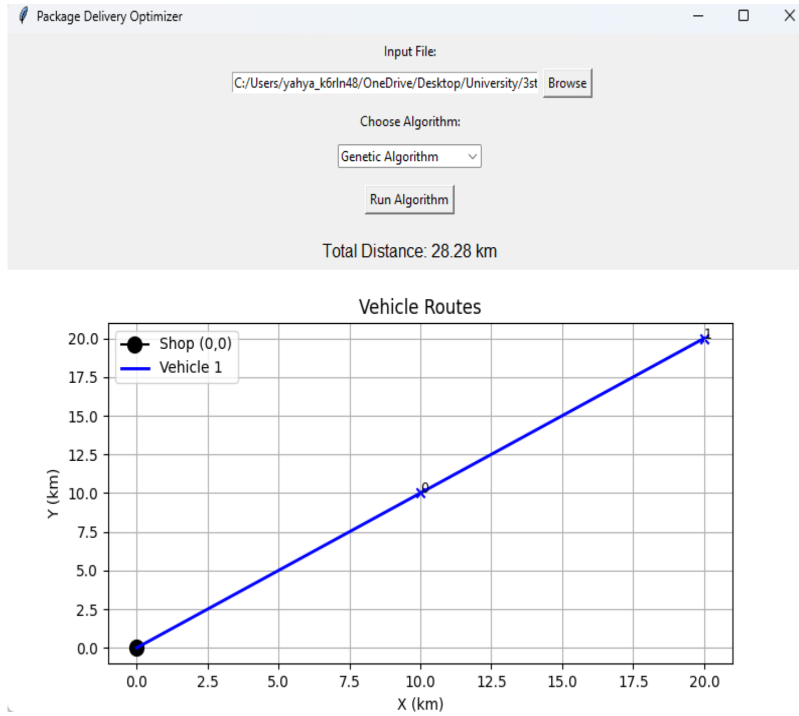
- ☑ **Simulated Annealing (SA)**
- ☑ **Genetic Algorithm (GA**

Metric	**Simulated Annealing**	**Genetic Algorithm**
Total Distance	**28.28 km**	**28.28 km**
Packages Assigned	**2**	**2**
Capacity Violations	**None**	**None**
Packages Skipped	**1**	**1**

Observations:

- Since the vehicle can only carry two out of the three 50 kg packages, both algorithms were forced to skip one.





Test Case 3: Distance Optimization Test

Vehicles: 2 vehicles, each with 100 kg capacity

Total packages = 6
Packages:

10 10 20 1

20 20 30 2

80 80 25 3

85 85 30 4

15 15 10 2

90 90 20 5

Algorithms Applied:

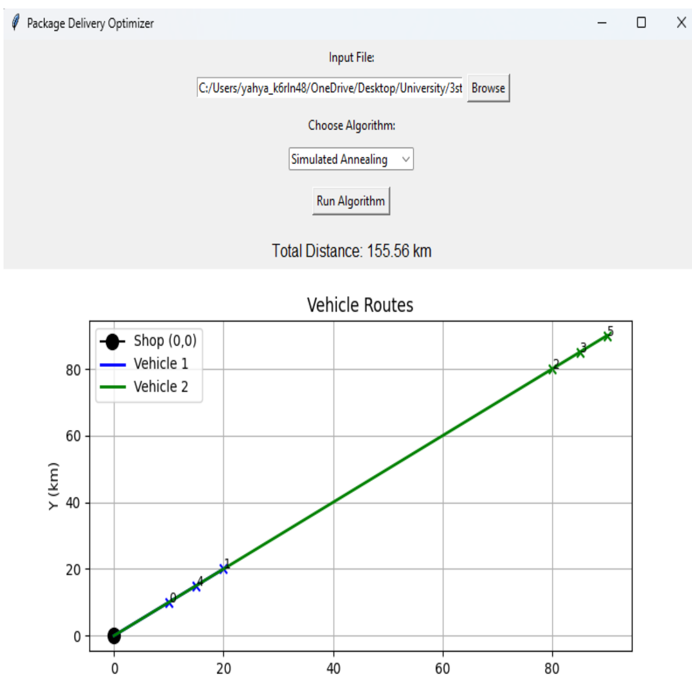
- ☑ **Simulated Annealing (SA)**
- ☑ **Genetic Algorithm (GA**

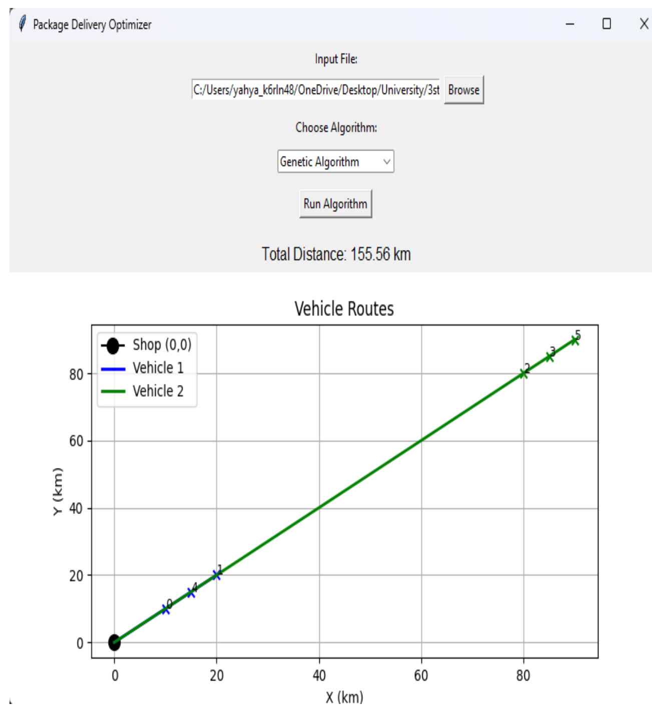
Metric	Simulated Annealing	Genetic Algorithm
Total Distance	155.56 km	155.56 km
Packages Assigned	ALL	ALL
Capacity Violations	None	None
Packages Skipped	None	None

Observations:

1. Both algorithms achieved the **same total distance**, which suggests this may be close to the optimal route.
2. The GUI route plots show:
   - Vehicle 1 handled the close-range deliveries.
   - Vehicle 2 took the longer ones, balancing the load and distance well.

3.This case shows that both algorithms are capable of **finding efficient routes**, even when package distances vary greatly.





Test Case 4: Edge Case - Overcapacity Package

Vehicles: 2 vehicles, each with 100 kg capacity

Total packages = 1
Packages:

50 50 150 1

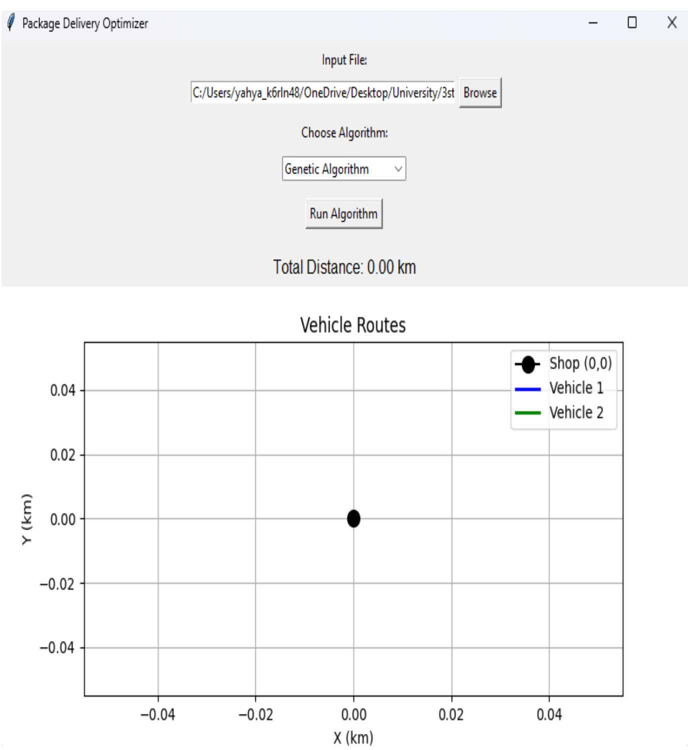
Algorithms Applied:

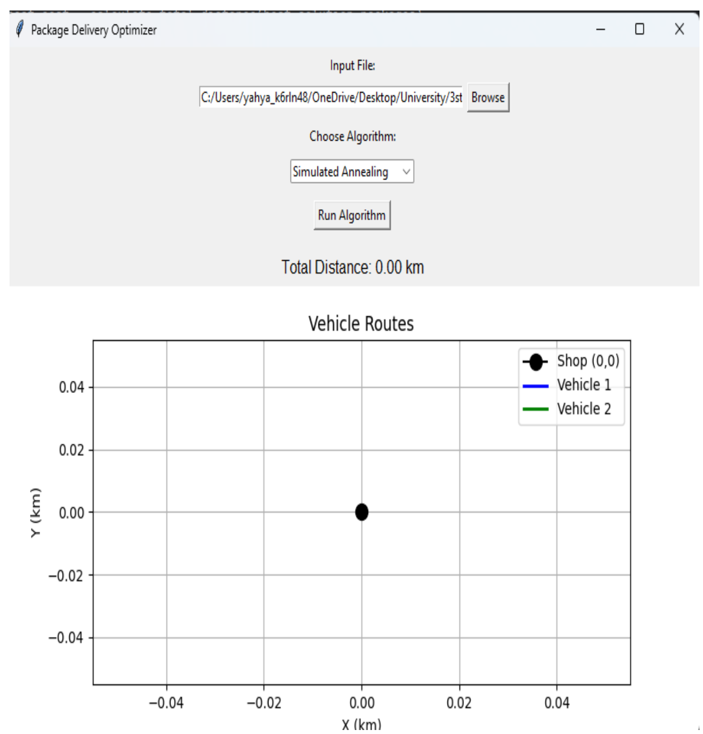
- ☑ **Simulated Annealing (SA)**
- ☑ **Genetic Algorithm (GA)**

Metric	Simulated Annealing	Genetic Algorithm
Total Distance	**155.56 km**	**155.56 km**
Packages Assigned	**None**	**None**
Capacity Violations	**None**	**None**
System Crash	**NO**	**NO**

Observations:

- This test verifies that neither algorithm crashed and that it rejected the large package correctly.
- Since there were no deliveries, the total distance is zero kilometers.
- As expected, the plot just displays the store at (0,0), with no delivery routes.
- The system's ability to carefully handle extra data is confirmed by this test.





Test Case 5: Simulated Annealing vs. Genetic Algorithm Comparison

Vehicles: 3 vehicles, each with 100 kg capacity

Total packages = 10
Packages:

10 10 20 1

20 30 15 2

40 10 30 3

60 60 25 4

15 15 10 2

80 90 35 1

25 25 15 3

50 50 30 2

35 45 10 1

45 65 20 5

Algorithms Applied:

- ☑ **Simulated Annealing (SA)**
- ☑ **Genetic Algorithm (GA)**

Metric	**Simulated Annealing**	**Genetic Algorithm**
Total Distance	**250.23 km**	**252.51 km**
Packages Assigned	**ALL**	**ALL**
Capacity Violations	**None**	**None**

Observations:

All packages were successfully assigned and routed by both algorithms while staying under the vehicles' capacity restrictions.
In this case, Simulated Annealing performed somewhat better than Genetic Algorithm, covering around 2.3 km less ground overall.
According to the route plots,

SA took a more structured, spread-out approach.

GA created more clustered assignments, but still valid.

This test confirms that both algorithms are effective and perform closely on medium-sized inputs.

Results & Discussion

I was able to observe the performance of both algorithms in various delivery problem types after executing the test cases using both Simulated Annealing (SA) and Genetic Algorithm (GA). The results showed that, depending on the situation at hand, each algorithm offers advantages and disadvantages.

Both SA and GA produced correct answers in straightforward test cases, such as Test Case 1, however GA outperformed SA by determining a shorter total distance. This is due to GA's greater ability to use its population to look into multiple possibilities simultaneously.

Test Case 3 showed that even when packages are dispersed over several locations, both SA and GA can produce excellent results. They discovered the same overall distance in this instance, which may be almost ideal.

Both methods correctly rejected the package and avoided crashing in the edge situation (Test case 4), where the package was too heavy for any vehicle. This showed how safely the program manages odd inputs.

Test Case 5 was the most interesting comparison. SA slightly outperformed GA, showing that sometimes SA is better at fine-tuning the solution, especially in medium-sized problems. However, the difference was small.

Based on these results, I can say that genetic algorithms are better for exploring a large number of alternatives, particularly when the problem is complex or has several alternative combinations. Simulated Annealing is greater at making small, intelligent adjustments and occasionally produces better outcomes in straightforward or basic situations.

In every test, both algorithms respected the vehicle's capacity, and the GUI displayed every route in detail. It was quite easy to comprehend how the products were assigned and delivered thanks to the visual display.

Conclusion:

In this project, i worked on solving a delivery optimization problem using two smart AI algorithms: Simulated Annealing and Genetic Algorithm. The goal was to assign packages to vehicles in a way that minimizes the total distance traveled, while making sure no vehicle goes over its weight limit and optionally considering package priorities.

I implemented both algorithms from scratch using Python, and built a user interface that allows the user to load input files, choose which algorithm to run, and see the delivery routes on a map. This helped make the project more interactive and easier to test.

I realized that each method has advantages after testing and contrasting them in different scenarios:

- Because it considers many different options simultaneously, the genetic algorithm performs better when the problem is large or complex.
- Simulated Annealing is effective at gradually enhancing a single solution, and it often produces better outcomes in smaller or intermediate cases.

Both algorithms produced reliable answers while keeping to all of the problem's limitations. Additionally, they managed edge scenarios without having any crashes.

Additional real-world features like delivery window times, fuel prices, traffic jams, or map-based coordinates could be added to this project in the future to make it better. All things considered, though, this project was a fantastic way to learn how AI can be used to intelligently and effectively address real-world issues like package delivery.