Faculty of Engineering & Technology

Electrical & Computer Engineering Department

Artificial Intelligence – ENCS3340

Project #1

Optimization Strategies for Local Package Delivery Operations

---

Student #1: Khaled Abu Lebdeh

Student Num.: 1220187

Student #2: Abdalraheem Shuaibi

Student Num.: 1220148

Table of Contents

Table of Figures

Test Cases

- Test Case #1: Basic Feasibility Test

o Vehicles: 2 vehicles, each with a capacity of 100 kg.

o Packages: 4 packages with the following weights: 30 kg, 40 kg, 50 kg, and 60 kg.

```
=== Initial vehicle assignments ===
Vehicle v1: used 100.0/100.0, free 0.0
   - p2: loc=(10,70), w=40, pr=3
   - p4: loc=(14,78), w=60, pr=2
Vehicle v2: used 80.0/100.0, free 20.0
   - p1: loc=(20,12), w=30, pr=1
   - p3: loc=(20,12), w=50, pr=1
====================================
```

Figure 1: Basic Feasibility Test

o Packages are distributed among vehicles such that no vehicle carries more than 100 kg. (achieved)

o All packages are assigned to vehicles. (achieved)

o Total weight per vehicle ≤ 100 kg. (achieved)

o No unassigned packages remain.

- Test Case #2: Priority Handling Test

o Vehicles: 1 vehicle with a capacity of 100 kg.

o Packages:
- ▪ Package #1: 50 kg, Priority 1, destination: (20,12)
- ▪ Package #2: 50 kg, Priority 2, destination: (10,70)
- ▪ Package #3: 50 kg, Priority 3, destination: (90,90)

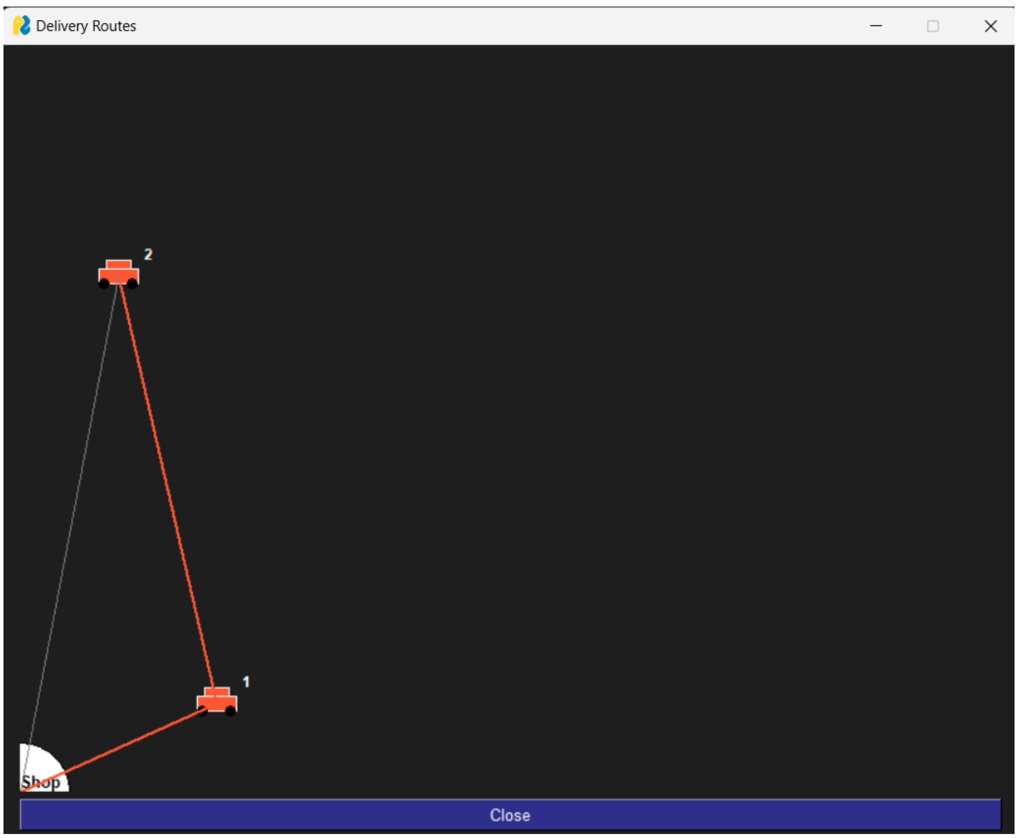


Figure 2: Priority Handling Test

o Packages #1 and #2 are selected for delivery due to higher priority.

o Package #3 is deferred or unassigned due to capacity constraints.

- Test Case #3: Distance Optimization Test

o Vehicles: 2 vehicles, each with a capacity of 100 kg.

o Packages: 6 packages located at varying distances from the depot.

Packages List

package_id	dest_x	dest_y	weight	priority	is_delivered
p1	1	12	7	1	False
p2	8	40	6	2	False
p3	7	80	11	3	False
p4	80	10	4	1	False
p5	90	50	6	2	False
p6	95	89	3	3	False

Figure 3: packages in case #3

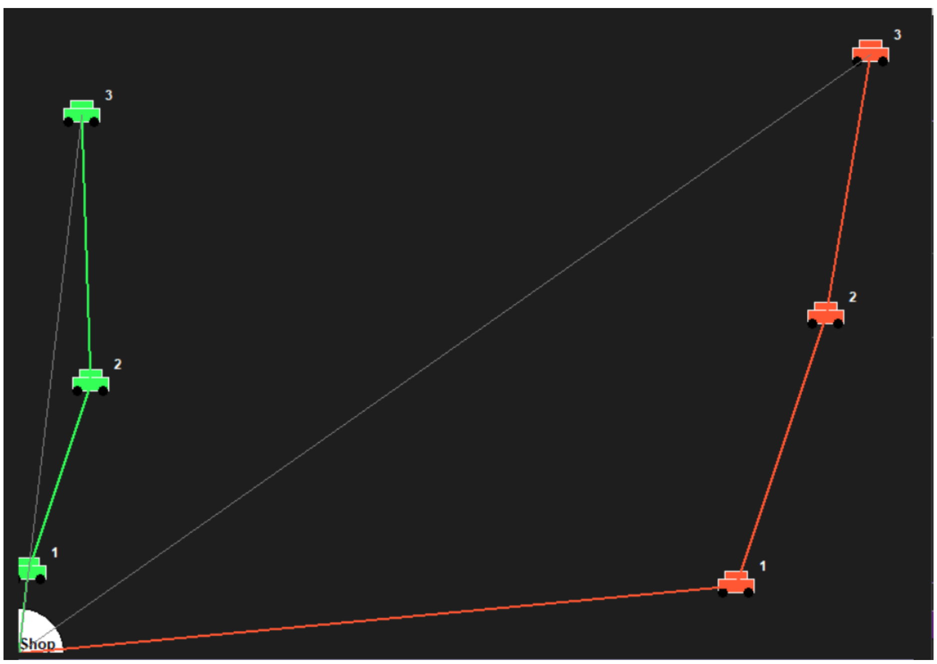


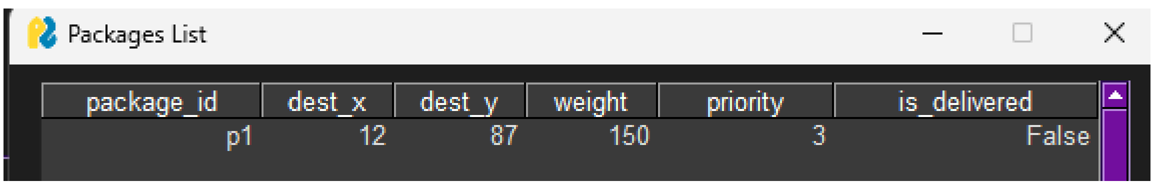
Figure 4: vehicles in test case #3

Figure 5: total distance in test case #3

o Packages are assigned and routed to minimize the combined distance travelled by both vehicles.

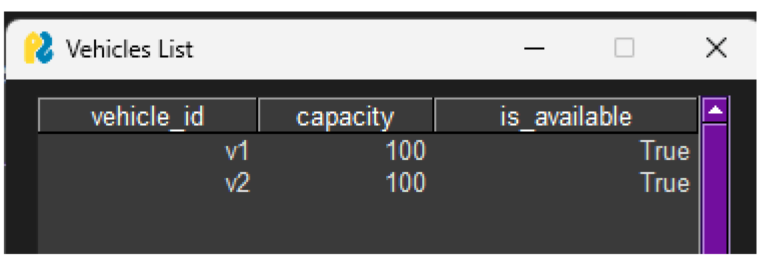
Test Case #4: Edge Case - Overcapacity Package

- **Input**:
  - **Vehicles**: 2 vehicles, each with a capacity of 100 kg.
  - **Packages**: 1 package weighing 150 kg.



package_id	dest_x	dest_y	weight	priority	is_delivered
p1	12	87	150	3	False

Figure 6: test case - 150kg Pack



vehicle_id	capacity	is_available
v1	100	True
v2	100	True

Figure 7: test case - 100kg (2) vehicles

- **Output:**

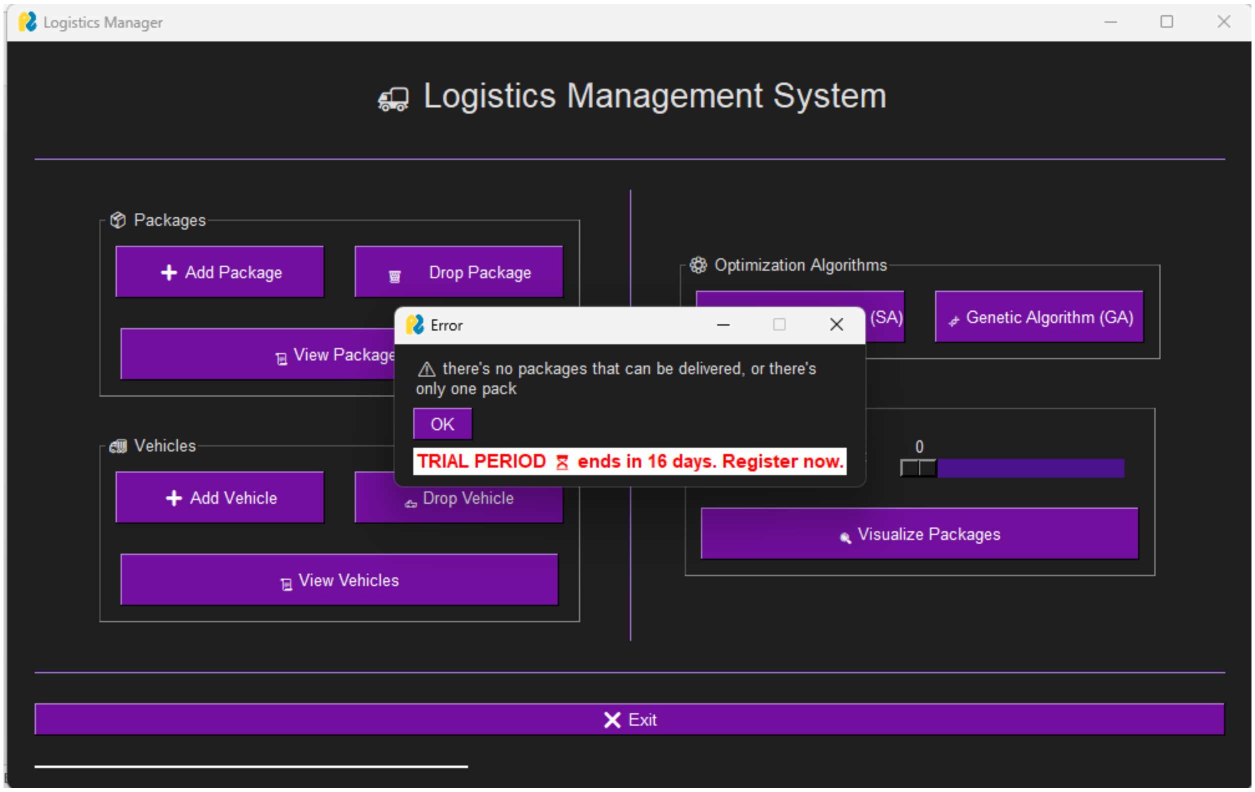


Figure 8: Error message

Test Case #5: Simulated Annealing vs. Genetic Algorithm Comparison

- **Input**:
  - **Vehicles**: 3 vehicles, each with a capacity of 100 kg.
  - **Packages**: 10 packages with varying weights and priorities.

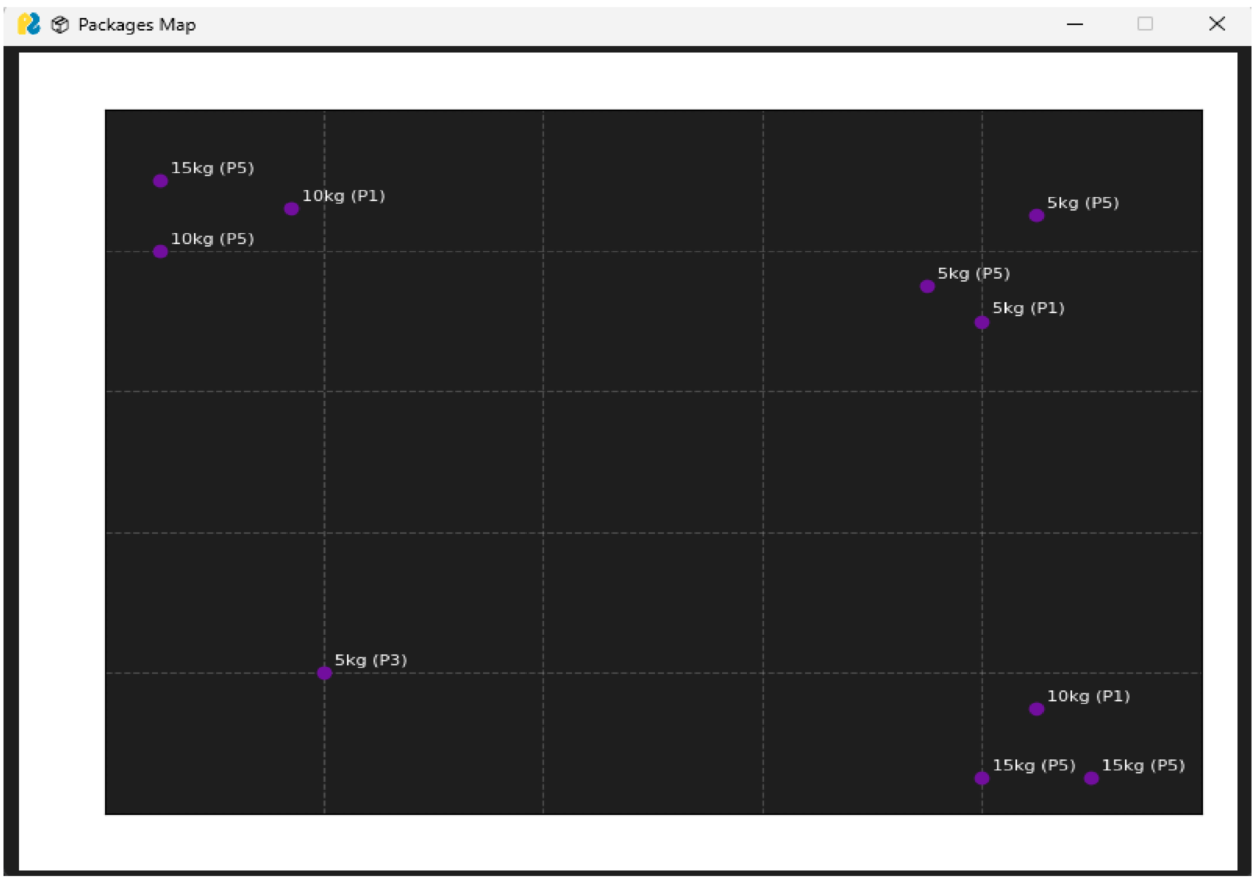
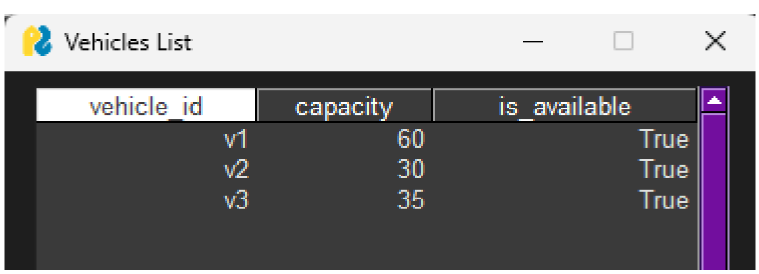


Figure 9: test case - 10 Packs


Vehicles List

vehicle_id	capacity	is_available
v1	60	True
v2	30	True
v3	35	True

Figure 10: test case - 3 different vehicles

- **Output:**

- **Simulated Annealing:**

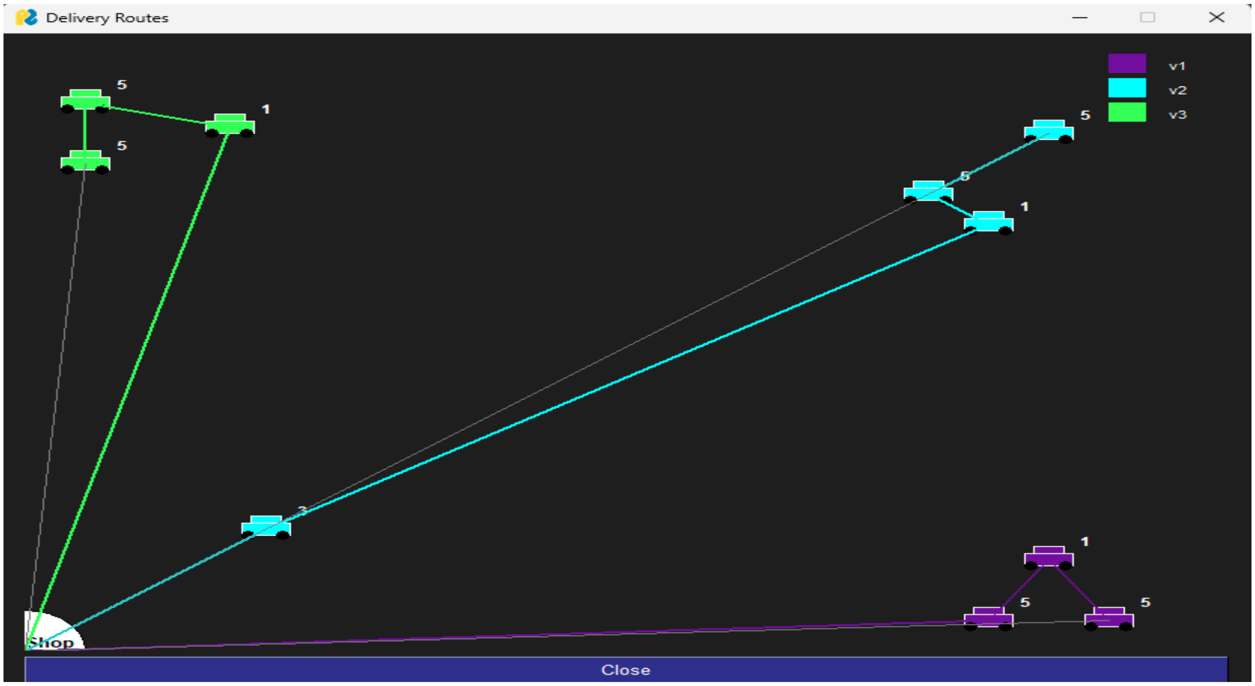


Figure 11: Output - 10 packs (SA)

- **Genetic Algorithm**

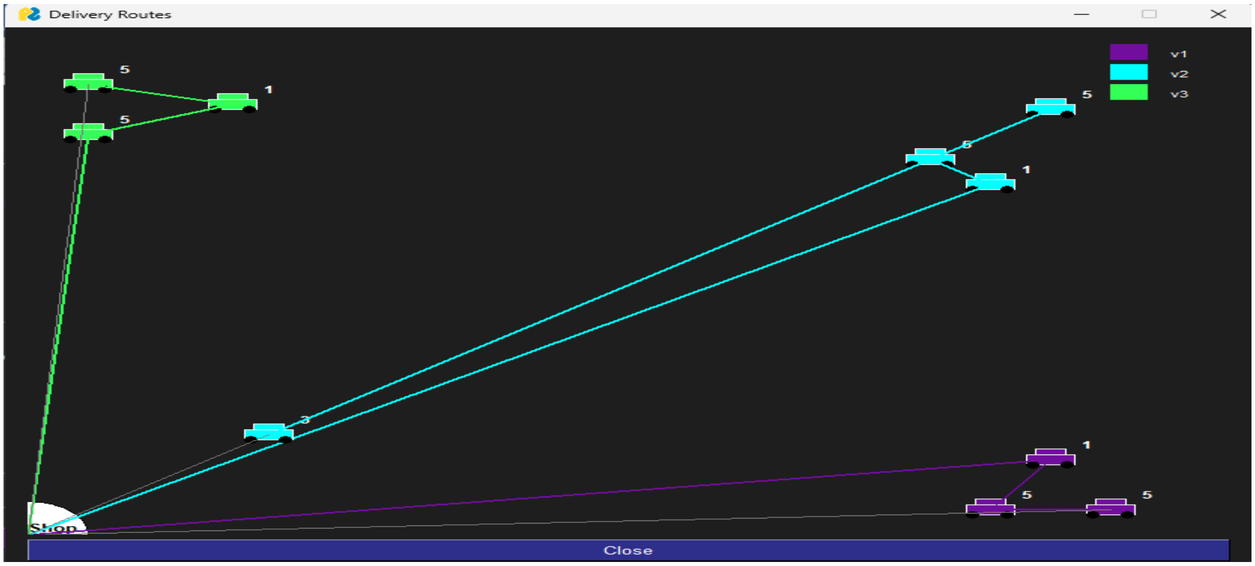


Figure 12: Output - 10 packs (GA)

Test Case #6: Scalability Test:

- **Input**:
  - **Vehicles**: 10 vehicles, each with a capacity of 100 kg.
  - **Packages**: 100 packages with random weights and priorities.

Packages List

package_id	dest_x	dest_y	weight	priority	is_delivered
p76	75.76	24.24	16	1	False
p77	76.77	23.23	17	2	False
p78	77.78	22.22	18	3	False
p79	78.79	21.21	19	4	False
p80	79.8	20.2	20	5	False
p81	80.81	19.19	1	1	False
p82	81.82	18.18	2	2	False
p83	82.83	17.17	3	3	False
p84	83.84	16.16	4	4	False
p85	84.85	15.15	5	5	False
p86	85.86	14.14	6	1	False
p87	86.87	13.13	7	2	False
p88	87.88	12.12	8	3	False
p89	88.89	11.11	9	4	False
p90	89.9	10.1	10	5	False
p91	90.91	9.09	11	1	False
p92	91.92	8.08	12	2	False
p93	92.93	7.07	13	3	False
p94	93.94	6.06	14	4	False
p95	94.95	5.05	15	5	False
p96	95.96	4.04	16	1	False
p97	96.97	3.03	17	2	False
p98	97.98	2.02	18	3	False
p99	98.99	1.01	19	4	False
p100	100.0	0.0	20	5	False

Figure 13: test case - 100 Pack

Vehicles List

vehicle_id	capacity	is_available
v1	100	True
v2	100	True
v3	100	True
v4	100	True
v5	100	True
v6	100	True
v7	100	True
v8	100	True
v9	100	True
v10	100	True

Figure 14: test case - 10 Vehicles

- **Output:**

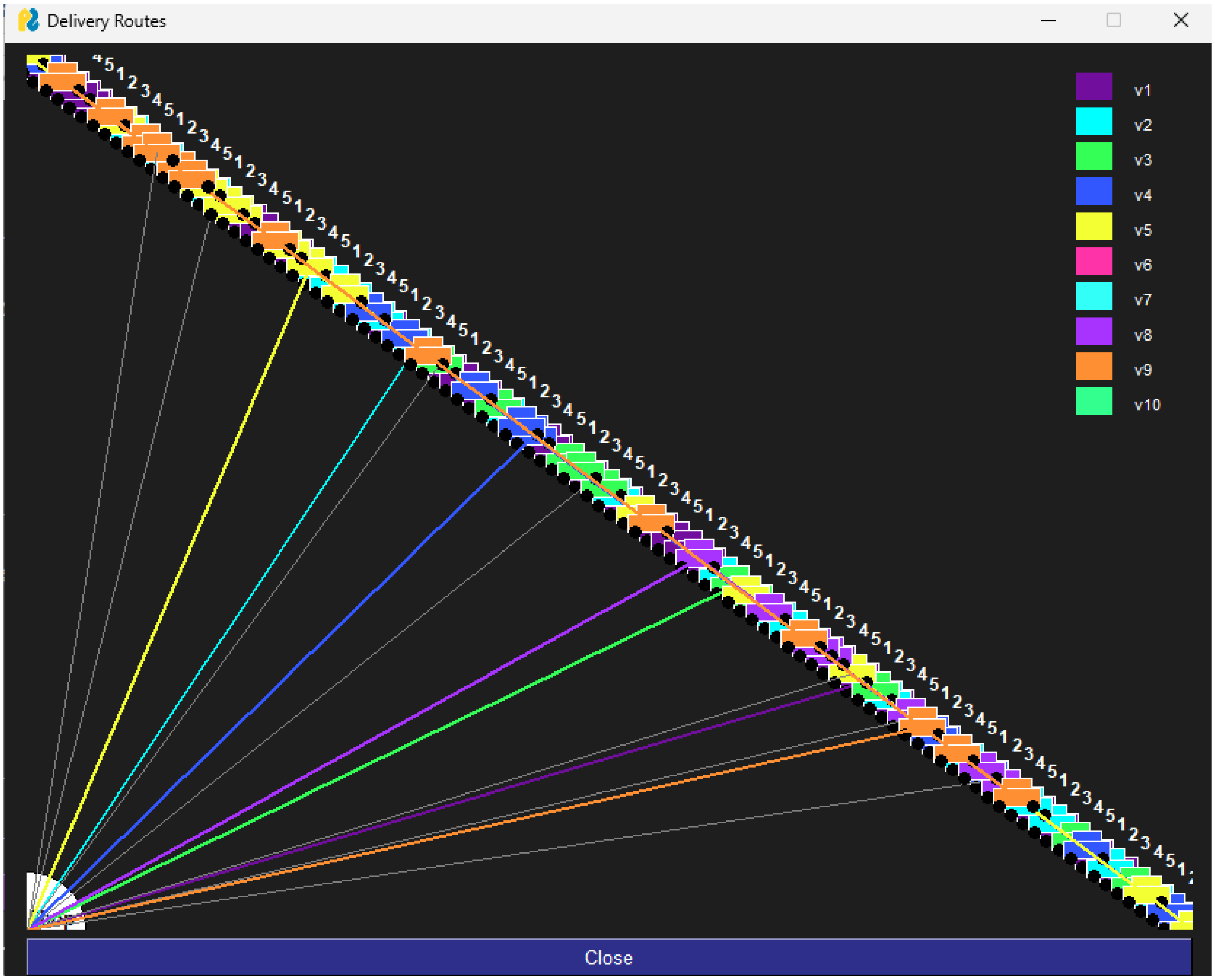


Figure 15: output - 100 Pack

Problem Formulation

- Genetic Algorithm Formulation

- Chromosome (state): A complete state, having packages assigned to vehicles. Each vehicle has a path to drive through it, to deliver all its assigned packages.

- Population: A hundred random generated chromosomes.

- Heuristic: The areal distance between two points on the (X, Y) coordinate.

- Fitness Function: It is the evaluation function in the search terminology.

    - Fitness= Summation for all packages [W1 * Direct Cost + W2 * (1 / priority) * Path Cost]
    - Lower fitness value indicates better chromosome

- Crossover: choose two parents, choose a random number of vehicles to exchange their packages. For example, 5 rounds, at each round, select a random vehicle from chromosome_1 and chromosome_2. Swap packages, and remove duplications, and rebuild tours.

- Mutation: Randomly, choose some children to mutate them. The mutation is applied by changing the tour of a random vehicle in the chromosome.

- Packages with higher priority are delivered first, except if it increases the cost too much.

- The higher number of generations, the better solutions you get (most probable), the higher execution time needed.

- Simulated Annealing Formulation

- **State**: A state represents which packages are assigned to each vehicle. It is stored as a dictionary data structure in the following format:
  {Vn: [Pn]} (e.g., {V1: [P1, P2], V2: [P3, P4], ……, Vn: [Pn]}).
- **Objective Function**: The objective function evaluates how optimal a state is. In this problem, the goal is to **minimize** the objective function.
  It considers two main factors:
  1. **Distance** – lower total distance results in lower cost.
  2. **Priority** – packages with higher priority (where 1 is the highest) should be delivered earlier.

  Formula:
  Summation for all packages: [W1 × Direct Cost + W2 × (1 / Priority) × Path Cost]

- **Next State**: The next state is generated randomly at each iteration by choosing one of the following methods:
  1. **Switching packages within the same vehicle** – two random packages from a randomly selected vehicle are swapped. This introduces variation in the route. No weight check is needed in this case.
  2. **Swapping packages between two different vehicles** – two random packages from two different vehicles are swapped. A weight constraint check is required.
  3. **Moving a package from one vehicle to another** – one random package is moved from a source vehicle to a different target vehicle. A weight constraint check is required.

Hyperparameters:

Table 1: Hyperparameters

Parameter	Value
Temperature	1000
Cooling Rate	0.99
Stopping Temperature	< 1
Number of iteration/ Temperature	1000