**ISE 247**

**Report**

**Optimization of Network and Facility Design**

**[](https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwjEotKp1vfdAhXmHDQIHXm7CP8QjRx6BAgBEAU&url=https://techcrunch.com/2013/07/19/san-jose-states-bold-experiment-in-online-ed-disappoints-suspends-pilot-with-udacity/&psig=AOvVaw1PiEQrTriKUo8qxKXlAOWu&ust=1539116601226785)**

**Submitted To**

**Professor Hongrui Liu**

**on**

**May 13, 2019**

**by**

**Group**

**Vivek Shah**

**Rohan Patil**

**TABLE OF CONTENTS**

1. **Executive Summary.**
2. **Network design with demand divisibility**

**2.1 Problem Statement & Formulation.**

**2.2 Program Design.**

**2.3 Computational Results**

1. **Quadratic Assignment for Facility Design with Departments of same area**

**3.1 Problem Statement & Formulation**

**3.2 Program Design**

**3.3 Computational Results**

**1. Executive Summary:**

The basic principle of Supply Chain is to help companies that face challenges when they buy, manufacture, move, sell and service products. Its is important that all these processes take place in a smooth manner with minimal cost and effort. The main objective of our project revolves around 2 specific tasks i.e. Network Design and Facility Design. The tasks that were performed was based on the knowledge imbibed during ISE 247 course of Logistics and Supply Chain, software skills were used such as Python for coding part and Cplex for solving the Linear Programming model.

**2. Network Design Formulation:**

**2.1 Problem Formulation:**

Our project is based on Demand Divisibility which means that the demand for a customer is divided and that 1 Customer can get materials from multiple Distribution Centers. Network Design problem provides an optimal cost of transportation, this cost includes the unit transportation cost from one distribution center to customer and includes the fixed cost which is provided for each Distribution Center. The problem is formulated in a such a way that the customer need gets satisfied keeping in mind the capacity of the Distribution Center.

800

1600

800

700

Fixed Cost

40

70

30

30

Capacity

1

2

4

3

8

7

15

16

12

4

6

4

Total Transportation Cost

9

8

6

6

Customer

1

2

3

40

30

40

Demand

**Decision Variable:**

Yi = 1 (if selected)

0(otherwise).

Xij = 1 (from DC i to customer j)

0 (otherwise).

Yij = 1 (from DC i for capacity j)

0 (otherwise).

**Objective:**

Minimize Total Cost: +

**Min:** 4X11 + 6X12 + 12X13 + 16X21 + 6X22 + 15X23 + 8X31+ 6X32 + 7X33 + 4X41 **+** 8X42 + 9X43 + 700Y1 + 800Y2 + 1600Y3 + 800Y4

**Subject To:**

**C1:** X11 + X12 + X13 ≤ 30Y1

**C2:** X21 + X22 + X23 ≤ 30Y2

**C3:** X31 + X32 + X33 ≤ 40Y3

**C4:** X41 + X42 + X43 ≤ 70Y4

**DC1:** X11 + X21 + X31 + X41 ≥ 40

**DC2:** X12 + X22 + X32 + X42 ≥ 30

**DC3:** X13 + X23 + X33 + X43 ≥ 40

**Binary:** X11, X21, X31, X41, X12, X22, X32, X42, X13, X23, X33, X43, Y1, Y2, Y3, Y4.

**2.2. Program Design:**

To design a network design problem the inputs that are required are the capacities of distribution centers, cost of distribution centers, demand of customers, unit transportation cost from each distribution center to the customer demand. These data’s available must be first read.

Eg:

Unit\_Cost\_Transportation = []

with open ('small\_example\_network design/unit\_tran\_cost.txt', 'r') as Trans\_Cost:

for line in Trans\_Cost:

inner\_list = [int(elt.strip()) for elt in line.split(',')]

Unit\_Cost\_Transportation.append(inner\_list)

Unit\_Cost\_Transportation

Once we read the data, the data is stored in a list where it is converted from a string to an integer using the for loop. In the above code unit transportation cost is stored as a matrix by converting the string into integers using for loop and forming an inner list of each rows. Similarly, distribution center capacity, cost and customer demand are read and converted into an integer.

Now to assign each value in a list a unique name a list of demand and distribution center (DC) is created. Assignment of unique names to each value in lists can be done by making dictionaries.

Eg.

Dic\_Trancosts = makeDict([Demandlist,DClist],Unit\_Cost\_Transportation,0)

In the above example, the transportation dictionary made is two-dimensional containing demand and DC list as keys. For instance, if there are two customer demands D1, D2 and two distribution centers S1, S2 then transportation cost to meet demand D1 from S1 will assign a value from unit transportation cost list. Similarly, the DC capacity, cost and customer demand dictionary will be made depending upon their dimension.

Then the linear programming model is defined where the objective is to minimize the transportation cost is mentioned.

In a similar fashion the LP variables x and y dictionary are made where the variable is 2 or 1 dimensional, category being binary or continuous and lower and upper bounds are mentioned and set.

Eg.

x = LpVariable.dicts("X", (Demandlist,DClist), lowBound = 0, cat ="Integer")

After defining the variables, the route is defined between the customer demand and DC list using the route function. This route function is used to sum the all the possible transportation cost and the DC capacity is added to it to form the objective function. Once the objective function is done, the constraints are made in such a way that when we read the LP file the constraints match the manual formulation.

The problem.solve() or Cplex should be used to solve the lp problem. The Linear programming problem should not have any non-linear terms as lp solver cannot solve non-linear terms. The code should be tested on a small example and should be made in a general form.

**2.3 Computational Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case | No. of DC | No. of Customers | Instance | Objective Function Value | Optimal | Run Time |
| 1 | 20 | 2000 | 1 | $2,314,680.00 | Yes | 17.15 secs |
| 20 | 2000 | 2 | $2,501,864.00 | Yes | 18.67 secs |
| 20 | 2000 | 3 | $2,236,759.00 | Yes | 16.17 secs |
| 2 | 40 | 4000 | 1 | $3,876,294.00 | Yes | 1.05 mins |
| 40 | 4000 | 2 | $4,492,178.00 | Yes | 1.32 mins |
| 40 | 4000 | 3 | $4,463,406.00 | Yes | 1.25 mins |
| 3 | 50 | 5000 | 1 | $4,840,198.00 | Yes | 2 hours |
| 50 | 5000 | 2 | $5,149,412.00 | Yes | 2 hours |
| 50 | 5000 | 3 | $5,387,545.00 | Yes | 2 hours |
| 4 | 60 | 6000 | 1 | $5,749,405.00 | Yes | 4 hours |

*Table 1. The following table shows the result of each instance.*

**3. Facility Design Problem**

**3.1 Problem Formulation:**

Quadratic Assignment is used to find the cost of transportation of units between the departments and minimize this cost. Our project is based on a multiple row facility design problem. The constraints are formulated in such a way that every department is allocated only one location. Below is a small example of how the formulation is done.

|  |  |  |
| --- | --- | --- |
|  | Unit Cost |  |
| 0 | 1 | 1 |
| 1 | 0 | 2 |
| 1 | 2 | 0 |

|  |  |  |
| --- | --- | --- |
|  | Flows |  |
| 0 | 5 | 4 |
| 5 | 0 | 4 |
| 4 | 4 | 0 |

**Decision Variables:**

Xij = 1 (Department i assigned to location j, i=1,2,3 & j=1,2,3)

0 (otherwise).

Flow ik = Flow of units from departure i to department k.

Cost jl = Unit cost of flow from location j to location l.

minimize cost:

**Minimize** (5\*1) X11 X22 + (5\*1) X11 X23 + (5\*1) X12 X21 + (5\*2) X12 X23 + (5\*1) X13 X22 + (5\*2) X13 X22 + (4\*1) X11 X32 + (4\*1) X11 X33 + (4\*1) X12 X31 + (4\*2) X12 X33 + (4\*1) X13 X31 + (4\*2) X13 X32 + (4\*1) X21 X32 + (4\*1) X21 X33 + (4\*1) X22 X31 + (4\*2) X22 X33 + (4\*1) X23 X31 + (4\*2) X23 X32

**Subject To:**

D1: X11 + X12 + X13 = 1

D2: X21 + X22 + X23 = 1

D3: X31 + X32 + X33 = 1

L1: X11 + X21 + X31 = 1

L2: X12 + X22 + X32 = 1

L3: X13 + X23 + X33 = 1

**Binary:**

X11, X21, X31, X12,X22, X32, X13, X23, X33.

Linearizing the Non-Linear Terms

Zijkl

Zijkl >= Xij + Xkl -1

Zijkl <= Xij

Zijkl <= Xkl

5 Z1122 + 5 Z1123 + 5 Z1221 + 10 Z1223 + 5 Z1322 + 5 Z1322 + 4 Z1132 + 4 Z1133 + 4 Z1231 + 8 Z1233 + 4 Z1331 + 8 Z1332 + 4 Z2132 + 4 Z2133 + 4 Z2231 + 8 Z2233 + 4 Z2331 + 8 Z2332.

**3.2 Program Design:**

Quadratic Assignment Problem for Facility Design with departments with same area.

Similar to network design for facility design to read the text file for the two inputs unit flow between departments and unit cost of transportation between location and converting the values into integer and putting them in a list and making inner list using for loop is obtained by the following code,

Eg.

Unit\_Cost = []

with open ('FacilityDesign\_SameArea\small\_example/unit\_cost.txt', 'r') as unit\_cost:

for line in unit\_cost:

inner\_list = [int(elt.strip()) for elt in line.split(',')]

Unit\_Cost.append(inner\_list)

Unit\_Cost

In the above code unit transportation cost between locations is stored as a matrix by converting the string into integers using for loop and forming an inner list of each rows. Similarly, flow between departments text file are is and converted into an integer.

In order to create a dictionary to assign unique names to the values obtained above in a list, a department list and location list is created. Assignment of unique names to each value in lists can be done by making dictionaries.

Eg.

Dic\_Flows = makeDict ([Departmentlist, Departmentlist], Flows,0)

In the above example a two-dimensional dictionary of flow of material between departments is made. Consider there are three departments D1, D2, D3 then the values from the flows list would have a unique name depending on the flow between departments.

Then the linear programming model is defined where the objective is to minimize the transportation cost between departments is mentioned.

In a similar fashion the LP variables x and y dictionary are made where the variables is 1,2,3,4-dimensional, category being binary or continuous and lower and upper bounds are mentioned and set.

As we see in the manual formulation there is a need to linearize the terms as they are in nonlinear form, we introduce a variable y that is four dimensional as mentioned below,

y = LpVariable.dicts("Y”, (Departmentlist, Locationlist, Departmentlist, Locationlist), lowBound = 0, cat ="Binary")

In the above-mentioned code y variables has 4- dimensions 2 department list and 2 location list.

After defining the decision variables, the objective function is formulated using lpSum and for loop function.

The constraints of assigning each department at any one location and each location having only one department is done. Other constraints of linearization of non-linear terms are carried out as done in manual formulation using for loops as lp solver cannot solve nonlinear terms.

Once the constraints are shown, then problem.solve() function is used to get the optimal solution. To write a Lp file “problem.writeLp function” and to check the status of the solution “LpStatus[Problem.status]” function is used.

**3.3 Computational Results:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Departments | Locations | Instance | Objective Function Value | Optimal | Run Time |
| 5 | 5 | 1 | $2274.0 | Yes | 10.73 secs |
| 5 | 5 | 2 | $2413.0 | Yes | 12.43secs |
| 5 | 5 | 3 | $2083.0 | Yes | 11.65 secs |