CS 451 - Computational Intelligence

Assignment 2

Swarm Intelligence

Sandesh Kumar & Shamsa Hafeez

Major: Computer Science

Class of 2023

19 March 2023

Contents

1 Capacitated Vehicle Routing using ACO **2**

1.1 Problem Formulation: 2

1.2 A-n32-k05.xml 3

1.3 A-n44-k06.xml 5

1.4 A-n60-k09.xml 6

1.5 A-n80-k10.xml 8

2 Building a simulator **9**

1 Capacitated Vehicle Routing using ACO

1.1 Problem Formulation:

- Each ant is a solution to the problem in the form of a list. The list consists of sublists. Each sublist represents a sequence of locations visited in that route such that the total request quantity for that route (calculated by adding the quantity requested in all the locations of that route) does not exceed the capacity of the truck.

  For instance : if an ant is [[1, 2, 3, 1], [1, 4, 1]] it means that it visits all the nodes in 2 routes such that the total capacity of the truck is greater than or equal to the sum of the requested load in each route.

- This is a minimization problem. We need to minimize the total distance covered by an ant. Thus notice that in the example mentioned earlier, the total distance shall be :
  $distance(1,2), distance(2,3), distance(3,1), distance(1,4), distance(4,1)$

- Fitness of an ant / solution is the total distance covered by an ant. We have to minimize it.

- The intensity of the trail of pheromones on an edge (i.e., tau) is trail left by the ant. The value of tau of an edge is calculated by summing the inverse of fitness of ants that passed through the edge.

- The visibility or desirability is the inverse of the length of the edge.

- Transition probability is calculated as follows: The transition probability from location i to location j for the kth ant is defined as

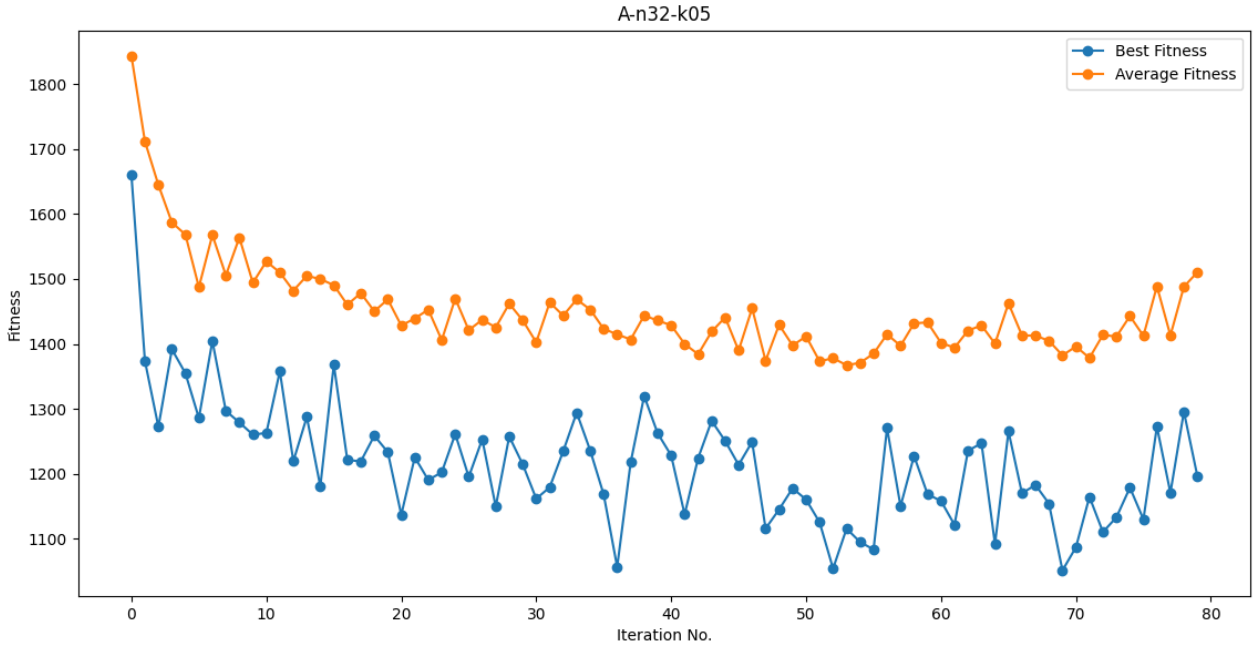
$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{k \in \text{allowed}_k} \left[\tau_{ik}(t)\right]^{\alpha} \cdot \left[\eta_{ik}\right]^{\beta}} & \text{if } j \in \text{allowed}_k \\ 0 & \text{otherwise} \end{cases}$$

- Note that the tabu list is implementation in a way that the ant can only go to the remaining requested nodes. The only node that can be repeated is the departure and arrival node (in this case 1) since all routes originate from it.

- With respect to code, we first make a dictionary of tau (to implement tau table) and a dictionary of map_ . Initially, all the entries of tau table are zero and the map_ entries correspond to a list of size 2 showing pheromone and distance between two towns.

- Note that the key (source, destination) is considered equivalent to (destination, source)

1.2 A-n32-k05.xml



Global Best Solution: [[1, 13, 7, 29, 5, 12, 9, 19, 15, 21, 1], [1, 24, 4, 3, 20, 18, 32, 22, 1], [1, 8, 11, 30, 10, 23, 16, 26, 6, 1], [1, 31, 17, 2, 25, 28, 27, 1], [1, 14, 1]]

Fitness: 1052.111757487989

Parameters: no of ants = 20

p = 0.6

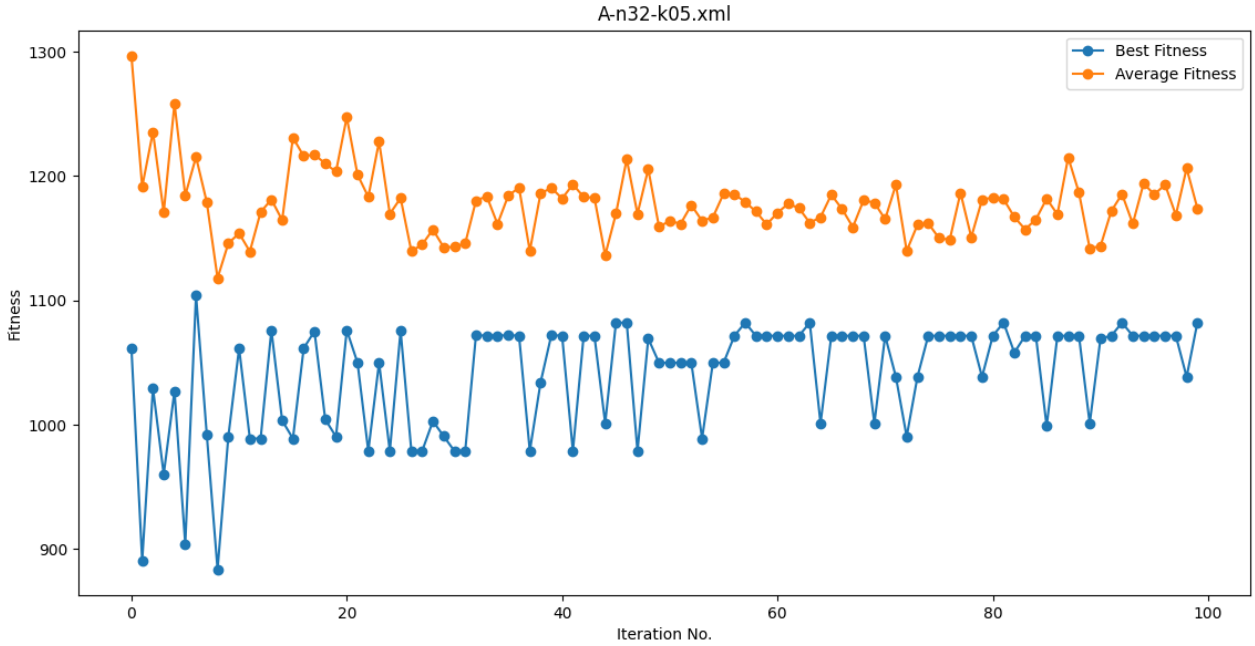
Q = 1

alpha = 0.8

beta = 0.8

Total number of iterations = 80

We changed the values of hyperparameters to see if we can obtain better results. Let us have a greater beta as compared to alpha (increase the importance of visibility as compared to trail). Following are the new parameters along with the graph obtained:



Best solution: [[1, 28, 25, 15, 27, 31, 17, 2, 1], [1, 7, 3, 4, 24, 29, 9, 10, 23, 19, 30, 6, 1], [1, 18, 20, 32, 22, 14, 8, 1], [1, 21, 26, 11, 16, 12, 5, 1], [1, 13, 1]]

Fitness of best solution: 883.5274847571451

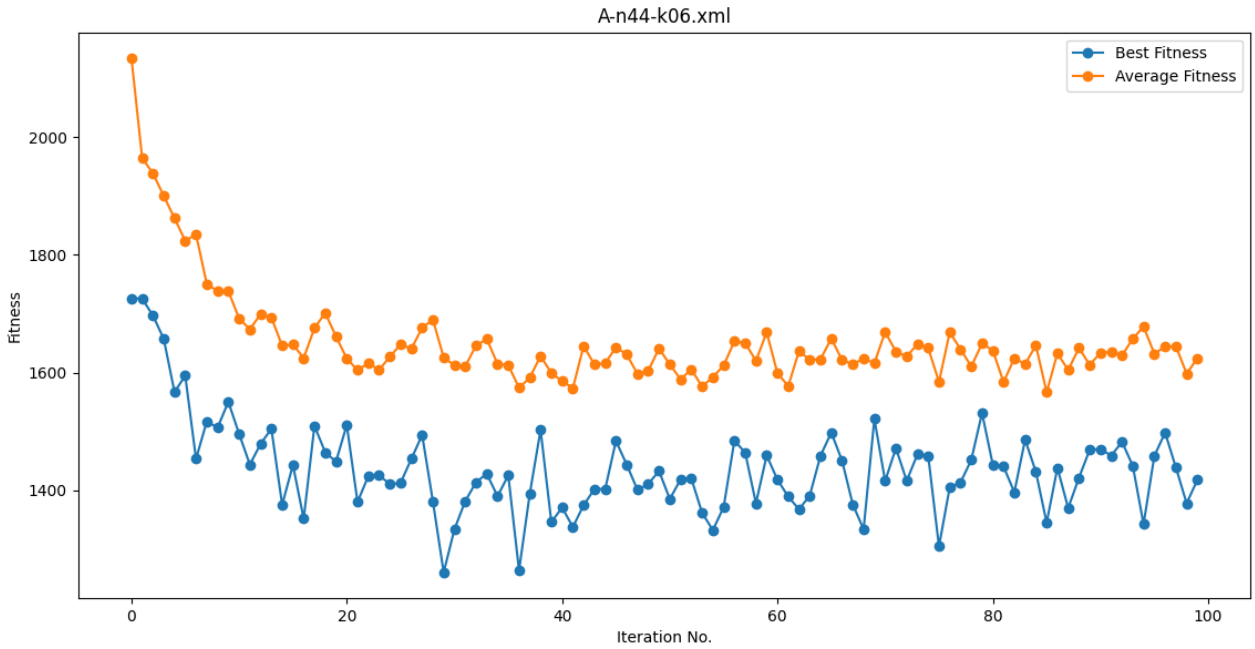
number of ants:20

p = 0.6, Q = 1, alpha = 2, beta = 3

number of iterations = 100

Note: The p value is the gamma such that 1-p represents the evaporation of trail

1.3 A-n44-k06.xml



Global Best Solution: [[1, 13, 40, 18, 35, 34, 6, 25, 27, 11, 12, 1], [1, 32, 29, 16, 8, 23, 37, 15, 42, 1], [1, 20, 28, 33, 38, 43, 22, 4, 1], [1, 19, 2, 36, 17, 21, 31, 10, 39, 1], [1, 3, 44, 41, 24, 30, 14, 1], [1, 7, 26, 5, 9, 1]]

Fitness: 1259.9656324851903

Parameters: no of ants = 20

p = 0.6

Q = 1

alpha = 0.8

beta = 0.8

Total number of iterations = 100

Now, as we did in last section, let us change the hyper parameter values in order to better suit our optimization problem. Note that in this report we are attaching only 2 graphs out of all the attempts that we made while tweaking the parameters.

Let us have a greater beta as compared to alpha (increase the importance of visibility as compared to trail). Follow- ing are the new parameters along with the graph obtained:

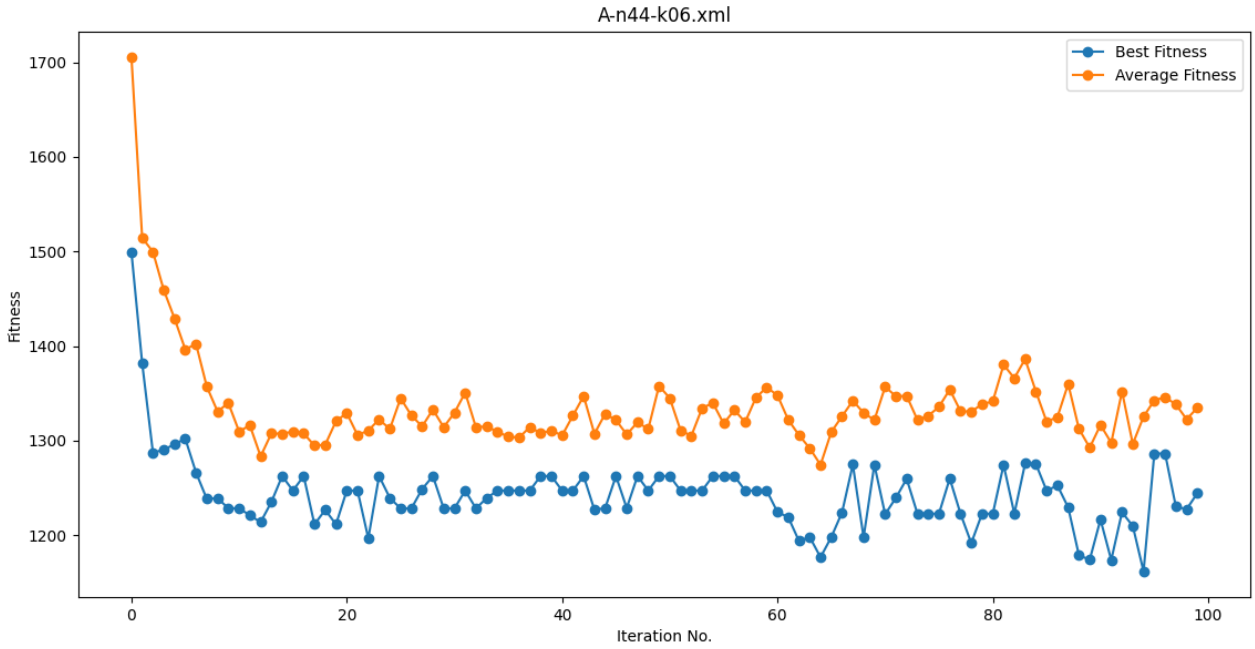
Best solution: [[1, 42, 15, 37, 39, 10, 26, 7, 13, 22, 1], [1, 32, 9, 29, 16, 28, 20, 25, 1], [1, 5, 35, 18, 40, 4, 23, 3, 1], [1, 8, 34, 6, 33, 43, 38, 11, 12, 1], [1, 24, 31, 41, 30, 44, 14, 1], [1, 27, 21, 17, 19, 2, 36, 1]]

Fitness of best solution: 1161.7826122276351

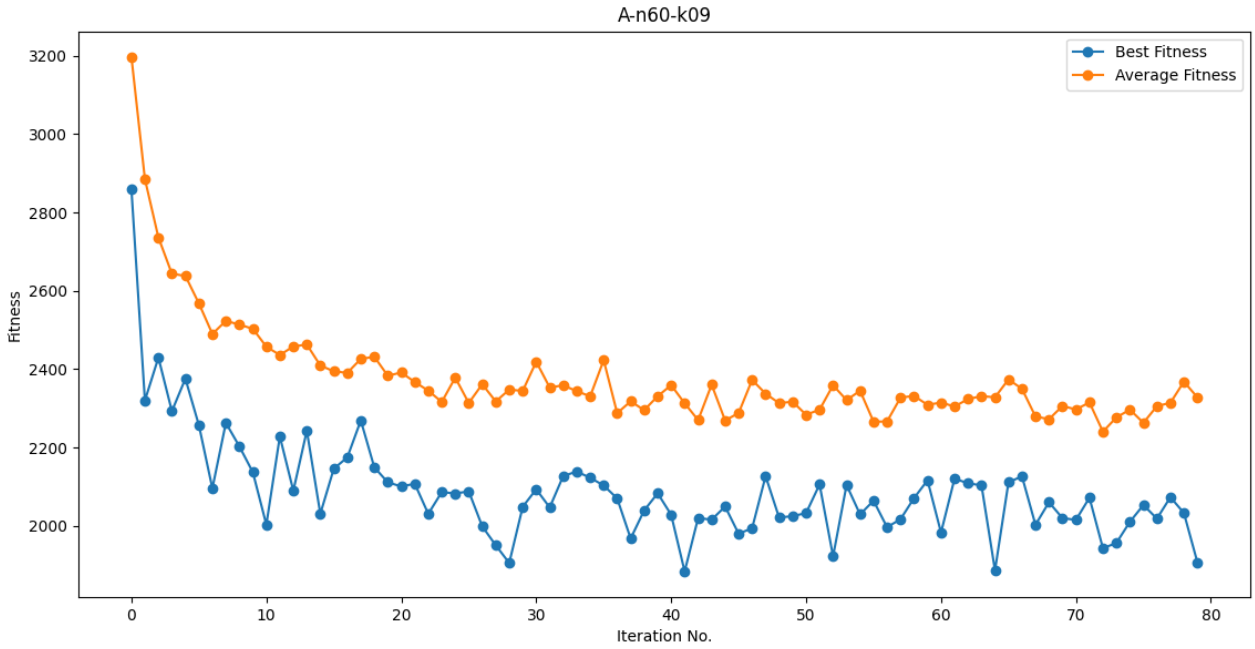
number of ants:20

number of iterations = 100

p = 0.6, Q = 1, alpha = 2, beta = 3



1.4 A-n60-k09.xml



Global Best Solution: [[1, 42, 34, 53, 19, 56, 16, 20, 15, 41, 17, 21, 1], [1, 58, 38, 28, 18, 24, 48, 35,

47, 1], [1, 52, 43, 55, 11, 23, 6, 37, 1], [1, 27, 44, 57, 13, 10, 33, 36, 51, 3, 1], [1, 22, 12, 5, 4, 26, 31, 54, 50, 1], [1, 49, 46, 59, 1], [1, 29, 2, 45, 32, 7, 40, 1], [1, 8, 14, 9, 30, 39, 1], [1, 60, 25, 1]]

Fitness: 1883.4383437775841

Parameters: no of ants = 25

p = 0.6

Q = 1

alpha = 0.8

beta = 0.8

Total number of iterations = 80

Notice that we tried to use greater number of ants since we need to explore more nodes before seeking to exploit the routes with minimum distance

Tweaking the hyper-parameters so as to improve its performance (just like we did in the previous 2 graphs). Also let us increase ants more to better explore the environment.

Parameters: no of ants = 30

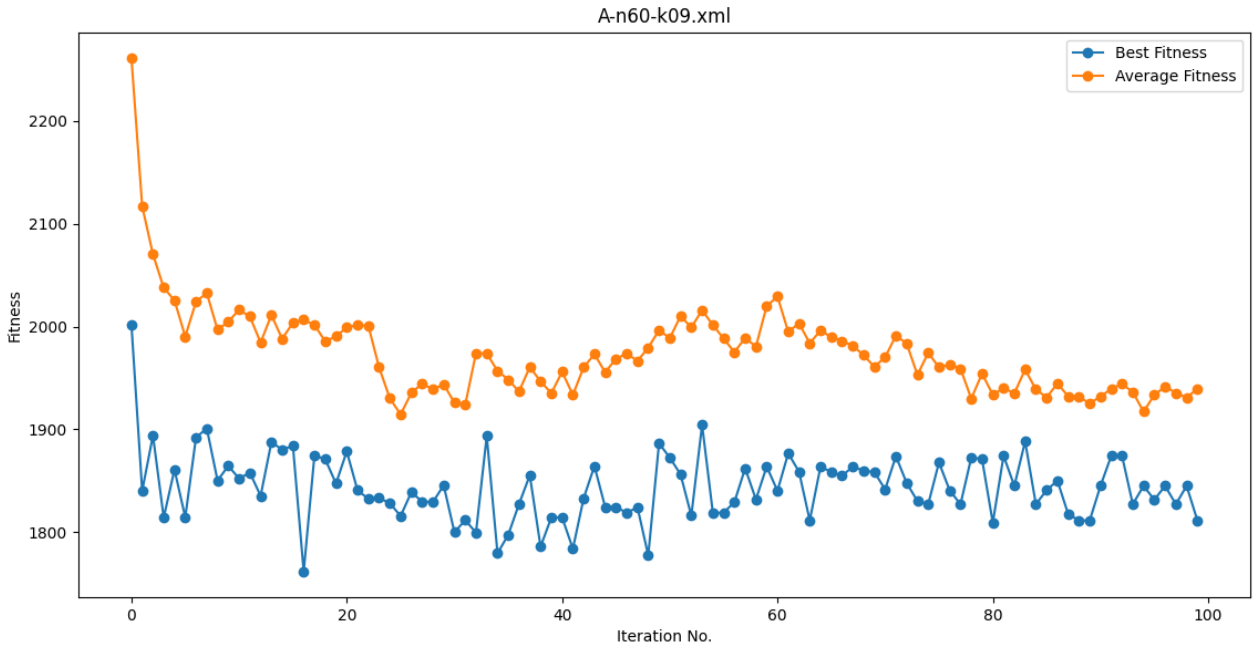
p = 0.6

Q = 1

alpha = 2

beta = 3

Total number of iterations = 100

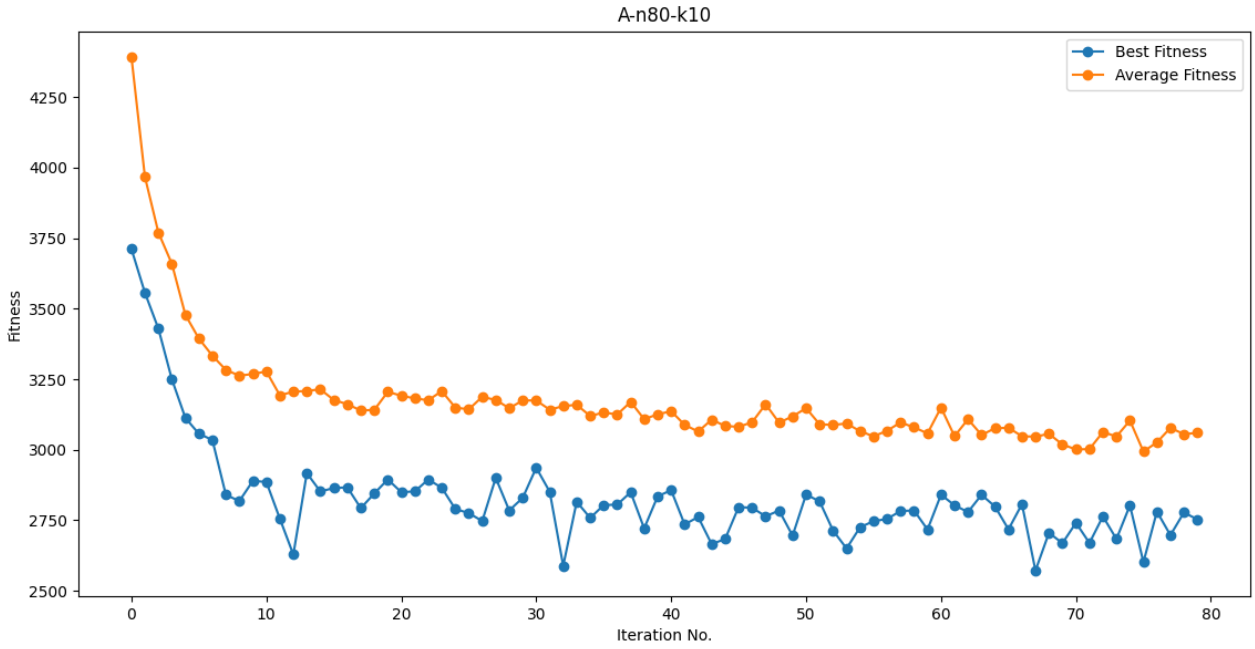


Best solution: [[1, 42, 15, 34, 26, 41, 47, 4, 17, 21, 19, 20, 36, 1], [1, 12, 5, 32, 54, 50, 45, 29, 1], [1, 46,

43, 23, 11, 22, 1], [1, 39, 60, 27, 40, 51, 16, 28, 38, 59, 3, 1], [1, 53, 8, 30, 14, 9, 1], [1, 56, 24, 48, 35, 7, 25, 1], [1, 13, 52, 10, 33, 57, 44, 1], [1, 6, 55, 49, 37, 2, 31, 1], [1, 18, 58, 1]]

Fitness of best solution: 1761.8269606038875

1.5 A-n80-k10.xml



Global Best Solution: [[1, 18, 32, 76, 21, 20, 42, 26, 47, 65, 16, 34, 1], [1, 53, 29, 3, 38, 9, 69, 44, 17, 79, 28, 35, 15, 1], [1, 22, 62, 58, 27, 57, 70, 56, 55, 73, 71, 1], [1, 74, 68, 52, 78, 4, 75, 45, 13, 54, 1], [1, 77, 59, 51, 46, 23, 5, 33, 48, 66, 36, 1], [1, 37, 43, 67, 10, 60, 31, 1], [1, 50, 8, 2, 19, 49, 1], [1, 30, 40, 61, 7, 41, 14, 25, 1], [1, 72, 64, 24, 12, 80, 6, 1], [1, 39, 11, 63, 1]]

Fitness: 2571.6867809567443

Parameters: no of ants = 35

p = 0.6

Q = 1

alpha = 0.8

beta = 0.8

Total number of iterations = 80

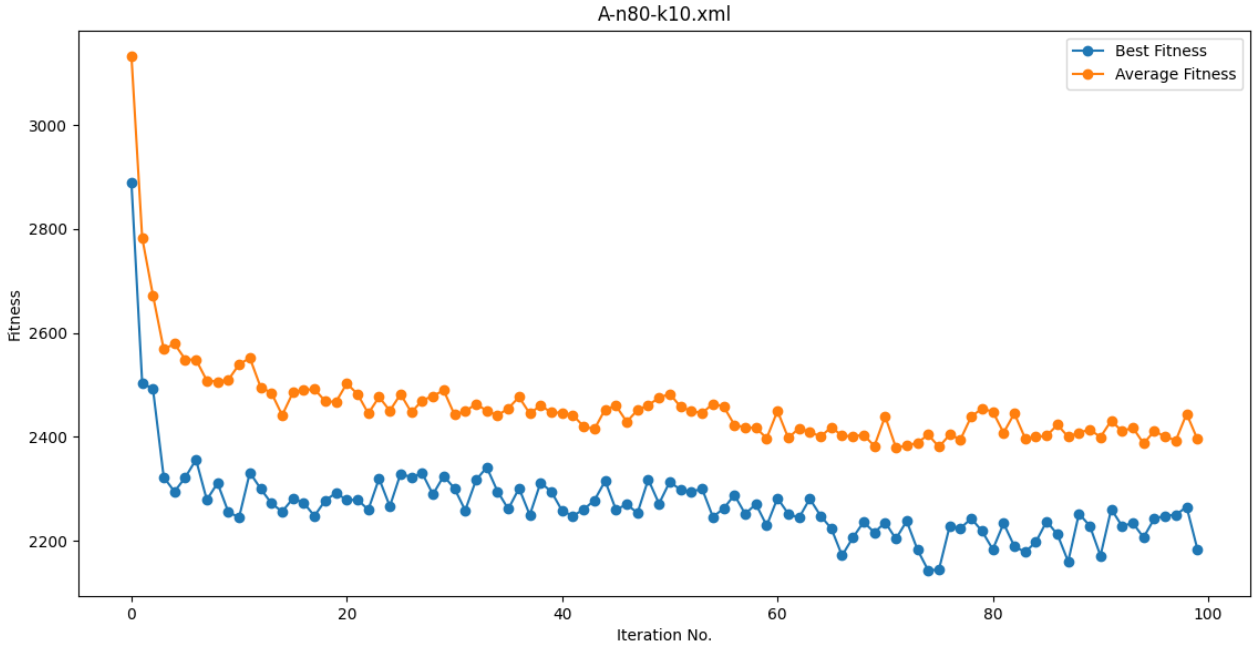
Let us use greater number of ants and tweak the hyper parameters in order to improve the optimization results as we did in previous cases:

p = 0.6, Q = 1, alpha = 2, beta = 3

number of iterations = 100

Best solution: [[1, 50, 74, 14, 43, 52, 54, 37, 68, 71, 78, 1], [1, 41, 22, 2, 8, 11, 12, 34, 1], [1, 16, 48, 27, 36, 66, 70, 57, 56, 10, 39, 67, 1], [1, 15, 72, 53, 29, 80, 49, 19, 35, 1], [1, 4, 61, 40, 75, 30, 6, 32, 1], [1, 18, 60, 28, 76, 21, 62, 69, 79, 1], [1, 59, 77, 51, 46, 23, 33, 5, 55, 73, 1], [1, 65, 47, 26, 42, 20, 58, 17, 44, 9, 25, 1], [1, 64, 63, 24, 13, 45, 31, 1], [1, 7, 3, 38, 1]]

Fitness of best solution: 2143.164539638669



2 Building a simulator

This question required us to learn any simulating library or language in case we don't know one. We tried doing this question in 2 ways. Firstly, a 2D visulization tool which is in file "Final2DPSOSimulator" is submitted which firstly asks for values on slider and then once you click on run, using those values it created a simulation. To make it a little more intersting, a 3D simulator for PSO is also being submitted in file "PSO3Dgif" which after running on the given parameters will create a gif on simulation.