

# Primal-Dual Algorithms for Network Design

Stefano Leonardi

Sapienza Università di Roma

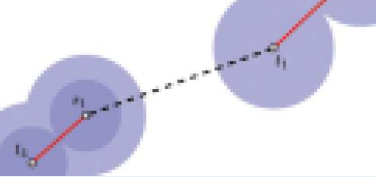
Spring 2010 – UPC, Barcelona



# Talk Outline

Part I The Facility Location problem

Part II Metric Facility Location



# The Facility Location problem



# Non Metric Facility location

## Input:

- undirected graph  $G = (V, E)$
- non-negative edge costs  $c : E \rightarrow \mathbb{R}^+$
- set of **facilities**  $F \subseteq V$
- facility  $i$  has facility opening cost  $f_i$
- set of **demand points**  $D \subseteq V$
- $c_{ij}$ : cost of connecting demand point  $j$  to facility  $i$ .

Distances between demand points and facilities do not form a metric. I.e., no triangle inequality:  $c_{ij} \leq c_{ik} + c_{kj}$

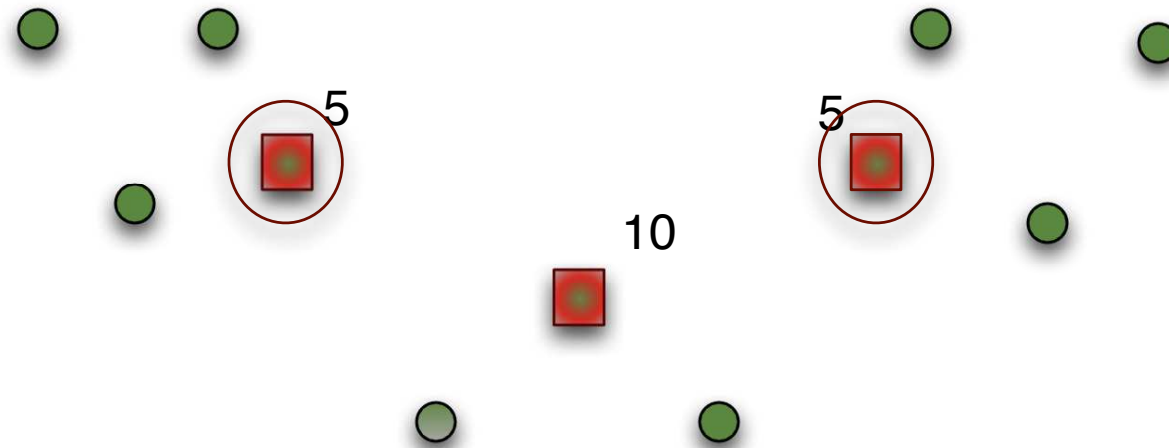
# Non Metric Facility location

**Goal:** Compute

- set  $F' \subseteq F$  of opened facilities; and
- function  $\phi : \mathcal{D} \rightarrow \mathcal{F}'$  assigning demand points to opened facilities that minimizes

$$\sum_{i \in F'} f_i + \sum_{j \in \mathcal{D}} c_{\phi(j)j}$$

# Example



Assume distance 3 from the two bottom vertices to the facilities of cost 5 and from the left and right vertices to the facility of cost 10

The two facilities of cost 5 are opened

Every demand is connected to the closest facility of cost 5.



# Non Metric Facility Location

- The facility location problem is a covering problem
- It also generalizes the Set Cover problem.
- Let  $c(S)$  be the cost of set  $S = \{e_1, \dots, e_k\}$ .
  - ◆ Associate a facility  $f_S$  to each set  $S$  with opening cost  $c(S)$
  - ◆ Associate a demand point to every element of the universe
  - ◆ The distance to facility  $f_S$  from each element  $e_j \in S$  is 0
  - ◆ The distance to facility  $f_S$  from each element  $e_j \notin S$  is  $\infty$
- Observe that the instance is non metric

# Approximation on Non-metric Facility location

- The non metric facility location problem can be approximated with a set cover-like greedy algorithm
- Consider for each facility  $i \in F$  all possible subsets  $S$  of demand points  $D$
- For facility  $i$  and subset  $S$  associate a cost  $f_i + \sum_{v \in S} c(v, i)$
- The reduction is non-polynomial but we can restrict consideration to a polynomial number of sets
- The algorithm will only consider for each facility  $i$  subsets of yet not covered vertices of  $D$  at non-decreasing distance from  $i$
- At each step at most  $|F| \times |D|$  subsets are considered

# Approximation on Non-metric Facility Location

---

## Algorithm 1: Algorithm for non-metric facility location.

---

**Data:**  $D, F, c: D \times F \rightarrow \mathbb{R}_{\geq 0}, f: F \rightarrow \mathbb{R}_{\geq 0}$ .

**while**  $D \neq \emptyset$  **do**

**let**  $i \in F$  and  $S \subseteq D$  **minimize**  $\frac{f_i + \sum_{v \in S} c(v, i)}{|S \cap D|}$ ;  
     $D \leftarrow D \setminus S$ ;

---

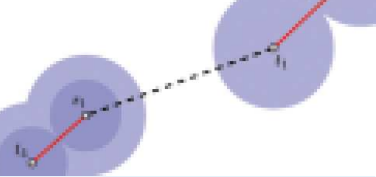
- Every step of the algorithm takes polynomial time since the most cost-effective facility is found between  $|D| \times |F|$  different sets.
- Let  $S_i$  be the demand set that is covered at the  $i$ th iteration of the algorithm,  $i = 1, \dots, k$ .
- Let  $|C_i|$  be the number of uncovered demands before set  $S_i$  is selected.
- Denote by  $c(S_i) = f_i + \sum_{v \in S_i} c(v, i)$  be the cost of the algorithm at the  $i$ th iteration.

# Approximation on Non-metric Facility Location

**Theorem:** The Greedy algorithm for Non-metric Facility Location is  $O(\log n)$  approximate, with  $n = |D|$ .

- The optimal solution will cover the demand set  $C_i$  at cost  $\frac{c(OPT)}{|C_i|}$  per demand. Therefore there exists a set in the optimal solution of cost effectiveness lower than  $\frac{c(OPT)}{|C_i|}$ .
- The cost of the algorithm is bounded by

$$\begin{aligned} C(ALG) &\leq \sum_{i=1}^k cost(S_i) \leq c(OPT) \sum_{i=1}^k \frac{|S_i \cap C_i|}{|C_i|} \\ &\leq c(OPT) \sum_{i=1}^{|D|} \frac{1}{i} = O(\log n)c(OPT) \end{aligned}$$



# Metric Facility Location



# Primal-Dual Network Design

- Metric Facility Location is an excellent example of application of Primal-Dual algorithms for Network Design
- Remember:
  - ◆ Primal-dual approximation algorithms construct a feasible dual together with an integral feasible solution for the problem.
  - ◆ Approximation guarantee is obtained by relating the cost of the integral solution to the feasible dual
- We'll present the algorithm of Jain and Vazirani [1999]

# Metric Facility location

## Input:

- undirected graph  $G = (V, E)$
- non-negative edge costs  $c : E \rightarrow \mathbb{R}^+$
- set of **facilities**  $F \subseteq V$
- facility  $i$  has facility opening cost  $f_i$
- set of **demand points**  $D \subseteq V$
- $c_{ij}$ : cost of connecting demand point  $j$  to facility  $i$ .  
Connection costs satisfy triangle inequality:  $c_{ij} \leq c_{ik} + c_{kj}$

## Goal: Compute

- set  $F' \subseteq F$  of opened facilities; and
- function  $\phi : \mathcal{D} \rightarrow \mathcal{F}'$  assigning demand points to opened facilities that minimizes

$$\sum_{i \in F'} f_i + \sum_{j \in \mathcal{D}} c_{\phi(j)j}$$

# LP formulation

$$\begin{aligned} \min \quad & \sum_{i \in F, j \in D} c_{ij} x_{ij} + \sum_{i \in F} f_i y_i \\ \text{s.t.} \quad & \sum_{i \in F} x_{ij} \geq 1 && j \in D \\ & y_i - x_{ij} \geq 0 && i \in F, j \in D \\ & x_{ij} \in \{0, 1\} && i \in F, j \in D \\ & y_i \in \{0, 1\} && i \in F \end{aligned}$$

- $y_i = 1$  if facility  $i$  is opened;
- $x_{ij} = 1$  if demand  $j$  connected to facility  $i$ .

# LP relaxation:

$$\begin{aligned} \text{Primal: } \min \quad & \sum_{i \in F, j \in D} c_{ij} x_{ij} + \sum_{i \in F} f_i y_i \\ \text{s.t.} \quad & \sum_{i \in F} x_{ij} \geq 1 && j \in D \\ & y_i - x_{ij} \geq 0 && i \in F, j \in D \\ & x_{ij} \geq 0 && i \in F, j \in D \\ & y_i \geq 0 && i \in F \end{aligned}$$

$$\begin{aligned} \text{Dual: } \max \quad & \sum_{j \in D} \alpha_j \\ \text{s.t.} \quad & \alpha_j - \beta_{ij} \leq c_{ij} && i \in F, j \in D \\ & \sum_{j \in D} \beta_{ij} \leq f_i && i \in F \\ & \alpha_j \geq 0 && j \in D \\ & \beta_{ij} \geq 0 && i \in F, j \in D \end{aligned}$$



# The Algorithm - Intuition

- We see the execution of the algorithm as a process in time
- Differently for previous algorithms all dual variables associated with yet not connected demand points are uniformly raised in time. E.g.,  $\alpha_j = t$  at time  $t$  if  $j$  yet not connected to an opened facility
- Dual variables pay for the cost of the solution
- $\alpha_j$  pays for connecting demand point  $j$  to an open facility. The connection cost of a demand point is paid only once
- $\beta_{ij}$  is the contribution of demand point  $j$  to the opening of facility  $i$ . Many  $\beta_{ij}$ 's associated to a demand point  $j$
- We should charge a demand point for opening only one facility



# The algorithm

At time 0, set all  $\alpha_j = 0$  and  $\beta_{ij} = 0$  and declare all demands unconnected.

While there is an unconnected demand:

- Raise uniformly all  $\alpha_j$ 's of unconnected demands
- If  $\alpha_j = c_{ij}$ , declare demand  $j$  **tight** with facility  $i$
- For a tight constraint  $ij$ , raise both  $\alpha_j$  and  $\beta_{ij}$
- If  $\sum_j \beta_{ij} = f_i$  at time  $t_i$ , declare:
  - ◆ Facility  $i$  *temporarily opened* at time  $t_i$ ;
  - ◆ All unconnected demands  $j$  that are tight with  $i$  *connected*;
- If  $\alpha_j = c_{ij}$  for a temporarily opened facility  $i$ , declare demand  $j$  **connected** with facility  $i$

[Jain and Vazirani, 1999][Metthu and Plaxton, 2000]



# The Algorithm

## Opening facilities:

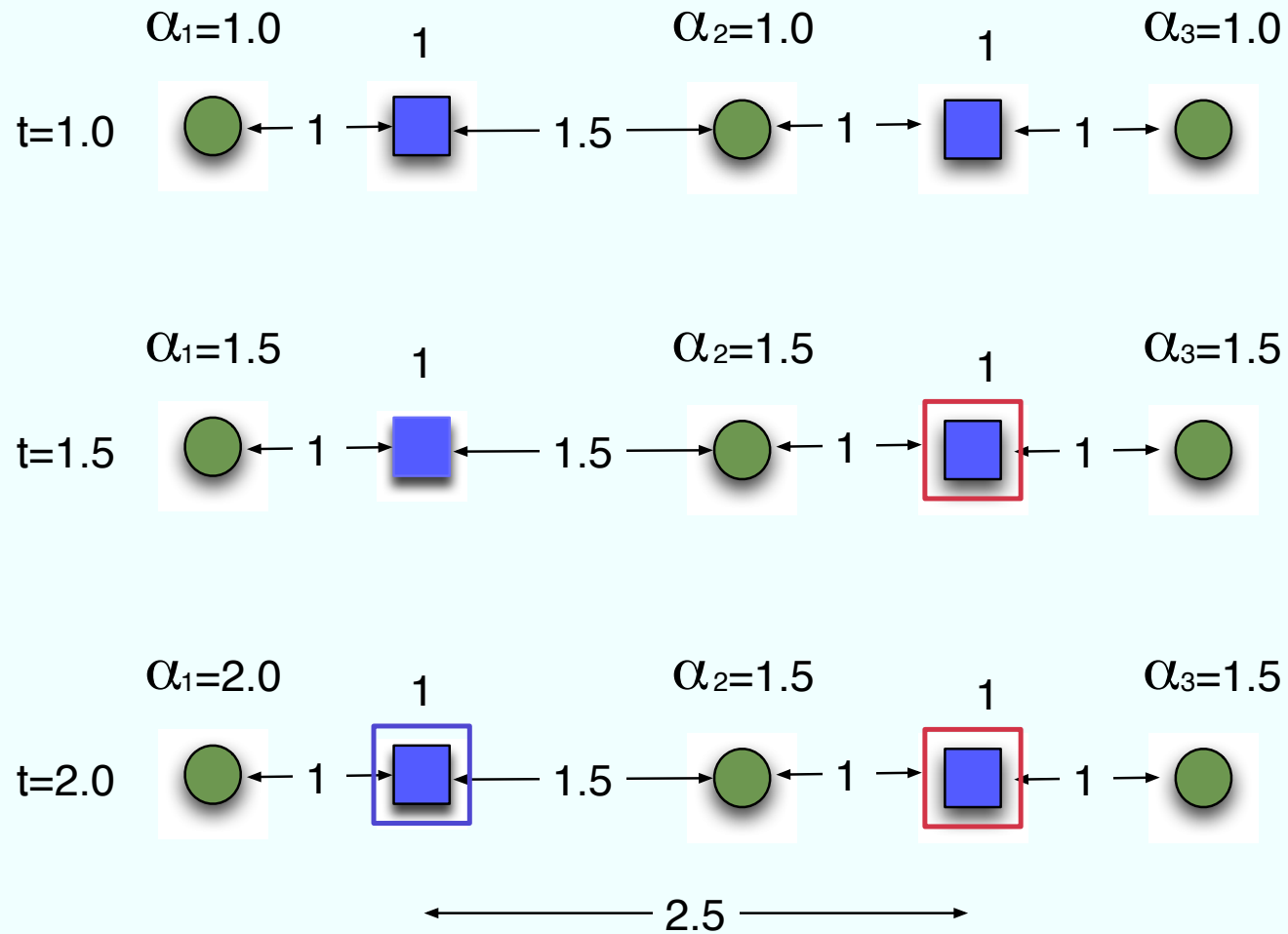
Demand points contribute to more permanently opened facilities. Not enough money for all of them.

- Facility  $i$  *temporarily opened* at time  $t_i$ ;
- Declare facility  $i$  *permanently opened* if there is no permanently opened facility within distance  $2t_i$ .

Open all permanently opened facilities.

Connect each demand to the nearest opened facility.

# Example of execution of the algorithm





# Analysis of the Algorithm

We'll prove the following for the solution  $x_{ij}$  provided by the algorithm:

**Theorem 1** 
$$\sum_{i \in F'} f_i + \sum_{j \in \mathcal{D}} c_{\phi(j)j} \leq 3 \times \sum_{j \in \mathcal{D}} \alpha_j$$

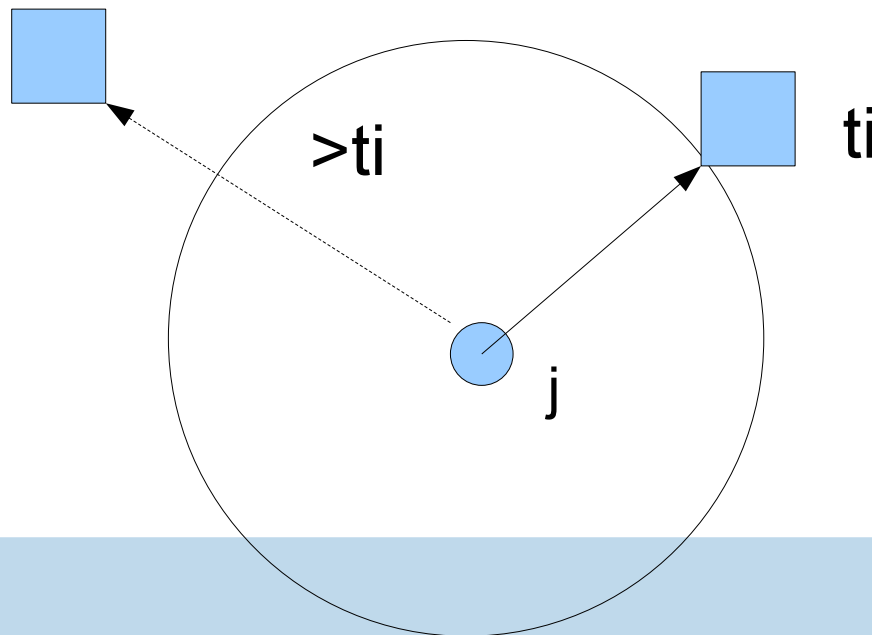
Line of action:

- For demand points connected to permanently opened facilities we prove that  $\alpha_j$  is enough to pay for the connection cost and to contribution to the opening cost
- For demand points connected to temporarily opened facilities we prove that  $\alpha_j$  can pay for 3 times the connection cost to the closest temporarily opened facility

# Proof of 3-approximation

## Demand points connected to opened facilities

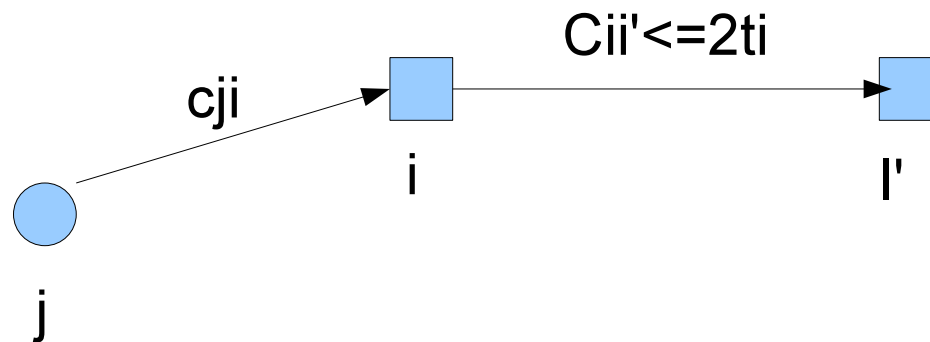
- $\alpha_j = c_{ij} + \beta_{ij}$  for demands connected to opened facility  $i$ .
- $\alpha_j$  pays for connection cost  $c_{ij}$  and contribute with  $\beta_{ij}$  to  $f_i$ .
- Since other opened facilities are at distance  $> t_i$ ,  $\alpha_j$  does not pay for opening any other facility.



# Proof of 3-approximation

## Demand points connected to temporarily opened facilities

- Demand  $j$  connected to temporarily opened facility  $i$ . There exists an opened facility  $i'$  with  $c_{ii'} \leq 2t_i$ .
- Since  $c_{ji} \leq \alpha_j$  and  $t_i \leq \alpha_j$ ,  $c_{ji'} \leq c_{ji} + c_{ii'} \leq 3\alpha_j$

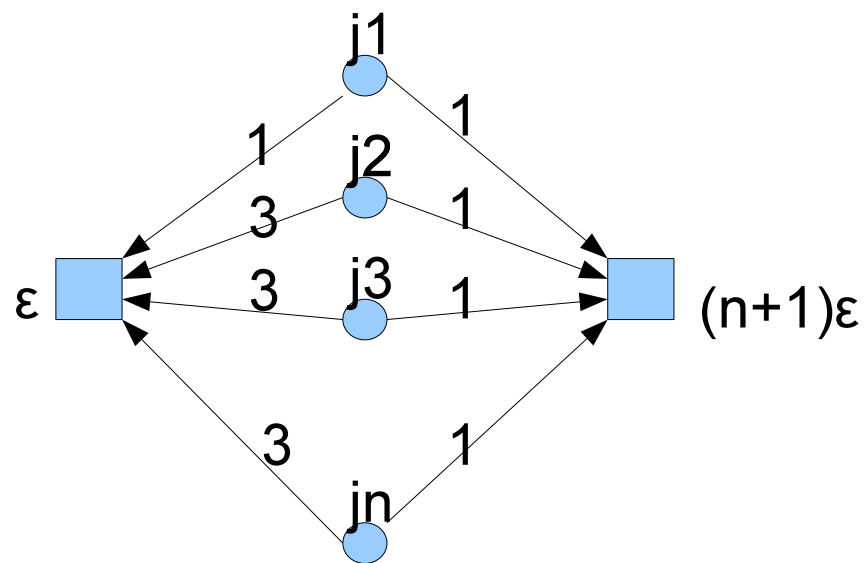


# A tight example

$$OPT = n + (n + 1) \times \epsilon$$

$$ALG = 3 \times (n - 1) + 1 + \epsilon$$

The facility of cost  $\epsilon$  is opened at time  $1 + \epsilon$  and prevents the other facility to be opened since it is at distance 2.





# Running Time of the Algorithm

- $n = |D| + |F|$ : number of nodes
- $m = |D| \times |F|$ : number of edges
- Sort all edges by non decreasing cost with in  $O(m \times \log m)$
- Store for each facility  $i$  the number of contributing demands and the current estimated time of opening
- Store facilities in a heap with priority equal to the estimated time of opening
- Update priority and find next facility to open in  $O(\log |F|)$  time



# Running Time of the Algorithm

Two types of events:

1 Edge  $(i, j)$  goes tight

1.1 If  $i$  not opened, increase number of contributing demands and update priority

1.2 If  $i$  opened, for each  $i'$  with  $(i', j)$  tight, decrease number of contributing demands and update priority

2 Facility  $i$  is fully paid. All demands  $j$  with  $(i, j)$  tight are connected. For each  $i'$  with  $(i', j)$  tight execute step [1.2]

■ Each edge is considered at most twice:

1. When  $(i, j)$  goes tight

2. When  $j$  becomes connected

■ We conclude with an  $O(m \log n)$  running time.