Location Problems

Basic Concepts of Location

Last IP modeling tricks

Dispel Misconception

Introduce NLP

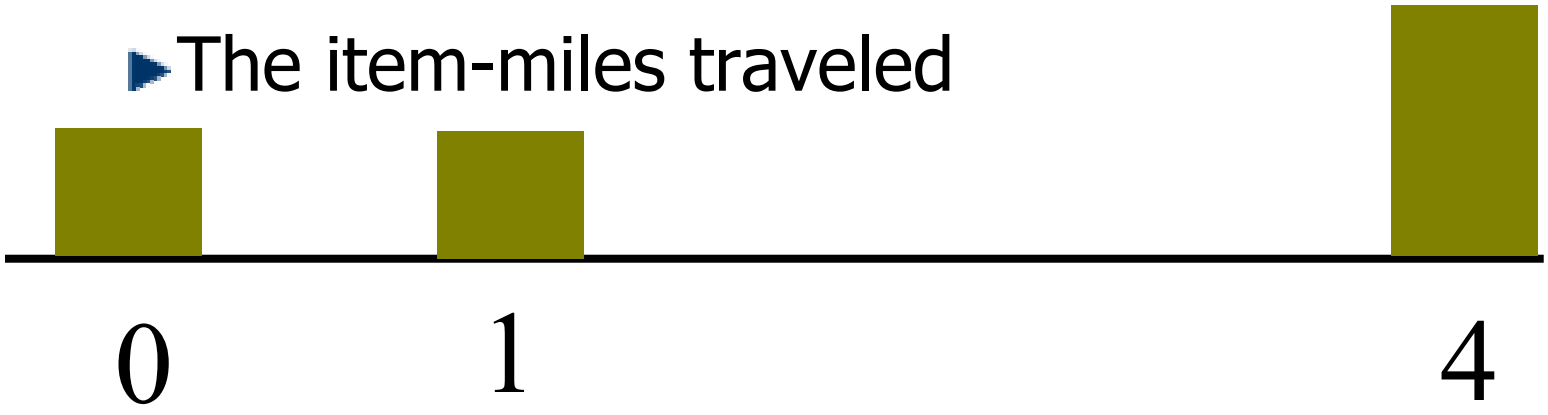
First Heuristic

Where to Locate Facilities

- Rectilinear Location Problems
- Euclidean Location Problems
- Location - Allocation Problems

Basic Intuition

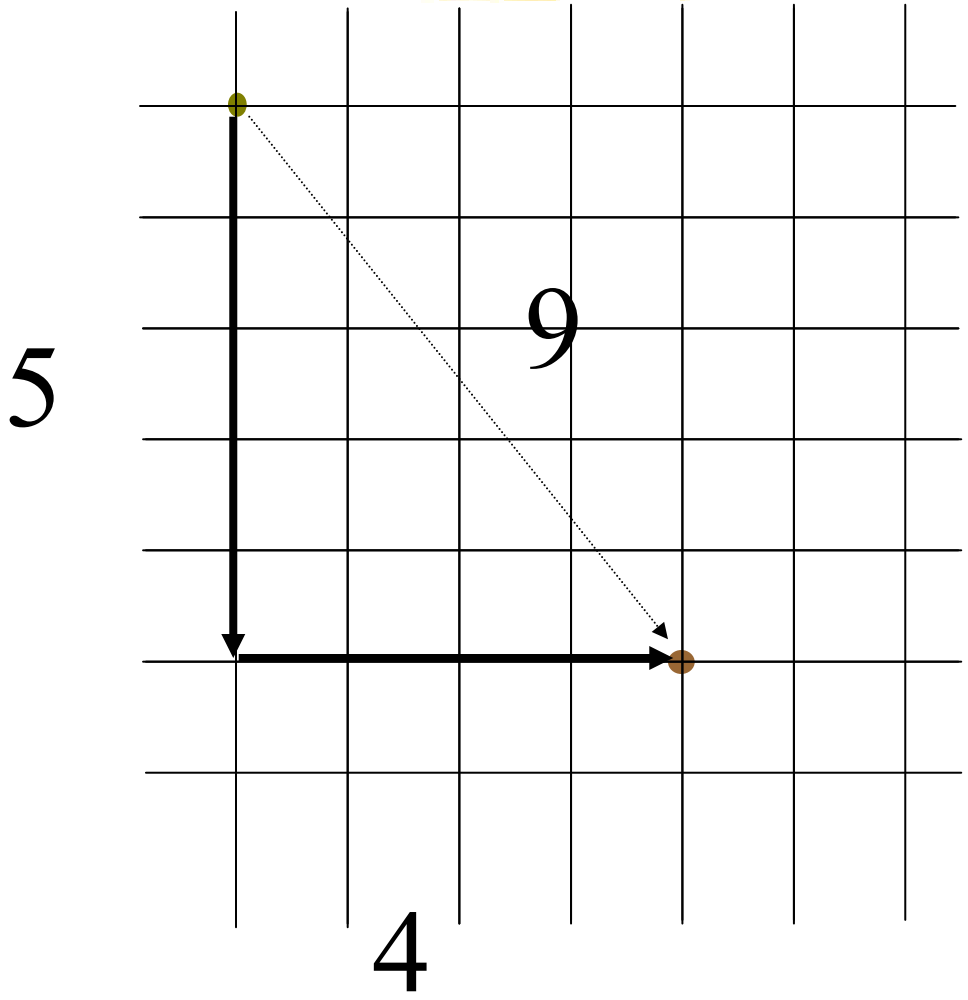
- On the line, if the objective is to min …
  - The maximum distance traveled
  - The maximum distance left + right
  - The distance traveled
    - there and back to each customer
  - The item-miles traveled



Rectilinear Distance

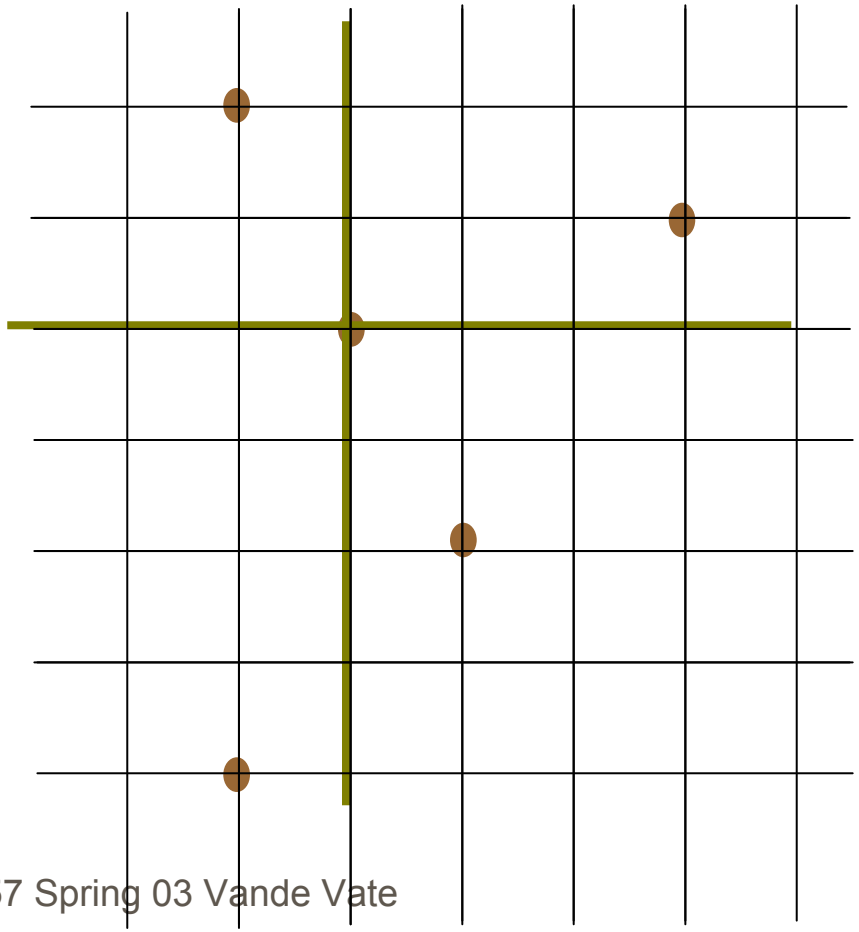
- Travel on the streets and avenues
- Distance =
  - number of blocks East-West +
  - number of blocks North-South
- Manhattan Metric
-

Rectilinear Distance



Locate a facility…

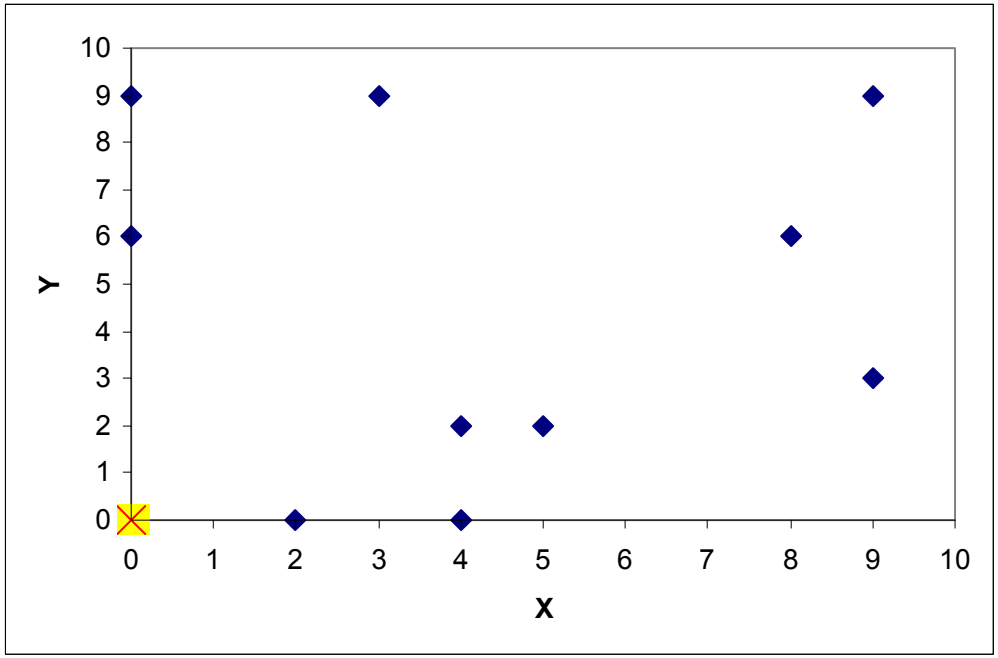
- To minimize the sum of rectilinear distances
- Intuition
- Where?
- Why?



Solver Model

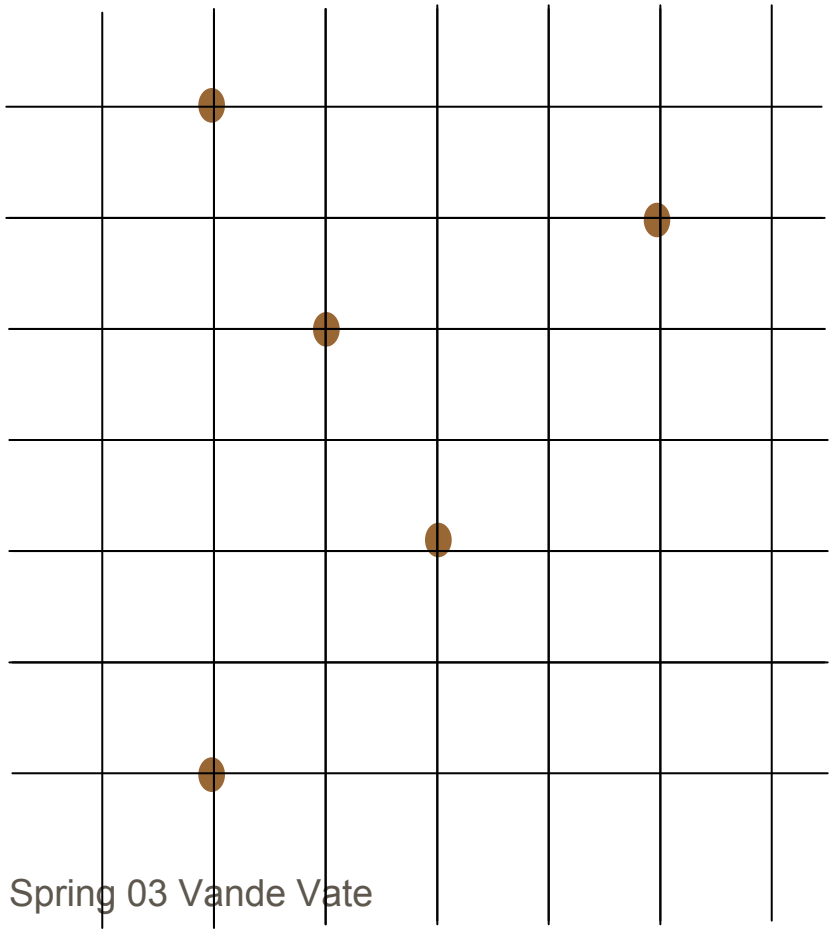
Minimize the sum of Rectilinear distances to customers

Cust.	X	Y	Xdist	Ydist	If Left	If Right	If Above	If Below
1	9	9			9.00	(9.00)	(9.00)	9.00
2	9	3			9.00	(9.00)	(3.00)	3.00
3	8	6			8.00	(8.00)	(6.00)	6.00
4	3	9			3.00	(3.00)	(9.00)	9.00
5	0	6			-	-	(6.00)	6.00
6	4	2			4.00	(4.00)	(2.00)	2.00
7	4	0			4.00	(4.00)	-	-
8	2	0			2.00	(2.00)	-	-
9	0	9			-	-	(9.00)	9.00
10	5	2			5.00	(5.00)	(2.00)	2.00
Facility	0	0						
Total		0	-	-				



Locate a facility…

- To minimize the max of rectilinear distances
- Intuition
- Where?
- Why?



Models

- Set Customers;
- Param X{Customer};
- Param Y{Customer};
- var Xloc >= 0;
- var Yloc >= 0;
- var Xdist{Customer};
- var Ydist{Customer};

Constraints

- DefineXdist1{c in Customer}:

  Xdist[c] >= X[c]-Xloc;
- DefineXdist2{c in Customer}:

  Xdist[c] >= Xloc-X[c];
- DefineYdist1{c in Customer}:

  Ydist[c] >= Y[c]-Yloc;
- DefineYdist2{c in Customer}:

  Ydist[c] >= Yloc-Y[c];

Objective

- Total Distance:
  - sum{c in Customer}(Xdist[c]+Ydist[c]);
- Maximum Distance?

Min the Max!

- Var Xloc;
- var Yloc;
- var Xdist{Customer}>= 0;
- var Ydist{Customer}>= 0;
- var dmax;
- min objective: dmax;
- s.t. DefineMaxDist{c in Custs}:

  dmax >= Xdist[c] + Ydist[c];

Min the Max Cont'd

- DefineXdist1{c in Customer}:

  Xdist[c] >= X[c]-Xloc;
- DefineXdist2{c in Customer}:

  Xdist[c] >= Xloc-X[c];
- DefineYdist1{c in Customer}:

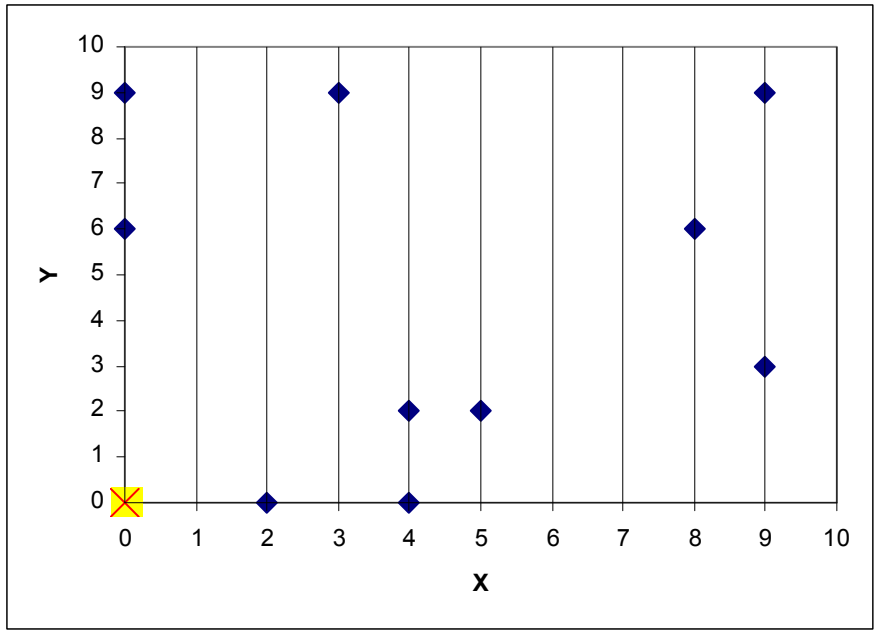
  Ydist[c] >= Y[c]-Yloc;
- DefineYdist2{c in Customer}:

  Ydist[c] >= Yloc-Y[c];

Solver Model

Mimimize the Maximum Rectilinear Distance

Customer	X	Y	Max	Total	Xdist	Ydist	If Left	If Right	If Above	If Below
1	9	9	0	0	0	0	9	-9	-9	9
2	9	3	0	0	0	0	9	-9	-3	3
3	8	6	0	0	0	0	8	-8	-6	6
4	3	9	0	0	0	0	3	-3	-9	9
5	0	6	0	0	0	0	0	0	-6	6
6	4	2	0	0	0	0	4	-4	-2	2
7	4	0	0	0	0	0	4	-4	0	0
8	2	0	0	0	0	0	2	-2	0	0
9	0	9	0	0	0	0	0	0	-9	9
10	5	2	0	0	0	0	5	-5	-2	2
Facility	0	0								
Total	0									



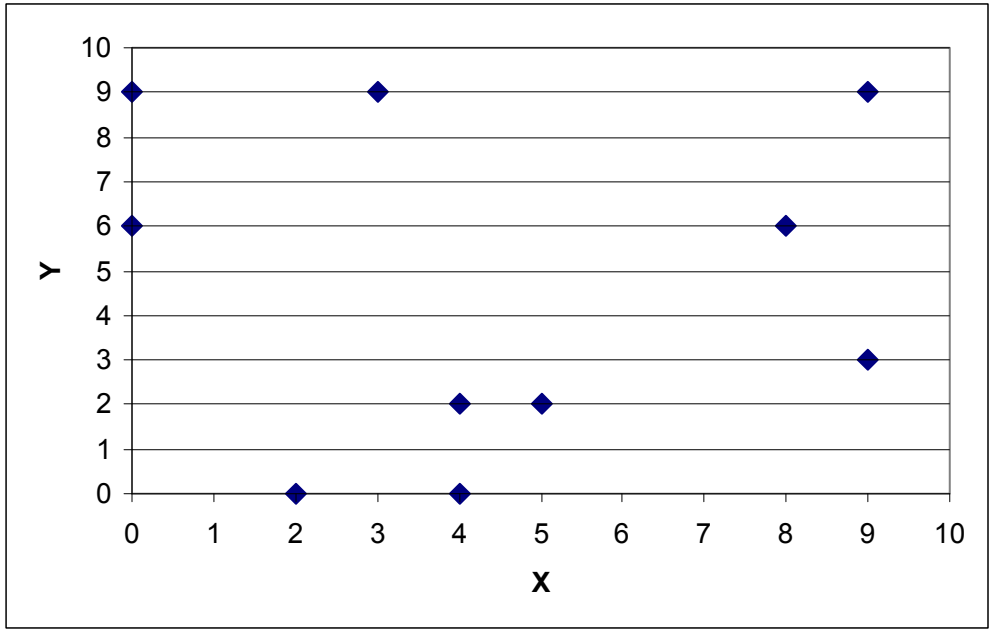
Challenge #3

- NIMBY….
- Maximize the Minimum Distance
- How did you do it?

NIMBY

Mimimize the Maximum Rectilinear Distance

Cust.	X	Y	Min Total	Xdist	Ydist	Left?	Above?	If Left	If Right	If Above	If Below
1	9	9	0	0				29	-9	11	9
2	9	3	0	0				29	-9	17	3
3	8	6	0	0				28	-8	14	6
4	3	9	0	0				23	-3	11	9
5	0	6	0	0				20	0	14	6
6	4	2	0	0				24	-4	18	2
7	4	0	0	0				24	-4	20	0
8	2	0	0	0				22	-2	20	0
9	0	9	0	0				20	0	11	9
10	5	2	0	0				25	-5	18	2
Facility Total											



Disjunctive Constraints

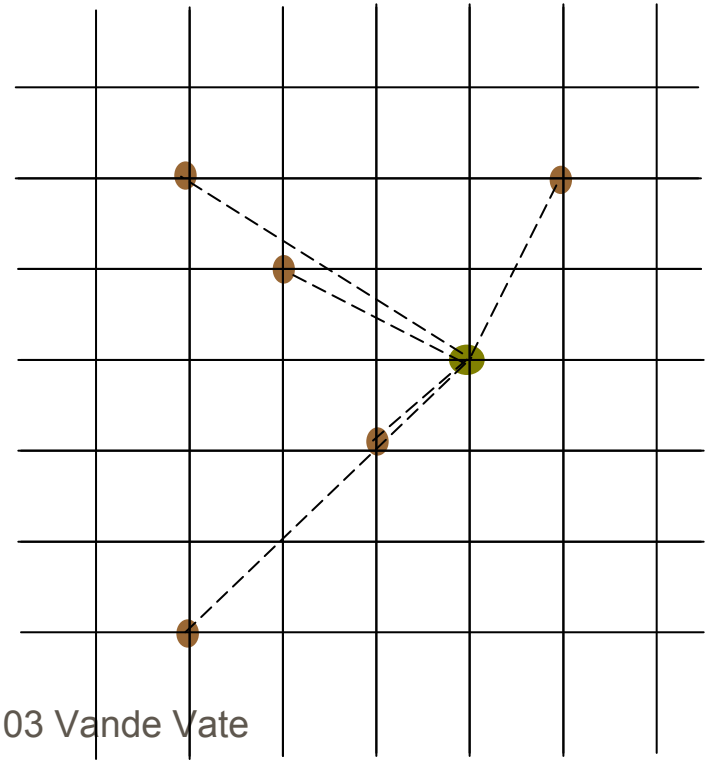
- One of Two Constraints holds
  - Ydist = TownY – DumpY
  - Ydist = DumpY – TownY
- Binary Variable Chooses 1
  - Ydist $\leq$ TownY – DumpY + 20*Left
  - Ydist $\leq$ DumpY – TownY + 20*(1-Left)
- Only 1 matters (20 is big)
- Objective drives the right choice
- These are hard for the solver

Outline

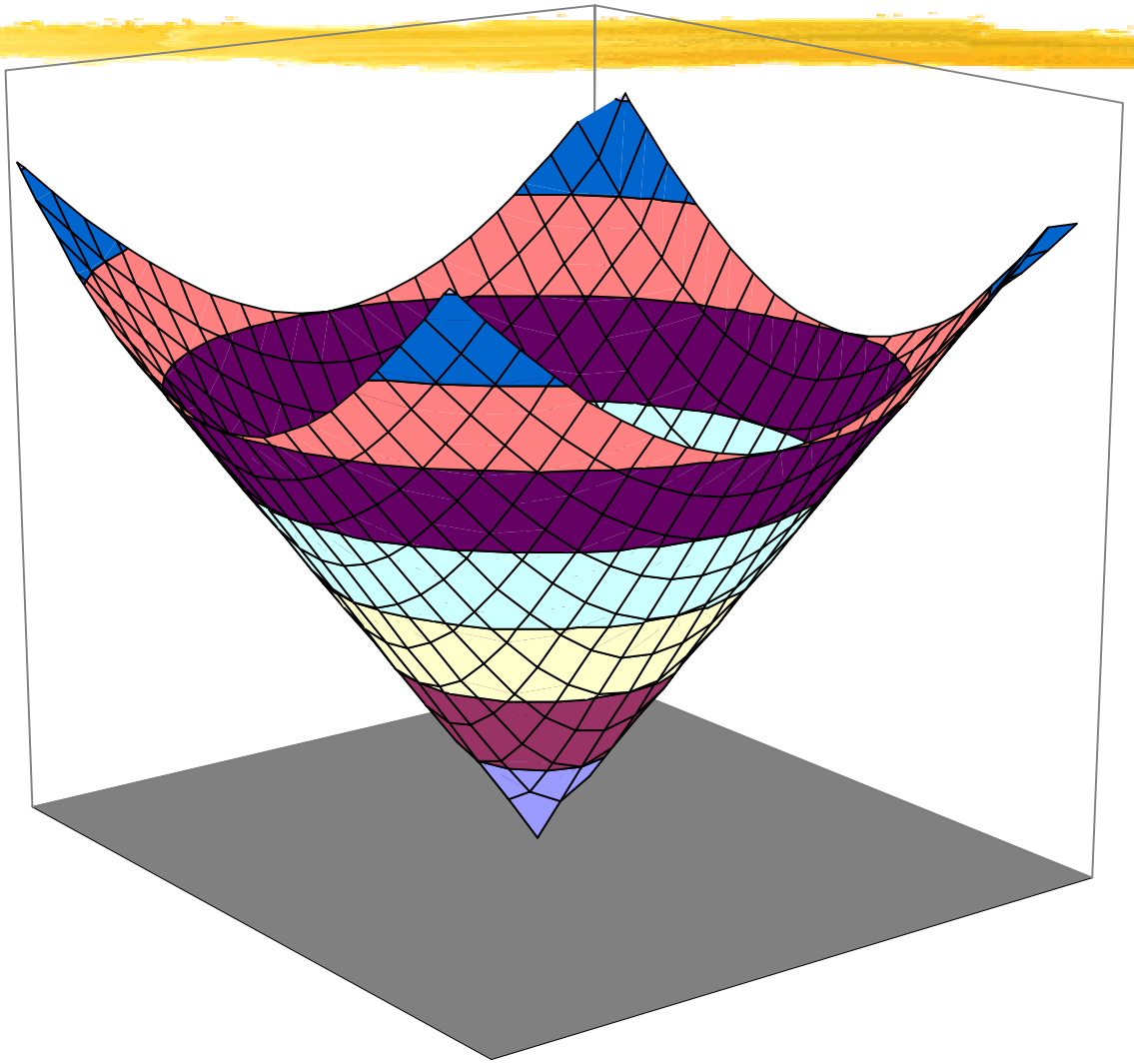
- Rectilinear Location Problems
- Euclidean Location Problems
- Location - Allocation Problems

Locating a single facility

- Distance is not linear
- Distance is a convex function
- Local Minimum is a global Minimum



What kind of function?



Where to Put the Facility

- Total Cost = $\Sigma\, c_k d_k(x,y)$
- $= \Sigma\, c_k \sqrt{(x_k - x)^2 + (y_k - y)^2}$
- ∂Total Cost/$\partial x = \Sigma\, c_k\, (x_k - x)/d_k(x,y)$
- ∂Total Cost/$\partial x = 0$ when
- $x = [\Sigma c_k x_k/d_k(x,y)]/[\Sigma c_k/d_k(x,y)]$
- $y = [\Sigma c_k y_k/d_k(x,y)]/[\Sigma c_k/d_k(x,y)]$
- But $d_k(x,y)$ changes with location...

Iterative Strategy

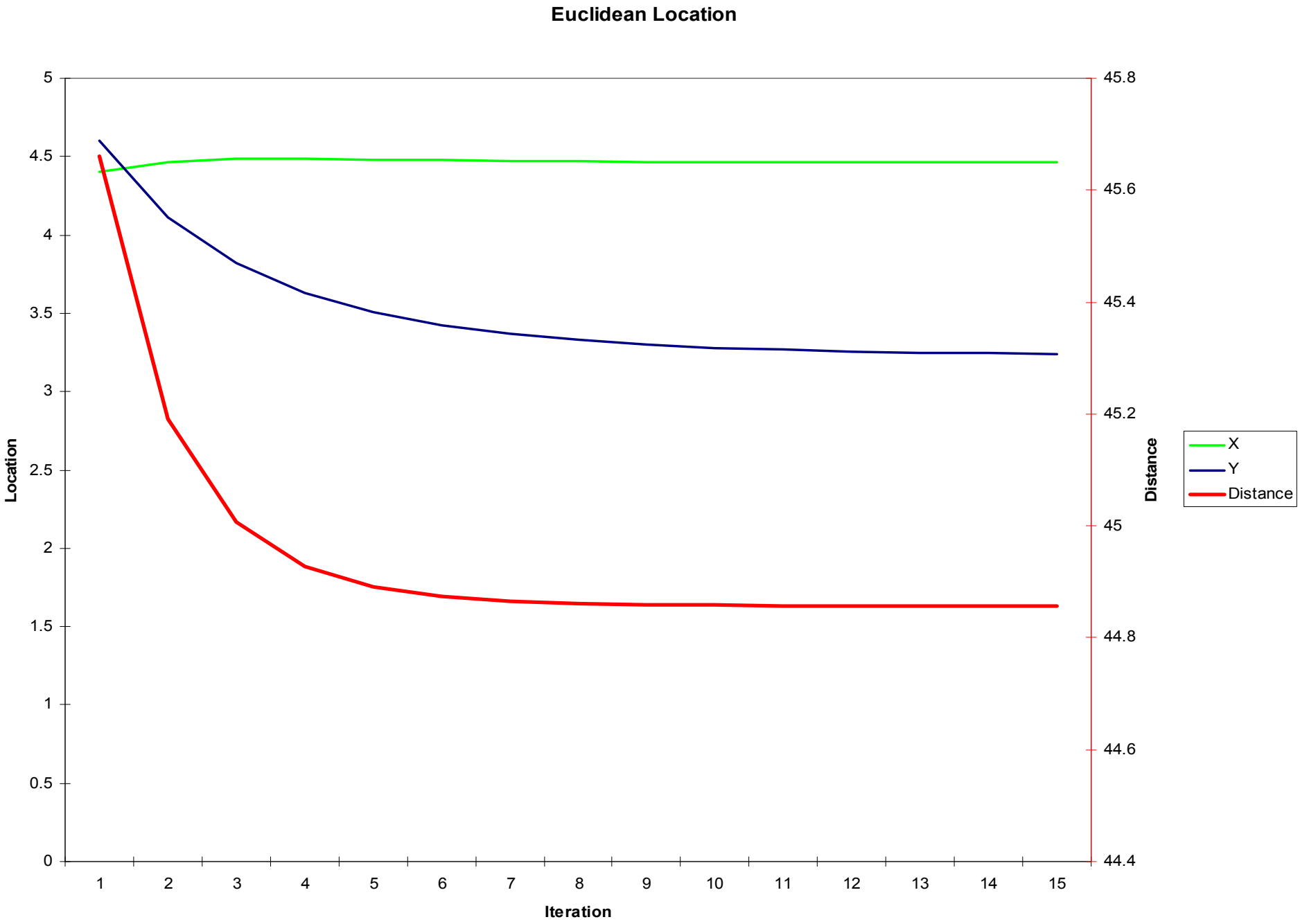
- Start somewhere, e.g.,
  - $x = [\Sigma c_k x_k]/[\Sigma c_k]$
  - $y = [\Sigma c_k y_k]/[\Sigma c_k]$
  - as though $d_k = 1$.
- Step 1: Calculate values of d_k
- Step 2: Refine values of x and y
  - $x = [\Sigma c_k x_k/d_k]/[\Sigma c_k/d_k]$
  - $y = [\Sigma c_k y_k/d_k]/[\Sigma c_k/d_k]$
- Repeat Steps 1 and 2. ...

Solver Model: Not Linear

Facility			
	4.5	4.6	50.88679
	4.283971	4.0907	50.35772
	4.241999	3.73981	50.11444
	4.239085	3.486694	49.9801
	4.239961	3.298874	49.90163
	4.238616	3.15693	49.85438
	4.235342	3.048107	49.82525
	4.23113	2.963646	49.80693
	4.226679	2.897385	49.79522
	4.222382	2.844914	49.78763
	4.21843	2.80302	49.78265
	4.214899	2.769336	49.77935
	4.211803	2.742087	49.77714
	4.209121	2.719928	49.77565
	4.206817	2.701827	49.77464

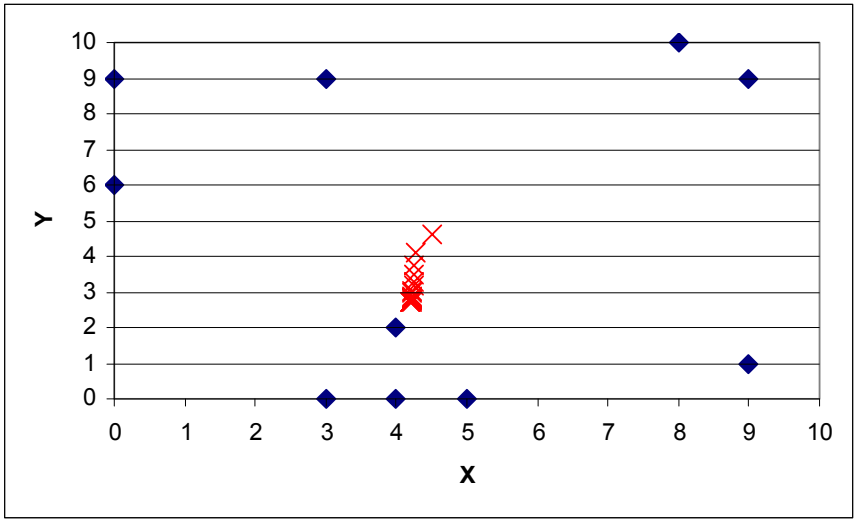
Center of Gravity

Better Answer


Euclidean Location
Location
Distance
Iteration
X
Y
Distance
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
44.4
44.6
44.8
45
45.2
45.4
45.6
45.8
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Euclidean Location

Customer	X	Y	d(x,y)	1/d(x,y)	d(x,y)	1/d(x,y)	d(x,y)	1/d(x,y)	d(x,y)	1/d(x,y)	d(x,y)	1/d(x,y)
1	9	9	6.293648	0.15889	6.807507	0.146897	7.092825	0.140988	7.284426	0.137279	7.427032	0.134643
2	9	1	5.762812	0.173526	5.638559	0.17735	5.490459	0.182134	5.371216	0.186178	5.286094	0.189176
3	8	10	6.43506	0.155399	6.980594	0.143254	7.301544	0.136957	7.521146	0.132958	7.683943	0.130142
4	3	9	4.648656	0.215116	5.074427	0.197067	5.404827	0.18502	5.65083	0.176965	5.83441	0.171397
5	0	6	4.712749	0.21219	4.690185	0.213211	4.806559	0.208049	4.928139	0.202916	5.027261	0.198915
6	4	2	2.64764	0.377695	2.109897	0.473957	1.75656	0.569294	1.505796	0.664101	1.320854	0.757086
7	4	0	4.627094	0.216118	4.100544	0.24387	3.747632	0.266835	3.494881	0.286133	3.30759	0.302335
8	3	0	4.838388	0.20668	4.287471	0.233238	3.940653	0.253765	3.70032	0.270247	3.524213	0.283751
9	0	9	6.293648	0.15889	6.515646	0.153477	6.757525	0.147983	6.954595	0.14379	7.104936	0.140747
10	5	0	4.627094	0.216118	4.152893	0.240796	3.815855	0.262064	3.568757	0.28021	3.385296	0.295395
Facility	4.5	4.6	50.88679	2.090624		2.223116		2.35309		2.480777		2.603587
	4.283971	4.0907	50.35772									
	4.241999	3.73981	50.11444									
	4.239085	3.486694	49.9801									
	4.239961	3.298874	49.90163									
	4.238616	3.15693	49.85438									
	4.235342	3.048107	49.82525									
	4.23113	2.963646	49.80693									
	4.226679	2.897385	49.79522									
	4.222382	2.844914	49.78763									
	4.21843	2.80302	49.78265									
	4.214899	2.769336	49.77935									
	4.211803	2.742087	49.77714									
	4.209121	2.719928	49.77565									
	4.206817	2.701827	49.77464									
	4.217658	2.805268										



Moving On

- Example of Convex Minimization (Non-linear)
- Example of a Heuristic: Not guaranteed to give us the best answer, but works well.

Locating Several Facilities

- Fixed Number of Facilities to Consider
- Single Sourcing
- Two Questions:
  - Location: Where
  - Allocation: Whom to serve
- Each is simple
- Together they are “harder”

Iterative Approach

- Put the facilities somewhere
- Step 1: Assign the Customers to the Facilities
- Step 2: Find the best location for each facility given the assignments (see previous method)
- Repeat Step 1 and Step 2 ….

Assign Customers to Facilities

- Uncapacitated (facilities can be any size)
  - "Greedy": Assign each customer to closest facility
- **Capacitated**
  - Use Optimization: Single-sourcing