Cost-sharing methods in approximation algorithms

Stefano Leonardi

Sapienza University of Rome
ADFOCS 2008, August 18 - 22, MPI Saarbrücken

Metric Facility location

Input:

- undirected graph $G = (V, E)$
- non-negative edge costs $c : E \to \mathbb{R}^+$
- 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 cost satisfy triangle inequality

Goal: Compute

- set $F' \subseteq F$ of opened facilities; and
- function $\phi : \mathcal{D} \to \mathcal{F}'$ 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.

Connected facility location

Input:

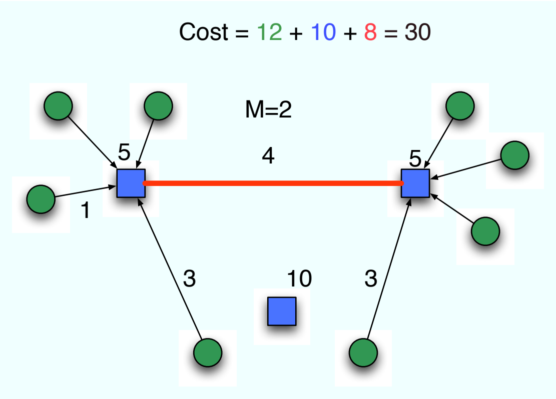
- Same as facility location; plus
- Parameter M

Goal: Compute

- set $F' \subseteq F$ of opened facilities; and
- function $\phi : \mathcal{D} \rightarrow \mathcal{F}'$ assigning cities to opened facilities; and
- Steiner tree T connecting the opened facilities that minimizes

$$\sum_{i \in F'} f_i + \sum_{j \in \mathcal{D}} c_{\phi(j)j} + M \sum_{e \in T} c_e$$

Example



Two facilities of cost 5 are opened and connected in a tree

Connected facility location

LP formulation:

Try all possible vertices facilities as root of the Steiner tree T.

$$
\begin{array}{lrcll}
\min & \sum_{i\in F} f_i y_i + \sum_{i\in F, j\in D} c_{ij}x_{ij} + M \sum_{e} c_e z_e & & & \\
\text{s.t.} & \sum_{i\in F} x_{ij} & \geq & 1 & j \in D \\
 & y_i - x_{ij} & \geq & 0 & i \in F, j \in D \\
 & y_v & = & 1 & \\
 & \sum_{i\in S} x_{ij} & \leq & \sum_{e\in\delta(S)} z_e & \forall S \subseteq V, v \notin S, j \\
 & x_{ij}, y_i, z_e & \in & \{0,1\} & i \in F, j \in D
\end{array}
$$

Primal-dual 9-apx [Swamy and Kumar, 2002].
Idea: Once a demand has contributed to open a facility, it starts paying for the Steiner cost.

Definition: Single-sink rent-or-buy

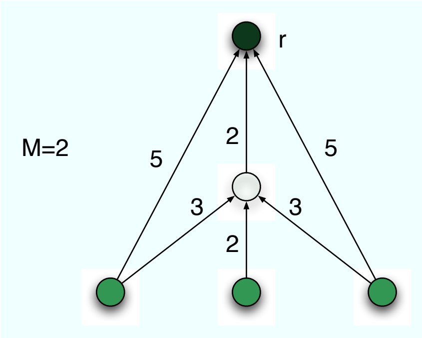
Input:

- Graph $G = (V, E)$, edge costs $c_e \geq 0$ for all $e \in E$
- root r
- Demand points $D = \{v_1, \ldots, v_k\} \subseteq V$
- Flows $f_1, \ldots, f_k$
  (here: assume $f_i = 1$ for all i)
- Economies of scale parameter $M \geq 1$

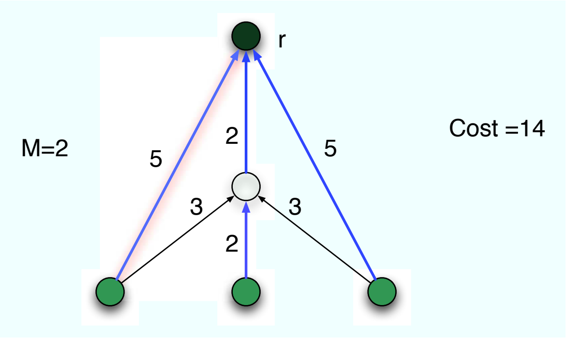
Goal: Find $E_b, E_r \subseteq E$ s.t.

- $F = E_b \cup E_r$ has an v_i, r-path for all i,
- $\sum_{e \in E_r} \lambda(e) \cdot c_e + \sum_{e \in E_b} M \cdot c_e$ is minimum

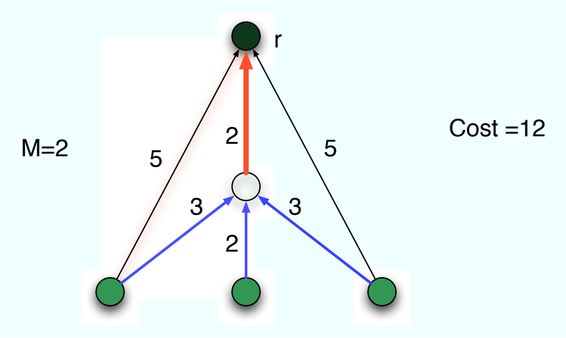
Example



Example



Example



Single-source Rent-or-buy network design

- SROB is a special case of Connected Facility Location
- Facilities have 0 opening cost
- Facilities can be opened at all vertices of the graph
- A 4.55 approximation primal-dual algorithm given in [Swamy and Kumar, 2002]
- Simple and elegant solution given in [Gupta, Kumar and Roughgarden, 2003] with 3.55 approximation
- Also applies to Multi-commodity rent-or-buy (later in this talk), CFL, Virtual Private Network design, Single-sink buy at bulk.

Special Cases

Steiner tree ($M = 1$):
Given a graph $G = (V, E)$, root r, k terminals $v_1, \ldots, v_k$ and non-negative edge costs c_e for all $e \in E$.
Find a minimum-cost tree T in G that contains an v_i, r-path for all i.

Shortest Paths ($M = \infty$):
An optimum solution will never buy any edge. Cheapest way of renting capacity f_i between s_i and t_i is along shortest path.

Special Cases

Steiner tree ($M = 1$):

Given a graph $G = (V, E)$, root r, k terminals $v_1, \ldots, v_k$ and non-negative edge costs c_e for all $e \in E$.
Find a minimum-cost tree T in G that contains an v_i, r-path for all i.

Shortest Paths ($M = \infty$):

An optimum solution will never buy any edge. Cheapest way of renting capacity f_i between s_i and t_i is along shortest path.

The GKR approach

Sample-Augment algorithm:

- Sample step Mark each demand with probability $1/M$
- Subproblem step Buy a forest F connecting the set of marked demands R
- Augmentation step Greedily rent capacity to produce a feasible solution.

GKR applied to SROB

Sample-Augment for SROB

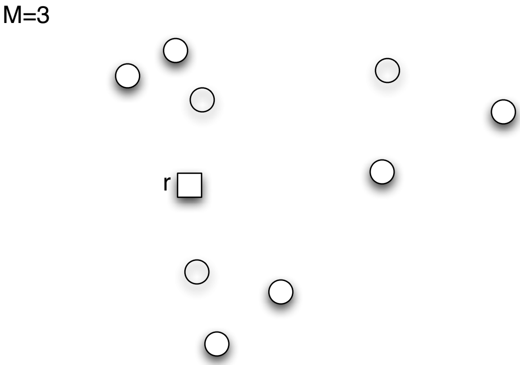
W.l.o.g, Consider demand flow $f_j = 1$.

- Sample step Mark each demand with probability $1/M$
- Subproblem step Buy a tree T connecting the set of marked demands R to the root r.
- Augmentation step Connect each demand in D/R to the closest vertex in T.

We separately bound:

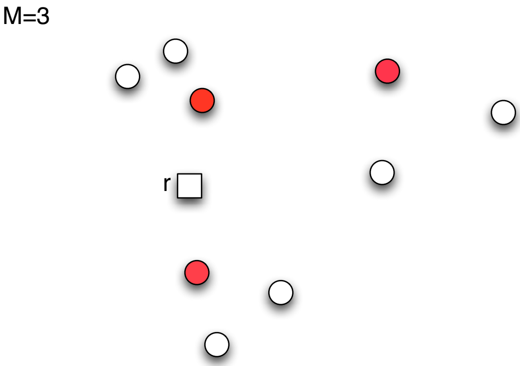
- Buying cost incurred in the Subproblem step
- Renting cost incurred in the Augmentation step

Sample-augment:Example



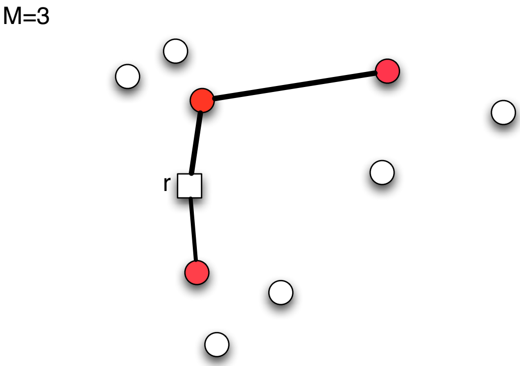
Demands sampled with pb $1/M$

Sample-augment:Example



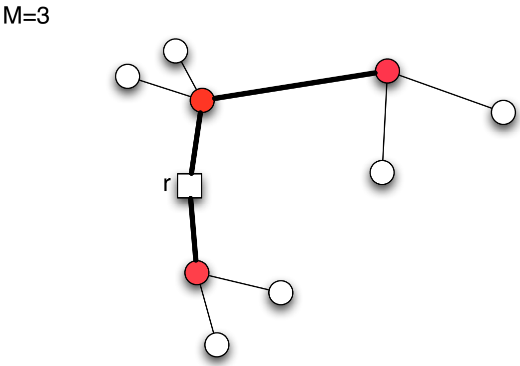
Three "facilities" openend

Sample-augment:Example



Build a Steiner tree over sampled demands

Sample-augment:Example



Connect demands to closest facilities

Approximation of SROB

Bounding the buying cost.

- T^{OPT}: Steiner tree in OPT spanning R^{OPT}.
- $OPT = M\ T^{OPT} + \sum_{v \in D/R^{OPT}} c(v, T^{OPT})$

Lemma

$E_R[T(R)] \leq OPT(D)$

Proof.

$E_R[T(R)] \leq M\ T^{OPT} + \sum_{v \in D/R^{OPT}} \frac{1}{M} M\ c(v, T^{OPT}) = OPT(D)$ □

Strict cost-shares

- We like to distribute in a fair manner between the demands the cost of the subproblem solution
- Every player should be charged proportionally to its contribution to the cost.

β-strictness

$\xi(v, R)$: cost share of vertex v on sapled set R.

Definition

Cost-shares $\xi(v, R)$ are β-strict if:

- $\sum_{v \in R} \xi(v, R) \leq T(R)$ competitiveness
- $c(v, T(R/v)) \leq \beta\xi(v, R)$ strictness

Strict cost-shares for SROB

Theorem

There exists 2-strict cost shares for Steiner tree.

- Let us run the Prim algorithm on the set of sampled demands
- MST is a 2-apx for Steiner tree.
- Let T_i be the tree constructed on the first i vertices selected by Prim's algorithm.
- If vertex v is connected by Prim at the $i+1$-th iteration, define $\xi_v(R) = \frac{1}{2}c(v, T_i)$.
- Prim's cost-shares are 2-strict for Steiner tree since:

$$c(v, T(R/v)) \leq c(v, T_i) \leq 2\xi_v(R).$$

Bounding the Renting cost

Proof.

- Renting cost $R_v = c(v, R)$ ($R_v = 0$ if $v \in R$.)
- Buying cost $B_v = M\xi(v, R)$ if $v \in R$ ($B_v = 0$ if $v \notin R$.)

Total buying cost: $\sum_{v \in D} B_v = \sum_{v \in R} M\xi(v, R) \leq M \; T(R)$

- Renting cost of v: $E[R_v|R] = (1 - \frac{1}{M})c(v, R)$
- Buying cost of v: $E[B_v|R] = \frac{1}{M}M\xi(v, R \cup v) = \xi(v, R \cup v)$
- It follows from β strictness: $E[R_v|R] \leq \beta E[B_v|R]$, and $E\left[\sum_{v \in D} R_v\right] \leq \beta \; E\left[\sum_{v \in D} B_v\right] \leq \beta \; E\left[M \; T(R)\right] \leq \beta OPT(D)$

□

Approximation via Cost-sharing

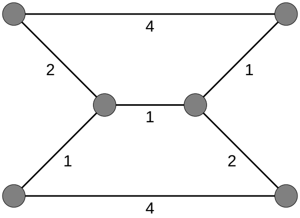
- One way to obtain strict cost shares is to add extra edges the solution of the subproblem.
- However, we like to obtain cost shares that are strict for a solution of good quality for the subproblem
- The approximation we achieve depends on the trade-off between the quality of the approximation to the subproblem and strictness

The strictness theorem:

Theorem

If there exist cost-shares that are competitive and β-strict for an α-approximate algorithm, then Sample-augment is $\alpha + \beta$-approximated.

Multi-commodity rent-or-buy (MROB)

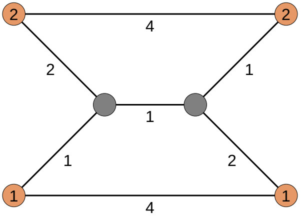


Given:

- Network $G = (V, E)$ with edge costs c_e for all $e \in E$
- Terminal pairs $(s_1, t_1), \ldots, (s_k, t_k)$
- Each terminal pair (s_i, t_i) wants to send f_i units of flow from s_i to t_i

Goal: Install capacities on edges such that all flows f_i can be routed simultaneously

Multi-commodity rent-or-buy (MROB)

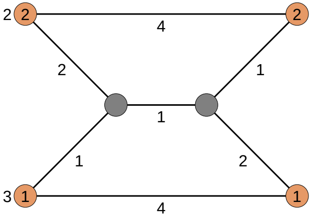


Given:

- Network $G = (V, E)$ with edge costs c_e for all $e \in E$
- Terminal pairs $(s_1, t_1), \ldots, (s_k, t_k)$
- Each terminal pair (s_i, t_i) wants to send f_i units of flow from s_i to t_i

Goal: Install capacities on edges such that all flows f_i can be routed simultaneously

Multi-commodity rent-or-buy (MROB)

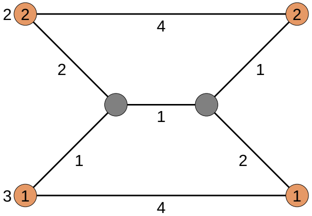


Given:

- Network $G = (V, E)$ with edge costs c_e for all $e \in E$
- Terminal pairs $(s_1, t_1), \ldots, (s_k, t_k)$
- Each terminal pair (s_i, t_i) wants to send f_i units of flow from s_i to t_i

Goal: Install capacities on edges such that all flows f_i can be routed simultaneously

Multi-commodity rent-or-buy (MROB)

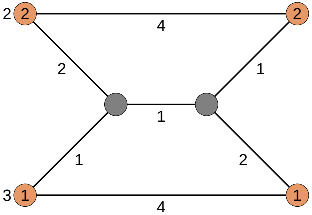


Given:

- Network $G = (V, E)$ with edge costs c_e for all $e \in E$
- Terminal pairs $(s_1, t_1), \ldots, (s_k, t_k)$
- Each terminal pair (s_i, t_i) wants to send f_i units of flow from s_i to t_i

Goal: Install capacities on edges such that all flows f_i can be routed simultaneously

MROB



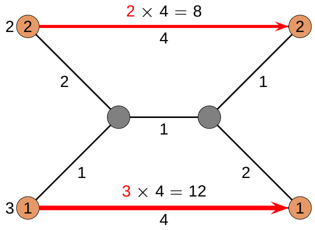
Rent-or-Buy: On each edge e

- we can either rent capacity $\lambda(e)$ at cost $\lambda(e) \cdot c_e$,
- or buy infinite capacity at cost $M \cdot c_e$

Example: $M = 4$

- Cost of capacity installation: 20
- Cost of capacity installation: 19

MROB



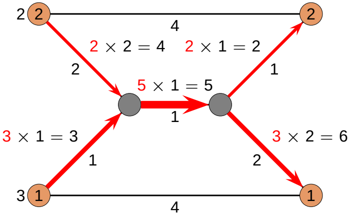
Rent-or-Buy: On each edge e

- we can either rent capacity $\lambda(e)$ at cost $\lambda(e) \cdot c_e$,
- or buy infinite capacity at cost $M \cdot c_e$

Example: $M = 4$

- Cost of capacity installation: 20
- Cost of capacity installation: 19

MROB



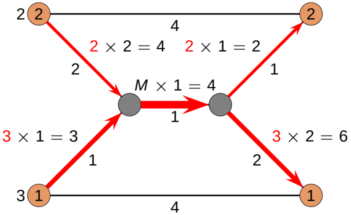
Rent-or-Buy: On each edge e

- we can either rent capacity $\lambda(e)$ at cost $\lambda(e) \cdot c_e$,
- or buy infinite capacity at cost $M \cdot c_e$

Example: $M = 4$

- Cost of capacity installation: 20
- Cost of capacity installation: 19

MROB



Rent-or-Buy: On each edge e

- we can either rent capacity $\lambda(e)$ at cost $\lambda(e) \cdot c_e$,
- or buy infinite capacity at cost $M \cdot c_e$

Example: $M = 4$

- Cost of capacity installation: 20
- Cost of capacity installation: 19

Definition: Multicommodity Rent-or-Buy

Input:

- Graph $G = (V, E)$, edge costs $c_e \geq 0$ for all $e \in E$
- Terminal pairs $R = \{(s_1, t_1), \ldots, (s_k, t_k)\} \subseteq V \times V$
- Flows $f_1, \ldots, f_k$
  (here: assume $f_i = 1$ for all i)
- Economies of scale parameter $M \geq 1$

Goal: Find $E_b, E_r \subseteq E$ s.t.

- $F = E_b \cup E_r$ has an s_i, t_i-path for all i,
- $\sum_{e \in E_r} \lambda(e) \cdot c_e + \sum_{e \in E_b} M \cdot c_e$ is minimum

Special Cases

Steiner Forests ($M = 1$):

Given a graph $G = (V, E)$, k terminal pairs $(s_1, t_1), \ldots, (s_k, t_k)$ and non-negative edge costs c_e for all $e \in E$.

Find a minimum-cost forest F in G that contains an s_i, t_i-path for all i.

Special Cases

Steiner Forests ($M = 1$):
Given a graph $G = (V, E)$, k terminal pairs $(s_1, t_1), \dots, (s_k, t_k)$ and non-negative edge costs c_e for all $e \in E$.
Find a minimum-cost forest F in G that contains an s_i, t_i-path for all i.

Results on MROB

Multicommodity Rent-or-Buy

Kumar, Gupta, Roughgarden ’02	O(1)
Gupta, Kumar, Pál, Roughgarden ’03	12
Becchetti, Könemann, L., Pál ’05	6.82

Theorem

There is a 5-apx for the multicommodity rent-or-buy problem. Fleischer, Könemann, L., Schäfer’06

Key features:

- Use the framework of Gupta et al. ’03.
- Alternate view of Steiner forest algorithm by Agrawal, Klein and Ravi ’95 gives much simpler analysis.

Results on MROB

Multicommodity Rent-or-Buy

Kumar, Gupta, Roughgarden '02	O(1)
Gupta, Kumar, Pál, Roughgarden '03	12
Becchetti, Könemann, L., Pál '05	6.82

Theorem

There is a 5-apx for the multicommodity rent-or-buy problem.
Fleischer, Könemann, L., Schäfer'06

Key features:

- Use the framework of Gupta et al. '03.
- Alternate view of Steiner forest algorithm by Agrawal, Klein and Ravi '95 gives much simpler analysis.

Results on MROB

Multicommodity Rent-or-Buy

Kumar, Gupta, Roughgarden '02	O(1)
Gupta, Kumar, Pál, Roughgarden '03	12
Becchetti, Könemann, L., Pál '05	6.82

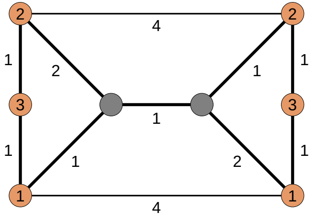
Theorem

There is a 5-apx for the multicommodity rent-or-buy problem.
Fleischer, Könemann, L., Schäfer'06

Key features:

- Use the framework of Gupta et al. '03.
- Alternate view of Steiner forest algorithm by Agrawal, Klein and Ravi '95 gives much simpler analysis.

Sample Augment for MROB[Gupta et al. ’03]



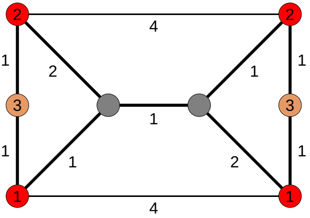
$M = 3$

1: Mark each terminal pair with probability $1/M$. Marked terminal pairs: D.
2: Buy the edges of a Steiner forest E_b for D.
3: Rent cheapest set E_r s.t. $F = E_b \cup E_r$ is feasible.

Total cost

$M \cdot c(E_b) + \sum_{e \in E_r} \lambda(F, e) c_e = 23.$

Sample Augment for MROB[Gupta et al. '03]



$M = 3$, ●: D

1: Mark each terminal pair with probability $1/M$. Marked terminal pairs: D.

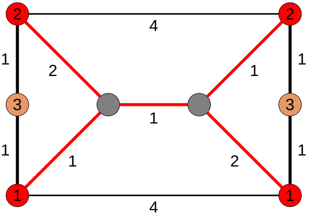
2: Buy the edges of a Steiner forest E_b for D.

3: Rent cheapest set E_r s.t. $F = E_b \cup E_r$ is feasible.

Total cost

$M \cdot c(E_b) + \sum_{e \in E_r} \lambda(F, e) c_e = 23.$

Sample Augment for MROB[Gupta et al. '03]



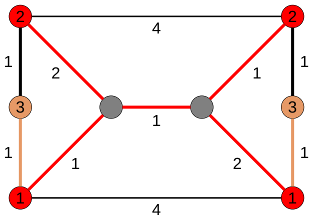
$M = 3$, ●: D , –: E_b

1: Mark each terminal pair with probability $1/M$. Marked terminal pairs: D.
2: Buy the edges of a Steiner forest E_b for D.
3: Rent cheapest set E_r s.t. $F = E_b \cup E_r$ is feasible.

Total cost

$M \cdot c(E_b) + \sum_{e \in E_r} \lambda(F, e) c_e = 23.$

Sample Augment for MROB[Gupta et al. '03]



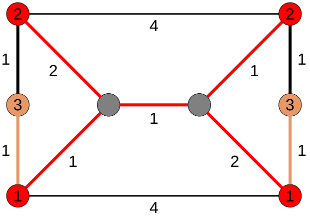
$M = 3$, ● : D , – : E_b

1: Mark each terminal pair with probability $1/M$. Marked terminal pairs: D.
2: Buy the edges of a Steiner forest E_b for D.
3: Rent cheapest set E_r s.t. $F = E_b \cup E_r$ is feasible.

Total cost

$M \cdot c(E_b) + \sum_{e \in E_r} \lambda(F, e) c_e = 23.$

Sample Augment for MROB[Gupta et al. '03]



$M = 3$, ● : D , – : E_b

1: Mark each terminal pair with probability $1/M$. Marked terminal pairs: D.
2: Buy the edges of a Steiner forest E_b for D.
3: Rent cheapest set E_r s.t. $F = E_b \cup E_r$ is feasible.

Total cost

$M \cdot c(E_b) + \sum_{e \in E_r} \lambda(F, e) c_e = 23.$

Sample Augment for MRoB

SimpleMRoB [Gupta et al. '03]:

1: Mark each terminal pair with probability $1/M$. Let D be set of marked terminal pairs.
2: Compute (approximate) Steiner forest $F' = E_b$ for D and buy all edges in E_b.
3: For all terminal pairs $(s, t) \notin D$: Rent unit capacity on shortest s, t-path in contracted graph $G|F'$.

Sample Augment for MRoB

SimpleMRoB [Gupta et al. '03]:

1: Mark each terminal pair with probability $1/M$. Let D be set of marked terminal pairs.
2: Compute (approximate) Steiner forest $F' = E_b$ for D and buy all edges in E_b.
3: For all terminal pairs $(s, t) \notin D$: Rent unit capacity on shortest s, t-path in contracted graph $G|F'$.

Sample Augment for MRoB

SimpleMRoB [Gupta et al. '03]:

1: Mark each terminal pair with probability $1/M$. Let D be set of marked terminal pairs.
2: Compute (approximate) Steiner forest $F' = E_b$ for D and buy all edges in E_b.
3: For all terminal pairs $(s,t) \notin D$: Rent unit capacity on shortest s,t-path in contracted graph $G|F'$.

Sample Augment for MRoB

SimpleMRoB [Gupta et al. '03]:

1: Mark each terminal pair with probability $1/M$. Let D be set of marked terminal pairs.
2: Compute (approximate) Steiner forest $F' = E_b$ for D and buy all edges in E_b.
3: For all terminal pairs $(s, t) \notin D$: Rent unit capacity on shortest s, t-path in contracted graph $G|F'$.

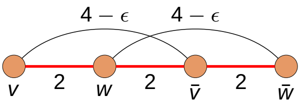
A Randomized Framework for MRoB

Theorem

Given an α-approximate and β-strict Steiner forest algorithm, SimpleMRoB returns a feasible solution $F = E_r \cup E_b$ such that

$$E\left[\sum_{e \in E_r} \lambda(e) \cdot c_e + \sum_{e \in E_b} M \cdot c_e\right] \leq (\alpha + \beta) \cdot \texttt{opt}.$$

Concept: Cost-Sharing



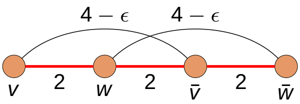
- An example with 2 terminal pairs $R = \{(v, \bar{v}), (w, \bar{w})\}$.
- Steiner forest returned by standard primal-dual algorithm AKR is $v, \bar{w}$-path.

$$\xi(v,\bar{v}) =$$
$$\xi(w,\bar{w}) =$$

Cost-Sharing Method:
Want algorithm to compute cost-share $\xi(u,\bar{u})$ for all $(u,\bar{u}) \in R$ s.t.

$$\sum_{(u,\bar{u})\in R} \xi(u,\bar{u}) \le \mathtt{opt}_R$$

Concept: Cost-Sharing



- An example with 2 terminal pairs $R = \{(v, \bar{v}), (w, \bar{w})\}$.
- Steiner forest returned by standard primal-dual algorithm AKR is $v, \bar{w}$-path.

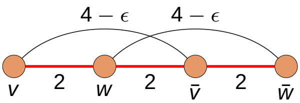
$$\xi(v, \bar{v}) \quad = $$
$$\xi(w, \bar{w}) \quad = $$

Cost-Sharing Method:

Want algorithm to compute cost-share $\xi(u, \bar{u})$ for all $(u, \bar{u}) \in R$ s.t.

$$\sum_{(u,\bar{u}) \in R} \xi(u, \bar{u}) \leq \texttt{opt}_R$$

Concept: Cost-Sharing



- An example with 2 terminal pairs $R = \{(v, \bar{v}), (w, \bar{w})\}$.
- Steiner forest returned by standard primal-dual algorithm AKR is $v, \bar{w}$-path.

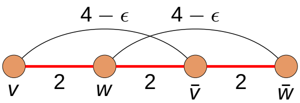
$$\begin{aligned} \xi(v, \bar{v}) &= 3 \\ \xi(w, \bar{w}) &= 3 \end{aligned}$$

Cost-Sharing Method:

Want algorithm to compute cost-share $\xi(u, \bar{u})$ for all $(u, \bar{u}) \in R$ s.t.

$$\sum_{(u,\bar{u}) \in R} \xi(u, \bar{u}) \leq \texttt{opt}_R$$

Concept: Cost-Sharing



- An example with 2 terminal pairs $R = \{(v, \bar{v}), (w, \bar{w})\}$.
- Steiner forest returned by standard primal-dual algorithm AKR is $v, \bar{w}$-path.

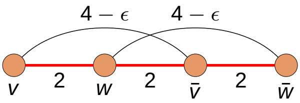
$$\xi(v, \bar{v}) = 1$$
$$\xi(w, \bar{w}) = 4$$

Cost-Sharing Method:

Want algorithm to compute cost-share $\xi(u, \bar{u})$ for all $(u, \bar{u}) \in R$ s.t.

$$\sum_{(u,\bar{u}) \in R} \xi(u, \bar{u}) \leq \mathtt{opt}_R$$

Concept: Strictness



$$\xi(v,\bar{v}) = 3$$
$$\xi(w,\bar{w}) = 3$$

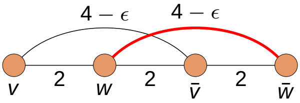
Notation:

- $R_{-u\bar{u}}$: all pairs except $(u,\bar{u})$
- $F_{-u\bar{u}}$: AKR forest for $R_{-u\bar{u}}$.

  Ex: $F_{-v\bar{v}}$.
- $c_{G|F_{-u\bar{u}}}(z,\bar{z})$: min-cost $z,\bar{z}$-path in G when edges in $F_{-u\bar{u}}$ are contracted.

  Ex: $c_{G|F_{-v,\bar{v}}}(v,\bar{v}) = 4-\epsilon$

Concept: Strictness



$$\xi(v,\bar{v}) = 3$$
$$\xi(w,\bar{w}) = 3$$

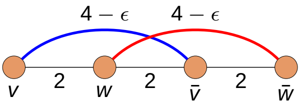
Notation:

- $R_{-u\bar{u}}$: all pairs except $(u,\bar{u})$
- $F_{-u\bar{u}}$: AKR forest for $R_{-u\bar{u}}$.

  Ex: $F_{-v\bar{v}}$.
- $c_{G|F_{-u\bar{u}}}(z,\bar{z})$: min-cost $z,\bar{z}$-path in G when edges in $F_{-u\bar{u}}$ are contracted.

  Ex: $c_{G|F_{-v,\bar{v}}}(v,\bar{v}) = 4 - \epsilon$

Concept: Strictness



$$\xi(v,\bar{v}) = 3$$
$$\xi(w,\bar{w}) = 3$$

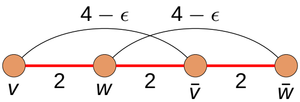
Notation:

- $R_{-u\bar{u}}$: all pairs except $(u,\bar{u})$
- $F_{-u\bar{u}}$: AKR forest for $R_{-u\bar{u}}$.

  Ex: $F_{-v\bar{v}}$.
- $c_{G|F_{-u\bar{u}}}(z,\bar{z})$: min-cost $z,\bar{z}$-path in G when edges in $F_{-u\bar{u}}$ are contracted.

  Ex: $c_{G|F_{-v,\bar{v}}}(v,\bar{v}) = 4-\epsilon$

Concept: Strictness



$$\xi(v,\bar{v}) = 3$$
$$\xi(w,\bar{w}) = 3$$

Definition: Cost-shares ξ are β-strict if

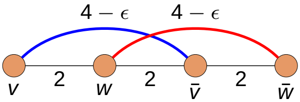
$$c_{G|F_{-u\bar{u}}}(u,\bar{u}) \le \beta \cdot \xi_{u,\bar{u}}$$

for all $(u,\bar{u}) \in R$.

$$c_{G|F_{-v\bar{v}}}(v,\bar{v}) = 4-\epsilon \le \frac{4}{3}\xi(v,\bar{v})$$
$$c_{G|F_{-w\bar{w}}}(w,\bar{w}) = 4-\epsilon \le \frac{4}{3}\xi(w,\bar{w})$$

Cost-shares in this example are $\frac{4}{3}$-strict.

Concept: Strictness



$$\xi(v,\bar{v}) = 3$$
$$\xi(w,\bar{w}) = 3$$

Definition: Cost-shares ξ are β-strict if

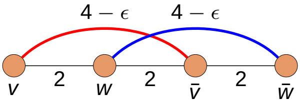
$$c_{G|F_{-u\bar{u}}}(u,\bar{u}) \leq \beta \cdot \xi_{u,\bar{u}}$$

for all $(u,\bar{u}) \in R$.

$$c_{G|F_{-v\bar{v}}}(v,\bar{v}) = 4-\epsilon \leq \frac{4}{3}\xi(v,\bar{v})$$
$$c_{G|F_{-w\bar{w}}}(w,\bar{w}) = 4-\epsilon \leq \frac{4}{3}\xi(w,\bar{w})$$

Cost-shares in this example are $\frac{4}{3}$-strict.

Concept: Strictness



$$\xi(v,\bar{v}) = 3$$
$$\xi(w,\bar{w}) = 3$$

Definition: Cost-shares ξ are β-strict if

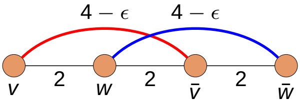
$$c_{G|F_{-u\bar{u}}}(u,\bar{u}) \leq \beta \cdot \xi_{u,\bar{u}}$$

for all $(u,\bar{u}) \in R$.

$$c_{G|F_{-v\bar{v}}}(v,\bar{v}) = 4-\epsilon \leq \frac{4}{3}\xi(v,\bar{v})$$
$$c_{G|F_{-w\bar{w}}}(w,\bar{w}) = 4-\epsilon \leq \frac{4}{3}\xi(w,\bar{w})$$

Cost-shares in this example are $\frac{4}{3}$-strict.

Concept: Strictness



$$\xi(v,\bar{v}) = 3$$
$$\xi(w,\bar{w}) = 3$$

Definition: Cost-shares ξ are β-strict if

$$c_{G|F_{-u\bar{u}}}(u,\bar{u}) \leq \beta \cdot \xi_{u,\bar{u}}$$

for all $(u,\bar{u}) \in R$.

$$c_{G|F_{-v\bar{v}}}(v,\bar{v}) = 4-\epsilon \leq \frac{4}{3}\xi(v,\bar{v})$$
$$c_{G|F_{-w\bar{w}}}(w,\bar{w}) = 4-\epsilon \leq \frac{4}{3}\xi(w,\bar{w})$$

Cost-shares in this example are $\frac{4}{3}$-strict.

Concept: Strictness

Definition

A Steiner forest algorithm AKR is β-strict if it returns a cost-share ξ_{st} for all $(s,t) \in R$ such that

1. $\sum_{(s,t)\in R} \xi_{st} \leq c(F^*)$
2. For any $(s,t) \in R$, $c_{G|F_{-st}}(s,t) \leq \beta \cdot \xi_{st}$

Notation:

- $F^* =$ min-cost Steiner forest for R
- $F_{-st} =$ apx Steiner forest for $R_{-st} = R \setminus \{(s,t)\}$ computed by AKR
- $G|F_{-st} =$ graph obtained if all components of F_{-st} are contracted.

Concept: Strictness

Definition

A Steiner forest algorithm AKR is β-strict if it returns a cost-share ξ_{st} for all $(s,t) \in R$ such that

1. $\sum_{(s,t)\in R} \xi_{st} \le c(F^*)$
2. For any $(s,t) \in R$, $c_{G|F_{-st}}(s,t) \le \beta \cdot \xi_{st}$

Notation:

- $F^* =$ min-cost Steiner forest for R
- $F_{-st} =$ apx Steiner forest for $R_{-st} = R \setminus \{(s,t)\}$ computed by AKR
- $G|F_{-st} =$ graph obtained if all components of F_{-st} are contracted.

Concept: Strictness

Definition

A Steiner forest algorithm AKR is β-strict if it returns a cost-share ξ_{st} for all $(s,t) \in R$ such that

1. $\sum_{(s,t)\in R} \xi_{st} \leq c(F^*)$
2. For any $(s,t) \in R$, $c_{G|F_{-st}}(s,t) \leq \beta \cdot \xi_{st}$

Notation:

- $F^* =$ min-cost Steiner forest for R
- $F_{-st} =$ apx Steiner forest for $R_{-st} = R \setminus \{(s,t)\}$ computed by AKR
- $G|F_{-st} =$ graph obtained if all components of F_{-st} are contracted.

Strictness: Once more...

- Run AKR on R to compute cost-shares ξ_{st} for all $(s,t) \in R$
- Cost-shares ξ_{st} must satisfy $\sum_{(s,t)\in R} \xi_{st} \leq c(F^*)$
- Pick an arbitrary terminal pair $(s,t) \in R$ and let $R_{-st} = R \setminus \{(s,t)\}$
- Run AKR on R_{-st} and let F_{-st} be the computed solution
- AKR β-strict implies:
  shortest s,t-path in $G|F_{-st}$ has cost at most $\beta \cdot \xi_{st}$

Strictness: Once more...

- Run AKR on R to compute cost-shares ξ_{st} for all $(s,t) \in R$
- Cost-shares ξ_{st} must satisfy $\sum_{(s,t)\in R} \xi_{st} \leq c(F^*)$
- Pick an arbitrary terminal pair $(s,t) \in R$ and let $R_{-st} = R \setminus \{(s,t)\}$
- Run AKR on R_{-st} and let F_{-st} be the computed solution
- AKR β-strict implies:
  shortest s,t-path in $G|F_{-st}$ has cost at most $\beta \cdot \xi_{st}$

Strictness: Once more...

- Run AKR on R to compute cost-shares ξ_{st} for all $(s,t) \in R$
- Cost-shares ξ_{st} must satisfy $\sum_{(s,t)\in R} \xi_{st} \leq c(F^*)$
- Pick an arbitrary terminal pair $(s,t) \in R$ and let $R_{-st} = R \setminus \{(s,t)\}$
- Run AKR on R_{-st} and let F_{-st} be the computed solution
- AKR β-strict implies:
  shortest s,t-path in $G|F_{-st}$ has cost at most $\beta \cdot \xi_{st}$

Strictness: Once more...

- Run AKR on R to compute cost-shares ξ_{st} for all $(s,t) \in R$
- Cost-shares ξ_{st} must satisfy $\sum_{(s,t)\in R} \xi_{st} \leq c(F^*)$
- Pick an arbitrary terminal pair $(s,t) \in R$ and let $R_{-st} = R \setminus \{(s,t)\}$
- Run AKR on R_{-st} and let F_{-st} be the computed solution
- AKR β-strict implies: shortest s,t-path in $G|F_{-st}$ has cost at most $\beta \cdot \xi_{st}$

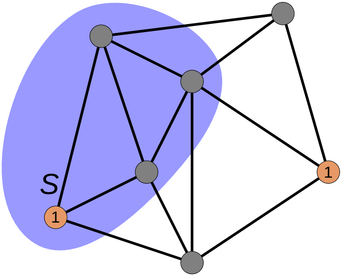
Strictness: Once more...

- Run AKR on R to compute cost-shares ξ_{st} for all $(s,t) \in R$
- Cost-shares ξ_{st} must satisfy $\sum_{(s,t)\in R} \xi_{st} \le c(F^*)$
- Pick an arbitrary terminal pair $(s,t) \in R$ and let $R_{-st} = R \setminus \{(s,t)\}$
- Run AKR on R_{-st} and let F_{-st} be the computed solution
- AKR β-strict implies:
  shortest s,t-path in $G|F_{-st}$ has cost at most $\beta \cdot \xi_{st}$

Remainder of this Lecture

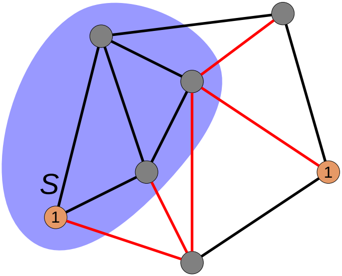
Show that the standard primal-dual algorithm for Steiner forests due to Agrawal, Klein and Ravi is 2-approximate and 4-strict.

Steiner Cuts



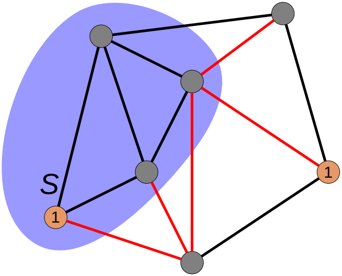
- A subset $S \subseteq V$ is called a Steiner cut if S separates at least one terminal pair
- Every feasible Steiner forest needs to have one edge crossing every Steiner cut
- Use $\mathcal{U}$ for the set of all Steiner cuts

Steiner Cuts



- A subset $S \subseteq V$ is called a Steiner cut if S separates at least one terminal pair
- Every feasible Steiner forest needs to have one edge crossing every Steiner cut
- Use $\mathcal{U}$ for the set of all Steiner cuts

Steiner Cuts



- A subset $S \subseteq V$ is called a Steiner cut if S separates at least one terminal pair
- Every feasible Steiner forest needs to have one edge crossing every Steiner cut
- Use $\mathcal{U}$ for the set of all Steiner cuts

Undirected Cut Relaxation

Primal LP Relaxation:

$$\texttt{min} \quad \sum_{e \in E} c_e \cdot x_e$$

$$\text{s.t.} \quad \sum_{e \in \delta(U)} x_e \geq 1 \quad \forall U \in \mathcal{U}$$

$$x_e \geq 0 \quad \forall e \in E$$

Dual LP:

$$\texttt{max} \quad \sum_{U \in \mathcal{U}} y_U$$

$$\text{s.t.} \quad \sum_{U : e \in \delta(U)} y_U \leq c_e \quad \forall e \in E$$

$$y_U \geq 0 \quad \forall U \in \mathcal{U}$$

($\delta(U)$: Edges in the cut defined by U)

Undirected Cut Relaxation

Primal LP Relaxation:

$$\min \sum_{e \in E} c_e \cdot x_e$$

$$\text{s.t.} \quad \sum_{e \in \delta(U)} x_e \geq 1 \quad \forall U \in \mathcal{U}$$

$$x_e \geq 0 \quad \forall e \in E$$

Dual LP:

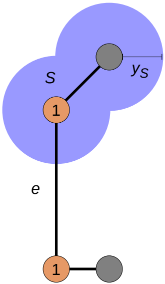
$$\max \sum_{U \in \mathcal{U}} y_U$$

$$\text{s.t.} \quad \sum_{U : e \in \delta(U)} y_U \leq c_e \quad \forall e \in E$$

$$y_U \geq 0 \quad \forall U \in \mathcal{U}$$

($\delta(U)$: Edges in the cut defined by U)

Visualizing the Dual

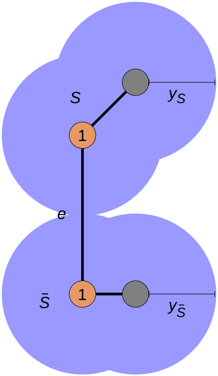


- The dual y_S of Steiner-cut S is visualized as moat around S of radius y_S
- The dual constraint for edge e is tight if

$$\sum_{U : e \in \delta(U)} y_U = c_e$$

Here: $y_S + y_{\bar{S}} = c_e$

Visualizing the Dual



- The dual y_S of Steiner-cut S is visualized as moat around S of radius y_S
- The dual constraint for edge e is tight if

$$\sum_{U:e\in\delta(U)} y_U = c_e$$

Here: $y_S + y_{\bar{S}} = c_e$

Primal-Dual Algorithm

- Algorithm starts with an empty (infeasible) primal solution F and dual (feasible) solution $y_U = 0$ for all Steiner cuts $U \in \mathcal{U}$
- Goal: Compute feasible primal/dual pair (F, y) such that cost of F is bounded within dual objective function value, e.g.,

$$c(F) \leq \alpha \cdot \sum_{U \in \mathcal{U}} y_U$$

- By weak duality, computed solution F is α-approximate Steiner forest

Primal-Dual Algorithm

- Algorithm starts with an empty (infeasible) primal solution F and dual (feasible) solution $y_U = 0$ for all Steiner cuts $U \in \mathcal{U}$
- Goal: Compute feasible primal/dual pair (F, y) such that cost of F is bounded within dual objective function value, e.g.,

$$c(F) \leq \alpha \cdot \sum_{U \in \mathcal{U}} y_U$$

- By weak duality, computed solution F is α-approximate Steiner forest

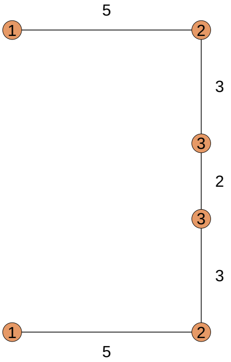
Primal-Dual Algorithm

- Algorithm starts with an empty (infeasible) primal solution F and dual (feasible) solution $y_U = 0$ for all Steiner cuts $U \in \mathcal{U}$
- Goal: Compute feasible primal/dual pair (F, y) such that cost of F is bounded within dual objective function value, e.g.,

$$c(F) \leq \alpha \cdot \sum_{U \in \mathcal{U}} y_U$$

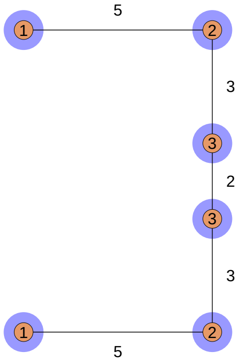
- By weak duality, computed solution F is α-approximate Steiner forest

Primal-Dual Algorithm



- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

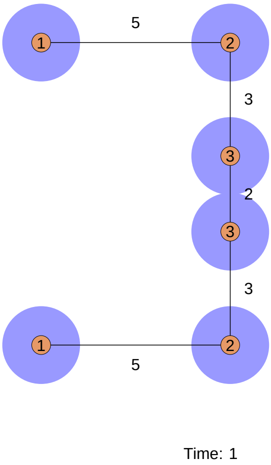
Primal-Dual Algorithm



Time: .5

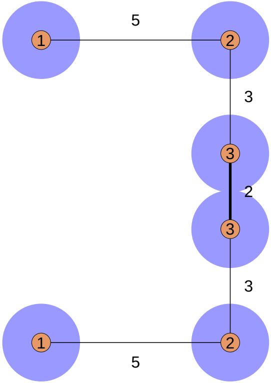
- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

Primal-Dual Algorithm



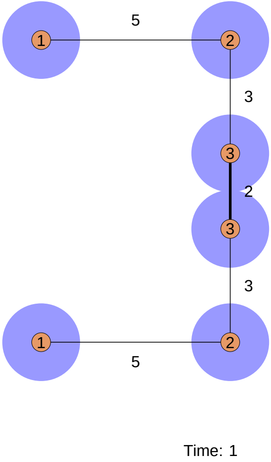
- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

Primal-Dual Algorithm



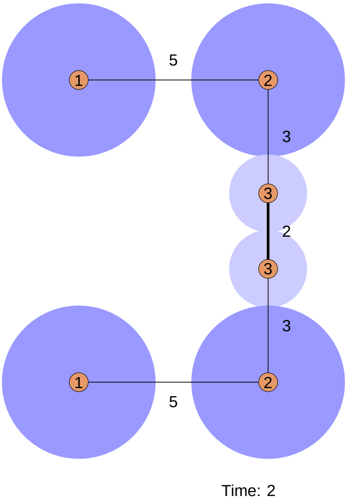
- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

Primal-Dual Algorithm



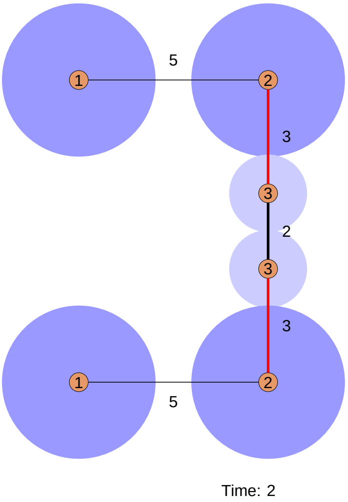
- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

Primal-Dual Algorithm



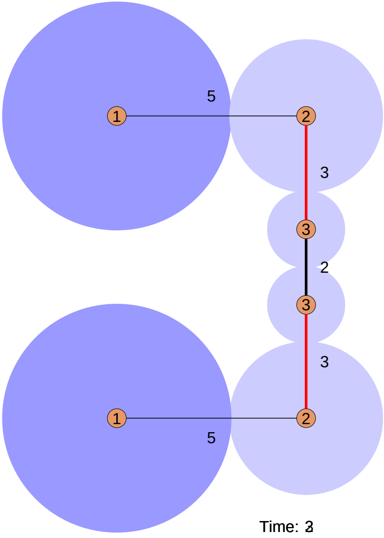
- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

Primal-Dual Algorithm



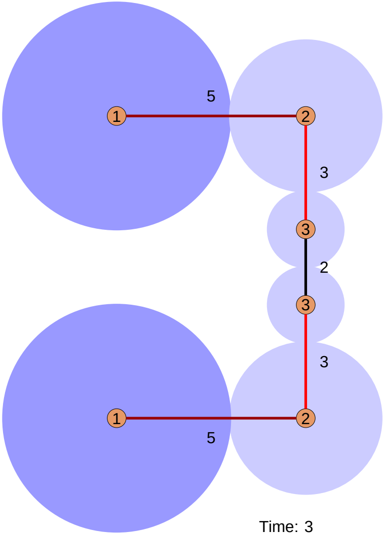
- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

Primal-Dual Algorithm



- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

Primal-Dual Algorithm



- Initially: Raise duals for all singleton Steiner cuts simultaneously... until some edge/path becomes tight
- Add tight segment to F
- Terminal is active if it is separated from its mate
- Raise the duals of active connected components

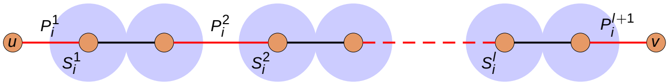
Approximation Guarantee

Theorem (Agrawal, Klein, Ravi '95)

The cost of the computed forest F is

$$c(F) \leq 2 \cdot \sum_{U \in \mathcal{U}} y_U \leq 2 \cdot \texttt{opt}$$

Primal-Dual Algorithm: Different View

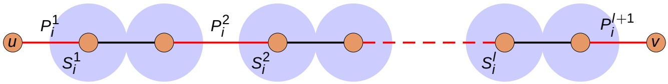


- Can view execution of algorithm AKR as picking paths

$$P_1, \ldots, P_q$$

- When path P_i becomes tight
  - passes through inactive moats $S_i^1, \ldots, S_i^l$
  - segments $P_i^1, \ldots, P_i^{l+1}$ are added to existing forest

Primal-Dual Algorithm: Different View



- Can view execution of algorithm AKR as picking paths

$$P_1, \ldots, P_q$$

- When path P_i becomes tight
  - passes through inactive moats $S_i^1, \ldots, S_i^l$
  - segments $P_i^1, \ldots, P_i^{l+1}$ are added to existing forest

Adding Strictness

1. Need to compute cost-shares ξ_{st} for all $(s,t) \in R$ such that

$$\sum_{(s,t)\in R} \xi_{st} \leq \texttt{opt}$$

- Final forest $F = P_1 \cup \ldots \cup P_q$ has cost at most $2 \cdot \texttt{opt}$
- Whenever a path P_i becomes tight, can distribute half of the cost of the added segments as cost-share
- This implies:
  Total cost-share distributed is $\frac{1}{2}c(F) \leq \texttt{opt}$.

Adding Strictness

1. Need to compute cost-shares ξ_{st} for all $(s,t) \in R$ such that

$$\sum_{(s,t)\in R} \xi_{st} \leq \texttt{opt}$$

- Final forest $F = P_1 \cup \ldots \cup P_q$ has cost at most $2 \cdot \texttt{opt}$
- Whenever a path P_i becomes tight, can distribute half of the cost of the added segments as cost-share
- This implies:
  Total cost-share distributed is $\frac{1}{2}c(F) \leq \texttt{opt}$.

Adding Strictness

1. Need to compute cost-shares ξ_{st} for all $(s,t) \in R$ such that

$$\sum_{(s,t)\in R} \xi_{st} \leq \mathtt{opt}$$

- Final forest $F = P_1 \cup \ldots \cup P_q$ has cost at most $2 \cdot \mathtt{opt}$
- Whenever a path P_i becomes tight, can distribute half of the cost of the added segments as cost-share
- This implies:
  Total cost-share distributed is $\frac{1}{2}c(F) \leq \mathtt{opt}$.

Adding Strictness

1. Need to compute cost-shares ξ_{st} for all $(s,t) \in R$ such that

$$\sum_{(s,t)\in R} \xi_{st} \leq \texttt{opt}$$

- Final forest $F = P_1 \cup \ldots \cup P_q$ has cost at most $2 \cdot \texttt{opt}$
- Whenever a path P_i becomes tight, can distribute half of the cost of the added segments as cost-share
- This implies:
  Total cost-share distributed is $\frac{1}{2}c(F) \leq \texttt{opt}$.

Adding Strictness

2. Need to augment forest F_{-st} at cost

$$c_{G|F_{-st}}(s,t) \leq \beta \cdot \xi_{st}$$

- Consider the unique s,t-path P_{st} in F
- Some segments of P_{st} might be missing in F_{-st}
- Use $\beta \cdot \xi_{st}$ to pay for adding missing segments

Adding Strictness

2. Need to augment forest F_{-st} at cost

$$c_{G|F_{-st}}(s,t) \leq \beta \cdot \xi_{st}$$

- Consider the unique s,t-path P_{st} in F
- Some segments of P_{st} might be missing in F_{-st}
- Use $\beta \cdot \xi_{st}$ to pay for adding missing segments

Adding Strictness

2. Need to augment forest F_{-st} at cost

$$c_{G|F_{-st}}(s,t) \leq \beta \cdot \xi_{st}$$

- Consider the unique s,t-path P_{st} in F
- Some segments of P_{st} might be missing in F_{-st}
- Use $\beta \cdot \xi_{st}$ to pay for adding missing segments

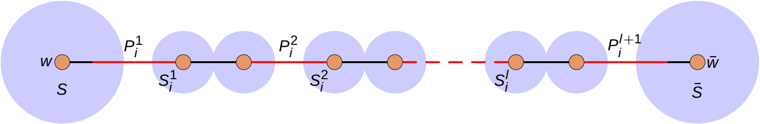
Adding Strictness

2. Need to augment forest F_{-st} at cost

$$c_{G|F_{-st}}(s,t) \leq \beta \cdot \xi_{st}$$

- Consider the unique s,t-path P_{st} in F
- Some segments of P_{st} might be missing in F_{-st}
- Use $\beta \cdot \xi_{st}$ to pay for adding missing segments

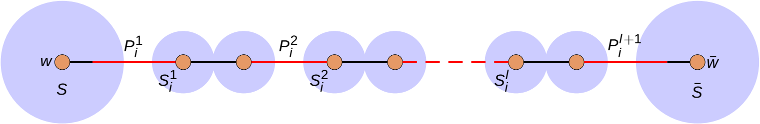
Crucial Notion: Witnesses



- Suppose, AKR adds path P_i to connect S and $\bar{S}$ at time τ_i.
- S and $\bar{S}$ are active $\implies$ both contain active terminals.

Witnesses:
Carefully chosen active terminals w and $\bar{w}$ in S and $\bar{S}$ that are closest to P_i.
For all $e \in P_i$, let $\mathcal{W}_e = \{w, \bar{w}\}$ be the set of its witnesses.

Crucial Notion: Witnesses



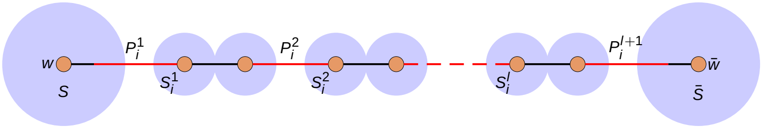
- Suppose, AKR adds path P_i to connect S and $\bar{S}$ at time τ_i.
- S and $\bar{S}$ are active $\implies$ both contain active terminals.

Witnesses:

Carefully chosen active terminals w and $\bar{w}$ in S and $\bar{S}$ that are closest to P_i.

For all $e \in P_i$, let $\mathcal{W}_e = \{w, \bar{w}\}$ be the set of its witnesses.

Crucial Notion: Witnesses

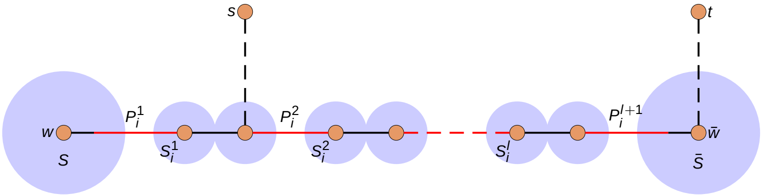


- Suppose, AKR adds path P_i to connect S and $\bar{S}$ at time τ_i.
- S and $\bar{S}$ are active $\implies$ both contain active terminals.

Witnesses:
Carefully chosen active terminals w and $\bar{w}$ in S and $\bar{S}$ that are closest to P_i.
For all $e \in P_i$, let $\mathcal{W}_e = \{w, \bar{w}\}$ be the set of its witnesses.

Witnesses

Suppose s and t meet at time $\tau_{st} \geq \tau_i$ and use (parts of) P_i:



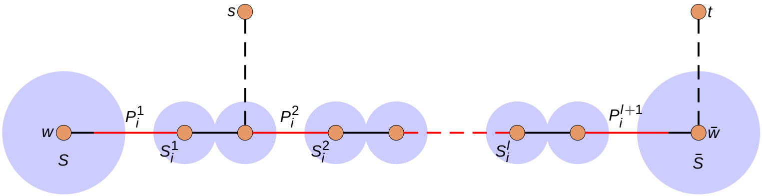
Witness Lemma:

For all edges e in F that have been added before time τ_{st}:

$$e \notin F_{-st} \Longrightarrow s \text{ or } t \text{ is a witness of } e$$

Witnesses

Suppose s and t meet at time $\tau_{st} \geq \tau_i$ and use (parts of) P_i:



Witness Lemma:

For all edges e in F that have been added before time τ_{st}:

$$e \notin F_{-st} \Longrightarrow s \text{ or } t \text{ is a witness of } e$$

Consequences of Witness Lemma

- Consider s, t-path P_{st} in F
- Any edge $e \in P_{st}$ must have been added at some time $\tau_e \leq \tau_{st}$ during AKR(R)
- Therefore: edge $e \in P_{st}$ missing $\Longrightarrow \{s, t\} \cap \mathcal{W}_e \neq \emptyset$

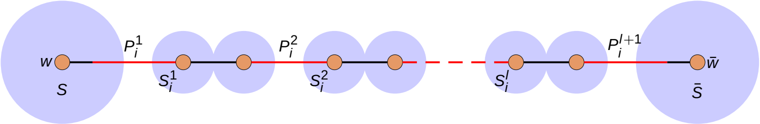
Consequences of Witness Lemma

- Consider s, t-path P_{st} in F
- Any edge $e \in P_{st}$ must have been added at some time $\tau_e \leq \tau_{st}$ during AKR(R)
- Therefore: edge $e \in P_{st}$ missing $\Longrightarrow \{s, t\} \cap \mathcal{W}_e \neq \emptyset$

Consequences of Witness Lemma

- Consider s, t-path P_{st} in F
- Any edge $e \in P_{st}$ must have been added at some time $\tau_e \leq \tau_{st}$ during AKR(R)
- Therefore: edge $e \in P_{st}$ missing $\implies \{s, t\} \cap \mathcal{W}_e \neq \emptyset$

Symmetric Cost-Sharing

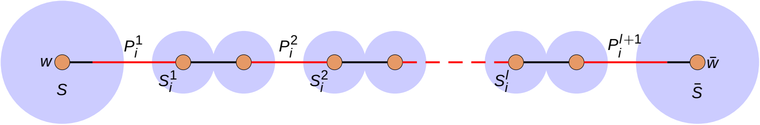


- $w, \bar{w}$: witnesses for the edges in e in P_i
- The cost-share of witness $v \in \{w, \bar{w}\}$ for each edge $e \in P_i^1 \cup \ldots \cup P_i^{l+1}$ is

$$\xi_v(e) := \frac{1}{4} c_e$$

- Cost-share of terminal pair (s,t): $\xi_{st} := \sum_{e \in F} (\xi_s(e) + \xi_t(e))$

Symmetric Cost-Sharing

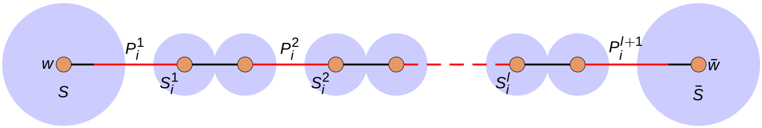


- $w, \bar{w}$: witnesses for the edges in e in P_i
- The cost-share of witness $v \in \{w, \bar{w}\}$ for each edge $e \in P_i^1 \cup \ldots \cup P_i^{l+1}$ is

$$\xi_v(e) := \frac{1}{4} c_e$$

- Cost-share of terminal pair (s, t): $\xi_{st} := \sum_{e \in F} (\xi_s(e) + \xi_t(e))$

Symmetric Cost-Sharing



- $w, \bar{w}$: witnesses for the edges in e in P_i
- The cost-share of witness $v \in \{w, \bar{w}\}$ for each edge $e \in P_i^1 \cup \ldots \cup P_i^{l+1}$ is

$$\xi_v(e) := \frac{1}{4} c_e$$

- Cost-share of terminal pair (s, t): $\xi_{st} := \sum_{e \in F} (\xi_s(e) + \xi_t(e))$

Wrapping Up

Theorem: AKR is 2-approximate and 4-strict.

Proof.

- $\bar{P}_{st}$: set of edges in P_{st} not contained in F_{-st}
- From Witness Lemma: $e \in \bar{P}_{st} \implies \{s,t\} \cap \mathcal{W}_e \neq \emptyset$
- Cost of edge e is at most $4 \cdot (\xi_e(s) + \xi_e(t))$
- Cost to rebuild the path P_{st} is at most

$$\sum_{e \in \bar{P}_{st}} c_e \leq 4 \cdot \sum_{e \in \bar{P}_{st}} (\xi_e(s) + \xi_e(t)) \leq 4 \cdot \xi_{st}$$

□

Wrapping Up

Theorem: AKR is 2-approximate and 4-strict.

Proof.

- $\bar{P}_{st}$: set of edges in P_{st} not contained in F_{-st}
- From Witness Lemma: $e \in \bar{P}_{st} \implies \{s,t\} \cap \mathcal{W}_e \neq \emptyset$
- Cost of edge e is at most $4 \cdot (\xi_e(s) + \xi_e(t))$
- Cost to rebuild the path P_{st} is at most

$$\sum_{e \in \bar{P}_{st}} c_e \leq 4 \cdot \sum_{e \in \bar{P}_{st}} (\xi_e(s) + \xi_e(t)) \leq 4 \cdot \xi_{st}$$

□

Wrapping Up

Theorem: AKR is 2-approximate and 4-strict.

Proof.

- $\bar{P}_{st}$: set of edges in P_{st} not contained in F_{-st}
- From Witness Lemma: $e \in \bar{P}_{st} \implies \{s,t\} \cap \mathcal{W}_e \neq \emptyset$
- Cost of edge e is at most $4 \cdot (\xi_e(s) + \xi_e(t))$
- Cost to rebuild the path P_{st} is at most

$$\sum_{e \in \bar{P}_{st}} c_e \leq 4 \cdot \sum_{e \in \bar{P}_{st}} (\xi_e(s) + \xi_e(t)) \leq 4 \cdot \xi_{st}$$

□

Wrapping Up

Theorem: AKR is 2-approximate and 4-strict.

Proof.

- $\bar{P}_{st}$: set of edges in P_{st} not contained in F_{-st}
- From Witness Lemma: $e \in \bar{P}_{st} \implies \{s,t\} \cap \mathcal{W}_e \neq \emptyset$
- Cost of edge e is at most $4 \cdot (\xi_e(s) + \xi_e(t))$
- Cost to rebuild the path P_{st} is at most

$$\sum_{e \in \bar{P}_{st}} c_e \leq 4 \cdot \sum_{e \in \bar{P}_{st}} (\xi_e(s) + \xi_e(t)) \leq 4 \cdot \xi_{st}$$

□

Wrapping Up

Theorem: AKR is 2-approximate and 4-strict.

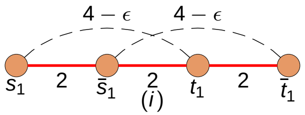
Proof.

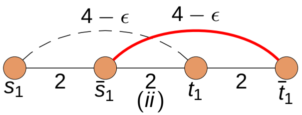
- $\bar{P}_{st}$: set of edges in P_{st} not contained in F_{-st}
- From Witness Lemma: $e \in \bar{P}_{st} \implies \{s,t\} \cap \mathcal{W}_e \neq \emptyset$
- Cost of edge e is at most $4 \cdot (\xi_e(s) + \xi_e(t))$
- Cost to rebuild the path P_{st} is at most

$$\sum_{e \in \bar{P}_{st}} c_e \leq 4 \cdot \sum_{e \in \bar{P}_{st}} (\xi_e(s) + \xi_e(t)) \leq 4 \cdot \xi_{st}$$

□

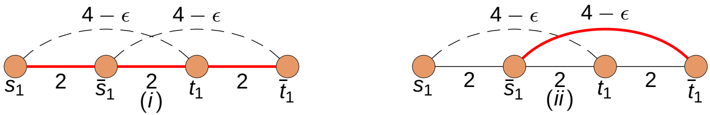
Bad Examples and Insights





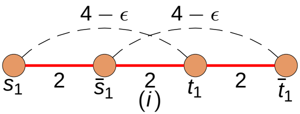
- Analysis is tight:
  Cost-share of (s_1, t_1) for path $\langle s_1, \bar{s}_1, t_1 \rangle$ is 1.
  Reconstruction of this path in $F_{-s_1 t_1}$ is 4.
- But we're not using $\xi_{(t_1 \bar{t}_1)}(t_1)$!
  Total cost-share of (s_1, t_1) in our algorithm is $\frac{3}{2}$.
  We could have shown $\frac{4}{3/2} = \frac{8}{3}$-strictness!
- Does the symmetric cost-sharing rule really lead to $\frac{8}{3}$-strictness?

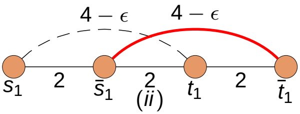
Bad Examples and Insights



- Analysis is tight:
  Cost-share of (s_1, t_1) for path $\langle s_1, \bar{s}_1, t_1 \rangle$ is 1.
  Reconstruction of this path in $F_{-s_1 t_1}$ is 4.
- But we're not using $\xi_{(t_1 \bar{t}_1)}(t_1)$!
  Total cost-share of (s_1, t_1) in our algorithm is $\frac{3}{2}$.
  We could have shown $\frac{4}{3/2} = \frac{8}{3}$-strictness!
- Does the symmetric cost-sharing rule really lead to $\frac{8}{3}$-strictness?

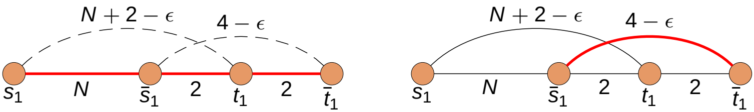
Bad Examples and Insights





- Analysis is tight:
  Cost-share of (s_1, t_1) for path $\langle s_1, \bar{s}_1, t_1 \rangle$ is 1.
  Reconstruction of this path in $F_{-s_1 t_1}$ is 4.
- But we're not using $\xi_{(t_1 \bar{t}_1)}(t_1)$!
  Total cost-share of (s_1, t_1) in our algorithm is $\frac{3}{2}$.
  We could have shown $\frac{4}{3/2} = \frac{8}{3}$-strictness!
- Does the symmetric cost-sharing rule really lead to $\frac{8}{3}$-strictness?

It gets worse...

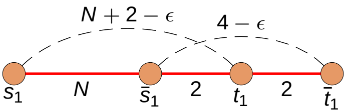


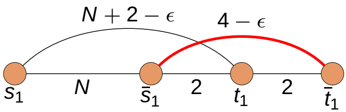
- Cost-shares of all terminal pairs

$$\xi_{s_1 t_1} = \xi_{\bar{s}_1 \bar{t}_1} = \frac{N}{4} + 1$$

- Reconnecting s_1 and t_1 costs $N + 2$.
- Reconnecting $\bar{s}_1$ and $\bar{t}_1$ costs $4 - \epsilon$
- Need to share cost of edges in asymmetric fashion!

It gets worse...



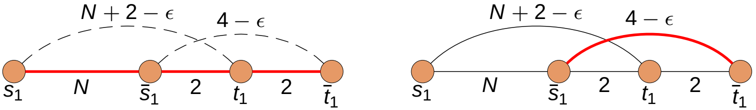


- Cost-shares of all terminal pairs

$$\xi_{s_1 t_1} = \xi_{\bar{s}_1 \bar{t}_1} = \frac{N}{4} + 1$$

- Reconnecting s_1 and t_1 costs $N+2$.
- Reconnecting $\bar{s}_1$ and $\bar{t}_1$ costs $4-\epsilon$
- Need to share cost of edges in asymmetric fashion!

It gets worse...

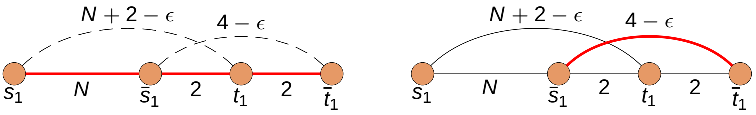


- Cost-shares of all terminal pairs

$$\xi_{s_1 t_1} = \xi_{\bar{s}_1 \bar{t}_1} = \frac{N}{4} + 1$$

- Reconnecting s_1 and t_1 costs $N + 2$.
- Reconnecting $\bar{s}_1$ and $\bar{t}_1$ costs $4 - \epsilon$
- Need to share cost of edges in asymmetric fashion!

It gets worse...



- Cost-shares of all terminal pairs

$$\xi_{s_1 t_1} = \xi_{\bar{s}_1 \bar{t}_1} = \frac{N}{4} + 1$$

- Reconnecting s_1 and t_1 costs $N + 2$.
- Reconnecting $\bar{s}_1$ and $\bar{t}_1$ costs $4 - \epsilon$
- Need to share cost of edges in asymmetric fashion!

Conclusion and Open Issues

- This lecture: AKR is 4-strict $\Longrightarrow$ 6-apx for MRoB
- The analysis is tight but can be strengthened: replacing symmetric by asymmetric cost-sharing rule leads to 3-strictness
- Can also show:
  Current Steiner forest algorithms are no better than $\frac{8}{3}$ strict.
  Conjecture: AKR is $\frac{8}{3}$-strict.