

Return of the Primal-Dual: Distributed Metric Facility Location

Saurav Pandit, Sriram Pemmaraju

presenter : Himan Assaf

Motivation

- Where to locate an hospital?
 - Build an hospital cost money.
 - Drive to each hospital cost money.
 - Minimum payment.
 - What to do ?



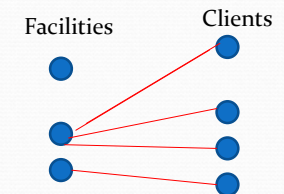
Outline

- Facility location problem definition.
- LP and dual problem relaxation.
- Linear dual problem definition.
- Logarithmic computing time algorithm
 - Algorithm Description.
 - Analysis.
- Constant computing time algorithm
 - Algorithm Description.
- Summary

Facility location problem

- Problem definition
 - Bipartite Graph $G=(F,C,E)$

- F – facility
- C – Clients
- E – graph edges



- $c \in E$ – connection cost.
- $f \in F$ – facility opening cost.
- $\Phi(j)$ – the facility connected to client j .
- Each client must be connected to at least one facility.

Goal : minimize the total cost in order to serve the entire clients.

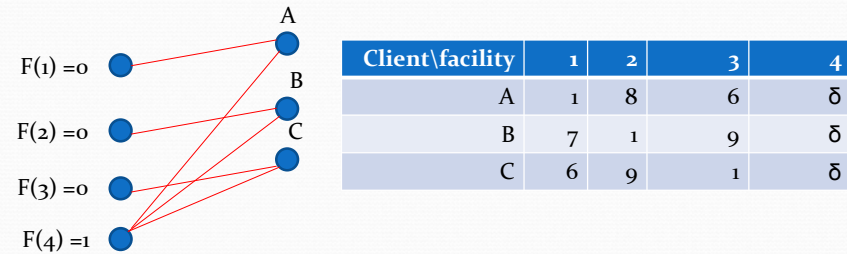
$$\sum_{i \in I} f(i) + \sum_{j \in C} c(j, \phi(j))$$

Facility location problem – greedy solution

- Intuitive and simple solution.
- for each client find $\min(f_i + c_{ij})$.
- Approximation - Worst case $O(n)$.

Facility location problem

- Worst case example



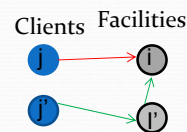
Alg : cost = 3

Opt: cost = 1+38

greedy isn't always the best choice !

Facility location problem

- The facility location problem comes in two main versions:
 - metric facility location
 - Satisfy the connection cost triangle inequality
 - For any $i, i' \in F, j, j' \in C$,
 - $c(i, j) \leq c(i', j') + c(i', j)$.
 - Number of sequential constant-factor approximation algorithm exist.
 - Non metric – more general problem.
 - The best known approximation factor is $O(\log n)$ and its optimal.



Our model

- $G = (F, C, E)$
- Synchronous algorithm.
- Each round clients\facilities send messages to all neighbors.
- m – number of facilities
- n – number of clients.
- Message size at most $O(\log(m+n))$ bits.
- 2 main results :
 - 7-approximation running in $O(\log m + \log n)$ rounds.
 - a k -rounds algorithm that yields an $o(m^{\frac{2}{k}}, n^{3/\sqrt{k}})$ approximation.
- Trade off, running time \rightarrow approximation rate.

CONGEST model

- Each node can send to its neighbor one message.
- Different message to different neighbors.
 - In Our model it is enough to send the same message.

Mathematical introduction

Integer problem :

$$\begin{aligned} &\text{minimize} && \sum_{i \in F, j \in C} c_{ij} \cdot x_{ij} + \sum_{i \in F} f_i \cdot y_i \\ &\text{subject to} && (1) \sum_{i \in F} x_{ij} \geq 1, && j \in C \\ &&& (2) y_i - x_{ij} \geq 0, && i \in F, j \in C \\ &&& x_{ij} \in \{0, 1\}, && i \in F, j \in C \\ &&& y_i \in \{0, 1\}, && i \in F \end{aligned}$$

y_i - whether facility i is open.

x_{ij} -if client j is connected to facility i .

Constraint (1) - each client connected to at least one facility.

Constraint(2) - Client connected to an open facility.

LP relaxation

- Convert into LP relaxation

$$\begin{aligned} &\text{minimize} && \sum_{i \in F, j \in C} c_{ij} \cdot x_{ij} + \sum_{i \in F} f_i \cdot y_i \\ &\text{subject to} && \sum_{i \in F} x_{ij} \geq 0, && j \in C \\ &&& y_i - x_{ij} \geq 0, && i \in F, j \in C \\ &&& x_{ij} \geq 0, && i \in F, j \in C \\ &&& y_i \geq 0, && i \in F \end{aligned}$$

y_i - whether facility i is open.

x_{ij} -if client j is connected to facility i .

- The value always smaller or equal than IP.

Linear dual problem

- Every linear programming problem, referred to as a primal problem, can be converted into a dual problem.
- Definition
 - Minimize $\mathbf{b}^T \mathbf{y}$ subject to $A^T \mathbf{y} \geq \mathbf{c}, \mathbf{y} \geq \mathbf{0}$.
 - The dual problem will be
Maximize $\mathbf{c}^T \mathbf{x}$ subject to $A \mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$
 - Weak duality - the value always greater or equal than the primal problem(will be use in our problem).

LP relaxation - dual problem

- Dual Problem

$$\begin{array}{ll} \text{maximize} & \sum_{j \in C} \alpha_j \\ \text{subject to} & \alpha_j - \beta_{ij} \leq c_{ij}, & i \in F, j \in C \\ & \sum_{j \in C} \beta_{ij} \leq f_i, & i \in F \\ & \alpha_j \geq 0, & j \in C \\ & \beta_{ij} \geq 0, & i \in F, j \in C \end{array}$$

- α_j amount that client j willing to pay in order to connect to a facility.
- β_{ij} payment of client j towards opening facility i .
- Target – finding upper bound –where c is the approximation rate.

$$\sum_{i \in F, j \in C} c_{ij} \cdot x_{ij} + \sum_{i \in F} f_i \cdot y_i \leq c \cdot \left(\sum_{j \in C} \alpha_j \right)$$

High level algorithm

- 3 main steps :
- Init : connect low-paying clients and open cheap facilities (constant time rounds), Initial α_j values for the remaining clients are computed.
- Primal-Dual phase : increments its dual variable α_i in each round until a facility i is found such that the connection cost c_{ij} is paid for, and facility pay f_i also paid, stop when all clients are satisfied ($\log(n)$ rounds).
- Sparsification phase : shut down some of the temporarily open facilities ($\log(m)$ rounds), and connect unconnected clients.

Initialization Phase

- For each $i \in C, j \in F$, find : $F_{i^*} + C_{i^*j} = \text{MIN}(F_i + C_{ij})$
- Define

$$\alpha_j = \frac{1}{n} \text{Min}_i (F_i + c_{i,j})$$

$$\beta_{ij} = \text{Max}(0, \alpha_j - c_{ij})$$

$$\phi_j = i^*$$
- Lemma : The solution constitute a feasible solution.

Low Paying Client and Cheap facility

- Lets define $\alpha^* = \text{MAX}_j \alpha_j$
 - Low pay client is $\alpha_j < \alpha^* / n^2$
 - Cheap facility $\phi(j)$
- Lemma : The total cost of opening all cheap facilities and connecting each low-paying client j to facility $\phi(j)$ is at most OPT.

Lemma Proof

- Definition
 - L - low paying clients group.
- Remainder $\alpha_j = \frac{1}{n} \text{Min}_i (F_i + c_{ij})$, $\alpha_j < \alpha^* / n^2$
- Total amount $\sum_{j \in L} (c_{\phi(j)j} + f_{\phi(j)}) = \sum \text{Min}_i (F_i + c_{ij}) = \sum_{j \in L} n \cdot \alpha_j \leq \sum_{j \in L} \frac{\alpha^*}{n}$
 - Of course, $|L| < n$ therefore $\sum_{j \in L} \frac{\alpha^*}{n} \leq \alpha^*$
 - $\alpha^* \leq \sum_{j \in C} \alpha_j$, where α_j are feasible dual problem solution.
 - from weak duality $\sum_{j \in C} \alpha_j \leq OPT$

The Primal-Dual Phase

- Removing the initial solution that aren't cheap facilities and low paying clients.
- Synchronous model, 3 step in each phase (Send, Receive, Send for Client or Revice, Send, Revice for facility).
- Clients Colors : grey – inactive, white – active.
- Facility status : {closed, temporarily-open, open}.
- Define $\alpha_{\min} := \min_{j \in L} \alpha_j$

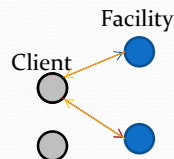
The Primal-Dual Phase

Client algorithm

Algorithm 2 Primal-Dual Client j Algorithm

- 1: **Init:** If j is not a low-paying client: $\alpha_j \leftarrow \alpha_{\min}$; \leftarrow
 $color(j) \leftarrow white.$
- 2: {Activity in each iteration:}
- 3: if ($color(j) = white$) then \leftarrow
- 4: $Send[\alpha_j]$ \leftarrow
- 5: $\forall i: \gamma_{ij} \leftarrow \alpha_j - c_{ij}$
- 6: $\forall i: Receive[status(i)]$ \leftarrow
- 7: if ($\exists i$ such that $(status(i) \in \{temporarily-open, open\} \wedge \gamma_{ij} \geq 0)$) then
- 8: $color(j) \leftarrow grey$ \leftarrow
- 9: $\phi(j) \leftarrow i$ \leftarrow
- 10: end if
- 11: $Send[color(j)]$
- 12: $\alpha_j \leftarrow 2 \cdot \alpha_j$
- 13: end if

Grey – inactive
White – active

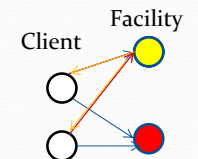


The Primal-Dual Phase

Facility algorithm

Algorithm 3 Primal-Dual Facility i Algorithm

- 1: **Init:** $status(i) \leftarrow closed$ if i is not a cheap facility \leftarrow
- 2: {Activity in each iteration:}
- 3: if ($status(i) = closed$) then \leftarrow
- 4: $\forall j: Receive[\alpha_j]$ \leftarrow
- 5: $\forall j$ such that $color(j) = white: \beta_{ij} \leftarrow \max\{\alpha_j - c_{ij}, 0\}$
- 6: if ($\sum_{j \in C} \beta_{ij} \geq f_i$) then
- 7: $status(i) \leftarrow temporarily-open$ \leftarrow
- 8: end if
- 9: $Send[status(i)]$ \leftarrow
- 10: $\forall j: Receive[color(j)]$
- 11: end if



Blue – close
Yellow – temporary open
Red – open

Primal Dual running time

- The Primal-Dual Algorithm runs for at most $9 \log n$ rounds.
- $\alpha_i^{(s)} - \alpha_i$ after s iteration,
- Reminder $\alpha_j^{(0)} = \alpha_{\min} > \alpha^* / n^2$, therefore, after $t = 3 \log(n)$ iteration $\alpha_j^{(t)} > n \cdot \alpha^*$.
- Client j , all by itself, is paying for more than the amount needed to open facility after $3 \log(n)$.
- Each iteration = 3 phases, there for $3 \cdot 3 \log(n) = 9 \log(n)$.

Primal dual approximation ratio

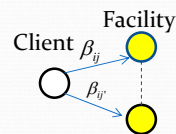
- Set $\bar{\alpha}_j = 0, j \in L$, and $\bar{\alpha}_j = \alpha_j / 2, j \notin L$.
- Set $\bar{\beta}_{ij} := \max\{c_{ij} - \bar{\alpha}_j, 0\}$.
- Feasible solution.
- Weak duality

$$\sum_{j \in C} \bar{\alpha}_j = \sum_{j \in L} \alpha_j / 2 \leq OPT$$

$$\sum_{j \in L} \alpha_j \leq 2OPT$$

Sparsification Phase(1)

- Improve solution.
- Work on Temporary open clients.
- Congest model.
- $O(\log(m))$ rounds.
- Define $H = (F_t, E_t)$
 - F_t - temporary open facilities
 - E_t - edge between facilities who have clients that are β_{ij}, β_{ir} to the facility are positive



Sparsification Phase(2)

- Directly connected and indirectly connected
 - Directly - case client connected from Primal Dual phase
 - In directly - case client connected from Sparsification phase.
- Two main goals
 - Bound for each client j , the size $K_j = \{i \in F_t \mid \beta_{ij} > 0, \text{ and } i \text{ is open}\}$
 - Make sure that for each client that its facility shut down, will have another facility reconnect at a cost $O(\alpha_j)$

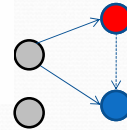
SparsificationPhase(2)

- Algorithm

Algorithm 4 Sparsification Phase

- 1: Compute a maximal independent set (MIS) M of H ←
- 2: Permanently open each facility in M , i.e., $\forall i \in M$: $status(i) \leftarrow open$ ←
- 3: Close each facility in $F_t \setminus M$, i.e., $\forall i \in F_t \setminus M$: $status(i) \leftarrow closed$ ←
- 4: for all clients $j \in C$ do
- 5: if $(status(\phi(j)) = closed)$ then
- 6: $\phi(j) \leftarrow i'$, where $i' \in M$ and is a neighbor in H of $\phi(j)$ ←
- 7: end if
- 8: end for

Blue – close
 Yellow – temporary open
 Red – open



Analysis(1)

- The algorithm terminates in $O(\log m + \log n)$ rounds.
- Approximation ratio
 - Init Phase, Cost less than OPT
 - Cost of Direct (A) and indirectly connected (B), Cost less than $6OPT$

$$\sum_{i \in I} f_i + \sum_{j \in C} c_{\phi(j)j} < 7OPT$$

- If j is indirectly connected to i , then $c_{ij} < 3\alpha_j$ (without proof).

Analysis(2)

- I' not "cheap" Facility, A – directed group, B – undirected group.

- Reminder $f_i < \sum_{j \in A} \beta_{ij}$, $\sum_{j \in L} \alpha_j \leq 2OPT$

- Now the phase 2 and phase 3 cost are

$$\begin{aligned} & \sum_{i \in I'} f_i + \sum_{j \in A} c_{\phi(j)j} + \sum_{j \in B} c_{\phi(j)j} (0) \\ & \leq \sum_{i \in I'} \sum_{j \in A} \beta_{ij} + \sum_{j \in A} (\alpha_j - \beta_{\phi(j)j}) + 3 \sum_{j \in B} \alpha_j (1) \\ & \leq \sum_{j \in A} \beta_{\phi(j)j} + \sum_{j \in A} (\alpha_j - \beta_{\phi(j)j}) + 3 \sum_{j \in B} \alpha_j (2) \\ & \leq \sum_{j \in A} \alpha_j + 3 \sum_{j \in B} \alpha_j \leq 3 \sum_{j \in L} \alpha_j \end{aligned}$$

- (0) to (1) – $\sum_{j \in A} c_{\phi(j)j} \leq \sum_{j \in A} \alpha_j - \beta_{\phi(j)j}$
- (1) to (2) – $\sum_{i \in I'} \sum_{j \in A} \beta_{ij} = \sum_{j \in A} \sum_{i \in I'} \beta_{ij} \leq \sum_{j \in A} \beta_{ij}$
- Total $3 \sum_{j \in L} \alpha_j \leq 6OPT$

K Round algorithm(1)

- $K = k_1 * k_2$ rounds.
- $O(n^{3/k_1} \cdot m^{2/k_2})$ approximation.
- Same initialization phase.
- Different primal dual phase.
- For every primal dual phase run k_2 rounds of Sparsification phase (total $k_1 * k_2$).
- Primal dual :
 - Iteration factor n^{3/k_1}
 - After K_1 rounds have enough to pay for a facility

$$\alpha_{\min} \cdot (n^{3/k_1})^{k_1} = \alpha_{\min} \cdot n^3 \leq \frac{\alpha^*}{n^2} \cdot n^3 = \alpha^* \cdot n$$
 - Approximation after k_1 rounds $\sum \alpha_j < n^{3/k_1} \cdot OPT$

K algorithm Sparsification(1)

- Probabilistic algorithm.
- Run on a graph $H=(F_t, E_t)$.
- Different from the logarithm Sparsification phase.
- Two main goals for **Sparsification** phase :
 - Bound for each client j , the size $K_j = \{i \in F_t \mid \beta_{ij} > 0, \text{ and } i \text{ is open}\}$
 - Make sure that for each client that its facility shut down, will have another facility reconnect at a cost $O(\alpha_j)$
- Degree after k_2 iteration of node subset $M < O(M^{2/k_2})$
 - $K_i = O(1)$, Implying upper bound of $O(M^{2/k_2})$

K algorithm Sparsification(2)

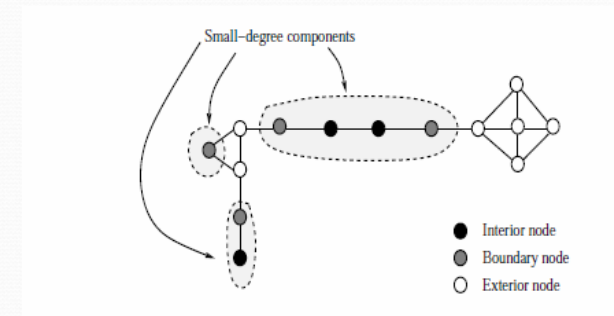
- Ensure that case facility shutdown its client will reconnect to a facility far as $2 \cdot k_2$ hops.
- Upper bound $O(\alpha_j)$ for $2k$ hops neighbored given upper bound $O(m^{2/k_2} \cdot \alpha_j)$

K algorithm Sparsification Phase(1)

- Define $M_0 = F_t, G_0 = H$
 - M_s - set of nodes selected in iteration s on graph G_{s-1}
 - $G_s = G[M_{s-1}]$
- The goal is to select M nodes that there max degree in $G[M_s]$ is smaller $m^{2/s+2}$
- Define
 - Degree smaller than $m^{2/s+2}$ called small degree.
 - Case all neighbor are also small degree, node called **interior**
 - else call **boundary** node.
 - Node that are not small called **exterior**.

K algorithm Sparsification Phase(2)

- Interior, boundary and exterior nodes, $\tau = 2$



K algorithm Sparsification Algorithm

- Algorithm

Algorithm 6 Sparsification Algorithm

```

1: Input:  $G = (V, E)$ ,  $|V| = m$ , positive integer  $t$ 
2:  $M_0 \leftarrow V$ 
3: for  $s = 1$  to  $t$  do
4:    $r \leftarrow s + 2$ 
5:    $\tau \leftarrow m^{\frac{2}{r}}$ 
6:    $\rho \leftarrow \frac{1}{m^{\frac{2}{r}}}$ 
7:   for all  $u \in M_{s-1}$ ,  $d_u \leftarrow$  degree of  $u$  in  $G[M_{s-1}]$ 
8:   for all  $u \in M_{s-1}$ ,  $\Delta_u \leftarrow \max\{d_v \mid v \in N[u]\}$ 
9:   if  $(d_u \leq \tau) \wedge (\Delta_u \leq \tau)$  then
10:     $type(u) \leftarrow interior$ 
11:     $u$  joins  $M_s$ 
12:  end if
13:  if  $(d_u \leq \tau) \wedge (\Delta_u > \tau)$  then
14:     $type(u) \leftarrow boundary$ 
15:  end if
16:  if  $(d_u > \tau)$  then
17:     $type(u) \leftarrow exterior$ 
18:  end if
19:  if  $type(u) == interior$  then
20:     $u$  joins  $M_s$  (deterministically)
21:  end if
22:  if  $type(u) \in \{boundary, exterior\}$  then
23:     $u$  independently joins  $M_s$  with probability  $\rho$ 
24:  end if
25: end for
26: Output:  $M_t$ 
    
```

K Round algorithm

- Algorithm

Algorithm 5 k -round Algorithm

```

1: Initialization Phase.
2: for  $p \leftarrow 1$  to  $k_1$  do
3:   Run an iteration of the Primal-Dual Algorithm (for each client and each facility); the output is the set (denoted  $T_p$ ) of temporarily open facilities
4:   if  $(\exists \text{ facility } i \in T_p) \wedge (\exists \text{ facility } i' \in (T_1 \cup T_2 \cup \dots \cup T_{p-1})) \wedge (\exists \text{ client } j: \beta_{ij} > 0 \wedge \beta_{i'j} > 0)$  then
5:      $status(i) \leftarrow closed$  (delete  $i$  from  $T_p$ )
6:      $\phi(j) \leftarrow i'$ 
7:   end if
8:   for  $q \leftarrow 1$  to  $k_2$  do
9:     Run an iteration of the Sparsification Algorithm
10:    {for each facility that remains in  $T_p$ }
11:   end for
12: end for
    
```

K algorithm Sparsification analysis(2)

- With high probability $MaxDeg(H[M]) \leq 6 \cdot m^{2/k_2}$, also $K_j = \{i \in F_i \mid \beta_{ij} > 0, \text{ and } i \text{ is open}\}, K_j = O(1)$, Then the $\sum \beta_{ij} = O(m^{2/k_2})$
- With high probability initially connected to facility i' that was closed will find a facility with new pay $O(m^{2/k_2})$
- Totally the whole algorithm will pay
 - Primal dual – $O(n^{3/k_1})$
 - Sparsification – $O(m^{2/k_2})$
 - Whole together $O(n^{3/k_1} \cdot m^{2/k_2})$
- Runing time $O(k_1 \cdot k_2)$

Summary

- We saw two main algorithms
 - Logarithm algorithm with 7 constant factor approximation.
 - Constant running time algorithm with $O(n^{3/k_1} \cdot m^{2/k_2} \cdot OPT)$ Approximation ratio.
- Trade off between running time to approximation ratio.

Questions

- questions ?

