Homework 1 (CMSC818B)

Author Name
Harsh Bharat Kakashaniya (116311236)

Homework 1 : TSP solution for 1 and K robots.

Date : 9/16/2019

Contents

1 Problem 1 **3**

1.1 Implementation MST 3

1.2 Implementation of Tour from MST: 4

1.3 Optimization of Tour: 4

1.3.1 Algorithm 1 (Swiping of 2 numbers): 4

1.3.2 Algorithm 2 (Swiping of series between 2 numbers): 5

1.4 Results for datasets: 7

1.4.1 Standard Data Set 7

1.4.2 Random 100 points: 7

1.4.3 Random 200 points: 8

1.4.4 Random 300 points: 8

2 Problem 2 **9**

2.1 Psudo Code: 9

2.2 Optimization Algorithm: 10

2.3 Algorithm 1: 10

2.4 Algorithm 2: 10

2.5 Results: 11

2.6 Videos Links : 12

List of Figures

1 MST for eil51.tsp 3

2 Tour for eil51.tsp without optimization 4

3 Psudo code algorithm 1 4

4 Tour for eil51.tsp with Algorithm 1 optimization 5

5 Psudo code algorithm 2 5

6 Tour for eil51.tsp with Algorithm 2 optimization 6

7 Psudo code Hybrid algorithm 6

8 Tour for eil51.tsp with Hybrid Algorithm 7

9 Psudo code for K robot 9

10 Psudo code Algorithm 1 10

11 Psudo code Algorithm 2 10

12 Multi robot solution 11

13 Multi robot solution 12

1 Problem 1

1.1 Implementation MST

1. **Pre-processing of input .tsp file:** Using csv given file was opened and converted to list with delimiter space. This gave output of list of all lines present in .tsp file.

   Finally, first 6 lines and last line of file were deleted then lines were converted to 2D matrix. So we received output of (Nx3) matrix where N is number of nodes.

2. **Distance matrix :** In order to process MST we need cost of each node from other node So we get a matrix using distance formula of size (NxN) where N will be number of nodes with its diagonal elements as infinity.

3. **Making of MST:** Distance matrix we created will be handy now. First of all we will find 2 nodes with minimum distance between each other. So here we have 1 connection and 2 nodes. All nodes with connections will be saved in visited list for example initially there will be 2 nodes in visited. Logic of making MST : **Find the nearest node from some node in visited list excluding visited node** this will avoid loops and will ensure we get a tree at the end. Following is the output after implementing that algorithm in terms of dictionary.

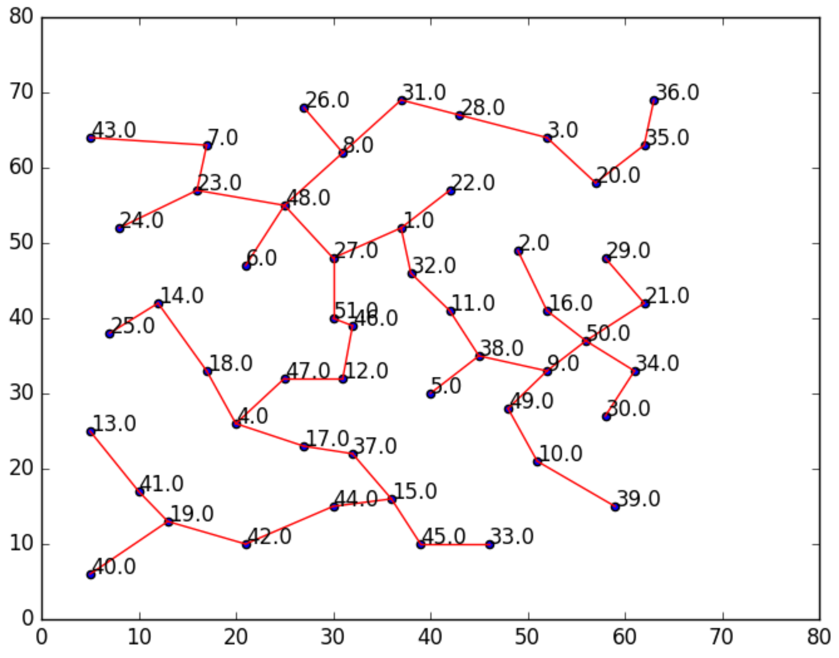


Figure 1: MST for eil51.tsp

4. **Representation of MST in dictionary**

   {1: [32, 22], 3: [20], 4: [17, 18], 7: [43], 8: [26, 31], 9: [50, 49], 10: [39], 11: [38], 12: [47], 14: [25], 15: [44, 45], 16: [2], 17: [37], 18: [14], 19: [41, 40], 20: [35], 21: [29], 23: [7, 24], 27: [1, 48], 28: [3], 31: [28], 32: [11], 34: [30], 35: [36], 37: [15], 38: [5, 9], 41: [13], 42: [19], 44: [42], 45: [33], 46: [51, 12], 47: [4], 48: [6, 8, 23], 49: [10], 50: [16, 34, 21], 51: [27]}

1.2 Implementation of Tour from MST:

Once we get MST we implement DFS and get a sequence of nodes which will result in a tour.

Note: Apply DFS to the top most point in the MST which is first element in MST.

So from the given MST we get a tour for first data set of eil51.tsp

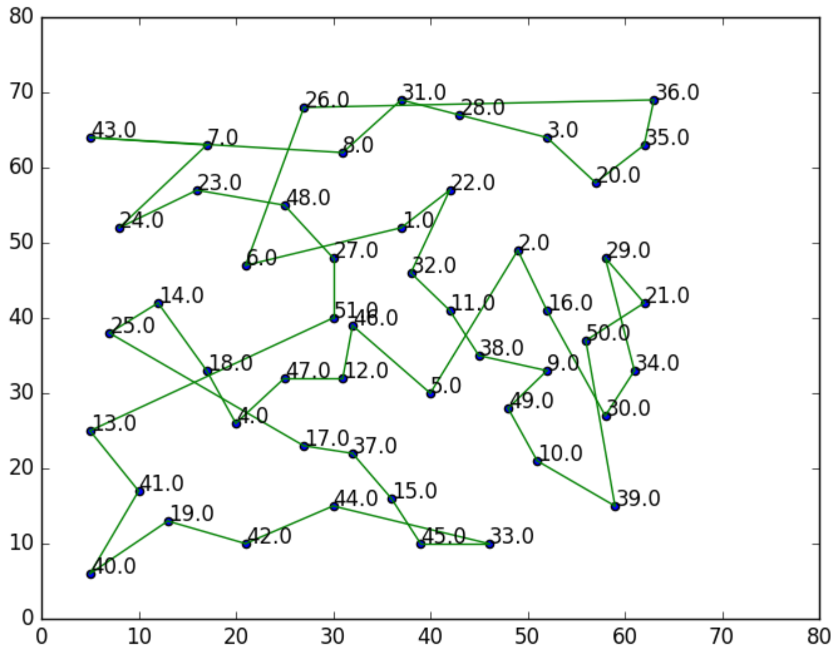


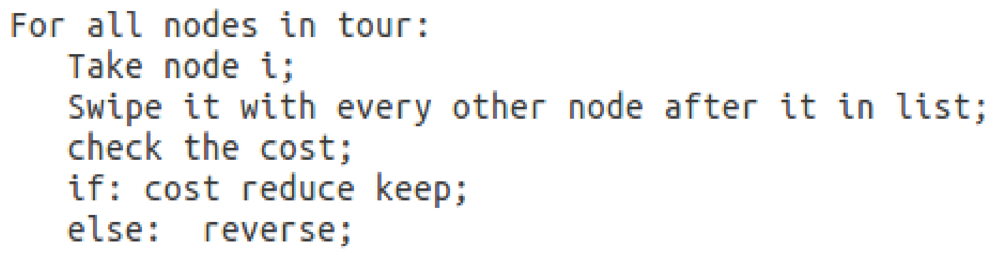
Figure 2: Tour for eil51.tsp without optimization

1.3 Optimization of Tour:

1.3.1 Algorithm 1 (Swiping of 2 numbers):

For improving the tour and decreasing the cost of tour following steps were taken:

Psudo code:



```
For all nodes in tour:
   Take node i;
   Swipe it with every other node after it in list;
   check the cost;
   if: cost reduce keep;
   else:  reverse;
```

Figure 3: Psudo code algorithm 1

Output of the following algorithm: Length of Tour: 498.426537872

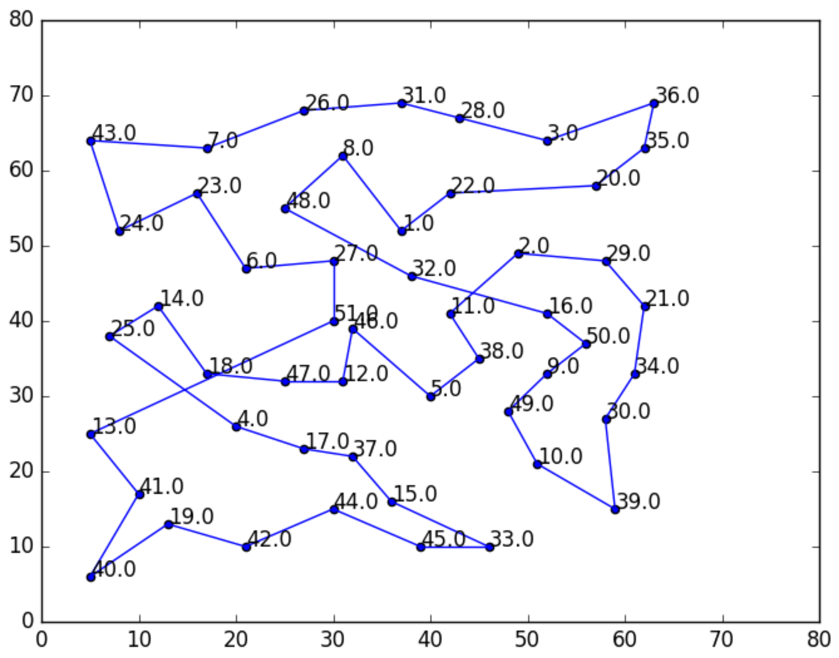
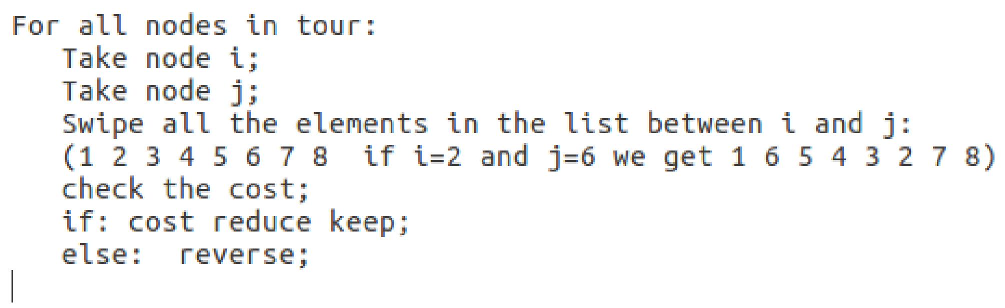


Figure 4: Tour for eil51.tsp with Algorithm 1 optimization

1.3.2 Algorithm 2 (Swiping of series between 2 numbers):

For further improving the tour and decreasing the cost of tour following steps were taken:

Psudo code:



```
For all nodes in tour:
   Take node i;
   Take node j;
   Swipe all the elements in the list between i and j:
   (1 2 3 4 5 6 7 8  if i=2 and j=6 we get 1 6 5 4 3 2 7 8)
   check the cost;
   if: cost reduce keep;
   else:  reverse;
```

Figure 5: Psudo code algorithm 2

Output of the following algorithm: Length of Tour: 456.557650342

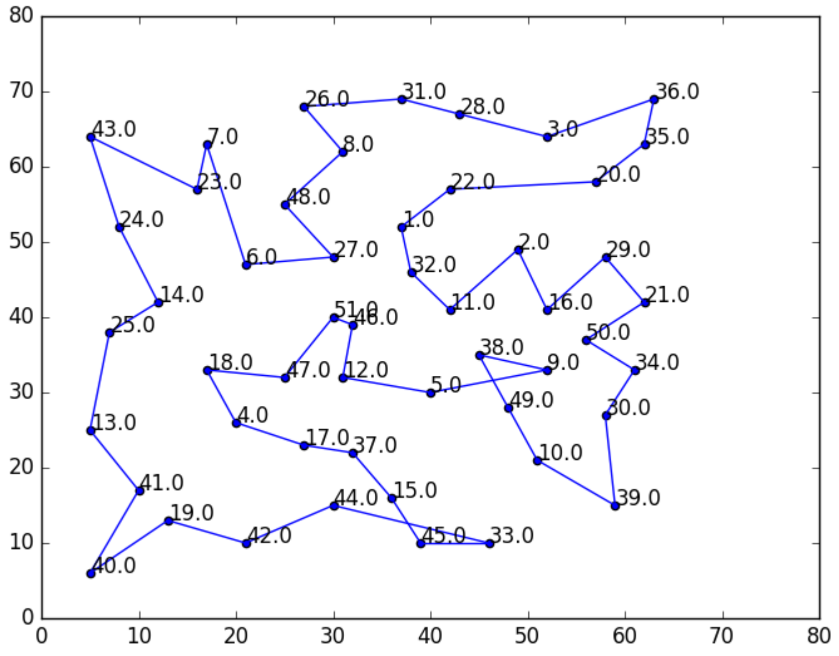
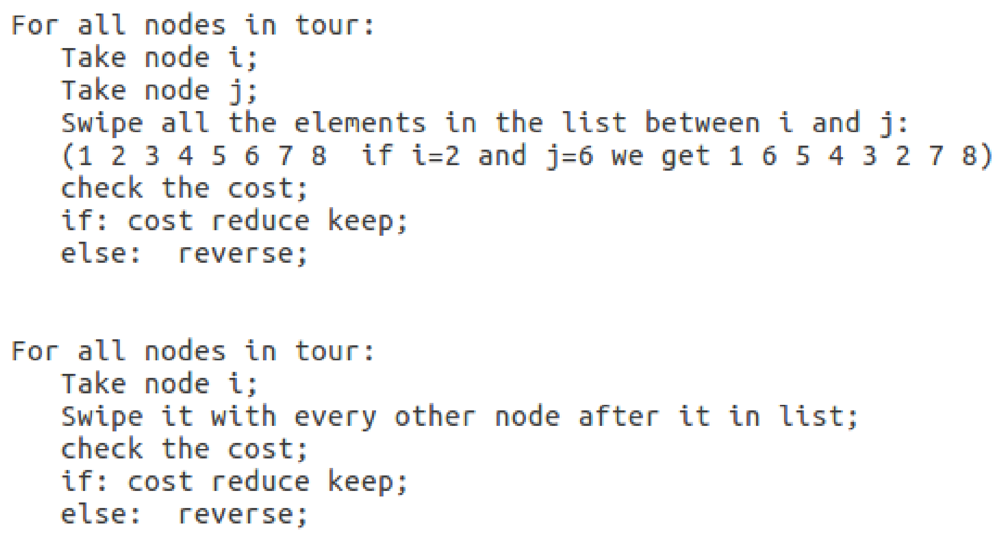


Figure 6: Tour for eil51.tsp with Algorithm 2 optimization

From the output of both the algorithms we infer that both algorithm complement each other.So we merge both the algorithm and take basic tour and pass it in algorithm 2 and output is passed in algorithm 1. Hence, this gives effect of best of both worlds. Following is the output of hybrid algorithm:

Psudo code:



```
For all nodes in tour:
   Take node i;
   Take node j;
   Swipe all the elements in the list between i and j:
   (1 2 3 4 5 6 7 8  if i=2 and j=6 we get 1 6 5 4 3 2 7 8)
   check the cost;
   if: cost reduce keep;
   else:  reverse;


For all nodes in tour:
   Take node i;
   Swipe it with every other node after it in list;
   check the cost;
   if: cost reduce keep;
   else:  reverse;
```

Figure 7: Psudo code Hybrid algorithm

Length of Tour: 441.429829814

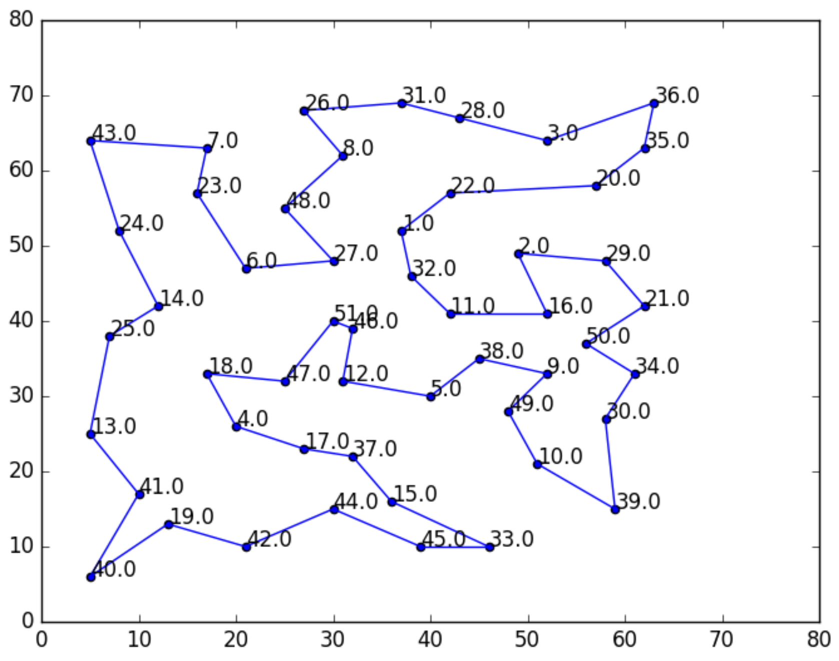


Figure 8: Tour for eil51.tsp with Hybrid Algorithm

So here we get an optimal result by merging both the algorithms.

1.4 Results for datasets:

1.4.1 Standard Data Set

Data Set	MST length	Raw Tour Length	Algorithm Tour Length	Optimal Tour length	Time Required
eil51	376.49	580.57	441.43	426	0.68
eil76	472.33	752.88	594.34	538	2.09
eil101	562.25	921.41	713.67	629	4.96

1.4.2 Random 100 points:

Random 100 points were taken between 0 to 80 in both X and Y direction following is the result obtained which shown that for 100 points average time lies around 4.9 to 5 seconds. Details can be seen in the table below which has all necessary tabs to check effectiveness of algorithm. Similar process will be applied for 200 and 300 point as well .

Note: Code has animations for understanding of optimization. While taking timings those plots were commented.

Sr. No.	MST length	Raw Tour Length	Algorithm Tour Length	Time Required(Seconds)
1	536.66	815.98	704.95	5.45
2	523.25	811.17	652.99	4.77
3	536.88	820.38	654.62	4.82
4	535.53	801.78	685.56	4.86
5	523.75	798.39	677.70	4.79
6	544.80	862.35	727.88	4.80
7	537.07	846.84	690.85	4.82
8	535.15	819.53	687.89	4.92
9	548.39	881.79	704.47	5.17
10	541.97	795.55	654.86	4.78

1.4.3 Random 200 points:

Sr. No.	MST length	Raw Tour Length	Algorithm Tour Length	Time Required(Seconds)
1	728.79	1192.40	914.0	41.20
2	780.79	1228.7	960.60	42.70
3	735.79	1160.40	934.79	42.29
4	737.20	1177.7	952.5	42.39
5	750.89	1188.7	943.60	41.10
6	767.79	1235.40	967.0	41.60
7	753.60	1227.90	953.60	41.60
8	724.70	1135.7	872.79	42.70
9	731.89	1193.2	919.10	42.5
10	731.10	1156.2	906.29	43.39

1.4.4 Random 300 points:

Sr. No.	MST length	Raw Tour Length	Algorithm Tour Length	Time Required(Seconds)
1	905.79	1431.7	1154.59	144.0
2	887.39	1399.2	1137.1	147.7
3	897.3	1444.8	1134.1	142.80
4	902.0	1413.3	1163.7	143.30
5	930.4	1484.6	1168.6	139.40
6	874.9	1439.0	1144.5	141.4
7	909.4	1462.1	1150.0	144.80
8	912.5	1449.90	1144.2	158.80
9	907.4	1425.8	1147.0	146.90
10	886.3	1440.1	1106.3	138.90

2 Problem 2

2.1 Psudo Code:

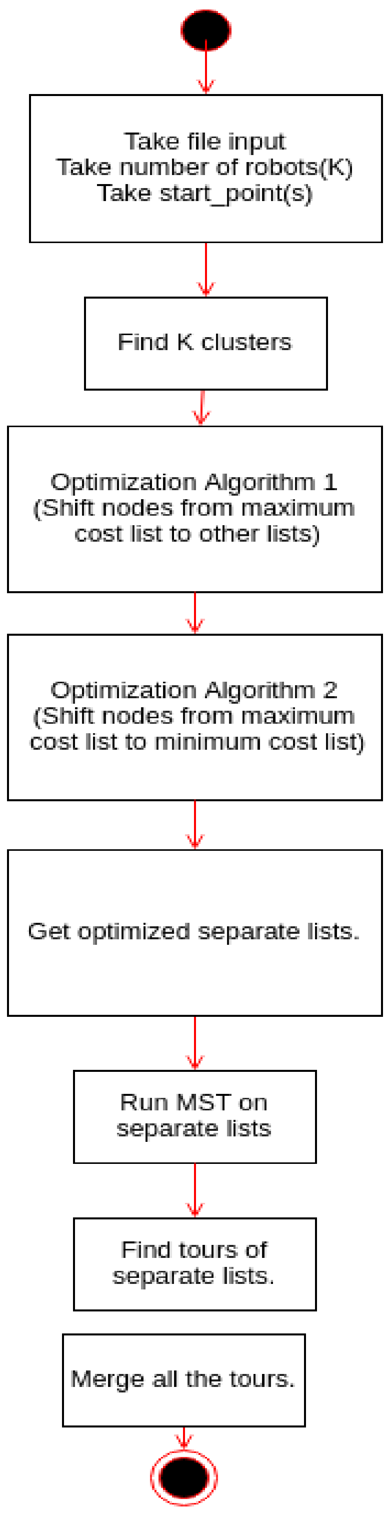

Take file input
Take number of robots(K)
Take start_point(s)
Find K clusters
Optimization Algorithm 1
(Shift nodes from maximum cost list to other lists)
Optimization Algorithm 2
(Shift nodes from maximum cost list to minimum cost list)
Get optimized separate lists.
Run MST on separate lists
Find tours of separate lists.
Merge all the tours.

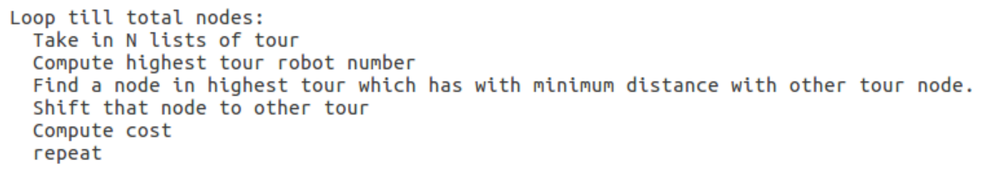
Figure 9: Psudo code for K robot

Above is the psudo code for K robot where we have 2 algorithms to optimize the clusters so that we can have some bound of minimizing the maximum tour.

2.2 Optimization Algorithm:

2.3 Algorithm 1:

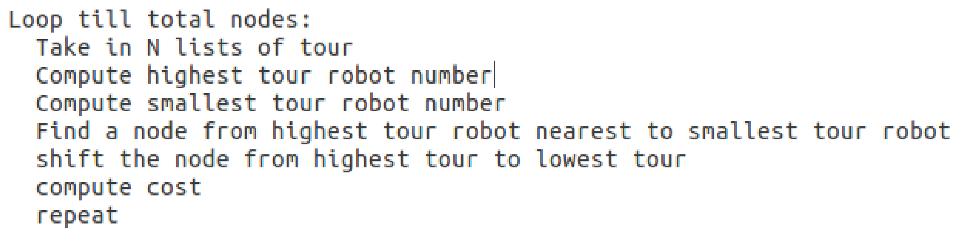
Following algorithm is used after getting k clusters to get costs nearly equal.



```
Loop till total nodes:
  Take in N lists of tour
  Compute highest tour robot number
  Find a node in highest tour which has with minimum distance with other tour node.
  Shift that node to other tour
  Compute cost
  repeat
```

Figure 10: Psudo code Algorithm 1

2.4 Algorithm 2:



```
Loop till total nodes:
  Take in N lists of tour
  Compute highest tour robot number
  Compute smallest tour robot number
  Find a node from highest tour robot nearest to smallest tour robot
  shift the node from highest tour to lowest tour
  compute cost
  repeat
```

Figure 11: Psudo code Algorithm 2

By using these algorithms we get a good output with cost of each tour approximately equal. Following are the output of the algorithm we built upon single robot.

After getting optimal output of points we can solve the problem with k instances of class for single robot.

2.5 Results:

Following algorithm was written and below are the output snaps for different scenarios as follows. For example,Following are the output for our multiple robot code on data set eil51.tsp in this case we have 3 robots which are solving the problem and minimizing the cost of maximum tour.

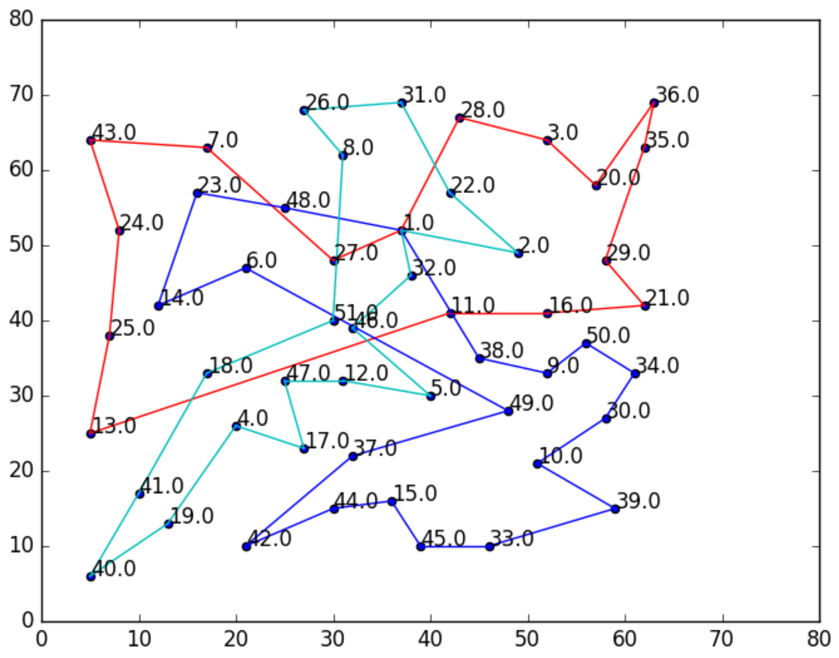


Figure 12: Multi robot solution

In this case we get Tours of individual robots as :

Robot_1(Red) = 214.672963035

Robot_2(Blue) = 221.861758172

Robot_3(Green) = 204.389007097

Now lets run one more instance of:

K = 4 ,Points = 76 , Start_point = 1

The graph below provides following tour paths:

Robot_1(Red) =243.25990294

Robot_2(Blue) = 254.037496922

Robot_3(Sky Blue) = 269.388657729

Robot_3(Green) = 194.103147566

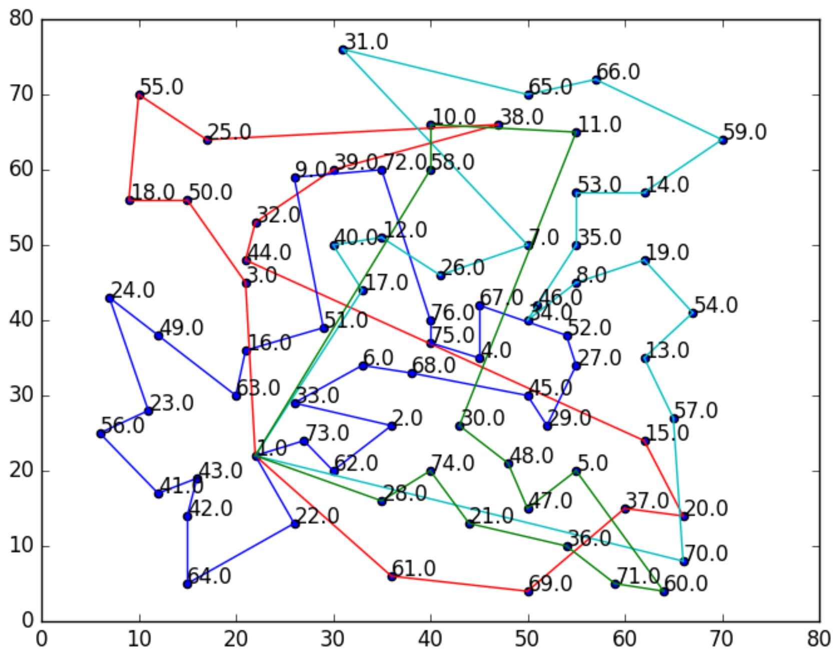


Figure 13: Multi robot solution

Following are the outputs generated by the code submitted we can see real time solution if we run the code. Follow read me to run the code.

2.6 Videos Links :

For 1 robot system:

https://youtu.be/74EUgazKQqM

For N robots system :

https://drive.google.com/drive/folders/1YCj1oz8CSLa$_{O}niN_{a}KBkrj9TeasQYti?usp = sharing$

.... Thank you ....