Discrete Optimization HW 1

Yichen Xu

September 2020

Part 1, Approach A

Assume that all the points can be distribution point, and for the distribution centers, they also need to be served by some points which could be itself.

Variables:

x_j: If node j is selected to be the distribution center.

y_{ij}: If node i is served by distribution center j.

Data:

c_{ij}: The cost of assigning distribution center j to node i. That is , in this question, the Euclidean distance between node i and node j.

d_i: The demand of node i, which is the population in this question.

p: The total number of distribution center required.

Objective:

$$Min \quad \sum_{i=1}^{241}\sum_{j=1}^{241} y_{ij}c_{ij}d_i$$

Subject to:

$$\sum_{j=1}^{241} y_{ij} = 1 \quad \forall i \in \{1, 2, \ldots, 241\}$$

$$y_{ij} \leq x_j \quad \forall i \in \{1, 2, \ldots, 241\}, j \in \{1, 2, \ldots, 241\}$$

$$\sum_{j=1}^{241} x_j = p$$

$$y_{ij}, x_j = 0, 1 \quad \forall i \in \{1, 2, \ldots, 241\}, j \in \{1, 2, \ldots, 241\}$$

code in Julia:

E.x.,p = 5

```
using JuMP,Gurobi
using CSV
data = CSV.read("data_transfered.csv")
I = [1:241;]
J = [1:241;]
de = 110.25
c = zeros((241,241))
for i = 1:241
    for j = 1:241
        c[i,j] = de*((data[i,8]-data[j,8])^2 + ((data[i,9]-data[j,9])*cos(data[
    j,8]))^2)^(1/2)
    end
end
p = 5
Model_p5 = Model(Gurobi.Optimizer)
@variable(Model_p5,x[i in I],Bin) # if city i is server
@variable(Model_p5,y[i in I,j in I],Bin) #if city i is served by city j
@objective(Model_p5,Min,sum(y[i,j]*c[i,j]*data[i,10] for i in I , j in I))
@constraints(Model_p5,begin
        [i in I], sum(y[i,j] for j in I) == 1
        [i in I , j in I], y[i,j]-x[j] <= 0
        #[i in I], y[i,i] == x[i]
        sum(x[i] for i in I) == p
    end)
optimize!(Model_p5)
print("Optimal value:")
println(objective_value(Model_p5))
print("node ")
for i in I
    if value(x[i]) > 0.5
        print(i," ")
    end
end
print("are selected as distribution centers")
```

Result:

p = 5

Optimal value: 370251583.81199

node 9 80 88 180 190 are selected as distribution centers

p = 10

Optimal value: 256201002.868491

node 9 51 80 88 103 161 167 180 190 208 are selected as distribution centers

p = 15

Optimal value: 192872310.991225

node 9 51 70 80 88 103 126 143 161 167 180 190 192 201 208 are selected as distribution centers

p = 20

Optimal value: 147620103.548795

node 9 46 47 51 64 70 80 88 103 104 112 122 126 129 143 161 167 180 192 201 are selected as distribution centers

p = 25

Optimal value: 118800405.624131

node 9 30 46 47 51 64 70 80 88 103 104 112 122 126 129 139 143 161 167 168 177 180 192 201 214 are selected as distribution centers

p = 30

Optimal value: 98245453.3119142

node 6 9 17 30 46 47 51 56 64 70 80 88 103 104 112 115 126 129 139 143 161 167 168 177 180 192 199 201 208 214 are selected as distribution centers

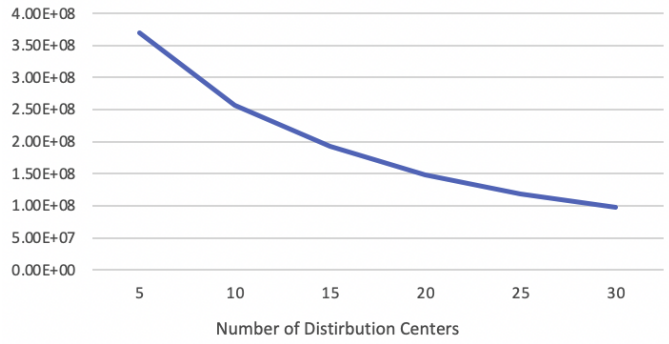

Figure 1: Cost of the system For Approach A

Part 1, Approach B

For the second method, as for initialization, a p-median model with p = 2 would be applied to the system. Then the whole network would be separated into two parts, and in each part there would be exactly one distribution center and all the nodes connected to that center. After getting two parts, one distribution center would be added to each of these areas using p-median model considering the existed one distribution center inside the area.

The preparation of programming (using packages, read data, etc.)

```
using JuMP, Gurobi
using CSV
data = CSV.read("data_transfered.csv")
de = 110.25
c = zeros((241,241))
for i = 1:241
    for j = 1:241
        c[i,j] = de*((data[i,8]-data[j,8])^2 + ((data[i,9]-data[j,9])*cos(data[
    j,8]))^2)^(1/2)
    end
end
```

Three functions would be used in this approach.

1.function of building and solving p-median model giving number of p and the network set.

```
# n stands for the SET of all points in the network
# p stands for the maximum number of distribution point
# assume that all the points could be distribution point
function pmedian(n, p)
    model = Model(Gurobi.Optimizer)
    @variable(model, x[i in n], Bin)
    @variable(model, y[i in n, j in n], Bin)
    @objective(model, Min, sum(c[i, j] * y[i, j] * data[i,10] for i in n, j in
    n))
    @constraints(model,begin
        [i in n], sum(y[i,j] for j in n) == 1
        [i in n , j in n], y[i,j]-x[j] <= 0
        sum(x[i] for i in n) == p
        end)
```

```
    status = optimize!(model)

    z_opt = JuMP.objective_value(model)
    x_opt = JuMP.value.(x)
    y_opt = JuMP.value.(y)

    return z_opt, x_opt, y_opt
end
```

2. function of adding one more distribution center in the area based on p-median model.

```
# this function is to add one more distribution center inside the network which
    already has ONE distribution center
# n stands for the SET of all points in the network
# p_1 stands for the existed distribution center
function plus_1_point(n,p_1)
    model = Model(Gurobi.Optimizer)
    @variable(model,x[i in n],Bin)
    @variable(model,y[i in n, j in n],Bin)
    @objective(model,Min, sum(c[i,j] * y[i,j] * data[i,10] for i in n, j in n))
    @constraints(model,begin
        [i in n], sum(y[i,j] for j in n)==1
        [i in n, j in n], y[i,j] - x[j] <= 0
        sum(x[i] for i in n) == 2
        x[p_1] == 1
        end)
    status = optimize!(model)

    z_opt = JuMP.objective_value(model)
    x_opt = JuMP.value.(x)
    y_opt = JuMP.value.(y)

    return z_opt, x_opt, y_opt
end
```

3. function of separating the result of p-median programming into two area with only one distribution center and all nodes connected to that center. The cost would also be separated.

```
# this function is to seperate the result from function pmedian(p = 2) and function plus_1_point
# k stands for the result of the function pmedian(p = 2) and function plus_1_point
# n stands for the set of nodes used in the function providing this result
# only 2 distribution center must be existed in the network after running the function pmedian and plus_1_point
function separate(k,n)
    distr=[]
    for i in n
        if k[2][i] > 0.5
            push!(distr,i)
        end
    end
    net_1 = []
    net_2 = []
    for i in n
        if k[3][i,distr[1]] > 0.5
            push!(net_1,i)
        end
    end
    for i in n
        if k[3][i,distr[2]] > 0.5
            push!(net_2,i)
        end
    end
    cost_1 = sum(c[i,j]*data[i,10] for i in net_1, j = distr[1])
    cost_2 = sum(c[i,j]*data[i,10] for i in net_2, j = distr[2])
    return distr[1],net_1,cost_1,distr[2],net_2,cost_2
end
# for the result of this function, element 1 & 4 are the two distribution points, 2 & 5 are the network connected to these two distribution centers, and 3 & 6 are the cost related to these two points
```

Process of getting 5, 10, 15, 20, 25 and 30 distribution centers using this approach:

1. Run pmedian function with n = [1:241], p = 2, then use separate function to get the result readable. The first two distribution centers are 88 and 190, and the

cost related to them are $353952706.583545 and $375813506.146286.

2. Run function plus_1_point to each of the two area provided by the step 1, then use function separate to get the result for 4 centers readable. Now the four distribution centers are 88, 161, 9 and 190, and the cost related to them are $183061396.362415, $70622604.7906699, $13321526.3169715 and $285151922.080468.

3. Run function plus_1_point to the area with the highest cost, which is area 190, then run function separate to the result. Now 5 distribution centers are selected, which are **88, 161, 9, 190 and 201**, and the cost related to them are $183061396.362415, $70622604.7906699, $13321526.3169715, $106522562.320426 and $114149595.436704. The total cost for getting 5 distribution centers in the network using the approach B is **$487677685.227186**.

4. As the number of nodes connected to point 9 is only 6 now, that area would be hold. Run function plus_1_point to all the other areas, then run function separate to the result. Now 7 distribution centers are provided, which are 80, 9, 161, 90, 88, 180, 201. The cost related to them are $35597964.44, $13321526.32, $29672504.16, $106522562.3, $66364824.16, $10188723.83 and $114149595.4.

5. Run function plus_1_point to the 3 areas with the highest cost which are area served by node 201, 88 and 80, then run function separate to the result. Now 10 distribution centers are selected, which are **168, 201, 70, 88, 80, 126, 9, 161, 190 and 180**. The total cost for using these 10 distribution centers in the network is **$319866100.3**.

6. Run function plus_1_point to all the other areas except for the one connected by node 9, then run function separate to the result. Now 13 points are selected as distribution centers which are 168, 201, 70, 88, 80, 126, 103, 161, 190, 208, 139, 180, 9. The cost for area with distribution center 103 and 139 are 0 because no other point is arranged to their area.

7. Run function plus_1_point to the areas with point 201 and 190 with function separate. Now 15 points are selected as distribution centers which are **51, 201, 167, 190, 168, 70, 88, 80, 126, 103, 161, 208, 139, 180 and 9**. The total cost of having 15 distribution centers in the system are **$229824899.3**.

8. Run function plus_1_point to the areas with point 88, 126, 161, 208 and 180 following with function separate. Now 20 points are selected as distribution centers

which are **51, 201, 167, 190, 88, 143, 126, 177, 17, 161, 199, 208, 100, 180, 168, 70, 80, 103, 139 and 9**. The total cost for getting 20 distribution centers are **$203561415.6**.

9. Run function plus_1_point to the areas with point 80. Now 21 points are selected with the new point 228. Now run function plus_1_point to the areas with point 51, 201, 190 and 88 following by function separate. Now we get 25 distribution centers in the system which are **167, 177, 17, 199, 100, 228, 168, 70, 103, 139, 9, 51, 188, 201, 206, 190, 192, 88, 112, 143, 126, 161, 208, 180 and 80**. The total cost for getting 25 distribution centers are **$167738839.1**.

10. Run function plus_1_point to the areas with point 143, 126, 161, 208 and 80, following by the function separate. Now there are 30 distribution centers in the system which are **167, 177, 17, 199, 100, 228, 168, 70, 103, 139, 9, 51, 188, 201, 206, 190, 192, 88, 112, 143, 126, 161, 208, 180, 80, 47, 86, 48, 241 and 58**. The total cost for having 30 distribution centers in the system is **$149116071.1**.

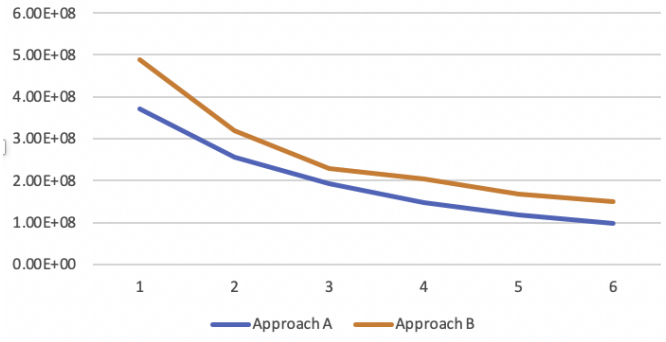

Figure 2: Cost of the System for Approach A B

The approach B would separate all the areas and add points to them, but the result of the separation might be unbalanced so that there are several areas with only one point inside and some area could be separated for more times. Thus, the classical p-median approach works better than this one.

Part 2

Variables:

x_j: If node j is selected to be the distribution center.

y_{ij}: If node i is served by distribution center j.

C: If the number of distribution centers in the first 20 cities is is more than 2.

Data:

c_{ij}: The cost of serving node i by distribution center j. That is , in this question, the Euclidean distance between point i and j.

d_i: The demand of node i, which is the population in this question.

p: The total amount of distribution center required.

Objective:

$$Min \quad \sum_{i=1}^{241}\sum_{j=1}^{241} y_{ij}c_{ij}d_i$$

Subject to:

$$\sum_{j=1}^{241} y_{ij} = 1 \quad \forall i \in \{1, 2, \ldots, 241\}$$

$$y_{ij} \leq x_j \quad \forall i \in \{1, 2, \ldots, 241\}, j \in \{1, 2, \ldots, 241\}$$

$$\sum_{j=1}^{241} x_j = 10$$

$$\sum_{j=1}^{20} x_j \leq 21 \sum_{j=30}^{40} x_j \quad \textbf{modification i}$$

$$\sum_{j=1}^{20} \geq 3C$$

$$\sum_{j=1}^{20} \leq 2 + 18C$$

$$\sum_{j=30}^{40} \geq C$$

$$\sum_{j=30}^{40} \leq 10C \quad \textbf{modification ii}$$

$$x_{39} + x_{230} \leq 1 \quad \textbf{modification iii}$$

$$x_{39} + x_{230} + x_{139} + x_{164} = 1 \quad \textbf{modification iv}$$

$$x_{112} + x_{103} + x_{139} + x_{70} + x_{47} + x_{17} + x_9 + x_{168} + x_{64} + x_{117} \leq 3 \quad \textbf{modification v}$$

$$y_{ij}, x_j, C = 0, 1 \quad \forall i \in \{1, 2, \ldots, 241\}, j \in \{1, 2, \ldots, 241\}$$

Code for the model with all the modifications

The other modifications would be disabled by # when one of them works.

```
using JuMP,Gurobi

using CSV
data = CSV.read("data_transfered.csv")

I = [1:241;]
J = [1:241;]
de = 110.25

c = zeros((241,241))
for i = 1:241
    for j = 1:241
        c[i,j] = de*((data[i,8]-data[j,8])^2 + ((data[i,9]-data[j,9])*cos(data[
    j,8]))^2)^(1/2)
    end
end

p = 10
K = [112 103 139 70 47 17 9 168 64 177]
Model_modify = Model(Gurobi.Optimizer)
@variable(Model_modify,x[i in I],Bin) # if city i is distribution center
@variable(Model_modify,y[i in I,j in J],Bin) #if city i is served by city j
@variable(Model_modify,C,Bin) #if there's more than 2 distributio centers in
    the first 20 cities
@objective(Model_modify,Min,sum(y[i,j]*c[i,j]*data[i,10] for i in I , j in J))
@constraints(Model_modify,begin
        [i in I], sum(y[i,j] for j in J) == 1
        [i in I , j in J], y[i,j]-x[j] <= 0
        sum(x[i] for i in I) == p
        # modification 1
        sum(x[i] for i in 1:20) <= 21*sum(x[i] for i in 30:40)
        # modification 2
        sum(x[i] for i in 1:20) >= 3C
        sum(x[i] for i in 1:20) <= 2+18C
        sum(x[i] for i in 30:40) >= C
        sum(x[i] for i in 30:40) <= 10C
        # modification 3
        x[39] + x[230] <= 1
```

```
        # modification 4
        x[39] + x[230] + x[139] + x[164] == 1
        # modification 5
        sum(x[i] for i in K) <= 3
    end)
```

The result for models with each modification:

Modification i:

Optimal value: $257614243.677668

node 46 51 80 88 103 161 167 180 190 208 are selected as distribution centers

Modification ii:

Optimal value: $256201002.868491

node 9 51 80 88 103 161 167 180 190 208 are selected as distribution centers

Modification iii:

Optimal value: $256201002.868491

node 9 51 80 88 103 161 167 180 190 208 are selected as distribution centers

Modification iv:

Optimal value: $267112828.69387776

node 9 51 80 88 139 161 167 180 190 208 are selected as distribution centers

Modification v:

Optimal value: $256201002.868491

node 9 51 80 88 103 161 167 180 190 208 are selected as distribution centers