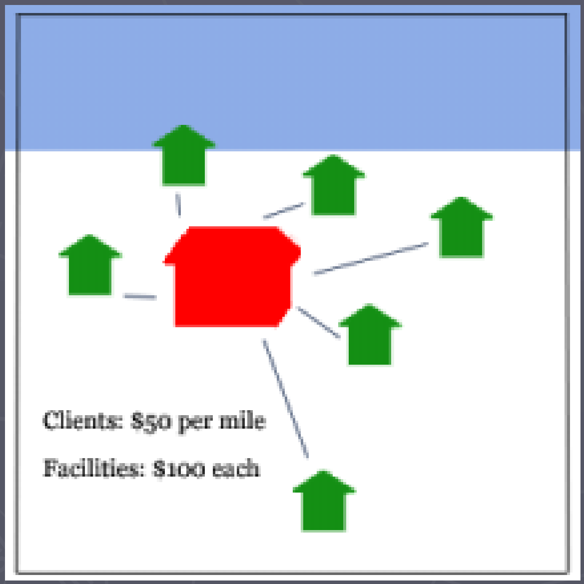
Facility Location



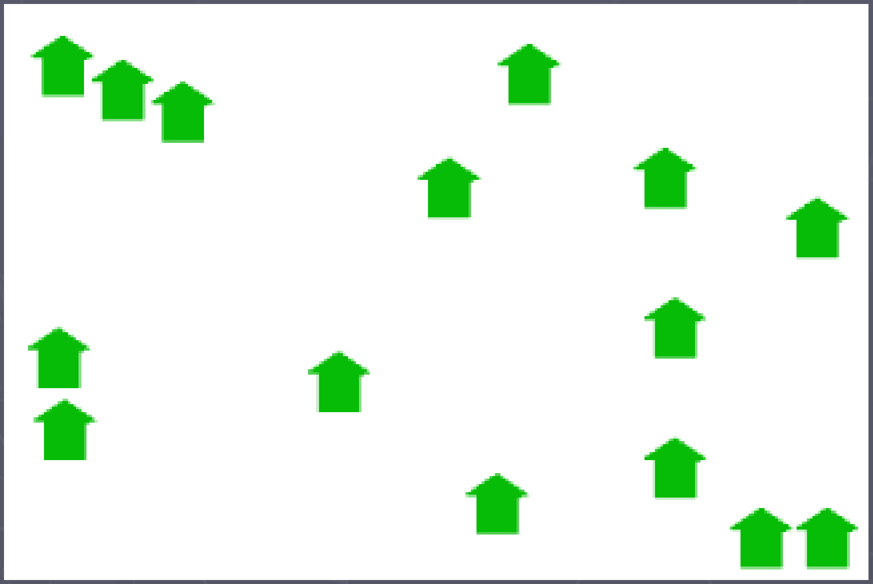
Lindsey Bleimes
Charlie Garrod
Adam Meyerson

The K-Median Problem

- Input: We're given a weighted, strongly connected graph, each vertex as a client having some demand
  - Demand is generally distance – it is a weight on the edges of the graph
- We can place facilities at any k vertices within our graph, which can then serve all the other clients

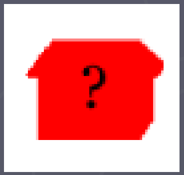
- At which vertices do we place our k facilities, in order to minimize total cost?

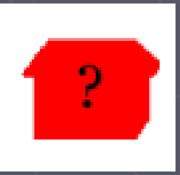
The K-Median Problem



Our 'Graph'

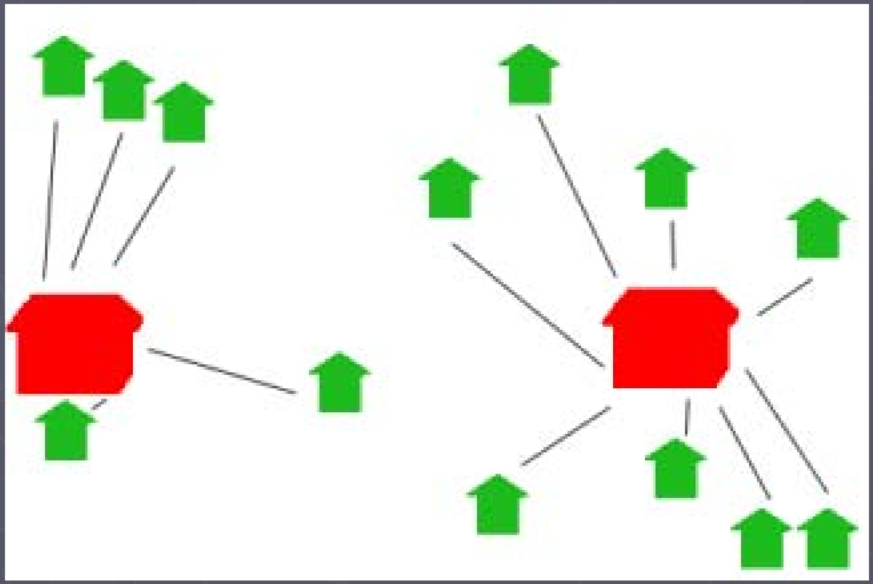
If we had 2 facilities to place, which vertices become Facilities?





We want to minimize average distance of each client to its closest facility

The K-Median Problem



How do we know which locations are really optimal, without testing every combination of k locations?

The K-Median Problem

- We want the facilities to be as efficient as possible, thus we want to minimize the distance from each client to its closest facility.
- There can be a cost associated with creating each facility that also must be minimized
  - otherwise if we were not limited to k facilities, all points could be facilities

Variations – Classic Facility Location

- We may not have a set number of facilities to place
- In that case, the cost of opening a facility is included in the total cost calculation which must be minimized

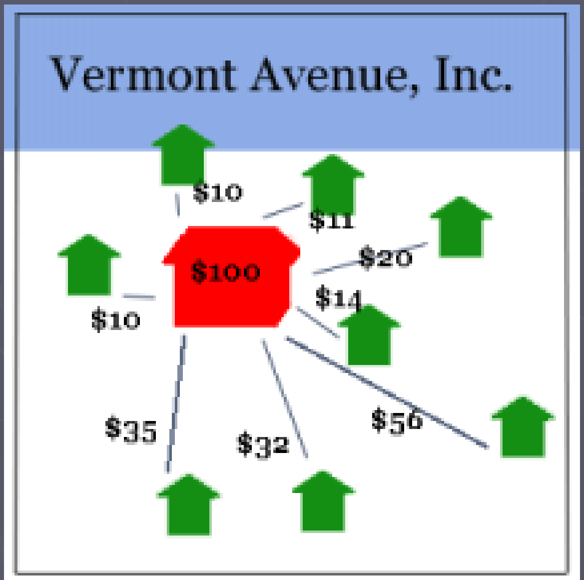
- Now the question is, how many facilities to we create, and where do we put them?

Variations – Online Facility Location

- We start with some graph and its solution, but we will have to add more vertices in the future, without disturbing our current setup
- The demands of incoming clients are based on some known function, generally of distance

- Our question: what do we do with each incoming point as it arrives?

Applications - Operations



- Stores and Warehouses
  - Where do we build our warehouses so that they are close to our stores?
  - And how many should we build to attain efficiency?

- Here, accuracy far outweighs speed

Applications - Clustering

- Databases
  - Data mining with huge datasets
  - Here, speed outweighs accuracy, to a point
- Finding Data patterns
  - 'Distances' measured either in space or in content
- Web Search clustering
- Medical Research
- And many other clustering problems

Limitations

- The problem of finding the best possible solution is NP-Hard
- It has been proved that the best upper-bound attainable is about the square root of 2 times the optimal solution cost – the best upper bound so far attained is around 1.5

← 50% extra cost – not so good when talking about millions of dollars, not so bad when talking about data clustering

Is It Really That Bad?

- Well ... on the average case, probably not.
- But that’s something we’re trying to find out

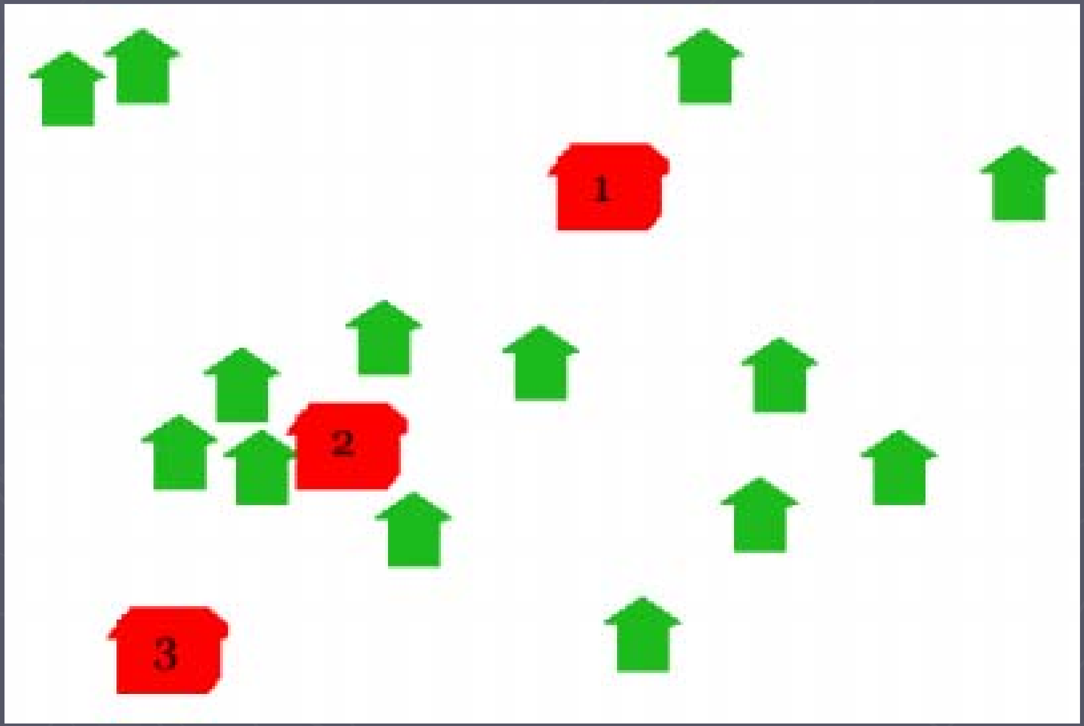
- Are the average-case solutions good enough for companies to use?
- Are online models fast enough and at least somewhat accurate for db/clustering applications?

Solution Techniques

- Local Search Heuristics for k-median and Facility Location Problems
  - V. Arya et al.
- Improved Approximation Algorithms for Metric Facility Location Problems
  - M. Mahdian, Y. Ye, J. Zhang
- Online Facility Location
  - A. Meyerson

Local Search / K-Median

Where do we place our k facilities?



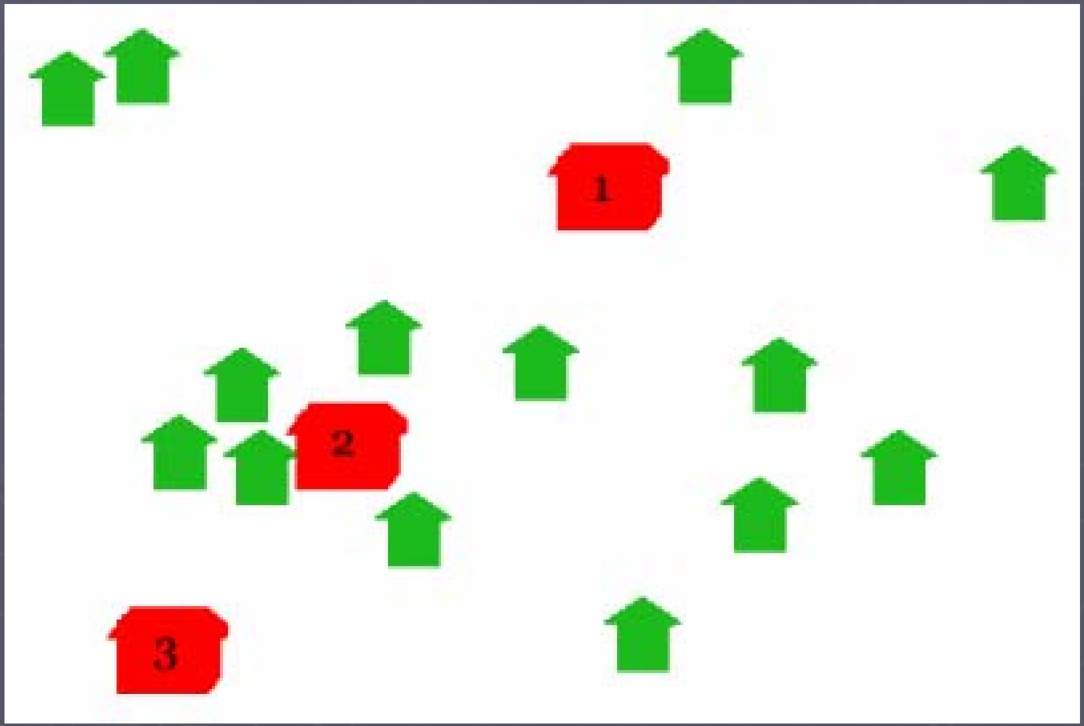
The Algorithm:

Choose some initial K points to be facilities, and calculate your cost

Initial points can be chosen by first choosing a random point, then successively choosing the point farthest from the current group of facilities until you have your initial K

Local Search / K-Median

Where do we place our k facilities?

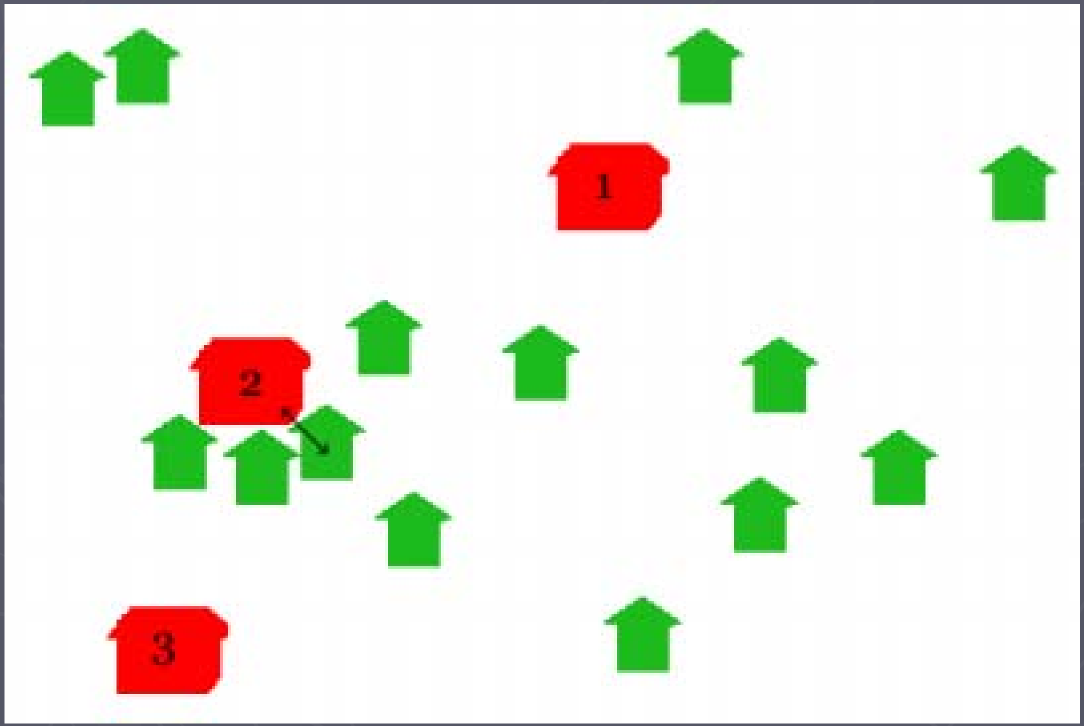


Now we swap

While there exists a swap between a current facility location and another vertex which improves our current cost, execute the swap

Local Search / K-Median

Where do we place our k facilities?

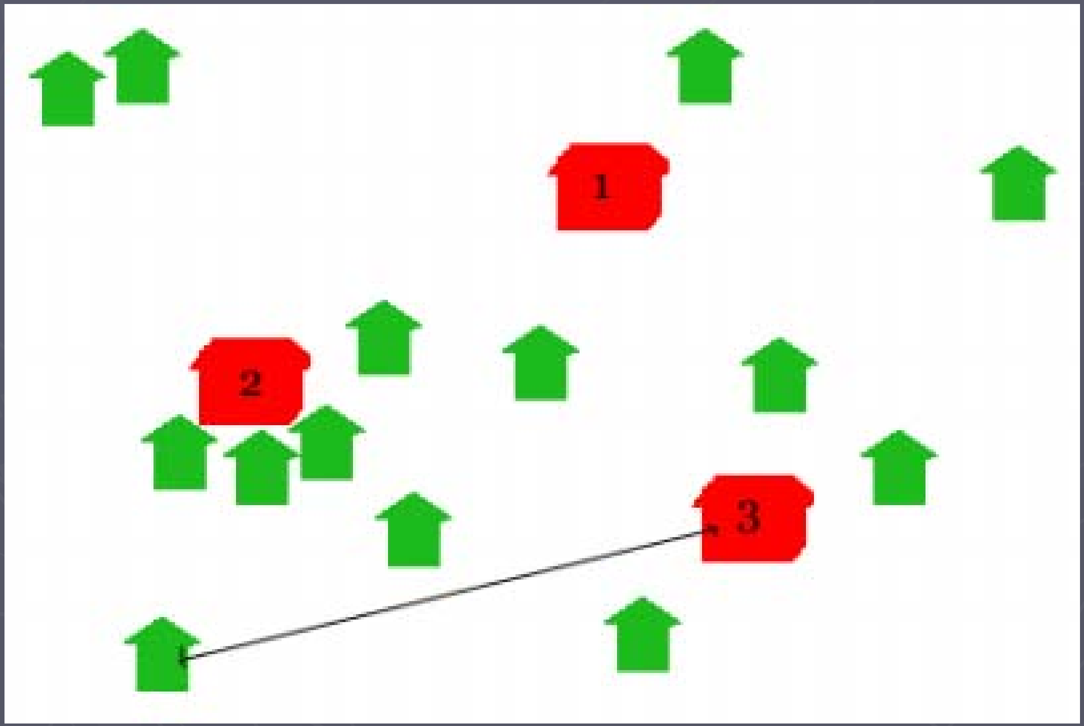


Now we swap

While there exists a swap between a current facility location and another point which improves our current cost, execute the swap

Local Search / K-Median

Where do we place our k facilities?



Now we swap

While there exists a swap between a current facility location and another point which improves our current cost, execute the swap

Etc.

Local Search / K-Median

- It is possible to do multiple swaps at one time
- In the worst case, this solution will produce a total cost of (3 + 2/p) times the optimal cost, where p is the number of swaps that can be done at one time

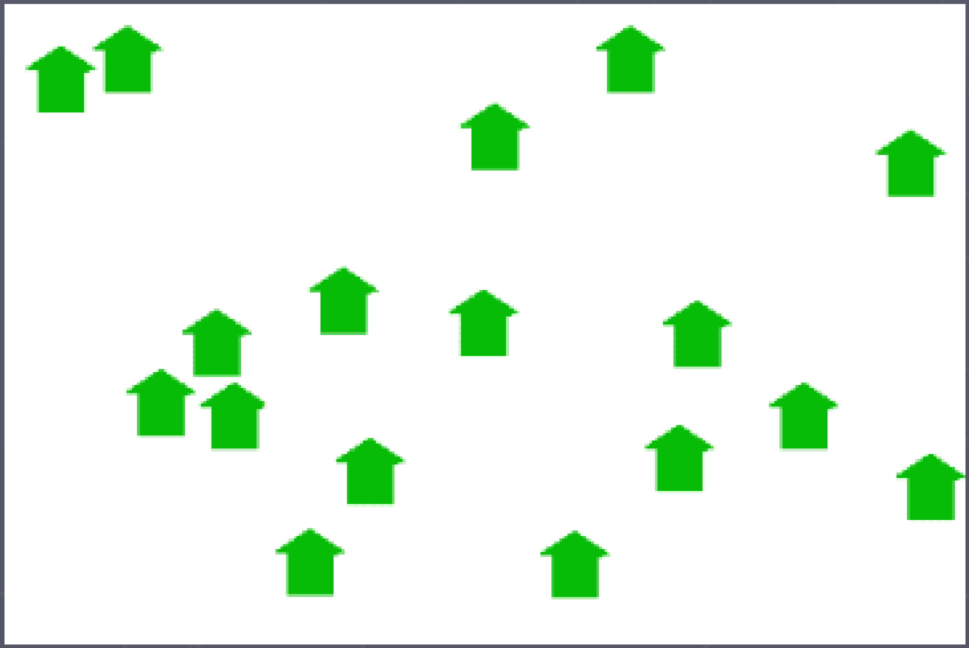
Facility Location

How many facilities do we need, and where?

The Algorithm:

Begin with all clients unconnected

All clients have a budget, initially zero

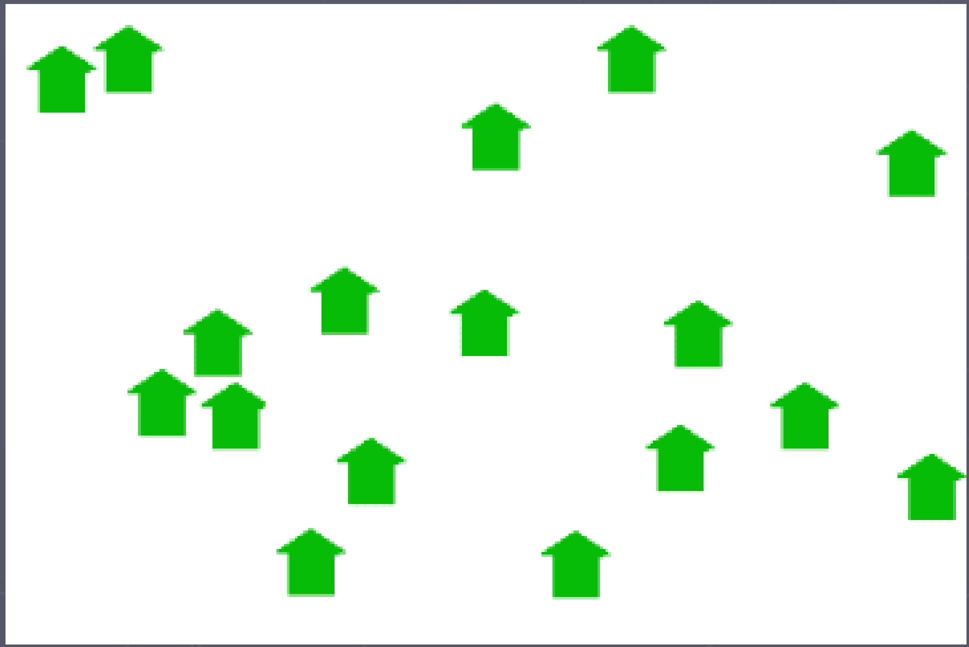


Facility Location

How many facilities do we need, and where?

Clients constantly offer some of their budget to open a new facility

This offer is:
max(budget-dist, 0) if unconnected, or
max(dist, dist') if connected
Where dist = distance to possible new facility,
and dist' = distance to current facility

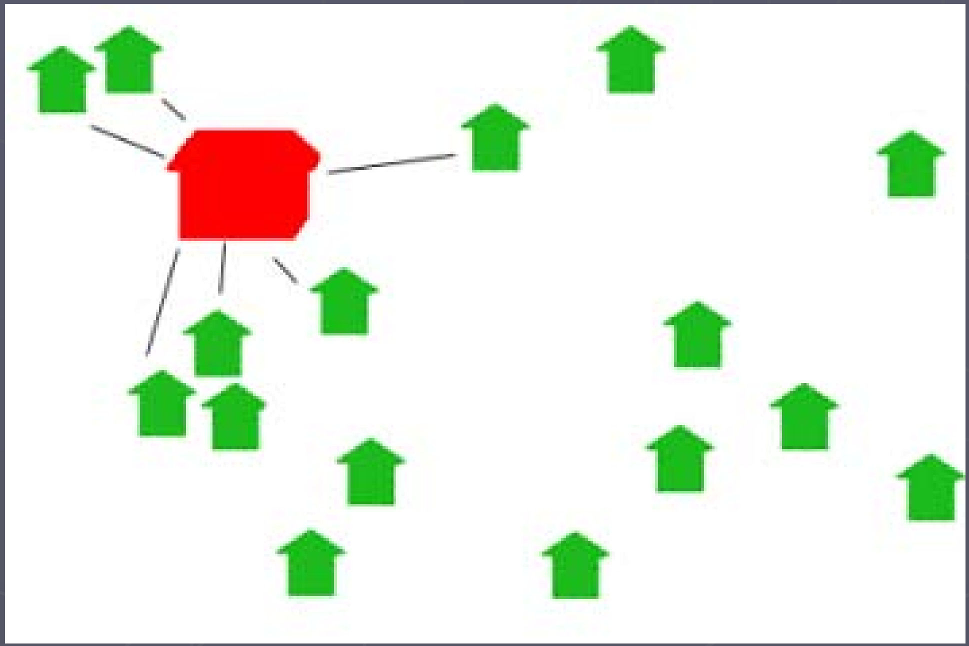


Facility Location

How many facilities do we need, and where?

- While there is an unconnected client, we keep increasing the budgets of each unconnected client at the same rate

- Eventually the offer to some new facility will equal the cost of opening it, and all clients with an offer to that point will be connected

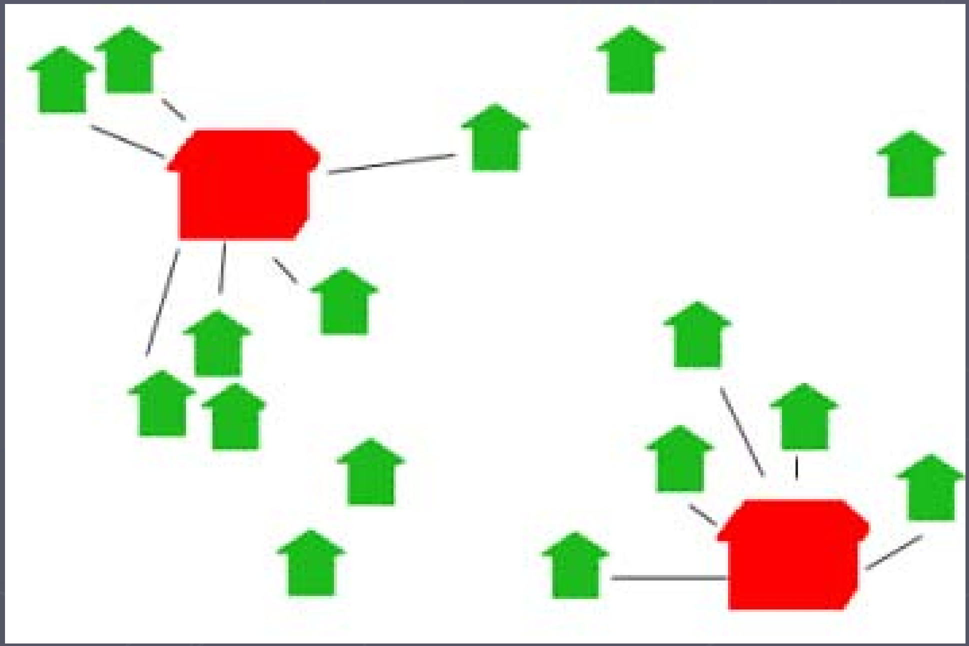


Facility Location

How many facilities do we need, and where?

While there is an unconnected client, we keep increasing the budgets of each unconnected client at the same rate

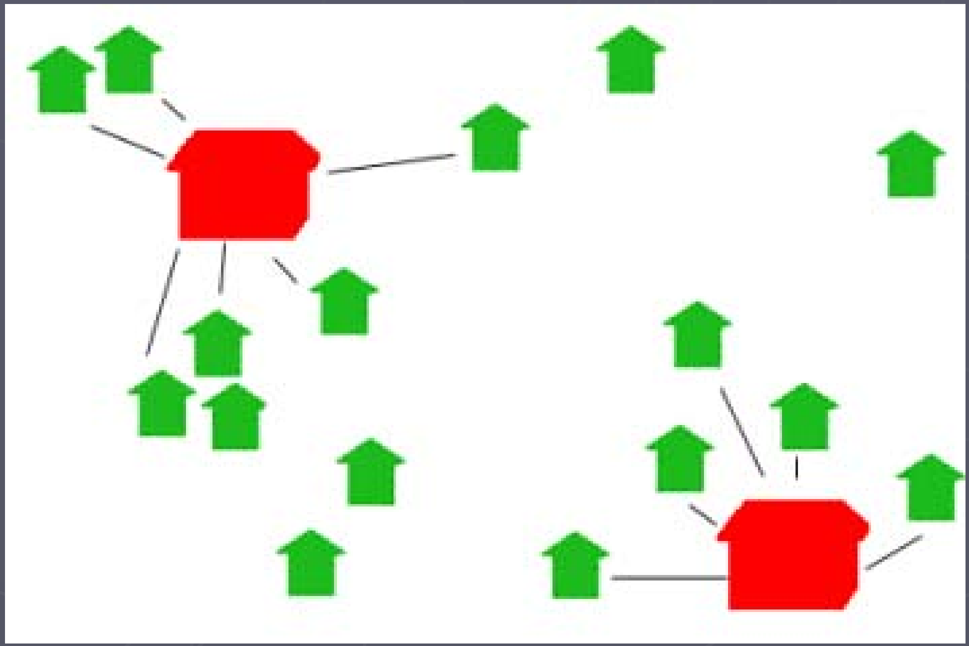
Eventually the offer to some new facility will equal the cost of opening it, and all clients with an offer to that point will be connected



Facility Location

How many facilities do we need, and where?

- Or, the increased budget of some unconnected client will eventually outweigh the distance to some already-opened facility, and can simply be connected then and there

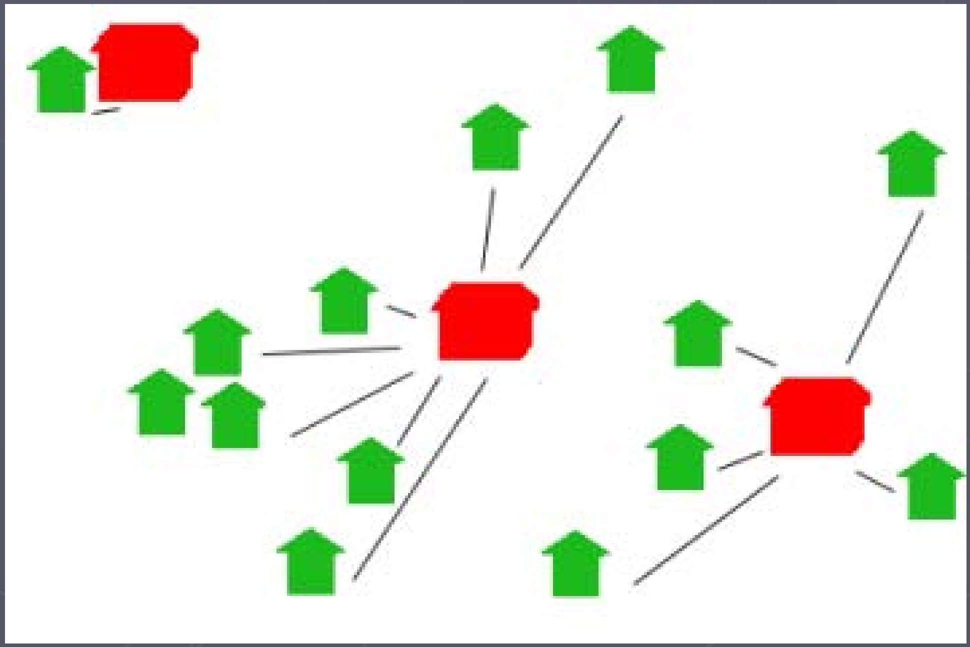


Facility Location – Phase 2

How many facilities do we need, and where?

- Now that everyone is connected, we scale back the cost of opening facilities at a uniform rate

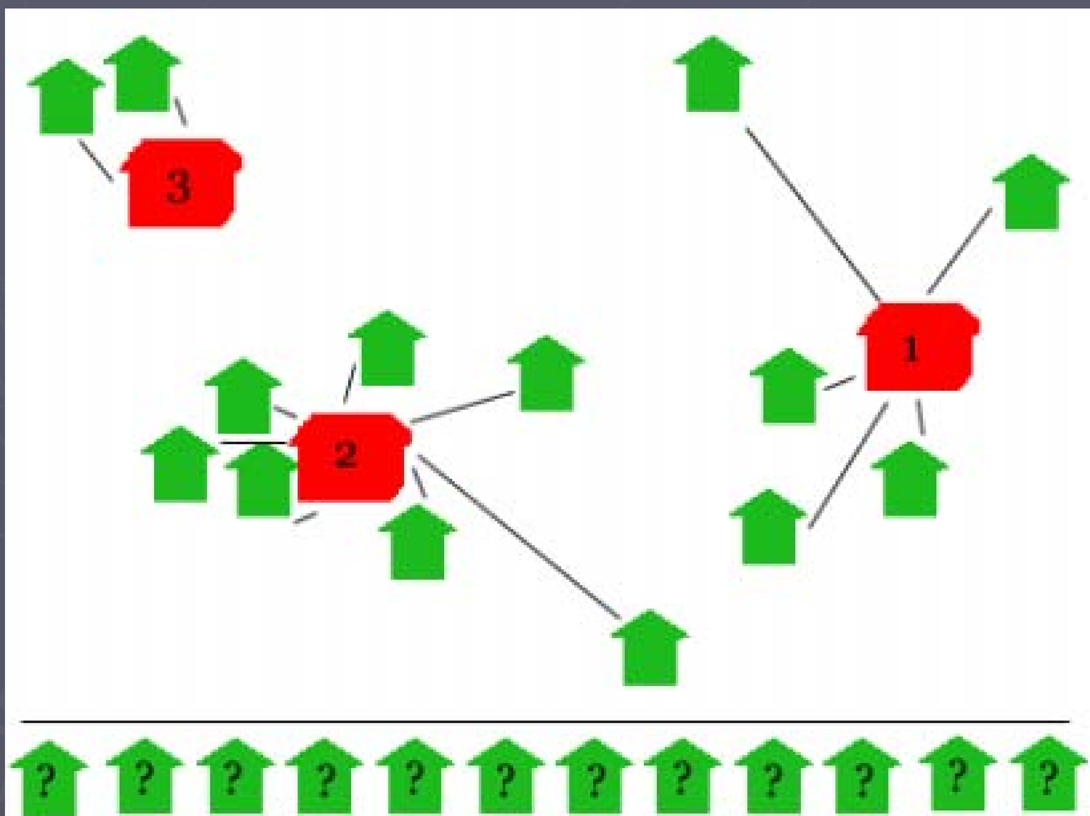
- If at any point it becomes cost-saving to open a new facility, we do so and re-connect all clients to their closest facility



Worst case, this solution is 1.52 times the optimal cost solution

Online Facility Location

What do we do with incoming vertices?

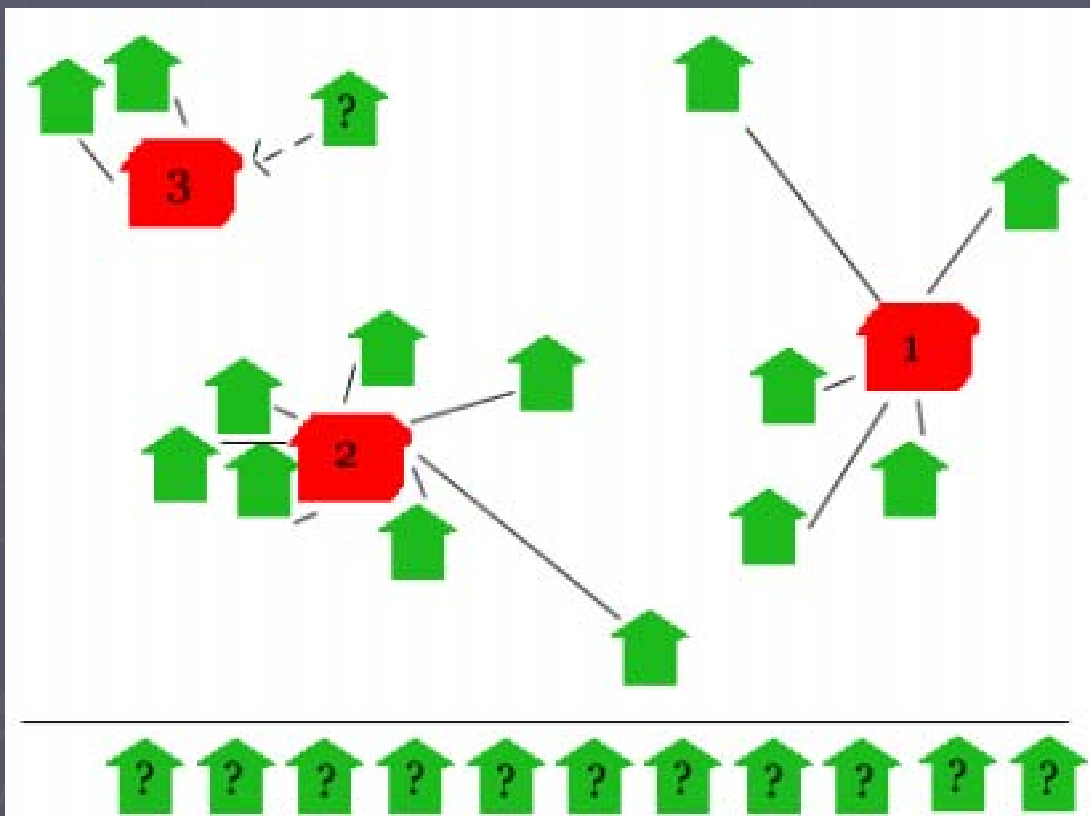


Here we start with an initial graph, but more clients will need to be added in the future, without wrecking our current scheme

As new clients arrive, we must evaluate their positions and determine whether or not to add a new facility

Online Facility Location

What do we do with incoming vertices?

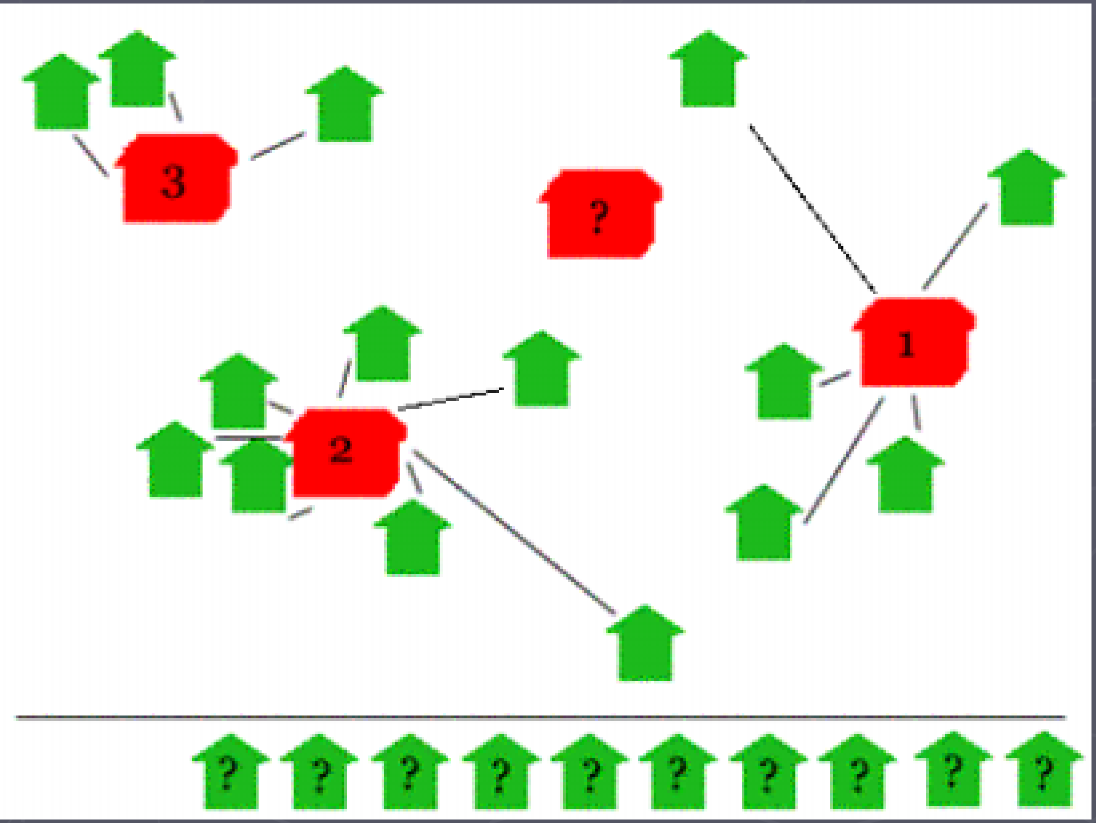


With each new client, we do one of two things:

(1) Connect our new client to an existing facility

Online Facility Location

What do we do with incoming vertices?



With each new client, we do one of two things:

(2) Connect our new client to an existing facility, or

(3) Make a new facility at the new point location

Online Facility Location

- The probability that a Facility is created out of a given incoming point is d/f
  - Where d = the distance to the nearest facility
  - And f = the cost of opening a facility
- Worst case cost is expected 8 times the optimal cost

Our Goal

- We’re not trying to solve the problem again
- Rather we’d like to know more about the realistic behavior of techniques we already have

- i.e. how often do we really see results at the upper/lower bounds of accuracy?
- How far off are streaming data models?

Our Goal

- We are trying to run simulations over both real and random data sets, to get average data on the performance of known algorithms for this problem

- Both speed and accuracy are important, but for different reasons and applications
- Realistic data will help determine how best to use these algorithms

Questions?