PROJECT TITLE

A MINI-PROJECT REPORT

of

BACHELOR OF TECHNOLOGY

in

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

at

DAYANANDA SAGAR UNIVERSITY

SCHOOL OF ENGINEERING, BANGALORE-560068

IV SEMESTER

(Course Code:16CS274)

(Design And Analysis Of Algorithms Lab)

DAYANANDA SAGAR UNIVERSITY

CERTIFICATE

This is to certify that the Design and Analysis of Algorithm Mini-Project report entitled **“URBAN INFRASTRUCTURE DEVELOPMENT PROGRAM”** being submitted by______________________to Department of Computer Science and Engineering, School of Engineering, Dayananda Sagar University, Bangalore, for the 4th semester B.Tech C.S.E of this university during the academic year 2019-2020.

Date:___________ ____________________________

Signature of the Faculty in Charge

____

Signature of the Chairman

ACKNOWLEDGEMENT

We are pleased to acknowledge the Faculty **Neeta M S, Assistant Professor ,**Department of Computer Science & Engineering for her invaluable guidance, support, motivation and patience during the course of this mini- project work.

We extend our sincere thanks to **Dr. Banga M.K** , **Chairman,** Department of Computer Science & Engineering who continuously helped us throughout the project and without his guidance, this project would have been an uphill task.

We have received a great deal of guidance and co-operation from our friends and we wish to thank one and all that have directly or indirectly helped us in the successful completion of this mini-project work.

Team Members

NIKHITA A (ENG18CS0192)

NISHA V (ENG18CS0194)

PAVANA M (ENG18CS0207)

PAVITHRA H A (ENG18CS0208)

ABSTRACT

The "URBAN INFRASTRUCTURE DEVELOPMENT PROGRAM" problem is that of choosing k vertices as centers in a weighted undirected graph in which the edge weights obey the triangle inequality so that the maximum distance of any vertex to its nearest center is minimized. From the given graph along with the distance, initially we select a random city of maximum distance .Then with respect to that we choose n-1 centres . The objective is to place the central centres such that all people have convenient access to them. We analyze the problem and compare different placement strategies and evaluate the number of required centers .The dynamic programming algorithm can be adapted in order to find the solution,which can also improve the time complexity compared to greedy approach which takes O(n3) time.We use recursive approach to find the minimum distance between centres.The algorithm runs in linear time. As per the performance of execution of the program, the time complexity is improved to O(*nk*). The "URBAN INFRASTRUCTURE DEVELOPMENT PROGRAM" has been designed to achieve the maximum profit and fulfill all the necessities.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Given n cities with specified distances, one wants to build k fire stations in different cities and minimize the maximum distance of a city to a fire station. According to graph theory this means finding a set of k vertices for which the largest distance of any point to its closest vertex in the k-set is minimum. The vertices must be in a metric space, providing a complete graph that satisfies the triangle inequality.

1.2 OBJECTIVE

1. To find the best possible accurate result for maximum inputs.
2. To provide solution to a problem where maximum distance objects can also access a particular requirement in a minimum length of complexity.
3. To choose k vertices as centers in a weighted undirected graph in which the edge weights obey the triangle inequality so that the maximum distance of any vertex to its nearest center is minimized.

Chapter 2

SYSTEM REQUIREMENTS

2.1 FUNCTIONAL REQUIREMENTS

No	FUNCTIONAL REQUIREMENTS	FUNCTIONAL REQUIREMENTS DESCRIPTION
1	FR 1	Used to accept values for adjacency matrix
2	FR 2	used to create an adjacency matrix.
3	FR 3	used to calculate minimum distance.
4	FR 4	Finding the k centres

2.2 SOFTWARE AND HARDWARE REQUIREMENTS

HARDWARE:

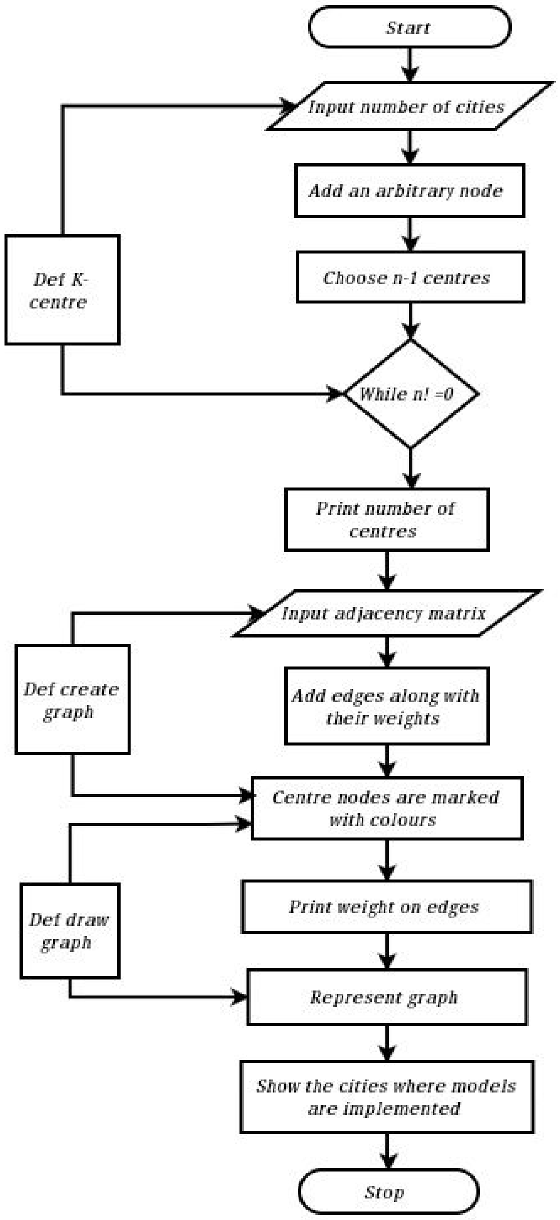
1. Processor Pentium 4
2. Hard disk drive 80GB
3. RAM 256 MB
4. Cache 512 KB

SOFTWARE:

1. Language used : Python 3.8
2. Libraries used : matplotlib and networkx
3. Operating system: Ubuntu or Windows 7 and above

Chapter 3
SYSTEM DESIGN

3.1 Architecture/Data Flow Diagram



3.2 MODULES

1. Networkx : NetworkX is a Python library for studying graphs and networks. NetworkX is a software released under the BSD-new license.
2. Matplotlib.pyplot : Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.
3. K_center(g,n) : To find the k centres where the infrastructure can be constructed.
4. Create graph : To create the original graph of cities with the distance between each in a numerical format.
5. Input : It's a text file where the inputs for the program are drawn from.
6. Draw_graph : Function that uses the numerical data to create a visual graph of the kcentres.
7. Main_function : Main part of the project which connects all the attributes to be called and executed.
8. Plt.show : plt. show() starts an event loop, looks for all currently active figure objects, and opens one or more interactive windows that display your figure or figures.

Chapter 4
SYSTEM IMPLEMENTATION

4.1 MODULE DESCRIPTION

We have many programs that we will interconnect by using them as modules which can be imported

The following are the important modules:

1.K_ centers module - To find k centres.

2.input module - To input adjacency matrix.

3.def Create Graph module - To create computational graph.

4.def Draw Graph module - To draw visual graph.

K_CENTERS

To find the k centres where the infrastructure can be constructed.The module implementation is given below:

```
def k_centers(G, n):
centers = []
cities = G.nodes()
#add an arbitrary node, here, the first node,to the centers list
centers.append((G.nodes())[0])
cities.remove(centers[0])
n = n-1 #since we have already added one center
#choose n-1 centers
```

```
while n!= 0:
city_dict = {}
for cty in cities:
min_dist = float("inf")
for c in centers:
min_dist = min(min_dist,G[cty][c]['length'])
city_dict[cty] = min_dist
#print city_dict
new_center = max(city_dict, key = lambda i: city_dict[i])
#print new_center
centers.append(new_center)
cities.remove(new_center)
n = n-1
#print centers
return centers
```

<u>def CREATE GRAPH</u>

To create the original graph of cities with the distance between each in a numerical format. The module implementation is given below:

```
#takes input from the file and creates a weighted undirected graph
def CreateGraph():
G = nx.Graph()
f = open('input.txt')
n = int(f.readline()) #n denotes the number of cities
wtMatrix = []
for i in range(n):
list1 = map(int, (f.readline()).split())
wtMatrix.append(list1)
```

```
 #Adds edges along with their weights to the graph
for i in range(n) :
for j in range(n)[i:] :
G.add_edge(i, j, length = wtMatrix[i][j])
noc = int(f.readline()) #noc,here,denotes the number of centers
return G, noc
```

<u>def DRAW GRAPH</u>

Function that uses the numerical data to create a visual graph of the kcentres.The module implementation is given below:

```
#draws the graph and displays the weights on the edges
def DrawGraph(G, centers):
pos = nx.spring_layout(G)
 color_map = ['blue'] * len(G.nodes())
 #all the center nodes are marked with 'red'
 for c in centers:
 color_map[c] = 'red'
nx.draw(G, pos, node_color = color_map, with_labels = True)
#with_labels=true is to show the node number in the output graph
edge_labels = nx.get_edge_attributes(G, 'length')
nx.draw_networkx_edge_labels(G, pos, edge_labels = edge_labels, font_size
=11)
 #prints weight on all the edges
```

4.2 PSEUDOCODE

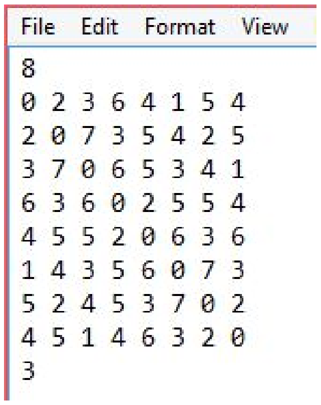
```
step 1 : START
step 2 : read an arbitrary node(the first node to the centers  list)
    step 2.1:choose n-1 centers.
    step 2.2:print the city distance
    step 2.3: print centers
step 3 : take an input from the file and create a weighted undirected graph.
    step 3.1:read number of cities.
    step 3.2:add edges along with their weights to the graph.
    step 3.3:return the number of centers.
step 4 : draw the graph and display the weights on the edges.
    step 4.1:all the center nodes are marked with color.
    step 4.2: with labels is equal to true is to show the node number in the
    output graph.
    step 4.3:print weight on all the edges.
step 5 : END
```

Chapter 5
OUTPUT SCREEN SHOTS

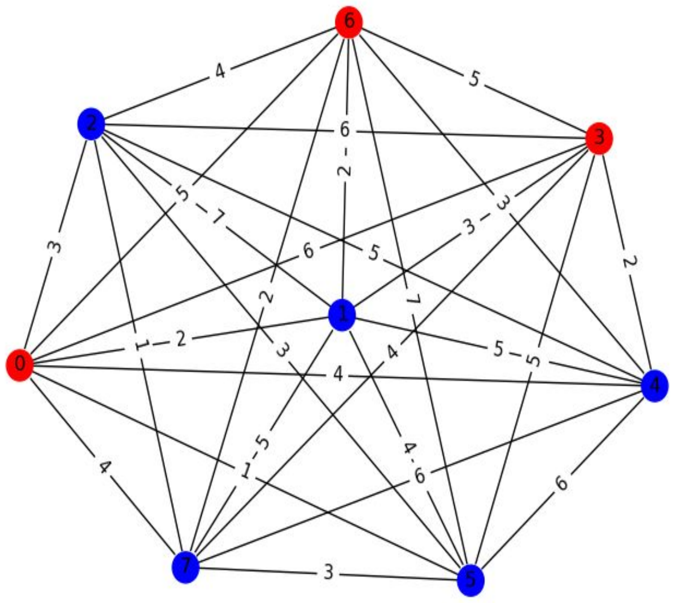
Example 1 :

Total number of cities : 8

Number of cities where the infrastructure has to be built : 3



```
8
0 2 3 6 4 1 5 4
2 0 7 3 5 4 2 5
3 7 0 6 5 3 4 1
6 3 6 0 2 5 5 4
4 5 5 2 0 6 3 6
1 4 3 5 6 0 7 3
5 2 4 5 3 7 0 2
4 5 1 4 6 3 2 0
3
```



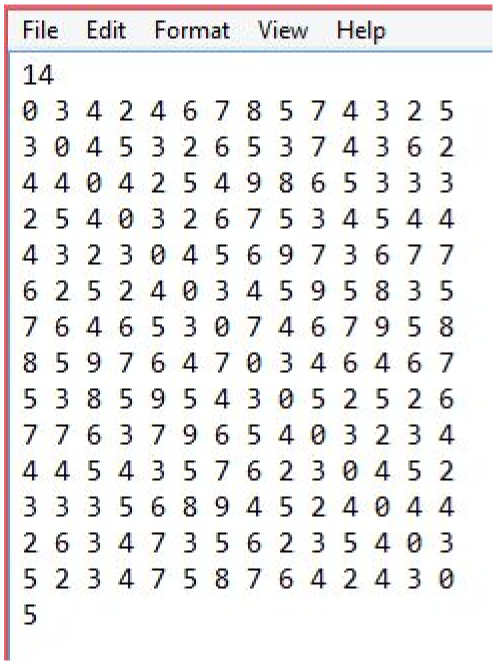
Cities where the infrastructure can be built are : City 0 ,City 3 ,City 6.

Total time taken: 0.003456788970

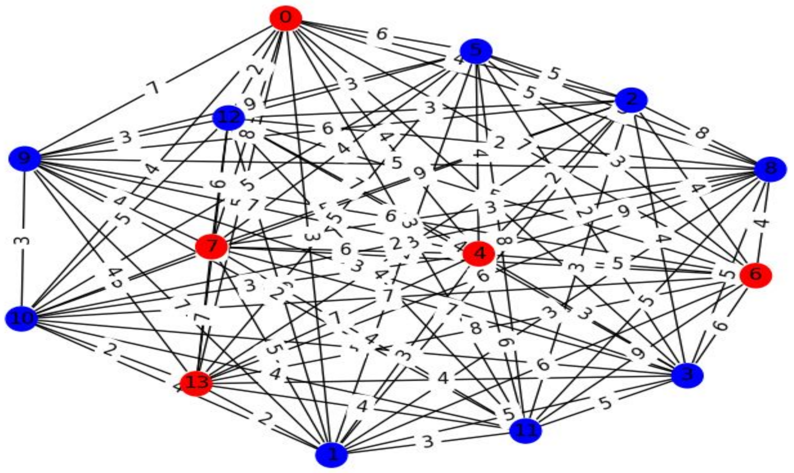
Example 2 :

Total number of cities : 14

Number of cities where the infrastructure has to be built : 5



```
14
0 3 4 2 4 6 7 8 5 7 4 3 2 5
3 0 4 5 3 2 6 5 3 7 4 3 6 2
4 4 0 4 2 5 4 9 8 6 5 3 3 3
2 5 4 0 3 2 6 7 5 3 4 5 4 4
4 3 2 3 0 4 5 6 9 7 3 6 7 7
6 2 5 2 4 0 3 4 5 9 5 8 3 5
7 6 4 6 5 3 0 7 4 6 7 9 5 8
8 5 9 7 6 4 7 0 3 4 6 4 6 7
5 3 8 5 9 5 4 3 0 5 2 5 2 6
7 7 6 3 7 9 6 5 4 0 3 2 3 4
4 4 5 4 3 5 7 6 2 3 0 4 5 2
3 3 3 5 6 8 9 4 5 2 4 0 4 4
2 6 3 4 7 3 5 6 2 3 5 4 0 3
5 2 3 4 7 5 8 7 6 4 2 4 3 0
5
```



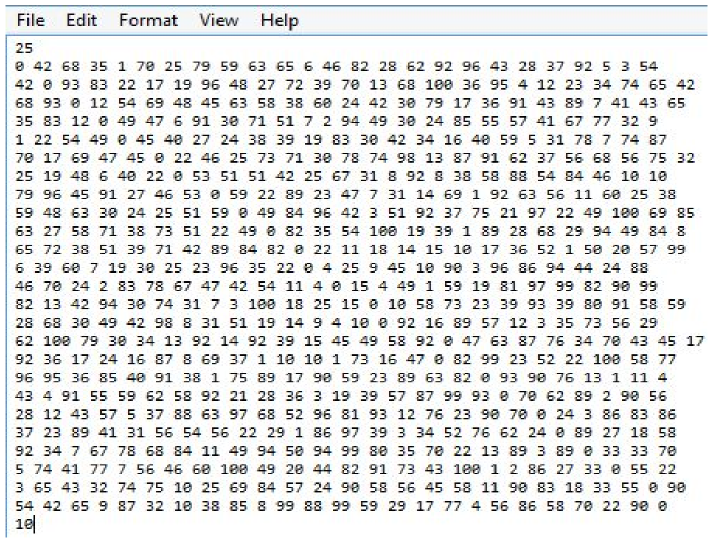
Cities where the infrastructure can be built are : City 0 ,City 4 ,City 6 ,City 7 and City 13.

Total time taken :0.39628076553344727

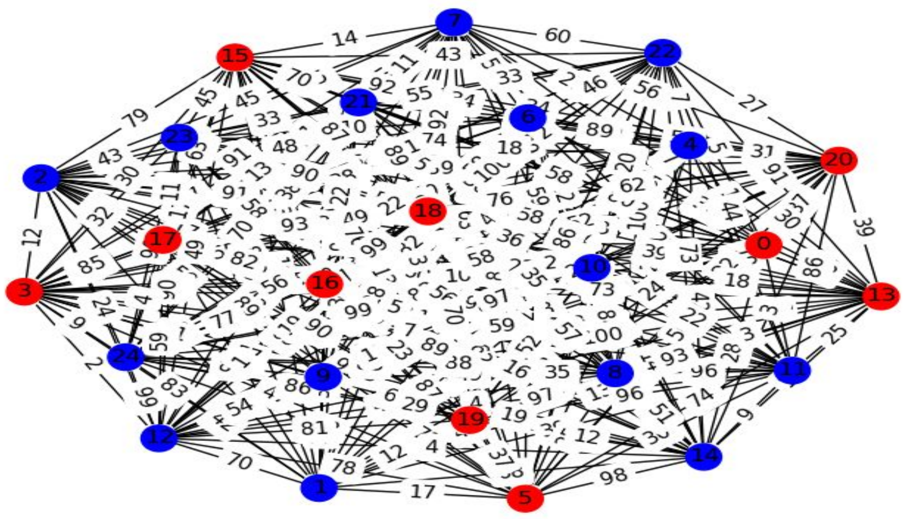
Example 3 :

Total number of cities : 25

Number of cities where the infrastructure has to be built : 10



```
25
0 42 68 35 1 70 25 79 59 63 65 6 46 82 28 62 92 96 43 28 37 92 5 3 54
42 0 93 83 22 17 19 96 48 27 72 39 70 13 68 100 36 95 4 12 23 34 74 65 42
68 93 0 12 54 69 48 45 63 58 38 60 24 42 30 79 17 36 91 43 89 7 41 43 65
35 83 12 0 49 47 6 91 30 71 51 7 2 94 49 30 24 85 55 57 41 67 77 32 9
1 22 54 49 0 45 40 27 24 38 39 19 83 30 42 34 16 40 59 5 31 78 7 74 87
70 17 69 47 45 0 22 46 25 73 71 30 78 74 98 13 87 91 62 37 56 68 56 75 32
25 19 48 6 40 22 0 53 51 51 42 25 67 31 8 92 8 38 58 88 54 84 46 10 10
79 96 45 91 27 46 53 0 59 22 89 23 47 7 31 14 69 1 92 63 56 11 60 25 38
59 48 63 30 24 25 51 59 0 49 84 96 42 3 51 92 37 75 21 97 22 49 100 69 85
63 27 58 71 38 73 51 22 49 0 82 35 54 100 19 39 1 89 28 68 29 94 49 84 8
65 72 38 51 39 71 42 89 84 82 0 22 11 18 14 15 10 17 36 52 1 50 20 57 99
6 39 60 7 19 30 25 23 96 35 22 0 4 25 9 45 10 90 3 96 86 94 44 24 88
46 70 24 2 83 78 67 47 42 54 11 4 0 15 4 49 1 59 19 81 97 99 82 90 99
82 13 42 94 30 74 31 7 3 100 18 25 15 0 10 58 73 23 39 93 39 80 91 58 59
28 68 30 49 42 98 8 31 51 19 14 9 4 10 0 92 16 89 57 12 3 35 73 56 29
62 100 79 30 34 13 92 14 92 39 15 45 49 58 92 0 47 63 87 76 34 70 43 45 17
92 36 17 24 16 87 8 69 37 1 10 10 1 73 16 47 0 82 99 23 52 22 100 58 77
96 95 36 85 40 91 38 1 75 89 17 90 59 23 89 63 82 0 93 90 76 13 1 11 4
43 4 91 55 59 62 58 92 21 28 36 3 19 39 57 87 99 93 0 70 62 89 2 90 56
28 12 43 57 5 37 88 63 97 68 52 96 81 93 12 76 23 90 70 0 24 3 86 83 86
37 23 89 41 31 56 54 56 22 29 1 86 97 39 3 34 52 76 62 24 0 89 27 18 58
92 34 7 67 78 68 84 11 49 94 50 94 99 80 35 70 22 13 89 3 89 0 33 33 70
5 74 41 77 7 56 46 60 100 49 20 44 82 91 73 43 100 1 2 86 27 33 0 55 22
3 65 43 32 74 75 10 25 69 84 57 24 90 58 56 45 58 11 90 83 18 33 55 0 90
54 42 65 9 87 32 10 38 85 8 99 88 99 59 29 17 77 4 56 86 58 70 22 90 0
10
```



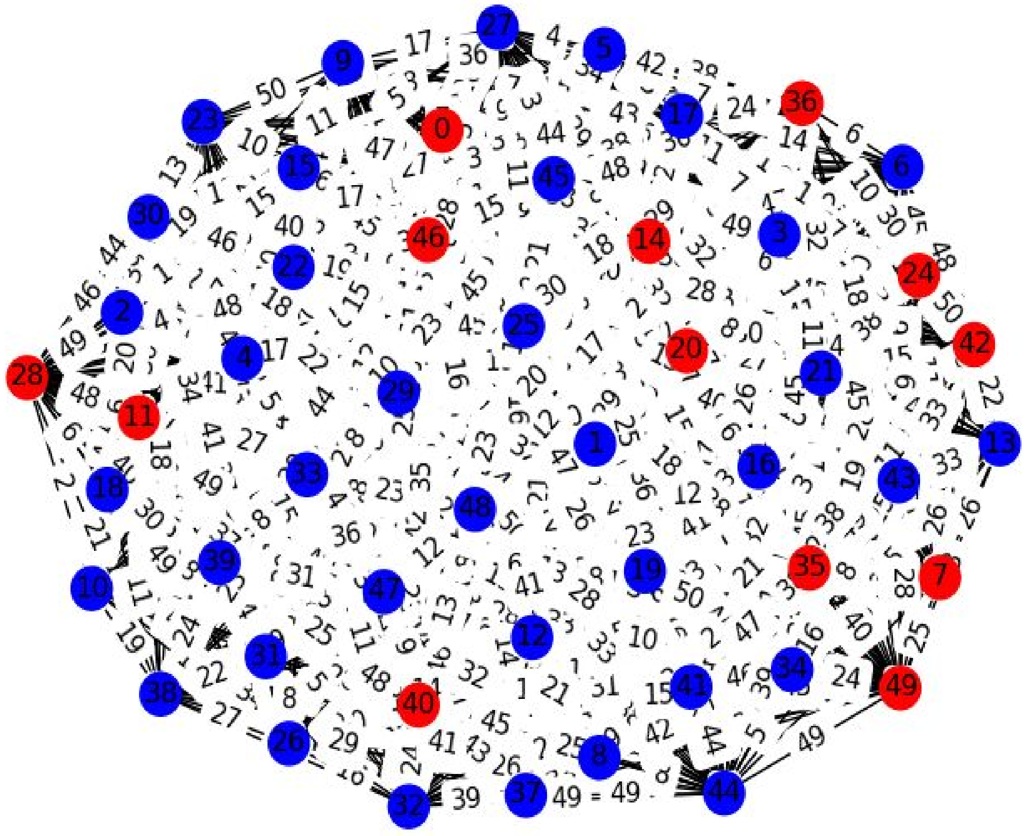
Cities where the infrastructure can be built are : City 0, City 3, City 5, City 13, City 15, City 16, City 17 , City 18, City 19, City 20.

Total time taken:0.67449760437011345

Example 4 :

Total number of cities : 50

Number of cities where the infrastructure has to be built : 13



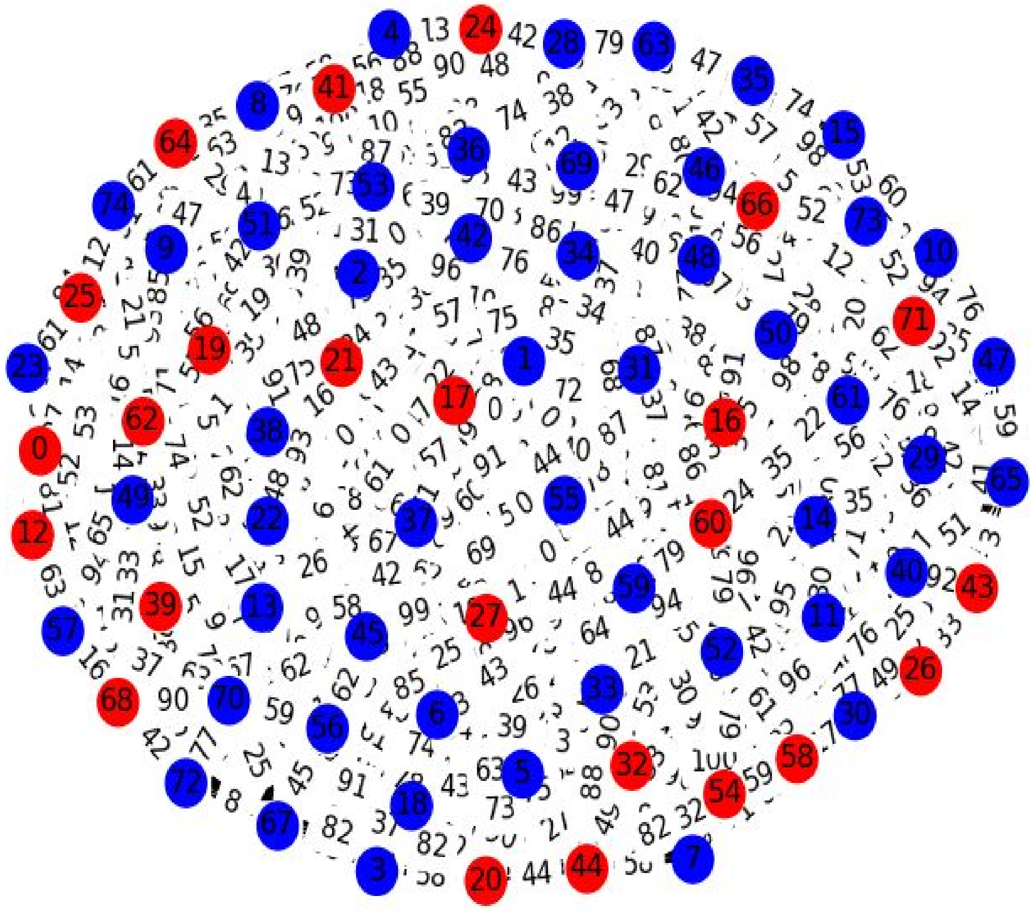
Cities where the infrastructure can be built are :City 0, City 7, City 11, City 14, City 20, City 24, City 28 , City 36, City 42, City 46, City 49.

Total time taken :1.7722418308258076540

Example 5 :

Total number of cities : 75

Number of cities where the infrastructure has to be built : 24



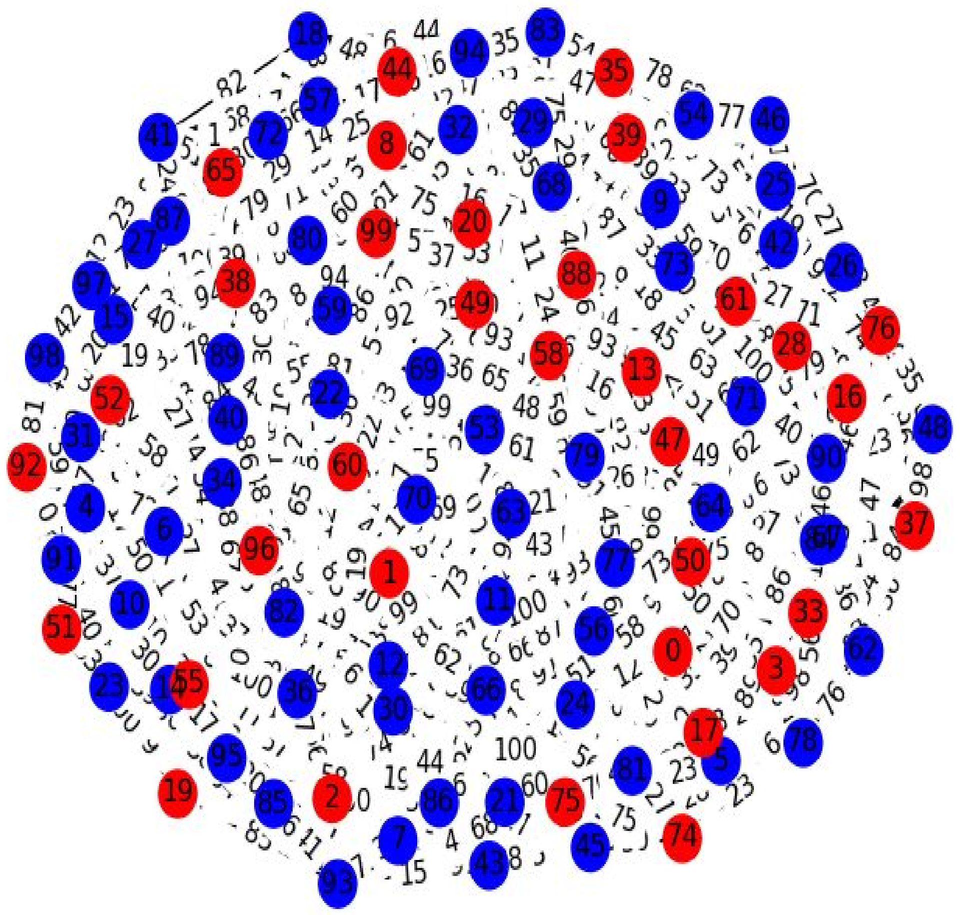
Cities where the infrastructure can be built are :City 0, City 12, City 16 ,City 17 , City 19, City 20, City 21, City 24, City 25, City 26, City 32 , City 39 ,City 41,City 43 , City 44, City 54,City 58, City 60, City 62, City 64, City 66, City 71.

Total time taken :3.65665888708976

Example 6:

Total number of cities : 100

Number of cities where the infrastructure has to be built : 34



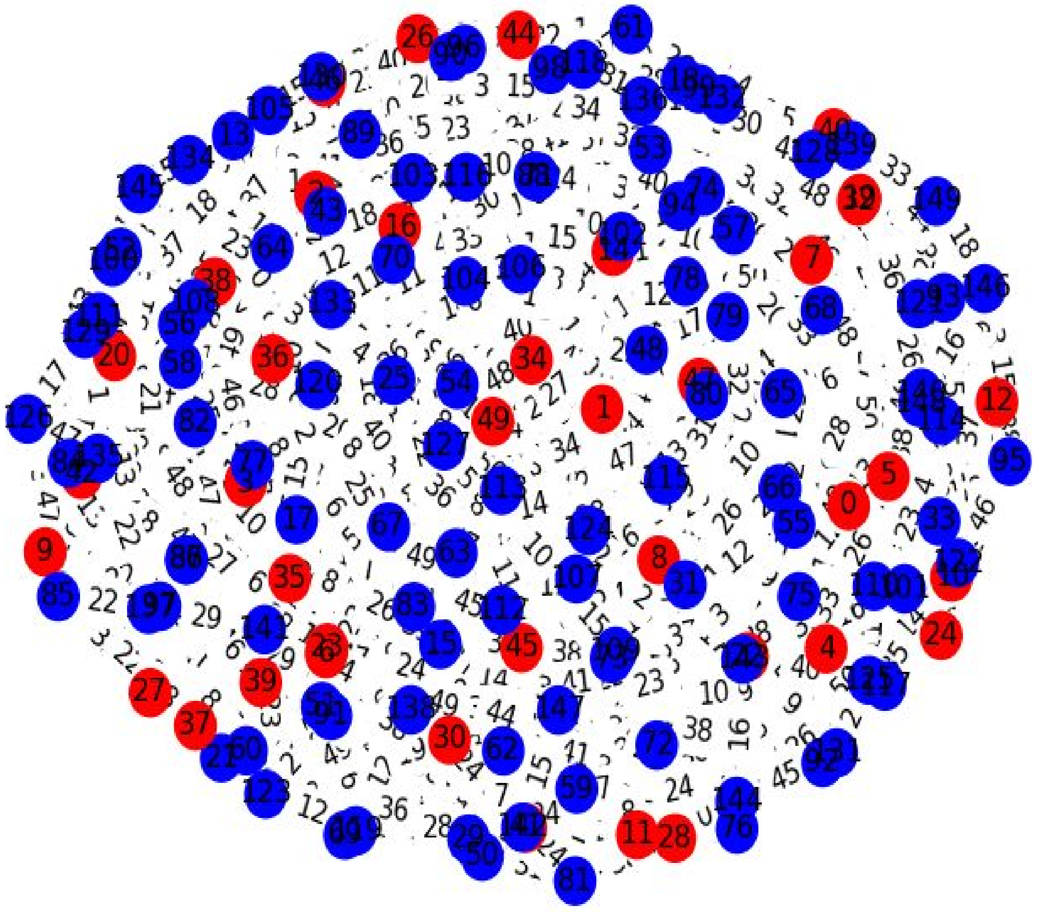
Cities where the infrastructure can be built are :City 0, City 1 , City2 ,City 3, City8 ,City 13 , City16 , City 17 , City 19 , City20 , City28 , City 33 ,City 35 ,City 37 , City 38 , City 39 ,City 44 , City 47 , City 49 , City 50 , City 51, City 52, City 55, City58, City60, City61, City65, City74, City75, City76, City88, City92, City96, City 99.

Total time taken :6.883880615234375

Example 7:

Total number of cities : 150

Number of cities where the infrastructure has to be built : 39



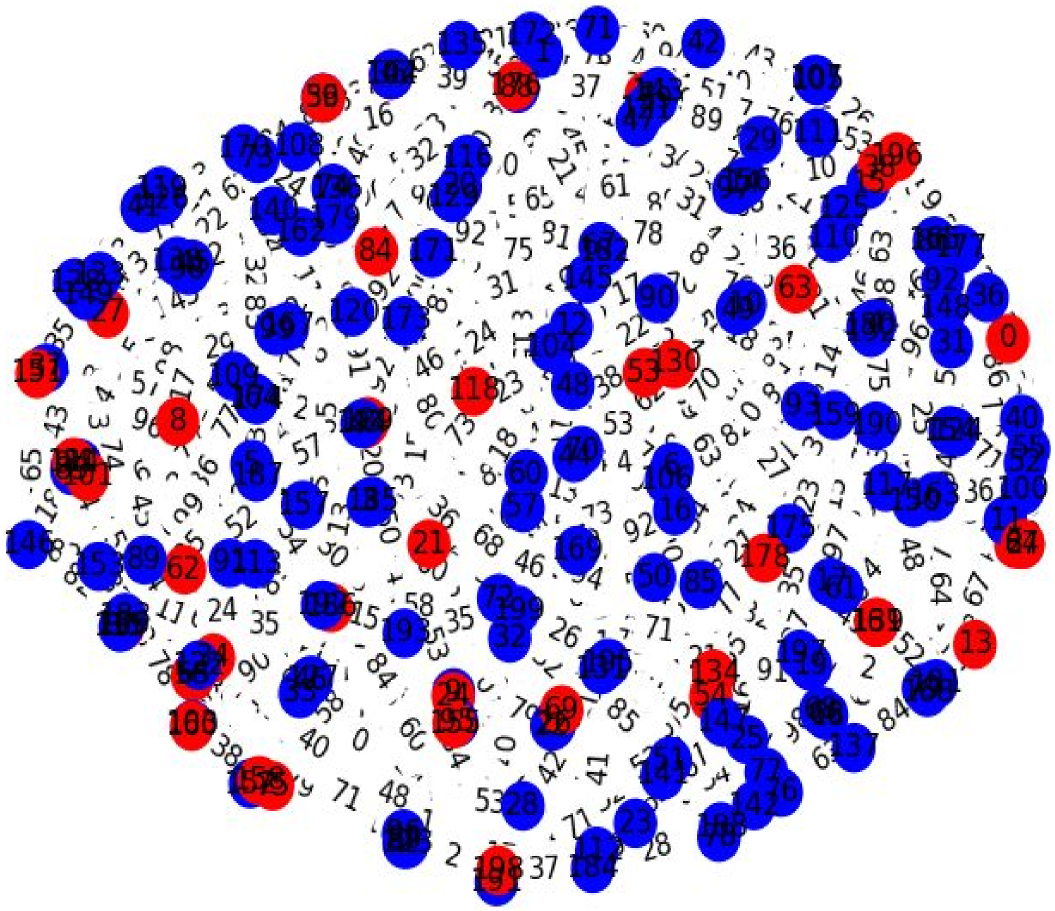
Cities where the infrastructure can be built are :City 0, City 1 ,City 2,City 3, City 4,City 5,City 6 ,City 7 ,City 8, City 9,City 10, City 11,City 12 ,City 14 City16 , City 20 , City 23 ,City24, City 26 , City 28,City 30 ,City 32 ,City 34,City 35 ,City 36 , City 37 , City 38 ,City 39 , City 40,City 42, City 44 , City 45 , City 49 ,City 72, City 122, City 135, City 136, City 140, City 146.

Total time taken:14.876568794233401

Example 8:

Total number of cities : 200

Number of cities where the infrastructure has to be built : 43



Cities where the infrastructure can be built are :City 0, City 1 , City 3 ,City 4 , City8 ,City 13 , City16 , City 19 , City21 , City27 , City 33 ,City 35 ,City 37 , City 38 , City 39 ,City 44 , City 47 , City 49 , City 50 , City 51, City 53, City 55, City 58, City 60, City 62, City 63, City 75, City 76, City 88, City 92, City 96, City 118, City 124, City 128, City 134, City 139, City 144, City 150, City 154 , City 160, City 178, City 182, City 190.

Total time taken: 16.3492665290854321

Time Complexity Analysis

The below given table represents the time complexity to find k number of centres out of the the total given centres using triangular inequality.This algorithm simply chooses the point farthest away from the current set of centers in each iteration as the new center.

- Pick an arbitrary point C1' into C1
- For every point v belongs to V compete d1[v] from C1'
- Pick the point C2' with highest distance from C1'
- Add it to the set of centers and denote this expanded set of centers as C2. Continue this till k centers are found.
- The ith iteration of choosing the ith center takes O(n)time.
- There are k such iterations.
- Thus, overall the algorithm takes O(nk)time

Thus the Time Complexity of the algorithm is O(nk) where k is the number of centres where the infrastructure needs to be set up.

No of cities	Total Time in Seconds
8	0.0034576896
14	0.396270809977654
25	0.6744978865430643
50	1.77225345879065
75	3.65667543890321
100	6.883880615234375
150	14 .876568794332928
200	16.349266529085641

Chapter 5

CONCLUSION

The project urban infrastructure development program helps us to set up a required number of infrastructure centers among the provided number of cities.The minimum formulation minimizes the sum of the weighted distances between facilities while the maximum formulation minimizes the overall maximum distance between facilities. The output graph which is the solution to this problem gives the highest profit choice that most efficiently serves the needs of all cities. The problem can be solved using a greedy approach which takes time complexity of O(n3). In order to improve the time complexity we have used a dynamic programming approach. Using dynamic programming we can analyze recursive functions into two pieces: which is the work done by each function that is to find the minimum distance from the centres and the amount of times that work is done , that is the final approximate k centres .Hence by using dynamic approach the time complexity of the program is improved to O(nk) where k is the number of centres. Hence ,our new optimization approach shows a significant improvement in finding the k centres in less amount of time.

REFERENCES

[1] https://ieeexplore.ieee.org/document/7408462

[2] https://ieeexplore.ieee.org/abstract/document/1017616 (By Tapas Kanungo ,David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, Angela Y Wu)

[3] https://ieeexplore.ieee.org/document/7367406

[4] https://ieeexplore.ieee.org/document/143543/authors#authors
(By Qingzhou Wang and Kam Hoi Cheng)

[5]https://www.geeksforgeeks.org/k-centers-problem-set-1-greedy-approximate-algorithm/

[6] https://en.wikipedia.org/wiki/Metric_k-center