Kushal Kharel

April 11, 2022

1 Warehouse Location Optimization

Let N be a set of candidate warehouse locations

Let M be a set of customer locations

Let $d_{n,m}$ be the cost of delivering product from warehouse n to customer location m

Objective is to find the optimal warehouse location that minimizes delivery cost

Let y_n be the binary variable representing whether or not to build a warehouse, 1 if warehouse n is selected, 0 otherwise

Let $x_{n,m}$ indicate the fraction of demand for customer m that can be fulfilled by warehouse n

P-median problem which has special property that x-variables will converge to $\{0, 1\}$ even though they are not specified as binary variables

Formulation:

$$\min \sum_{n \in N} \sum_{m \in M} d_{n,m} x_{n,m}$$

minimize the total cost associated with delivering items

subject to constraints:

$$\sum_{ninN} x_{n,m} = 1, \forall m \in M$$

ensures that each customer's demand is fully met

$$x_{n,m} \leq y_n, \forall n \in N, m \in M$$

ensures that a warehouse can deliver product to customers only if that warehouse is selected to built

$$\sum_{n \in N} y_n \leq P$$

represents the number of warehouses that can be build should be less than or equal to the number P

$$0 \leq x \leq 1$$

represents x is a between 0 and 1

$$y \in \{0, 1\}$$

represents y is binary

Now, let's assume that P is 10

Customer Locations: M from excel file

Candidate warehouse locations: N from excel file

Cost: $d_{n,m}$ is also in excel file

[10]:
```
import pandas as pd
import sys
from pyomo.environ import *


df = pd.read_excel('Delivery_Cost.xlsx', header = 0, index_col = 0)


N = list(df.index.map(str))

M = list(df.columns.map(str))



d = {(r, c):df.at[r, c] for r in N for c in M}

P = 15

from pyomo.environ import *
from pyomo.opt import *
from pyomo.core import *
import cplex


def obj_rule(model):
    return sum(d[n,m]*model.x[n,m] for n in N for m in M)

def one_per_cust_rule(model,m):
    return sum(model.x[n,m] for n in N) == 1
```

```
def warehouse_active_rule(model, n , m):
    return model.x[n,m] <= model.y[n]

def num_warehouses_rule(model):
    return sum(model.y[n] for n in N) <= P

def SolveUsingPyomo():

    model = ConcreteModel(name  = "(Warehouse Location)")


    model.x = Var(N,M, bounds = (0,1))
    model.y = Var(N, within = Binary)

    model.obj = Objective(rule = obj_rule)


    model.one_per_cust = Constraint(M, rule = one_per_cust_rule)

    model.warehouse_active = Constraint(N,M,rule=warehouse_active_rule)

    model.num_warehouses = Constraint(rule = num_warehouses_rule)

    opt = SolverFactory("cplex")

    results = opt.solve(model, tee = True)

    print( "The objective value is: " + str(model.obj.expr()))


    model.y.pprint() #print the optimal warehouse locations

    for warehouselocation in N:
        if value(model.y[warehouselocation]) > 0.5:
            customers = [str(cl) for cl in M if value(model.
 ↪x[warehouselocation,cl] > 0.5)]
            print(str(warehouselocation) + ' serves customers: '␣
 ↪+str(customers))
        else:
            print(str(warehouselocation) + ": do not build")

SolveUsingPyomo()
```

```
Welcome to IBM(R) ILOG(R) CPLEX(R) Interactive Optimizer 20.1.0.0
  with Simplex, Mixed Integer & Barrier Optimizers
5725-A06 5725-A29 5724-Y48 5724-Y49 5724-Y54 5724-Y55 5655-Y21
```

```
Copyright IBM Corp. 1988, 2020.  All Rights Reserved.

Type 'help' for a list of available commands.
Type 'help' followed by a command name for more
information on commands.

CPLEX> Logfile 'cplex.log' closed.
Logfile 'C:\Users\kkhar\AppData\Local\Temp\tmp4ng399u3.cplex.log' open.
CPLEX> Problem 'C:\Users\kkhar\AppData\Local\Temp\tmp3vxsk9q5.pyomo.lp' read.
Read time = 0.00 sec. (0.25 ticks)
CPLEX> Problem name         :
C:\Users\kkhar\AppData\Local\Temp\tmp3vxsk9q5.pyomo.lp
Objective sense      : Minimize
Variables            :    2549  [Nneg: 1,  Box: 2499,  Binary: 49]
Objective nonzeros   :    2499
Linear constraints   :    2552  [Less: 2500,  Equal: 52]
  Nonzeros           :    7547
  RHS nonzeros       :      53

Variables            : Min LB: 0.000000         Max UB: 1.000000
Objective nonzeros   : Min   : 500.0000         Max   : 2998.000
Linear constraints   :
  Nonzeros           : Min   : 1.000000         Max   : 1.000000
  RHS nonzeros       : Min   : 1.000000         Max   : 15.00000
CPLEX> Version identifier: 20.1.0.0 | 2020-11-10 | 9bedb6d68
Found incumbent of value 88485.000000 after 0.00 sec. (0.21 ticks)
Tried aggregator 1 time.
MIP Presolve eliminated 1 rows and 1 columns.
Reduced MIP has 2551 rows, 2548 columns, and 7546 nonzeros.
Reduced MIP has 49 binaries, 0 generals, 0 SOSs, and 0 indicators.
Presolve time = 0.00 sec. (3.45 ticks)
Probing time = 0.00 sec. (0.32 ticks)
Tried aggregator 1 time.
Detecting symmetries…
Reduced MIP has 2551 rows, 2548 columns, and 7546 nonzeros.
Reduced MIP has 49 binaries, 0 generals, 0 SOSs, and 0 indicators.
Presolve time = 0.00 sec. (4.47 ticks)
Probing time = 0.00 sec. (0.32 ticks)
MIP emphasis: balance optimality and feasibility.
MIP search method: dynamic search.
Parallel mode: deterministic, using up to 8 threads.
Root relaxation solution time = 0.00 sec. (4.27 ticks)

        Nodes                                         Cuts/
   Node  Left     Objective  IInf  Best Integer    Best Bound    ItCnt     Gap

*     0+    0                        88485.0000        0.0000           100.00%
*     0+    0                        35008.0000        0.0000           100.00%
```

```
      0     0    30252.5000    12    35008.0000    30252.5000      120   13.58%
      0     0    30263.3333    14    35008.0000       Cuts: 6      132   13.55%
      0     0    30288.4643    15    35008.0000       Cuts: 3      148   13.48%
*     0+    0                          30532.0000    30288.4643            0.80%
*     0+    0                          30309.0000    30288.4643            0.07%
      0     0        cutoff             30309.0000    30309.0000      148    0.00%
Elapsed time = 0.06 sec. (63.35 ticks, tree = 0.01 MB, solutions = 4)

Lift and project cuts applied:  2
Gomory fractional cuts applied:  2

Root node processing (before b&c):
  Real time             =    0.06 sec. (63.52 ticks)
Parallel b&c, 8 threads:
  Real time             =    0.00 sec. (0.00 ticks)
  Sync time (average)   =    0.00 sec.
  Wait time (average)   =    0.00 sec.
                          ------------
Total (root+branch&cut) =    0.06 sec. (63.52 ticks)

Solution pool: 4 solutions saved.

MIP - Integer optimal solution:  Objective =  3.0309000000e+04
Solution time =    0.06 sec.  Iterations = 148  Nodes = 0
Deterministic time = 63.53 ticks  (1008.44 ticks/sec)

CPLEX> Incumbent solution written to file
'C:\Users\kkhar\AppData\Local\Temp\tmpl54_h0g5.cplex.sol'.
CPLEX> The objective value is: 30309.0
y : Size=49, Index=y_index
    Key            : Lower : Value : Upper : Fixed : Stale : Domain
            Albany :     0 :  -0.0 :     1 : False : False : Binary
         Annapolis :     0 :   1.0 :     1 : False : False : Binary
           Atlanta :     0 :   0.0 :     1 : False : False : Binary
           Augusta :     0 :  -0.0 :     1 : False : False : Binary
            Austin :     0 :   1.0 :     1 : False : False : Binary
       Baton Rouge :     0 :   1.0 :     1 : False : False : Binary
          Bismarck :     0 :   1.0 :     1 : False : False : Binary
             Boise :     0 :   1.0 :     1 : False : False : Binary
            Boston :     0 :   0.0 :     1 : False : False : Binary
       Carson City :     0 :   1.0 :     1 : False : False : Binary
        Charleston :     0 :   0.0 :     1 : False : False : Binary
          Cheyenne :     0 :   0.0 :     1 : False : False : Binary
          Columbia :     0 :   1.0 :     1 : False : False : Binary
          Columbus :     0 :   0.0 :     1 : False : False : Binary
           Concord :     0 :   1.0 :     1 : False : False : Binary
            Denver :     0 :  -0.0 :     1 : False : False : Binary
        Des Moines :     0 :   0.0 :     1 : False : False : Binary
```

```
            Dover :     0 :   0.0 :     1 : False : False : Binary
        Frankfort :     0 :   0.0 :     1 : False : False : Binary
       Harrisburg :     0 :   0.0 :     1 : False : False : Binary
         Hartford :     0 :   1.0 :     1 : False : False : Binary
           Helena :     0 :  -0.0 :     1 : False : False : Binary
         Honolulu :     0 :   1.0 :     1 : False : False : Binary
     Indianapolis :     0 :  -0.0 :     1 : False : False : Binary
          Jackson :     0 :  -0.0 :     1 : False : False : Binary
   Jefferson City :     0 :  -0.0 :     1 : False : False : Binary
          Juneau :     0 :  -0.0 :     1 : False : False : Binary
         Lansing :     0 :   0.0 :     1 : False : False : Binary
         Lincoln :     0 :   1.0 :     1 : False : False : Binary
      Little Rock :     0 :   0.0 :     1 : False : False : Binary
          Madison :     0 :  -0.0 :     1 : False : False : Binary
       Montpelier :     0 :  -0.0 :     1 : False : False : Binary
        Nashville :     0 :  -0.0 :     1 : False : False : Binary
    Oklahoma City :     0 :  -0.0 :     1 : False : False : Binary
         Olympia :     0 :  -0.0 :     1 : False : False : Binary
         Phoenix :     0 :  -0.0 :     1 : False : False : Binary
          Pierre :     0 :  -0.0 :     1 : False : False : Binary
       Providence :     0 :  -0.0 :     1 : False : False : Binary
          Raleigh :     0 :   1.0 :     1 : False : False : Binary
         Richmond :     0 :  -0.0 :     1 : False : False : Binary
       Sacramento :     0 :   1.0 :     1 : False : False : Binary
       Saint Paul :     0 :   1.0 :     1 : False : False : Binary
            Salem :     0 :   0.0 :     1 : False : False : Binary
   Salt Lake City :     0 :   0.0 :     1 : False : False : Binary
         Santa Fe :     0 :   1.0 :     1 : False : False : Binary
      Springfield :     0 :   0.0 :     1 : False : False : Binary
      Tallahassee :     0 :   0.0 :     1 : False : False : Binary
           Topeka :     0 :   0.0 :     1 : False : False : Binary
          Trenton :     0 :  -0.0 :     1 : False : False : Binary
Juneau: do not build
Phoenix: do not build
Little Rock: do not build
Sacramento serves customers: ['Alaska', 'Idaho', 'New Hampshire', 'New Jersey',
'Oregon']
Denver: do not build
Hartford serves customers: ['Georgia', 'Nebraska']
Dover: do not build
Tallahassee: do not build
Atlanta: do not build
Honolulu serves customers: ['Hawaii', 'West Virginia']
Boise serves customers: ['Arkansas', 'Maryland']
Springfield: do not build
Indianapolis: do not build
Des Moines: do not build
Topeka: do not build
```

```
Frankfort: do not build
Baton Rouge serves customers: ['Kansas', 'Virginia', 'Wisconsin']
Augusta: do not build
Annapolis serves customers: ['Arizona', 'Nevada', 'New York', 'North Carolina',
'Wyoming']
Boston: do not build
Lansing: do not build
Saint Paul serves customers: ['Montgomery', 'Florida', 'Indiana', 'New Mexico',
'Tennessee']
Jackson: do not build
Jefferson City: do not build
Helena: do not build
Lincoln serves customers: ['Alabama', 'Iowa', 'Massachusetts', 'Vermont']
Carson City serves customers: ['Colorado', 'Kentucky', 'Washington']
Concord serves customers: ['Missouri', 'Montana', 'Ohio', 'Oklahoma',
'Pennsylvania']
Trenton: do not build
Santa Fe serves customers: ['California', 'Delaware', 'Minnesota', 'Utah']
Albany: do not build
Raleigh serves customers: ['Rhode Island']
Bismarck serves customers: ['Connecticut', 'Illinois', 'Maine', 'Mississippi']
Columbus: do not build
Oklahoma City: do not build
Salem: do not build
Harrisburg: do not build
Providence: do not build
Columbia serves customers: ['North Dakota', 'South Dakota', 'Texas']
Pierre: do not build
Nashville: do not build
Austin serves customers: ['Louisiana', 'Michigan', 'South Carolina']
Salt Lake City: do not build
Montpelier: do not build
Richmond: do not build
Olympia: do not build
Charleston: do not build
Madison: do not build
Cheyenne: do not build
```