Approximation Algorithms for the Fault-Tolerant Facility Placement Problem

Li Yan

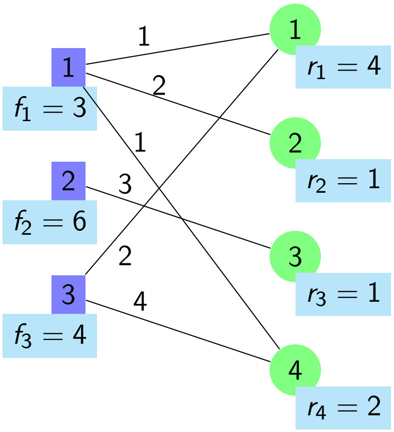
Computer Science
University of California Riverside

06/10/2013

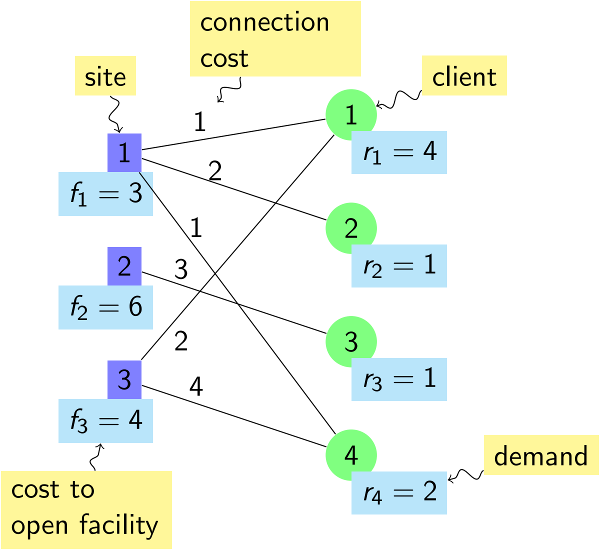
Outline

Table of Contents

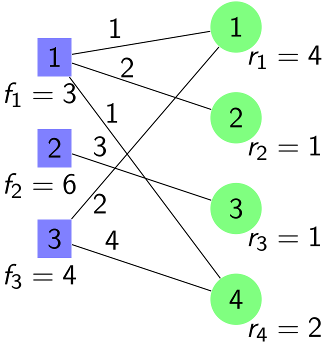
Fault-Tolerant Facility Placement Problem (FTFP)



Fault-Tolerant Facility Placement Problem (FTFP)

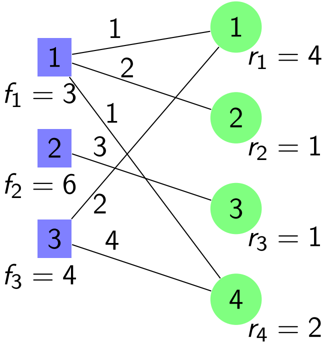


Feasible Integral Solution

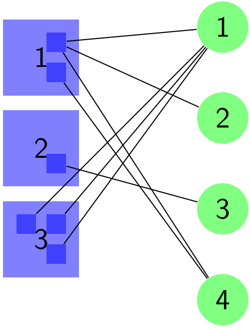


Instance

Feasible Integral Solution

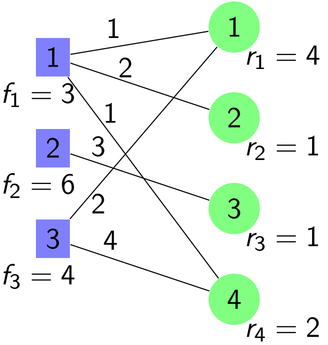


Instance

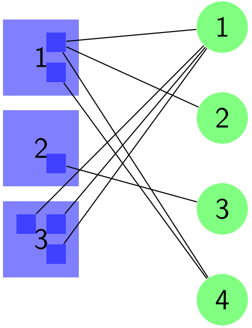


Solution

Feasible Integral Solution



Instance

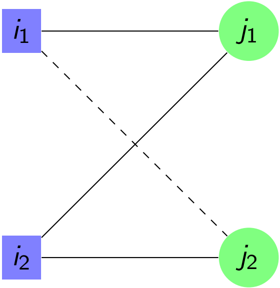


Solution

Cost

$$2f_1 + f_2 + 3f_3 + d_{11} + d_{12} + 2d_{14} + d_{23} + 3d_{31} = 38$$

Metric Distances: Triangle Inequality



Triangle Inequality

$$d(i_1, j_2) \leq d(i_1, j_1) + d(i_2, j_1) + d(i_2, j_2)$$

Needed when estimating distances...

Table of Contents

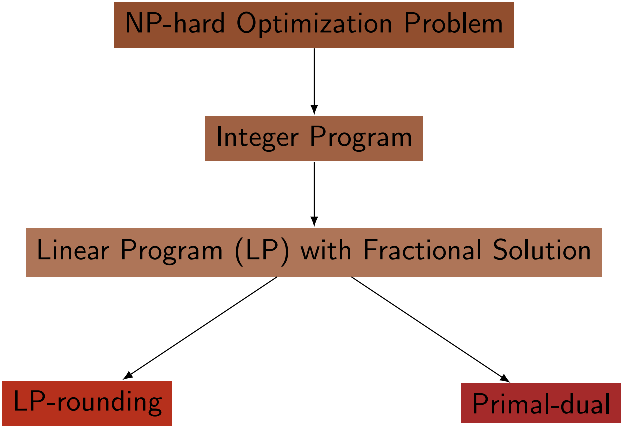
Hardness

How hard is FTFP?

FTFP is NP-hard

FTFP is MaxSNP-hard

Best ratio ≥ 1.463 unless $\mathrm{P} = \mathrm{NP}$


NP-hard Optimization Problem
Integer Program
Linear Program (LP) with Fractional Solution
LP-rounding
Primal-dual

Results Highlight

- LP-rounding: 1.575-approximation
- LP-rounding: asymptotic ratio of 1 when all demands large
- Primal-dual: H_n-approximation
- Primal-dual: Example of $\Omega(\log n / \log\log n)$ for dual-fitting

Relation between Problems

FTFP	$r_j \geq 1$	$< \infty$ facility per site
UFL	$r_j = 1$	≤ 1 facility per site
FTFL	$r_j \geq 1$	≤ 1 facility per site

Relation between Problems

FTFP	$r_j \geq 1$	$< \infty$ facility per site
UFL	$r_j = 1$	≤ 1 facility per site
FTFL	$r_j \geq 1$	≤ 1 facility per site

$$\text{UFL} \preceq \text{FTFP} \preceq \text{FTFL}$$

Relation between Problems

FTFP	$r_j \geq 1$	$< \infty$ facility per site
UFL	$r_j = 1$	≤ 1 facility per site
FTFL	$r_j \geq 1$	≤ 1 facility per site

$$\text{UFL} \preceq \text{FTFP} \preceq \text{FTFL}$$

LP-rounding	
UFL, FTFP	1.575
FTFL	1.7245

Relation between Problems

FTFP	$r_j \geq 1$	$< \infty$ facility per site
UFL	$r_j = 1$	≤ 1 facility per site
FTFL	$r_j \geq 1$	≤ 1 facility per site

$$\text{UFL} \preceq \text{FTFP} \preceq \text{FTFL}$$

LP-rounding	
UFL FTFP	1.575
FTFL	1.7245

Primal-dual	
UFL	1.52
FTFP FTFL	$O(\log n)$

Table of Contents

Related Work for UFL

Approximation Results for UFL

Shmoys, Tardos and Aardal	1997	3.16	LP-rounding
Chudak	1998	1.736	LP-rounding
Sviridenko	2002	1.58	LP-rounding
Jain and Vazirani	2001	3	primal-dual
Jain *et al.*	2002	1.61	greedy
Mahdian *et al.*	2002	1.52	greedy
Arya *et al.*	2004	3	local search
Byrka	2007	1.5	hybrid
Li	2011	1.488	hybrid

Lower Bound

Guha and Khuller	1998	1.463

Related Work for FTFL

Approximation Algorithms for FTFL

Jain and Vazirani	2000	$3 \ln \max_j r_j$	primal-dual
Guha *et al.*	2001	4	LP-rounding
Swamy, Shmoys	2008	2.076	LP-rounding
Byrka *et al.*	2010	1.7245	LP-rounding

No primal-dual algorithms for FTFL with constant ratio.

Work on FTFP (Dissertation Topic)

Approximation Algorithms for FTFP

Xu and Shen	2009		Introduced FTFP
Liao and Shen	2011	1.861	Dual-fitting (for special case)
Yan and Chrobak	2011	3.16	LP-rounding
Yan and Chrobak	2012	1.575	LP-rounding
Yan and Chrobak	preliminary results		Dual-fitting (for general case)

Table of Contents

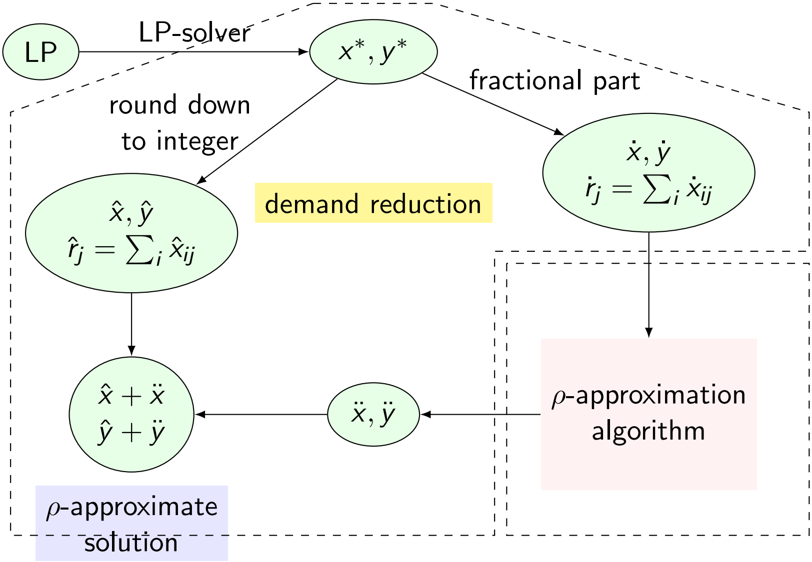
Algorithm for FTFP — LP

- y_i = number of facilities open at site $i \in F$
- x_{ij} = number of connections from client $j \in \mathbb{C}$ to site $i \in F$

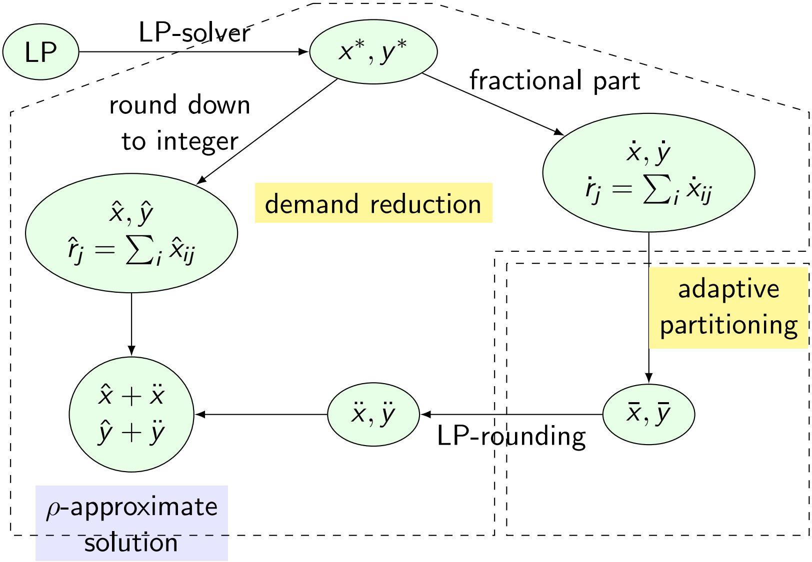
$$\begin{array}{llll} \text{(Primal)} & \text{minimize} & \sum f_i y_i + \sum d_{ij} x_{ij} & \\ & \text{subject to} & y_i - x_{ij} \geq 0 & \forall i, j \\ & & \sum x_{ij} \geq r_j & \forall j \\ & & x_{ij} \geq 0, y_i \geq 0 & \forall i, j \end{array}$$

$$\begin{array}{llll} \text{(Dual)} & \text{maximize} & \sum r_j \alpha_j & \\ & \text{subject to} & \sum \beta_{ij} \leq f_i & \forall i \\ & & \alpha_j - \beta_{ij} \leq d_{ij} & \forall i, j \\ & & \alpha_j \geq 0, \beta_{ij} \geq 0 & \forall i, j \end{array}$$

Algorithm for FTFP — Demand Reduction



Algorithm for FTFP — Adaptive Partitioning



Techniques

- Demand Reduction
  - Reduce all r_j to polynomial values (to ensure polynomial time of rounding)
  - ρ-approx for reduced instance $\Rightarrow$ ρ-approx for original instance
- Adaptive Partitioning
  - Split sites into facilities and clients into unit demands
  - Split associated fractional values
  - Properties ensure rounding similar to UFL can be applied

Demand Reduction

Implementation

- Solving LP for $(\mathbf{x}^*, \mathbf{y}^*)$
- $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) = (\mathbf{x}^*, \mathbf{y}^*)$ round down to integer
- $(\dot{\mathbf{x}}, \dot{\mathbf{y}}) = (\mathbf{x}^*, \mathbf{y}^*) - (\hat{\mathbf{x}}, \hat{\mathbf{y}})$, fractional part
- $\hat{r}_j = \sum_i \hat{x}_{ij}$ for $\hat{\mathcal{I}}$, $\dot{r}_j = r_j - \hat{r}_j$ for $\dot{\mathcal{I}}$
- $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ (integral) feasible and optimal for $\hat{\mathcal{I}}$
- $(\dot{\mathbf{x}}, \dot{\mathbf{y}})$ (fractional) feasible and optimal for $\dot{\mathcal{I}}$

Properties

- $\dot{r}_j = \text{poly}(|F|)$
- ρ-approx for $\dot{\mathcal{I}}$ implies ρ-approx for $\mathcal{I}$

Demand Reduction: Consequences

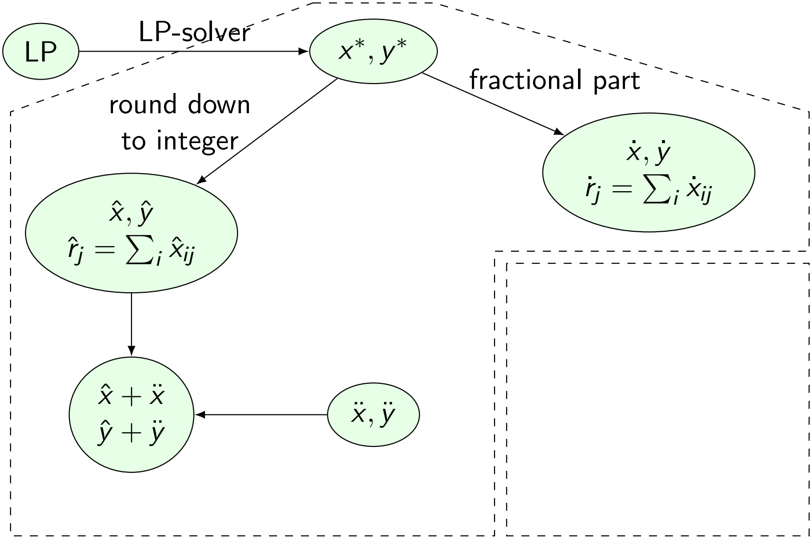
FTFP to FTFL, 1.7245-approximation

- Sites into facilities
- Clients with demand r_j
- FTFL size polynomial because of demand reduction

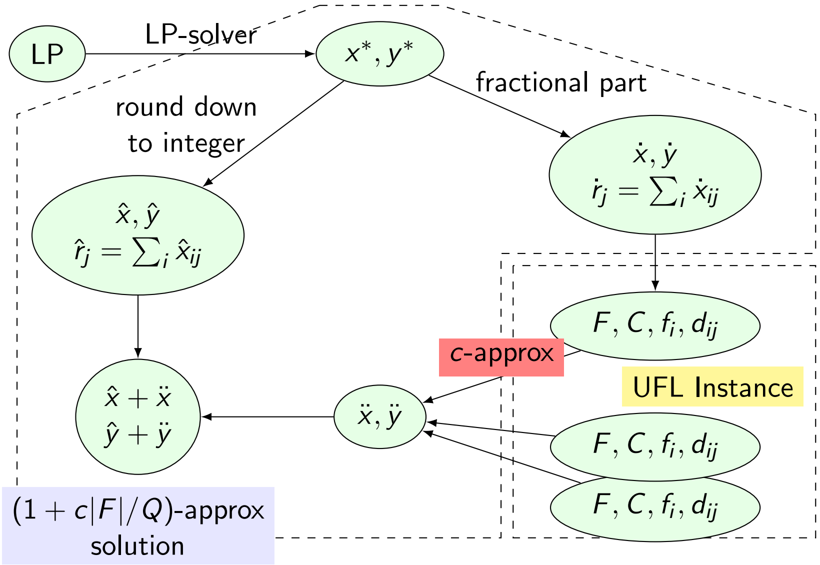
Ratio $1 + O(|F|/Q)$ for $Q = \min_j r_j$, approaches 1 when Q is large

- Next slide

Ratio $1 + O(|F|/Q)$ for FTFP



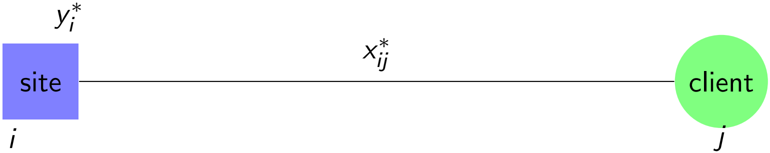
Ratio $1 + O(|F|/Q)$ for FTFP



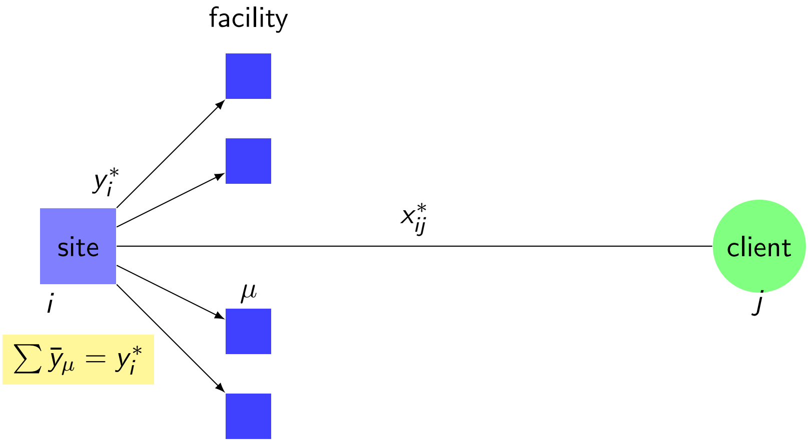
Techniques

- Demand Reduction
  - Reduce all r_j to polynomial values (to ensure polynomial time of rounding)
  - ρ-approx for reduced instance $\Rightarrow$ ρ-approx for original instance
- Adaptive Partitioning
  - Split sites into facilities and clients into unit demands
  - Split associated fractional values
  - Properties ensure rounding similar to UFL can be applied

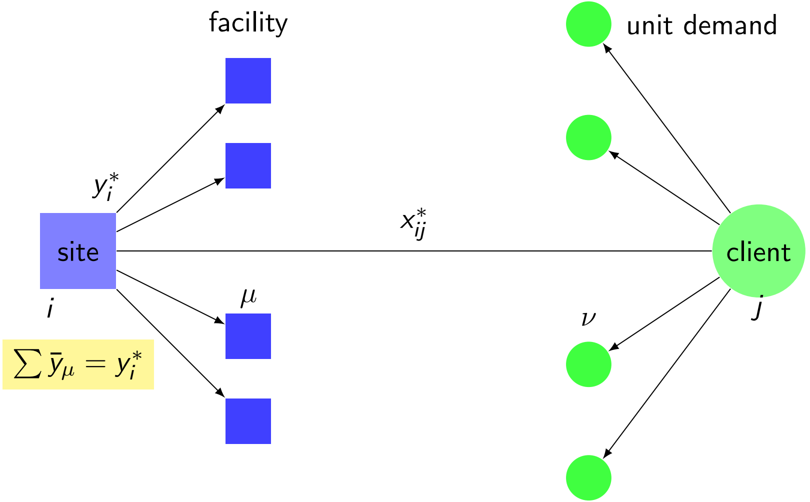
Adaptive Partitioning



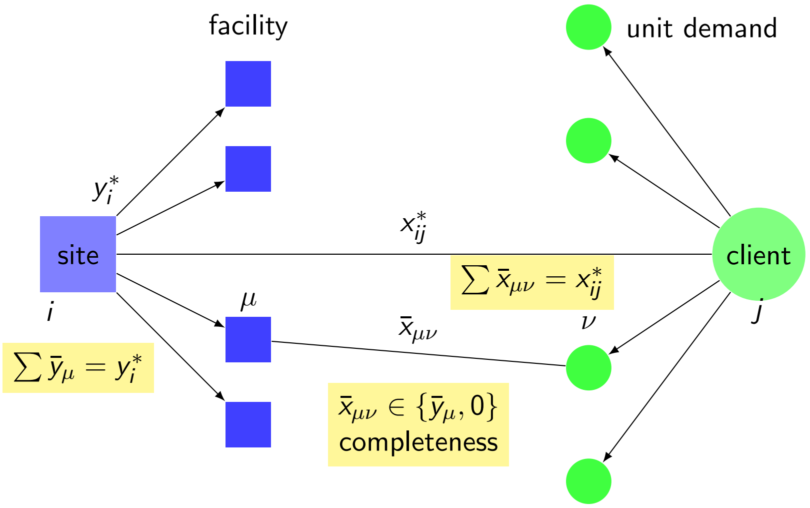
Adaptive Partitioning



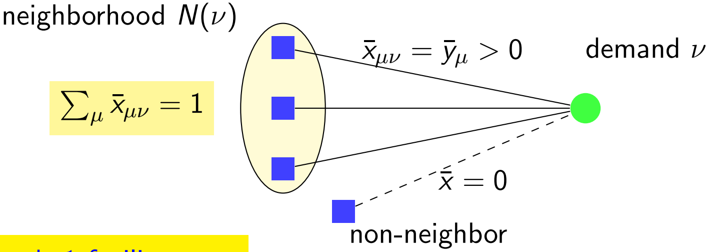
Adaptive Partitioning



Adaptive Partitioning

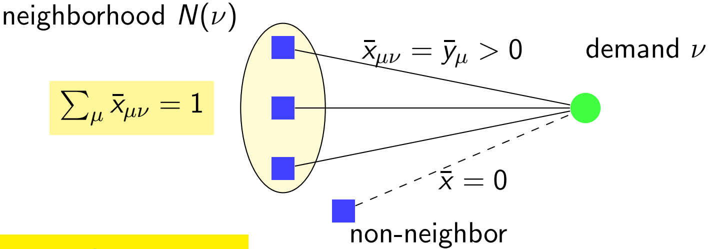


Neighborhood of Demand



ν needs 1 facility...

Neighborhood of Demand



ν needs 1 facility...

Strategy 1: for each ν, open one $\mu \in N(\nu)$ with prob. $\bar{y}_\mu$

- optimal connection cost
- large facility cost

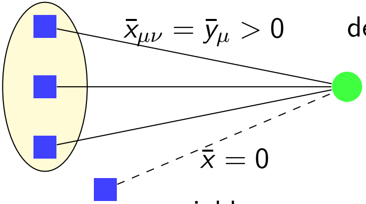
Strategy 2: open facility only for demands with disjoint neighborhoods

- optimal facility cost
- large connection cost

Neighborhood of Demand

neighborhood $N(\nu)$

$\sum_{\mu} \bar{x}_{\mu\nu} = 1$



ν needs 1 facility...

Strategy 1: for each ν, open one $\mu \in N(\nu)$ with prob. $\bar{y}_{\mu}$

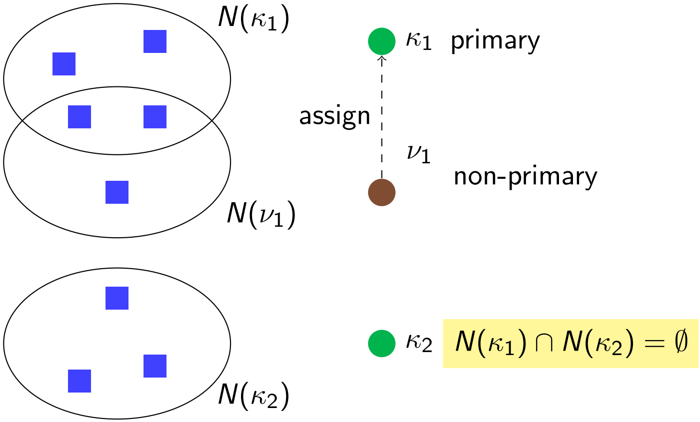
- optimal connection cost
- large facility cost

Strategy 2: open facility only for demands with disjoint neighborhoods

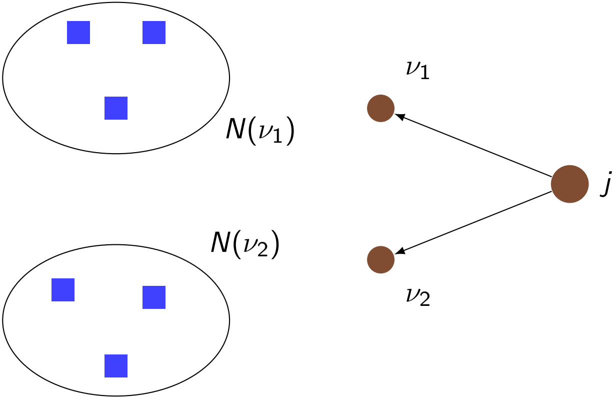
- optimal facility cost
- large connection cost

How to balance these two costs?

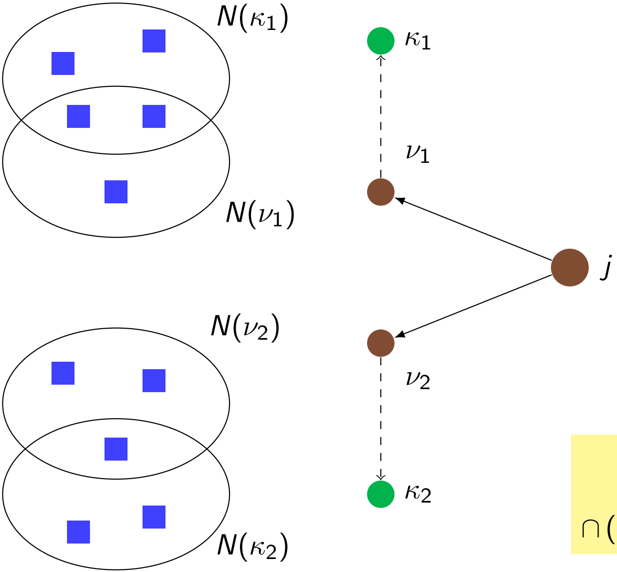
Two Types of Demands: Primary and Non-primary



Neighborhood Structure for Siblings

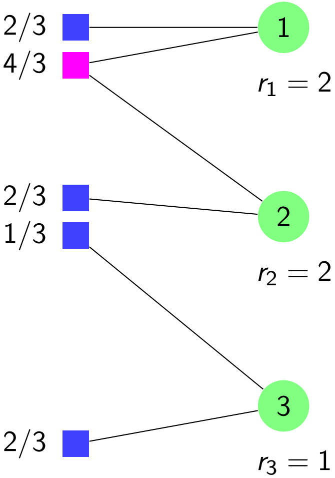


Neighborhood Structure for Siblings



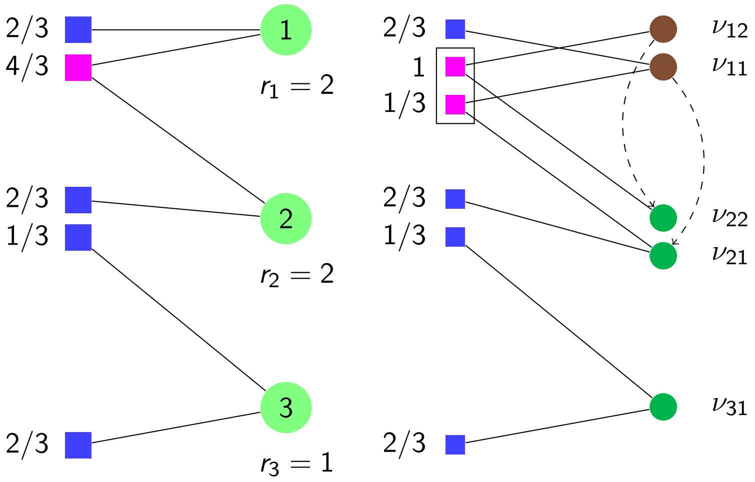
For siblings
$$(N(\kappa_1) \cup N(\nu_1)) \cap (N(\kappa_2) \cup N(\nu_2)) = \emptyset$$

Example of Partitioning



Before Partitioning

Example of Partitioning



Before Partitioning After Partitioning

Summary of Partitioning

Partitioning:

- Clients $\rightarrow$ demands
- Sites $\rightarrow$ facilities
- $(x^*, y^*) \rightarrow (\bar{x}, \bar{y})$
- $\sum_{\mu} \bar{x}_{\mu\nu} = 1$
- $\bar{x}_{\mu\nu} = \bar{y}_{\mu}$ or 0

Structure:

- If κ_1, κ_2 primary then $N(\kappa_1) \cap N(\kappa_2) = \emptyset$
- Each non-primary ν assigned to κ with
  - $N(\kappa) \cap N(\nu) \neq \emptyset$
  - priority$(\kappa) \leq$ priority (ν)
- $(N(\kappa_1) \cup N(\nu_1)) \cap (N(\kappa_2) \cup N(\nu_2)) = \emptyset$

Summary of Partitioning

Partitioning:

- Clients $\rightarrow$ demands
- Sites $\rightarrow$ facilities
- $(x^*, y^*) \rightarrow (\bar{x}, \bar{y})$
- $\sum_\mu \bar{x}_{\mu\nu} = 1$
- $\bar{x}_{\mu\nu} = \bar{y}_\mu$ or 0

Structure:

- If κ_1, κ_2 primary then $N(\kappa_1) \cap N(\kappa_2) = \emptyset$

small facility cost

- Each non-primary ν assigned to κ with
  - $N(\kappa) \cap N(\nu) \neq \emptyset$
  - priority$(\kappa) \leq$ priority (ν)
- $(N(\kappa_1) \cup N(\nu_1)) \cap (N(\kappa_2) \cup N(\nu_2)) = \emptyset$

Summary of Partitioning

Partitioning:

- Clients $\rightarrow$ demands
- Sites $\rightarrow$ facilities
- $(x^*, y^*) \rightarrow (\bar{x}, \bar{y})$
- $\sum_{\mu} \bar{x}_{\mu\nu} = 1$
- $\bar{x}_{\mu\nu} = \bar{y}_{\mu}$ or 0

Structure:

- If κ_1, κ_2 primary then $N(\kappa_1) \cap N(\kappa_2) = \emptyset$

small facility cost

- Each non-primary ν assigned to κ with
  - $N(\kappa) \cap N(\nu) \neq \emptyset$
  - priority$(\kappa) \leq$ priority (ν)

small connection cost of ν

- $(N(\kappa_1) \cup N(\nu_1)) \cap (N(\kappa_2) \cup N(\nu_2)) = \emptyset$

Summary of Partitioning

Partitioning:

- Clients $\rightarrow$ demands
- Sites $\rightarrow$ facilities
- $(x^*, y^*) \rightarrow (\bar{x}, \bar{y})$
- $\sum_\mu \bar{x}_{\mu\nu} = 1$
- $\bar{x}_{\mu\nu} = \bar{y}_\mu$ or 0

Structure:

- If κ_1, κ_2 primary then $N(\kappa_1) \cap N(\kappa_2) = \emptyset$

small facility cost

- Each non-primary ν assigned to κ with
  - $N(\kappa) \cap N(\nu) \neq \emptyset$
  - priority$(\kappa) \leq$ priority (ν)

small connection cost of ν

- $(N(\kappa_1) \cup N(\nu_1)) \cap (N(\kappa_2) \cup N(\nu_2)) = \emptyset$

fault-tolerance

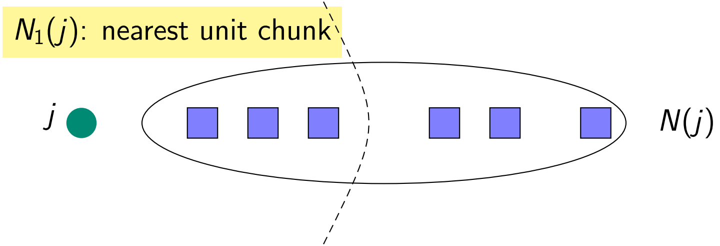
Partitioning Implementation

Partitioning implementation: two phases

- Phase 1, the partitioning phase
  - Define demands
  - Allocate facilities
- Phase 2, the augmenting phase
  - Add facilities to make neighborhood unit

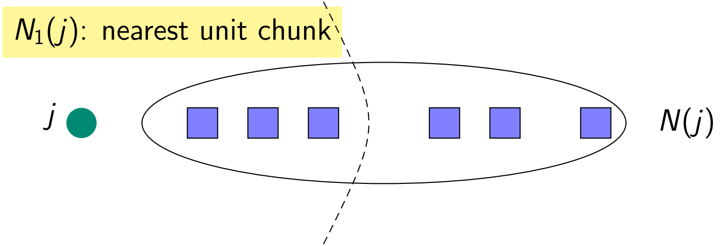
Phase 1, Step 1: Choose Best Client

In each iteration, create one demand for best client



Phase 1, Step 1: Choose Best Client

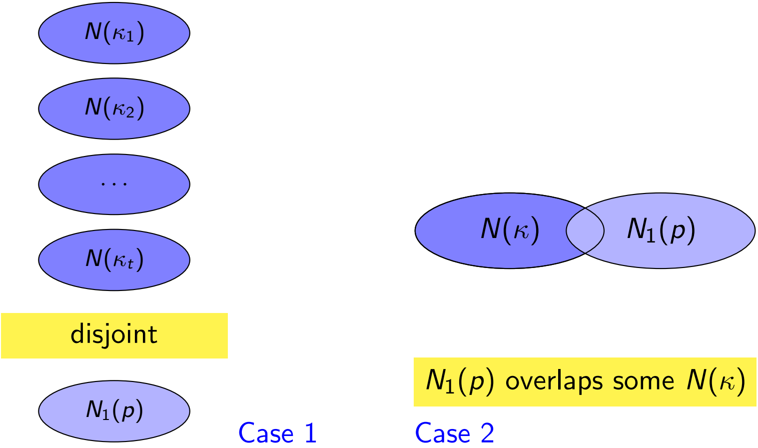
In each iteration, create one demand for best client



- $\text{bid}(j) = \text{avgdist}(N_1(j)) + \alpha_j^*(\text{dual value})$
- Best bid client p selected to create a demand

Phase 1, Step 2: Decide Neighborhood

Best client p creates demand ν, to decide $N(\nu)$, two cases:



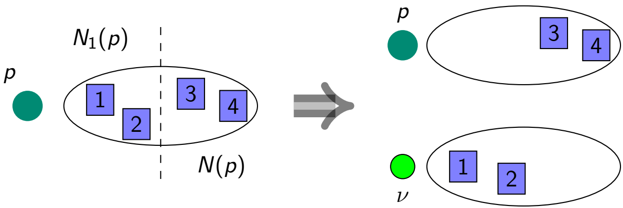
Phase 1, Step 2 Contd.

Best client p creates demand ν, to decide $N(\nu)$, two cases:

Phase 1, Step 2 Contd.

Best client p creates demand ν, to decide $N(\nu)$, two cases:

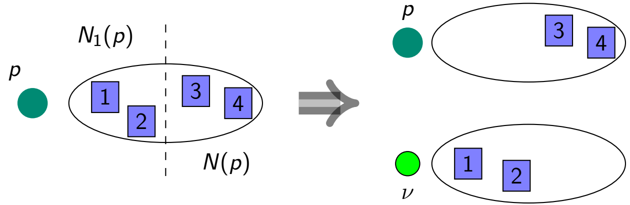
- Case 1: disjoint, $N(\nu)$ gets $N_1(p)$



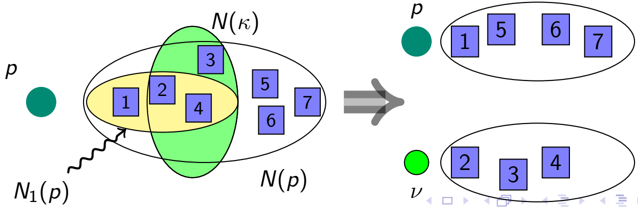
Phase 1, Step 2 Contd.

Best client p creates demand ν, to decide $N(\nu)$, two cases:

- Case 1: disjoint, $N(\nu)$ gets $N_1(p)$

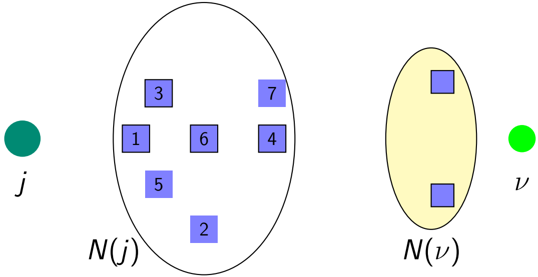


- Case 2: overlap, $N(\nu)$ gets $N(p) \cap N(\kappa)$



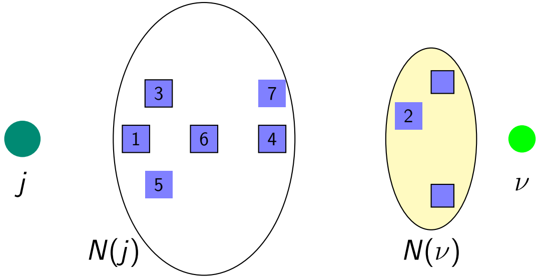
Phase 2

Add facilities from $N(j)$ to $N(\nu)$ until total value 1



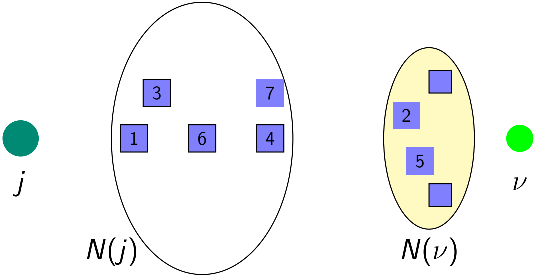
Phase 2

Add facilities from $N(j)$ to $N(\nu)$ until total value 1



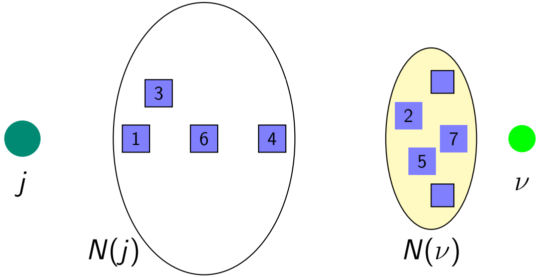
Phase 2

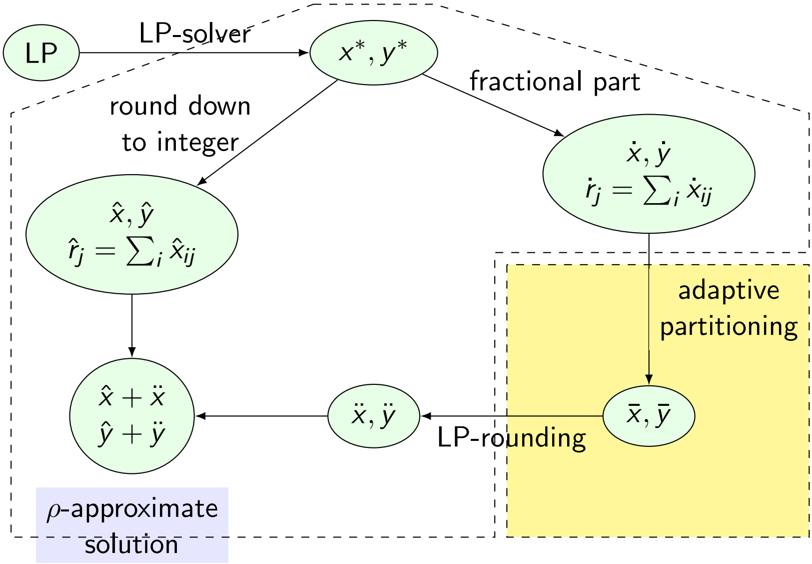
Add facilities from $N(j)$ to $N(\nu)$ until total value 1



Phase 2

Add facilities from $N(j)$ to $N(\nu)$ until total value 1





Done with partitioning, next to rounding

Table of Contents

3-Approximation for FTFP

Rounding: round each $\bar{y}_\mu$ and $\bar{x}_{\mu\nu}$ to 0 or 1

- Facilities: each primary κ opens one $\mu \in N(\kappa)$
- Connections: non-primary demands ν assigned to κ connect to μ

Analysis

- Fault-Tolerance: ν uses only facilities in $N(\nu) \cup N(\kappa)$
- Cost: $\leq 3 \cdot \mathrm{LP}^*$, because
  - Facility cost $\leq F^*$
  - Connection cost $\leq C^* + 2 \cdot \mathrm{LP}^*$

1.736-Approximation for FTFP

Rounding: round each $\bar{y}_\mu$ and $\bar{x}_{\mu\nu}$ to 0 or 1

- Facilities:
  - Each primary κ opens random $\mu \in N(\kappa)$
  - Other facilities open randomly independently
- Connections:
  - If a neighbor open, connect to nearest neighbor
  - Else connect via assigned primary demand

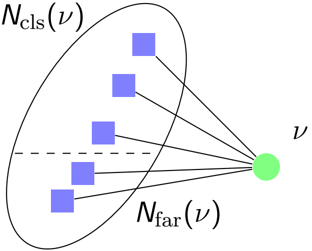
Analysis

- Fault-Tolerance: ν uses only facilities in $N(\nu) \cup N(\kappa)$
- Cost: $\leq (1 + 2/e)\,\mathrm{LP}^*$, because
  - Facility cost $\leq F^*$
  - Connection cost $\leq C^* + (2/e) \cdot \mathrm{LP}^*$

1.575-Approximation for FTFP — Partitioning

More intricate neighborhood structure

- Two neighborhoods: close and far, $N(\nu) = N_{\text{cls}}(\nu) \cup N_{\text{far}}(\nu)$
- $N_{\text{cls}}(\nu) =$ nearest $(1/\gamma)$-fraction of $N(\nu)$
- $N_{\text{cls}}(\nu) \cap N_{\text{cls}}(\kappa) \neq \emptyset$, if ν assigned to κ
- For siblings ν_1, ν_2, $N_{\text{cls}}(\kappa_1) \cup N(\nu_1)$ and $N_{\text{cls}}(\kappa_2) \cup N(\nu_2)$ disjoint
- ...



1.575-Approximation for FTFP — Rounding

Rounding: boost $(\mathbf{x}^*, \mathbf{y}^*)$ by γ and apply demand reduction and adaptive partitioning, then round by

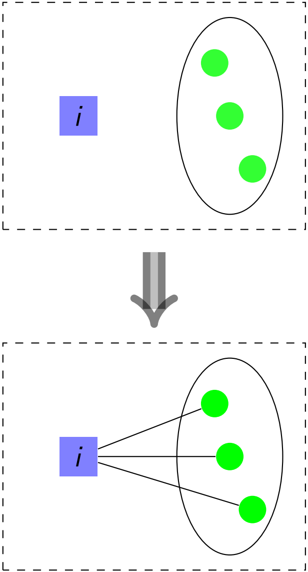
- Facilities:
  - Each primary κ opens random $\mu \in N_{\mathrm{cls}}(\kappa)$
  - Other facilities open randomly independently
- Connections:
  - If a neighbor open, connect to nearest neighbor
  - Else connect via assigned primary demand

Analysis

- Fault-Tolerance: ν uses only facilities in $N(\nu) \cup N_{\mathrm{cls}}(\kappa)$
- Cost: $\leq \gamma \cdot \mathrm{LP}$ for $\gamma = 1.575$, because
  - Facility cost $\leq \gamma \cdot F^*$
  - Connection cost $\leq \gamma \cdot C^*$

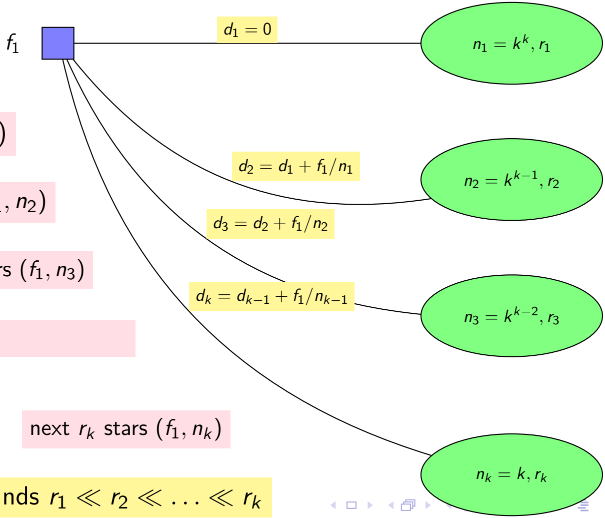
Greedy and Dual-fitting

- Greedy in polynomial time
  - Best star can be found quickly
  - Best star remains best
- Ratio H_n (Wolsey's result): Greedy is H_n-approx for
  - Minimizing a linear function
  - Subject to submodular constraints
- Lower bound $\Omega(\log n / \log\log n)$ for dual-fitting
  - Example has k groups, $n = k^k$
  - Shrinking factor is $k/2$



Dual-fitting Example

Dual constraints force a ratio of $k/2$, number of clients $n = k^k$



first r_1 stars (f_1, n_1)

next r_2 stars (f_1, n_2)

next r_3 stars (f_1, n_3)

next r_k stars (f_1, n_k)

demands $r_1 \ll r_2 \ll \ldots \ll r_k$

Table of Contents

Summary

Results

- 1.575-approximation algorithm for FTFP
- Technique for extending LP-rounding algorithms for UFL to FTFP

Summary

Results

- 1.575-approximation algorithm for FTFP
- Technique for extending LP-rounding algorithms for UFL to FTFP

Open Problems

- Can FTFL be approximated with the same ratio?
- LP-free algorithms for FTFP or FTFL with constant ratio?
- Close the 1.463 – 1.488 gap for UFL!