Faculty of Engineering &Technology

Electrical & Computer Engineering Department

Artificial Intelligence-ENCS3340

Project#1

Prepared by:

Name: Mohammad Omar Azam **ID**:1212429

Name: Abdallah Khawaja **ID**:1220152

Instructor: Dr. Yazan Abu Farha

Section: 4

Date:4/5/2025

Abstract

The objective of this project is to develop a system that efficiently assigns packages to available delivery vehicles in a way that minimizes the total travel distance covered by all vehicles. Each package must be delivered from a central shop location to its designated destination, with both locations represented as two dimensional coordinates (x, y).

To solve this optimization problem, the project will implement and compare two metaheuristic approaches: **Genetic Algorithms (GA)** and **Simulated Annealing (SA)**.

These techniques are well-suited for exploring large solution spaces and finding near-optimal solutions for complex, constraint-driven problems like our project.

Table of Contents

Table of Figures :

Table of Tables:

Explain our code:

1.Initiation

We have a main equation that we will use in the two algorithms, and this equation is the basis of the project. When we want to balance between priority and distance, we cannot multiply the distance by the priority because that will give us a number that has no proportion and we want the priority equation to have importance, as well as the distance to get the near-optimal solution.

This equation is called the function →**Score**

Score = α*Distance+ β*Priority

- The advantage of this equation lies in its balance between long distances and priority. According to this project, the distance is defined as between 0 and 100 kilometers.

  We calculate distance by using Euclidean distance formula

  This makes the selection rate for a package with a low priority and a long distance sometimes higher than the selection rate for a package with a slightly smaller distance and a higher priority.
- The basis for this is balance. To achieve this balance, we assume the presence of alpha and set its value equal to one, since the distances are large. We also assume beta equals five, so that the priority is close to the distance ratio, since priority is a small number, while distance is a large number.

At the beginning of the algorithms, we make a point of difference in our approximation to the best cost, so we start distributing the packages randomly to each car, taking into account the capacity of each car. After that, a car enters the function score so that the packages are arranged perfectly. Then, the array is called for each car, and within each array are the packages, and from here the work of the algorithms begins.

2.Simulated Annealing

Explain how simulated annealing in our code working:

After the first stage, which is the random distribution of packages and their ordering for each car, all cars enter the algorithm.

Within this algorithm, a temperature of 1,000 is defined, a cooling ratio of 0.95, and the number of iterations for each time the temperature drops is 100.

In each main cycle, the temperature is multiplied by the cooling ratio, resulting in 100 iterations in which we perform a change in the packages.

When the cars enter, four random operations are performed:

1. Transfer a package from one car to another, taking into account the capacity of the car.

2. Swap two packages between two different cars.

3. Randomly shuffle the packages inside the car.

4. Reverse the packages within specific car.

The cars enter the first round and one hundred iterations are performed on them.

In each iteration, one of the four steps mentioned previously occurs, when that occur the code check the capacity of cars, thus the score value for each car changes with each iteration by sending all cars to score function, then the current value is compared with the new value, and we begin to examine the lowest percentage.

We assume three cases:

Case1: In any case where the temperature is as high as possible or as low as possible and the new score value is lower than the old score value, the order is taken immediately.

Case2: If the new score is higher than the old score and the temperature is also high, it is possible to take the new score based on the probability equation. In this case, it is likely to take the new value despite its badness because the probability will be high due to the high temperature (at first)

Case3: If the new score is higher than the old score and the temperature is low, it is not possible to take the new score because the probability will be low due to the low temperature (after time)

- When temperature equal one after10000-13500 iteration the SA will plot each car with its near-optimal path

Test Cases

1st test case:

Vehicle 1 = 20 kg Vehicle 2= 15 kg Vehicle 3 = 25 kg

Package Number	(x , y)	Weight	Priority
P1	(20,30)	5 k	1
P2	(10,10)	3 k	2
P3	(70,50)	8 k	1
P4	(50,60)	4 k	2
P5	(80,80)	9 k	3
P6	(5,15)	6 k	1
P7	(60,10)	7 k	4
P8	(90,90)	5 k	5
P9	(30,25)	3 k	2
P10	(45,55)	4 k	3

Table 1: Example #1 (SA)

Result →

Final Delivery Cost = 550.24 km

Vehicle 1: used 19.0/20 kg → [6, 1, 10, 4]
Vehicle 2: used 13.0/15 kg → [2, 9, 7]
Vehicle 3: used 22.0/25 kg → [3, 5, 8]

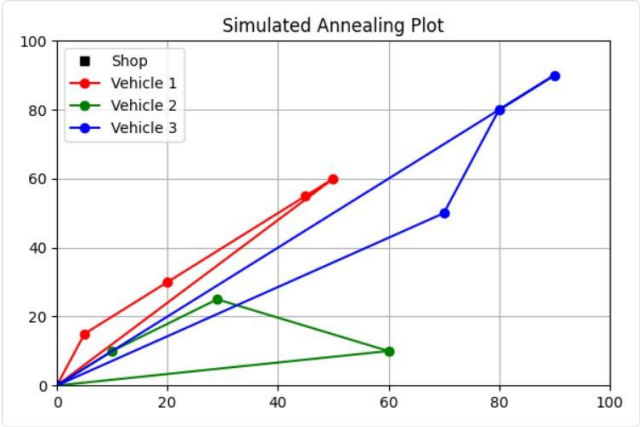

Figure 1: Example #1 (SA)

- We note in the figure that the distribution of the packages is appropriate for the vehicle capacity, and the score equation yields a result.
- Consequently, the vehicles are distributed according to the algorithm and the score equation, and cost to return to their starting point(origin) is calculated. The total cost is calculated by adding the distance according to the function(true_cost) and adding the return cost for each vehicle.

2nd test case:

Vehicle 1 = 35 kg Vehicle 2= 34 kg Vehicle 3 = 30 kg

Package Number	(x, y)	Weight (kg)	Priority
P_1	(28, 41)	3.2	2
P_2	(75, 62)	5.7	3
P_3	(33, 77)	6.1	1
P_4	(41, 38)	4.4	2
P_5	(64, 64)	8.0	1
P_6	(71, 25)	7.3	4
P_7	(52, 28)	9.8	3
P_8	(26, 61)	5.2	2
P_9	(48, 44)	2.9	1
P_{10}	(36, 70)	6.6	2
P_{11}	(73, 31)	3.5	3
P_{12}	(60, 55)	4.1	1
P_{13}	(44, 29)	6.4	2
P_{14}	(57, 69)	8.7	4
P_{15}	(22, 46)	7.1	3

Table 2: Example #2 (SA)

Result →

Final Delivery Cost = 561.16 km

Vehicle 1: used 33.700/35 kg → [8, 15, 3, 10, 14]
Vehicle 2: used 31.500/34 kg → [9, 12, 5, 2, 11, 6]
Vehicle 3: used 23.800/30 kg → [1, 4, 13, 7]

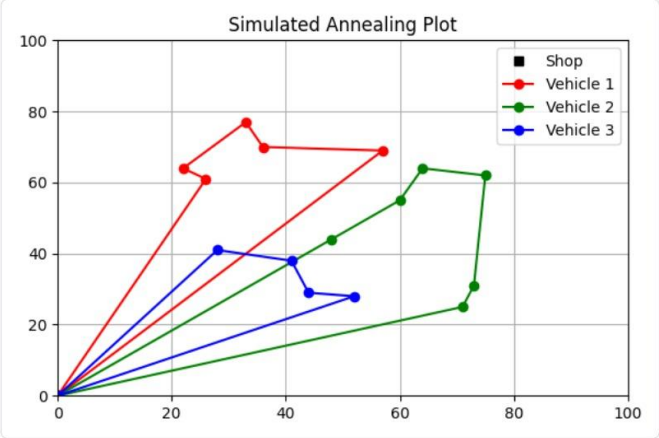

Figure 2: Example #2 (SA)

3.Genetic Algorithm (GA)

In this algorithm, we apply the first step (Initiation) by randomly distributing the packages to the cars, and then arranging each car according to the score equation. Then, the algorithm (GA) takes the following variables:

1.Population size = 80

2.mutation rate = 0.05

3.Genaration = 500

The cars enter our algorithm, and generate group of guessing (population) each guess named as chromosome, after that we need to get the best chromosomes of our population according to the fitness which is our score equation, before doing crossover we check the capacity of the car, then we do crossover between two packages from different best chromosome that we get, then get the child from the parent (best chromosomes).

After we get children from crossover, we need avoid getting stuck in local optima, thus we need a random swap between two packages according to mutation rate.

We put the mutation rate 0.05 because we have a chance with 0.05 to swap them. that means we take a randomly number arranged between 0-1, if the number is less than 0.05, we swap else we don't do anything.

After that we generate the next generation and repeated it 500 time.

Finally, we get the best final solution.

If the mutation rate is above our value assume 0.6 the algorithm will become explorative (try more random changes), it can be good early to discovered diverse solution but it slow or break good patterns. assume mutation rate equal 1 it will have very high randomness like pure search but its bad stability and there are no exceptions. because of that our value is balance mutation have best stability, very save and it will not break good solution keeps getting stuck in same score.

Test Cases

1st test case:

Vehicle 1 = 20 kg Vehicle 2= 15 kg Vehicle 3 = 21 kg

Package Number	(x, y)	Weight (kg)	Priority
P_1	(30, 58)	2.6	1
P_2	(68, 44)	4.8	3
P_3	(42, 79)	6.9	2
P_4	(50, 33)	3.1	1
P_5	(74, 69)	7.7	4
P_6	(61, 21)	5.5	2
P_7	(29, 65)	2.9	1
P_8	(38, 46)	4.2	2
P_9	(65, 59)	6.3	3
P_{10}	(57, 40)	3.8	2

Table 3: Example #1 (GA)

Result →

Final Delivery Cost = 565.35 km

Vehicle 1: used 18.300/20 kg → [8, 10, 2, 6]
Vehicle 2: used 12.400/15 kg → [1, 7, 3]
Vehicle 3: used 17.100/21 kg → [4, 9, 5]

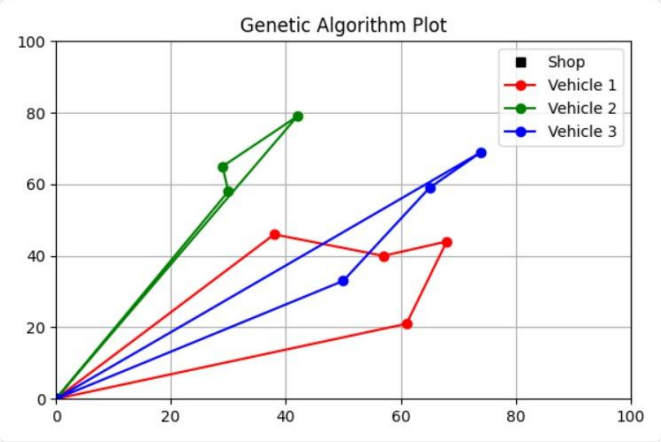

Figure 3: Example #1 (GA)

2nd test case:

Vehicle 1 = 30 kg Vehicle 2= 25 kg Vehicle 3 = 26 kg

Package Number	(x, y)	Weight (kg)	Priority
P_1	(35, 73)	4.7	3
P_2	(60, 41)	3.9	2
P_3	(78, 33)	5.2	4
P_4	(25, 67)	6.8	1
P_5	(40, 52)	2.3	1
P_6	(70, 38)	4.5	3
P_7	(59, 61)	7.6	2
P_8	(26, 45)	3.4	1
P_9	(63, 77)	8.2	4
P_{10}	(31, 55)	4.0	2
P_{11}	(47, 69)	6.1	3
P_{12}	(53, 29)	3.3	2
P_{13}	(22, 63)	2.8	1
P_{14}	(76, 58)	5.7	4
P_{15}	(44, 36)	6.0	2

Table 4: Example #2 (GA)

Result →

Final Delivery Cost = 613.94 km

Vehicle 1: used 28.600/30 kg → [15, 12, 2, 6, 3, 14]
Vehicle 2: used 24.200/25 kg → [5, 7, 11, 9]
Vehicle 3: used 21.700/26 kg → [8, 10, 13, 4, 1]

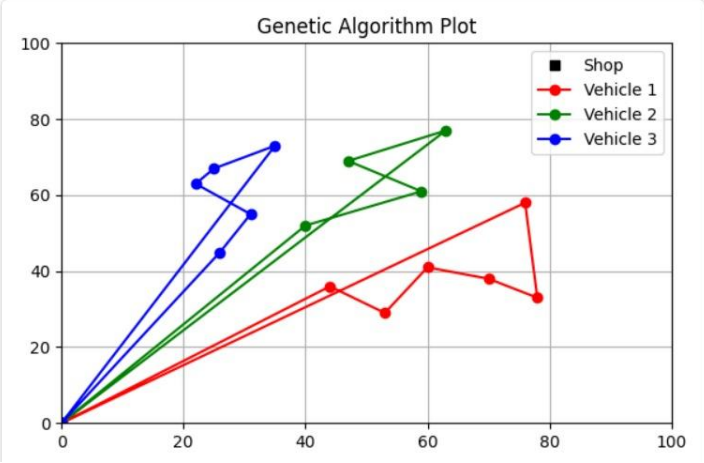

Figure 4: Example #2 (GA)

Handle Cases

These cases handling many situations according to the capacities of cars and the priority of packages dependent on score function and algorithm.

Case #1

Vehicle #1 capacity(kg)= 100 kg Vehicle #2 capacity(kg)=100 kg

Package Number	(x, y)	Weight(kg)	Priority
P1	(20,25)	30	3
P2	(10,30)	40	2
P3	(60,60)	50	2
P4	(5,5)	60	1

Table 5: Handle Case #1

Result →

Figure 5: Handle Case #1 using (SA)

Case #2

Vehicle #1 capacity(kg)=100 kg

Package Number	(x, y)	Weight(kg)	Priority
P1	(12,20)	50	1
P2	(8,18)	50	2
P3	(10,15)	50	3

Table 6: Handle Case #2

Result →

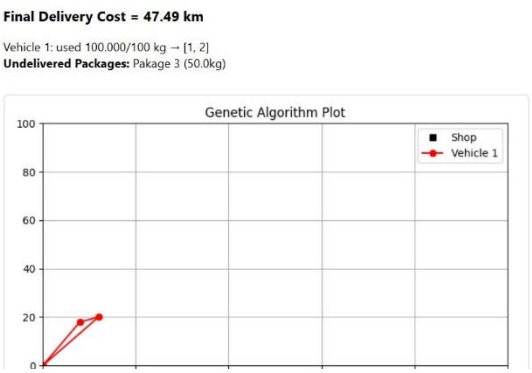

Figure 6: Handle Case #2 using (GA)

Case #3

Vehicle #1 capacity(kg)=100 kg Vehicle #2 capacity(kg)=100 kg

Package Number	(x, y)	Weight(kg)	Priority
P1	(5,9)	20	1
P2	(20,23)	30	3
P3	(81,79)	25	2
P4	(85,90)	15	3
P5	(50,55)	10	1
P6	(64,65)	10	2

Table 7: Handle Case #3

Result →

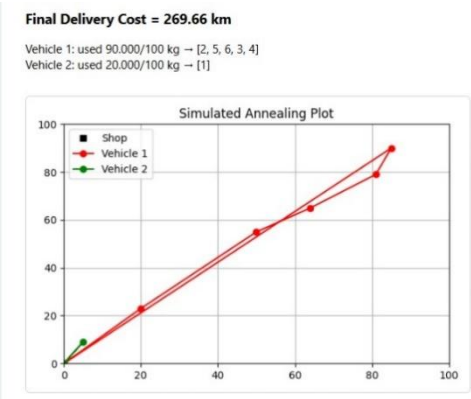

Figure 7: Handle Case #3 using (SA)

Case #4

Vehicle #1 capacity(kg)=100 kg Vehicle #2 capacity(kg)=100 kg

Package Number	(x, y)	Weight(kg)	Priority
P1	(25,14)	150	1

Table 8: Handle Case #4

Result →

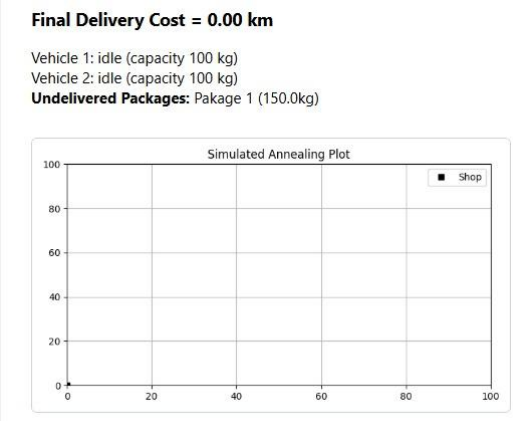

Figure 8: Handle Case #4 using (SA)