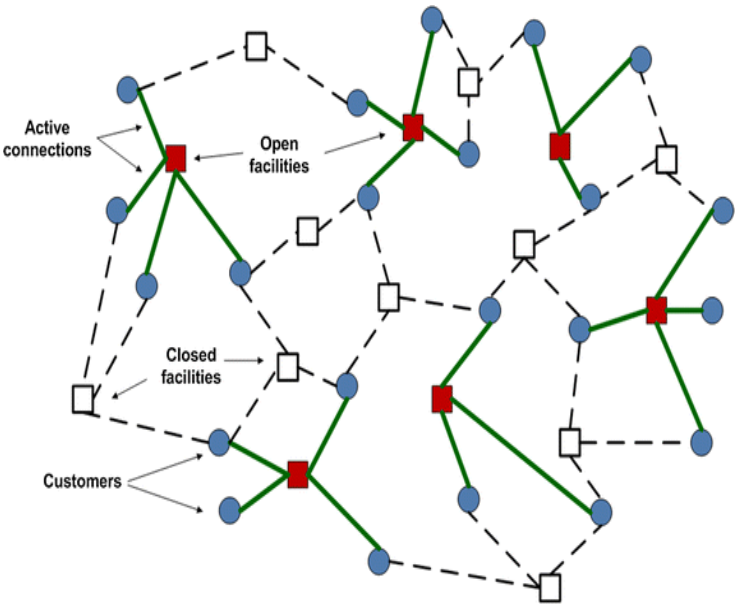
Uncapacitated Facility Location Problem

Problem Overview - The goal of this problem is to choose a subset of facilities(plants, warehouses. etc) to open from a given set of facilities to minimize the sum of transportation costs while meeting the demands of a client and the fixed costs of setting up the facilities. The key point to note is that **each facility has enough capacity to meet the demands of all clients.**



Notations used -

- n = Number of candidate locations to open facility.
- m = Number of clients
- f_i = Fixed cost of opening a facility at location i.
- d_j = Demand of client j.
- D = Sum of demands from all clients.
- g_{ij} = Per unit transportation cost from facility i to client j

Decision variables -

- x_i = Binary variable, 1 if facility i is open, 0 if closed
- z_{ij} = Quantity shipped from facility i to client j.

Integer Program Problem formulation -

Minimize - **Fixed costs + Transportation costs**

Objective function - $\text{Minimize} \sum\sum f_i x_i + \sum\sum g_{ij} z_{ij}$

Constraints -

$$\sum_i z_{ij} = d_j \quad \forall j$$
$$\sum_j z_{ij} <= Dx_i \quad \forall i$$
$$z_{ij} >= 0 \quad \forall (i,j)$$
$$x_i = 0.1 \quad \forall i$$

1) Implementation using CPLEX C++ API -

Data Structures used -

IloNumArray - 1D Numerical Array (f_i)
IloNumVarArray - Array for 1D decision variables(x_i)
IloArray<IloNumArray> - 2D Numerical Array. (g_{ij})
IloArray<IloNumVarArray> - Array for 2D decision variables (z_{ij})

Methods used -
.solve() - solves a given model
.getObjValue() - returns objective value of model.
.getValues(val, input) - returns optimized value of the input argument to val.

Algorithm -

- Construct a CPLEX environment using class IloEnv.
  IloEnv env;
  At the end of the code destroy the env object using
  env.end();
- Create a model using class IloModel.
  IloModel model(env);
- Define decision variables using class IloNumVarArray.
  IloNumVarArray(env, lower bound, upper bound, datatype)
- Define objective function and add it to the model.
  IloObjective obj = IloMinimize(env, objective function)
  model.add(obj)
- Define constraints and add them to the model.
  IloExpr constraint(env);
  model.add(constraint)
- Create an IloCplex object to solve the model.
  IloCplex cplex(model)

- Solve the model.
  cplex.solve();
- Query the results using
  cplex.getObjValue(); and cplex,getValues(locations, x)

Data Files description -

Format of the input data files used is -

number of potential facility locations(n), number of clients(m)
for each potential facility location (i = 1, 2 …..n)
capacity (ignored) fixed cost
for each client (j = 1,2, ….m)
demand
cost of allocating all of demand to facility i (i = 1,2,....n)

Output files format -

for each client (j = 1,2,...m)
jth client connected to which facility optimal_solution

Datasets collected from -
http://people.brunel.ac.uk/~mastjjb/jeb/orlib/uncapinfo.html