Facility Location with Capacitated and Length-Bounded Tree Connections

Andreas Bley   Jannik Matuschke   Benjamin Müller

Workshop on Algorithms and Dynamics for Games and Optimization
Playa Blanca, October 17, 2013

Motivation

Design of Optical Access Networks

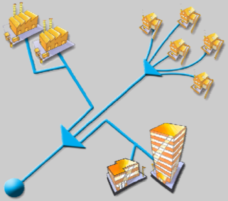
- determine sites for central offices with access hardware
- connect clients to central offices using optical fibres

Motivation

Design of Optical Access Networks

- determine sites for central offices with access hardware
- connect clients to central offices using optical fibres

Fiber Trees



Splitting

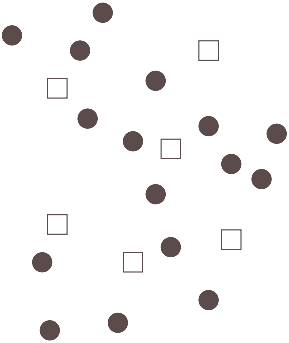
- single fiber at central office
- split into tree by splitters
- shared by many clients

Technological constraints

- clients share capacity of fiber
- limited signal strength

Outline

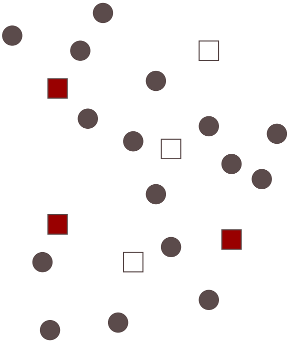
Problem Definition



UFL-CLT

- **input:** graph $G = (V, E)$,
  facilities $\mathcal{F} \subseteq V$, clients $\mathcal{C} \subseteq V$
  - client $v \in \mathcal{C}$: demand $d(v)$
  - facility $w \in \mathcal{F}$: opening cost $\phi(w)$
  - edge $e \in E$: length $\ell(e)$, cost $c(e)$
  - tree capacity U, length bound L
- **task:** Find facilities $F \subseteq \mathcal{F}$
  and trees $\mathcal{T}$ of minimum cost s.t.
  - every $T \in \mathcal{T}$ is rooted at $w_T \in F$,
  - every $v \in \mathcal{C}$ is served by $T_v \in \mathcal{T}$,
  - $\sum_{v : T_v = T} d(v) \leq U$,
  - $\sum_{e \in T[v, w_T]} \ell(e) \leq L$.

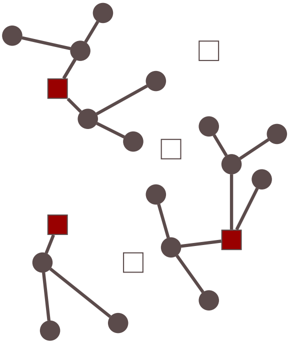
Problem Definition



UFL-CLT

- **input:** graph $G = (V, E)$, facilities $\mathcal{F} \subseteq V$, clients $\mathcal{C} \subseteq V$
  - client $v \in \mathcal{C}$: demand $d(v)$
  - facility $w \in \mathcal{F}$: opening cost $\phi(w)$
  - edge $e \in E$: length $\ell(e)$, cost $c(e)$
  - tree capacity U, length bound L
- **task:** Find facilities $F \subseteq \mathcal{F}$ and trees $\mathcal{T}$ of minimum cost s.t.
  - every $T \in \mathcal{T}$ is rooted at $w_T \in F$,
  - every $v \in \mathcal{C}$ is served by $T_v \in \mathcal{T}$,
  - $\sum_{v: T_v = T} d(v) \leq U$,
  - $\sum_{e \in T[v, w_T]} \ell(e) \leq L$.

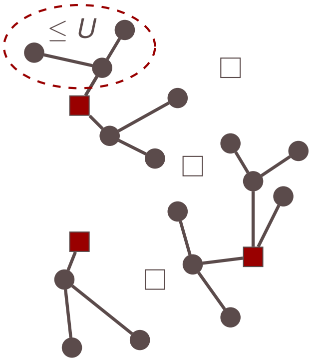
Problem Definition



UFL-CLT

- **input:** graph $G = (V, E)$,
  facilities $\mathcal{F} \subseteq V$, clients $\mathcal{C} \subseteq V$
  - client $v \in \mathcal{C}$: demand $d(v)$
  - facility $w \in \mathcal{F}$: opening cost $\phi(w)$
  - edge $e \in E$: length $\ell(e)$, cost $c(e)$
  - tree capacity U, length bound L
- **task:** Find facilities $F \subseteq \mathcal{F}$
  and trees $\mathcal{T}$ of minimum cost s.t.
  - every $T \in \mathcal{T}$ is rooted at $w_T \in F$,
  - every $v \in \mathcal{C}$ is served by $T_v \in \mathcal{T}$,
  - $\sum_{v:T_v=T} d(v) \leq U$,
  - $\sum_{e \in T[v,w_T]} \ell(e) \leq L$.

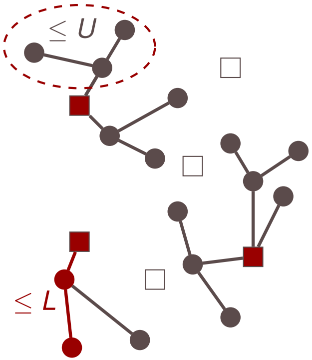
Problem Definition



UFL-CLT

- **input:** graph $G = (V, E)$, facilities $\mathcal{F} \subseteq V$, clients $\mathcal{C} \subseteq V$
  - client $v \in \mathcal{C}$: demand $d(v)$
  - facility $w \in \mathcal{F}$: opening cost $\phi(w)$
  - edge $e \in E$: length $\ell(e)$, cost $c(e)$
  - tree capacity U, length bound L
- **task:** Find facilities $F \subseteq \mathcal{F}$ and trees $\mathcal{T}$ of minimum cost s.t.
  - every $T \in \mathcal{T}$ is rooted at $w_T \in F$,
  - every $v \in \mathcal{C}$ is served by $T_v \in \mathcal{T}$,
  - $\sum_{v:T_v=T} d(v) \leq U$,
  - $\sum_{e \in T[v,w_T]} \ell(e) \leq L$.

Problem Definition



UFL-CLT

- **input:** graph $G = (V, E)$,
  facilities $\mathcal{F} \subseteq V$, clients $\mathcal{C} \subseteq V$
  - client $v \in \mathcal{C}$: demand $d(v)$
  - facility $w \in \mathcal{F}$: opening cost $\phi(w)$
  - edge $e \in E$: length $\ell(e)$, cost $c(e)$
  - tree capacity U, length bound L
- **task:** Find facilities $F \subseteq \mathcal{F}$
  and trees $\mathcal{T}$ of minimum cost s.t.
  - every $T \in \mathcal{T}$ is rooted at $w_T \in F$,
  - every $v \in \mathcal{C}$ is served by $T_v \in \mathcal{T}$,
  - $\sum_{v:T_v=T} d(v) \leq U$,
  - $\sum_{e \in T[v,w_T]} \ell(e) \leq L$.

Related Problems

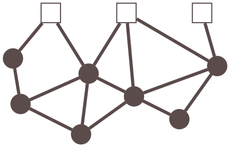
(α, β)-Approximation Algorithm

- approximates length bound by a factor of α
- approximates cost of optimal length-bounded solution by β

- Shallow-Light Trees
  - $(O(\log n), O(\log n))$-approximation for SL Steiner Tree [Marathe et al. 1998]
  - $(\alpha, 1 + \frac{2}{1-\alpha})$-Light Approximate Shortest Path Tree ($\ell = c$) [Khuller et al. 1995]
  - $(1, O(\log n))$-approximation for k-hop Spanning Tree ($\ell \equiv 1$) [Althaus et al. 2005]
- Capacitated Cable Facility Location
  - no length bound, splittable demands
  - $\rho_{\text{UFL}} + \rho_{\text{ST}}$-approximation [Ravi & Sinha 2006]

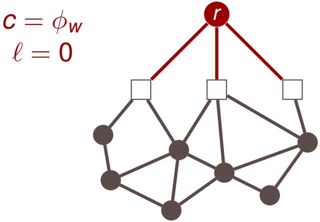
Tree Lower Bound

Construct graph G':



Tree Lower Bound

Construct graph G':

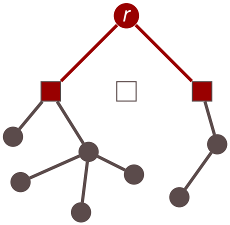


- add root vertex r
- connect r to all facilities with ϕ-cost and 0-length edges

Tree Lower Bound

Construct graph G':

$c = \phi_w$
$\ell = 0$

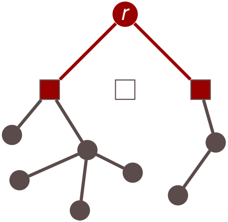


- add root vertex r
- connect r to all facilities with ϕ-cost and 0-length edges
- let S be a min-cost SLST in G' rooted at r spanning $\mathcal{C}$ with depth at most L

Tree Lower Bound

Construct graph G':

$c = \phi_w$
$\ell = 0$



- add root vertex r
- connect r to all facilities with ϕ-cost and 0-length edges
- let S be a min-cost SLST in G' rooted at r spanning $\mathcal{C}$ with depth at most L

Lemma

$c(S) \leq OPT$

UFL Lower Bound

Construct UFL instance:

- $\mathcal{C}$, $\mathcal{F}$, ϕ as before
- $\tilde{c}(v, w) = \frac{d(v)}{U} \min\{c(P) : P \text{ is a } v\text{-}w\text{-path}, \ell(P) \leq L\}$

UFL Lower Bound

Construct UFL instance:

- $\mathcal{C}$, $\mathcal{F}$, ϕ as before
- $\tilde{c}(v, w) = \frac{d(v)}{U} \min\{c(P) : P \text{ is a } v\text{-}w\text{-path}, \ell(P) \leq L\}$
- let F be an optimal solution to this UFL instance

UFL Lower Bound

Construct UFL instance:

- $\mathcal{C}$, $\mathcal{F}$, ϕ as before
- $\tilde{c}(v, w) = \frac{d(v)}{U} \min\{c(P) : P \text{ is a } v\text{-}w\text{-path}, \ell(P) \leq L\}$
- let F be an optimal solution to this UFL instance

Lemma

$\phi(F) + \sum_{v \in \mathcal{C}} \tilde{c}(v, F) \leq OPT$

UFL Lower Bound

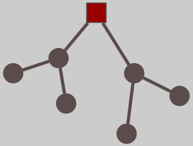
Construct UFL instance:

- $\mathcal{C}$, $\mathcal{F}$, ϕ as before
- $\tilde{c}(v, w) = \frac{d(v)}{U} \min\{c(P) : P \text{ is a } v\text{-}w\text{-path}, \ell(P) \leq L\}$
- let F be an optimal solution to this UFL instance

Lemma

$\phi(F) + \sum_{v \in \mathcal{C}} \tilde{c}(v, F) \leq OPT$

Proof



UFL Lower Bound

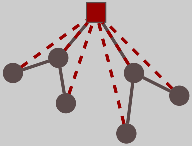
Construct UFL instance:

- $\mathcal{C}$, $\mathcal{F}$, ϕ as before
- $\tilde{c}(v,w) = \frac{d(v)}{U} \min\{c(P) : P \text{ is a } v\text{-}w\text{-path}, \ell(P) \leq L\}$
- let F be an optimal solution to this UFL instance

Lemma

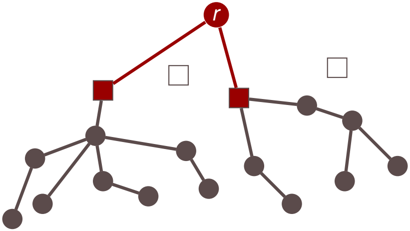
$\phi(F) + \sum_{v \in \mathcal{C}} \tilde{c}(v,F) \leq OPT$

Proof



$$\sum_{v:T_v=T} \tilde{c}(v,w) \leq \underbrace{\left(\sum_{v:T_v=T} \frac{d(v)}{U} \right)}_{\leq 1} c(T)$$

Algorithm

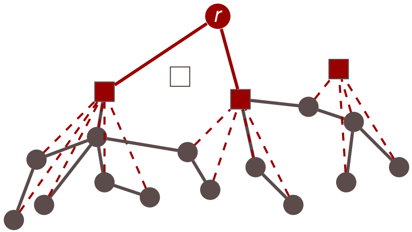


$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{\mathrm{ST}}, \beta_{\mathrm{ST}})$-approx. SLST T^*

Algorithm

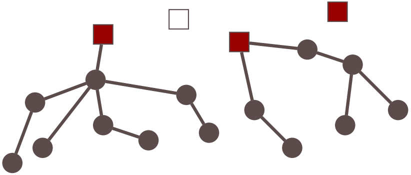


$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{\mathrm{ST}}, \beta_{\mathrm{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*

Algorithm

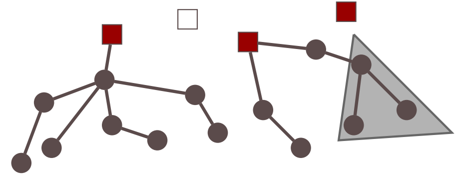


$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{\mathrm{ST}}, \beta_{\mathrm{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Algorithm

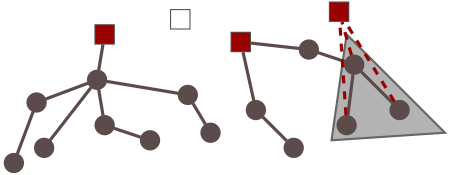


$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{\text{ST}}, \beta_{\text{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Algorithm

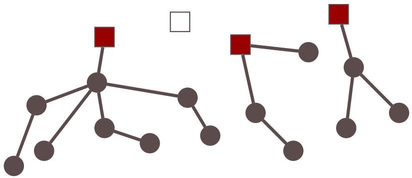


$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{\mathrm{ST}}, \beta_{\mathrm{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Algorithm

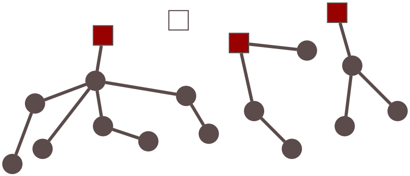


$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{\mathrm{ST}}, \beta_{\mathrm{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Algorithm



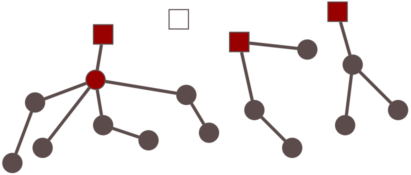
$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{\mathsf{ST}}, \beta_{\mathsf{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Subprocedure: Relieve Overloaded Subtree

Algorithm



$U = 3$, $d \equiv 1$

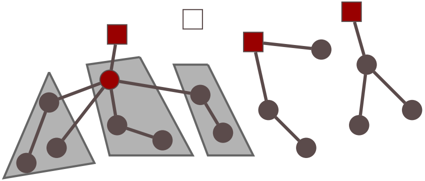
Algorithm

1. compute $(\alpha_{\mathrm{ST}}, \beta_{\mathrm{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Subprocedure: Relieve Overloaded Subtree

1. find node v with $d(T^*[v]) > U$ but $d(T^*[w]) \leq U$ for all children w of v

Algorithm



$U = 3$, $d \equiv 1$

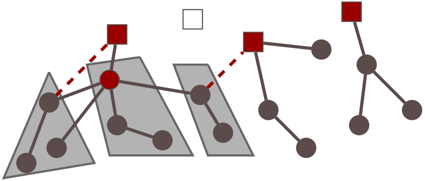
Algorithm

1. compute $(\alpha_{\mathsf{ST}}, \beta_{\mathsf{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Subprocedure: Relieve Overloaded Subtree

1. find node v with $d(T^*[v]) > U$ but $d(T^*[w]) \leq U$ for all children w of v
2. partition subtrees into groups with demand between $U/2$ and U

Algorithm



$U = 3$, $d \equiv 1$

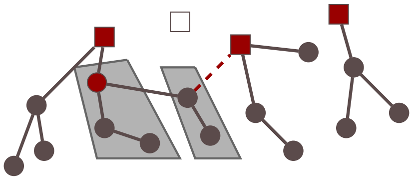
Algorithm

1. compute $(\alpha_{\text{ST}}, \beta_{\text{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Subprocedure: Relieve Overloaded Subtree

1. find node v with $d(T^*[v]) > U$ but $d(T^*[w]) \leq U$ for all children w of v
2. partition subtrees into groups with demand between $U/2$ and U
3. find c-closest client-facility pairs

Algorithm



$U = 3$, $d \equiv 1$

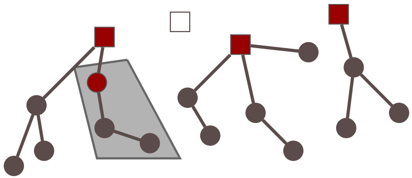
Algorithm

1. compute $(\alpha_{\text{ST}}, \beta_{\text{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Subprocedure: Relieve Overloaded Subtree

1. find node v with $d(T^*[v]) > U$ but $d(T^*[w]) \leq U$ for all children w of v
2. partition subtrees into groups with demand between $U/2$ and U
3. find c-closest client-facility pairs

Algorithm



$U = 3$, $d \equiv 1$

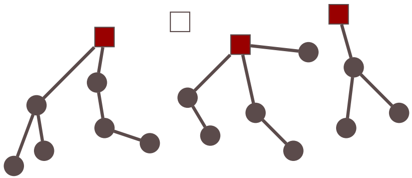
Algorithm

1. compute $(\alpha_{\mathsf{ST}}, \beta_{\mathsf{ST}})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Subprocedure: Relieve Overloaded Subtree

1. find node v with $d(T^*[v]) > U$ but $d(T^*[w]) \leq U$ for all children w of v
2. partition subtrees into groups with demand between $U/2$ and U
3. find c-closest client-facility pairs

Algorithm



$U = 3$, $d \equiv 1$

Algorithm

1. compute $(\alpha_{ST}, \beta_{ST})$-approx. SLST T^*
2. compute β_{UFL}-approx. UFL solution F^*
3. relieve overloaded subtrees using F^*

Subprocedure: Relieve Overloaded Subtree

1. find node v with $d(T^*[v]) > U$ but $d(T^*[w]) \leq U$ for all children w of v
2. partition subtrees into groups with demand between $U/2$ and U
3. find c-closest client-facility pairs

Analysis

- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*

capacity: ?

length: ?

cost: ?

Analysis

- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*

capacity: relieve
length: ?
cost: ?

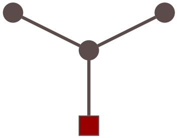
Analysis

- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*

capacity:	$d(T) \leq U$	for all $T \in \mathcal{T}$
length:	?	
cost:	?	

Analysis

- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*



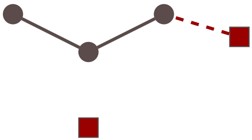
capacity: $d(T) \leq U$ for all $T \in \mathcal{T}$

length: $\ell(T^*[v, w]) \leq \alpha_{\mathsf{ST}} L$

cost: ?

Analysis

- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*



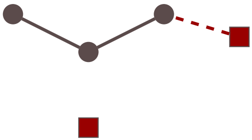
capacity:	$d(T) \leq U$	for all $T \in \mathcal{T}$
length:	$\ell(T[v,w]) \leq (2\alpha_{\mathrm{ST}} + 1)L$	for all $T \in \mathcal{T}$
cost:	?	

Analysis

- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*

capacity:	$d(T) \leq U$	for all $T \in \mathcal{T}$
length:	$\ell(T[v,w]) \leq (2\alpha_{\mathrm{ST}} + 1)L$	for all $T \in \mathcal{T}$
cost:	$c(\mathcal{T}) + \phi(F) \leq c(T^*) + \phi(F^*) + 2\sum_{v \in \mathcal{C}} \tilde{c}(v, F^*)$	

Analysis

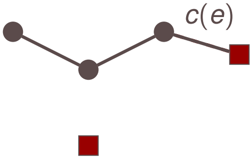


- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*

capacity:	$d(T) \leq U$	for all $T \in \mathcal{T}$
length:	$\ell(T[v,w]) \leq (2\alpha_{\mathrm{ST}} + 1)L$	for all $T \in \mathcal{T}$
cost:	$c(\mathcal{T}) + \phi(F) \leq c(T^*) + \phi(F^*) + 2\sum_{v \in \mathcal{C}} \tilde{c}(v, F^*)$	

Analysis

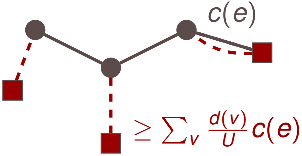
- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*



capacity:	$d(T) \leq U$	for all $T \in \mathcal{T}$
length:	$\ell(T[v,w]) \leq (2\alpha_{\mathrm{ST}} + 1)L$	for all $T \in \mathcal{T}$
cost:	$c(\mathcal{T}) + \phi(F) \leq c(T^*) + \phi(F^*) + 2\sum_{v \in \mathcal{C}} \tilde{c}(v, F^*)$	

Analysis

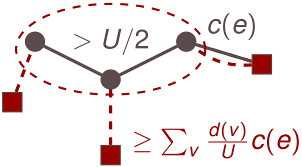
- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*



$$
\begin{aligned}
\textbf{capacity:}\quad & d(T) \leq U && \text{for all } T \in \mathcal{T} \\
\textbf{length:}\quad & \ell(T[v,w]) \leq (2\alpha_{\mathrm{ST}} + 1)L && \text{for all } T \in \mathcal{T} \\
\textbf{cost:}\quad & c(\mathcal{T}) + \phi(F) \leq c(T^*) + \phi(F^*) + 2\sum_{v \in \mathcal{C}} \tilde{c}(v, F^*)
\end{aligned}
$$

Analysis

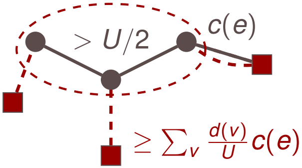
- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*



capacity:	$d(T) \leq U$	for all $T \in \mathcal{T}$
length:	$\ell(T[v,w]) \leq (2\alpha_{\mathrm{ST}} + 1)L$	for all $T \in \mathcal{T}$
cost:	$c(\mathcal{T}) + \phi(F) \leq c(T^*) + \phi(F^*) + 2\sum_{v \in \mathcal{C}} \tilde{c}(v, F^*)$	

Analysis

- returned solution $(\mathcal{T}, F)$
- initial tree T^*, UFL solution F^*



capacity:	$d(T) \leq U$	for all $T \in \mathcal{T}$
length:	$\ell(T[v, w]) \leq (2\alpha_{\mathrm{ST}} + 1)L$	for all $T \in \mathcal{T}$
cost:	$c(\mathcal{T}) + \phi(F) \leq c(T^*) + \phi(F^*) + 2\sum_{v \in \mathcal{C}} \tilde{c}(v, F^*)$	

Theorem

There is an $(O(\log n), O(\log n))$-approximation for UFL-CLT.

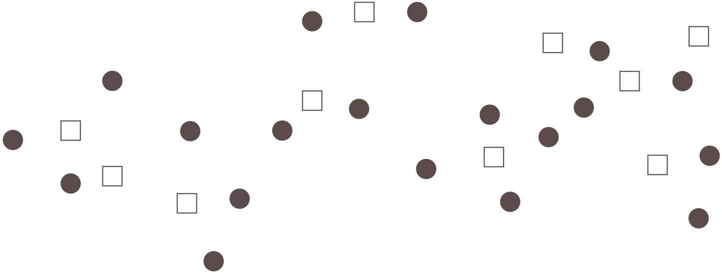
- use SLST matching augmentation and UFL-greedy

Approximation Guarantees

	length	cost	
general	$O(\log n)$	$O(\log n)$	
	$3+\varepsilon$	$O(\log^2 n)$	$(*)$
$\ell = c$, metric	3α	$(1+\frac{2}{\alpha-1})(\beta_{\mathrm{ST}}+\beta_{\mathrm{UFL}}^{\alpha})+1$	
$\ell \equiv 1$, c metric	$1+\varepsilon$	$O(\log n)$	

$(*)$ quasi-polynomial run-time

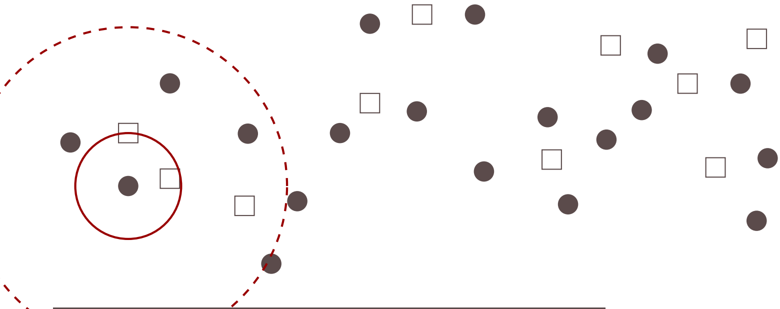
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

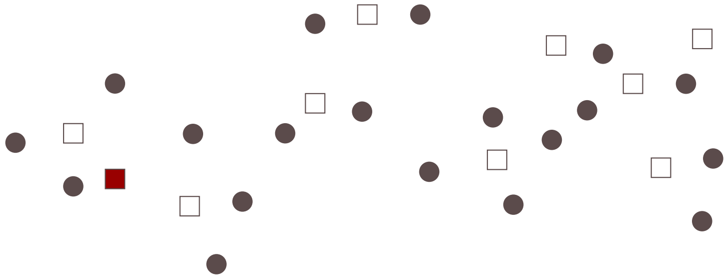
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

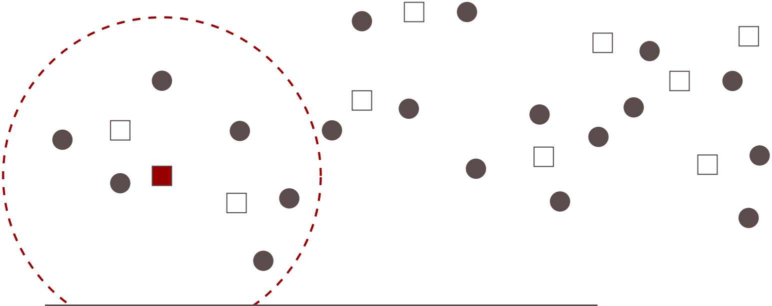
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

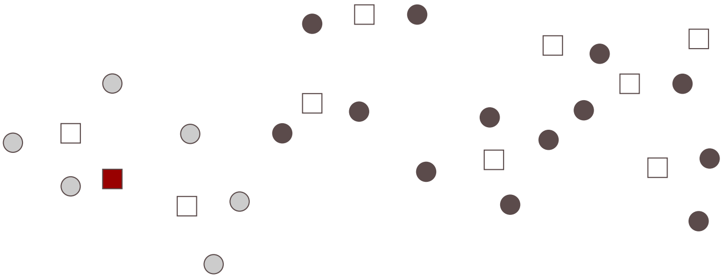
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

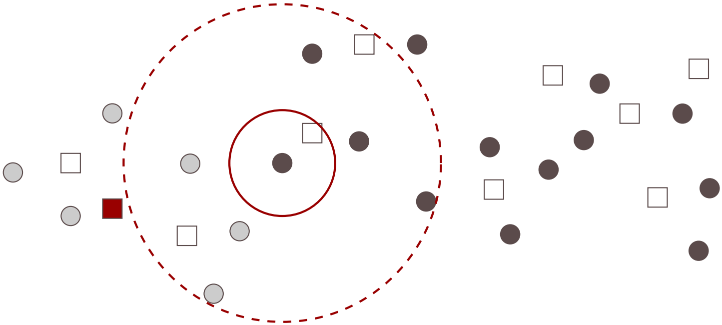
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

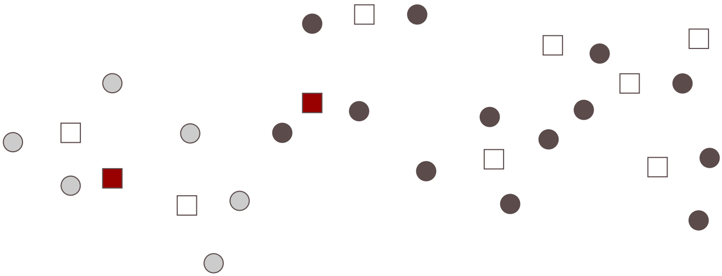
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

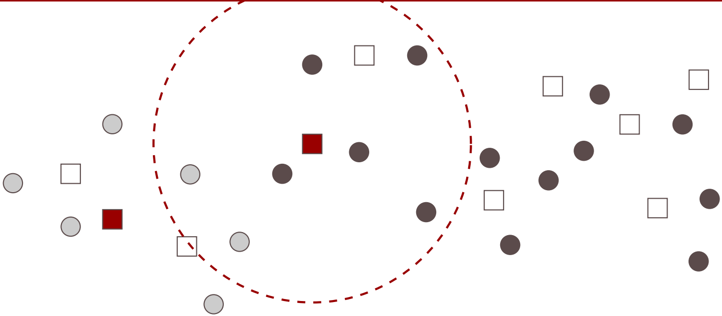
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

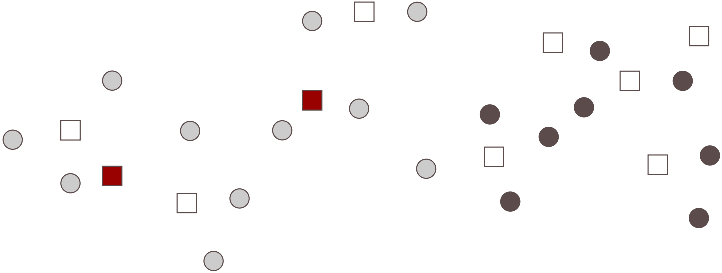
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

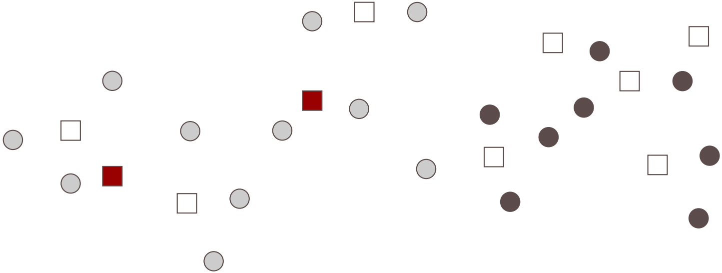
Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

Greedy Cover



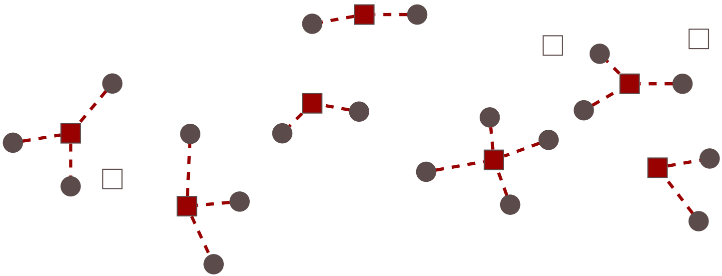
Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

Lemma

If ℓ is a metric, then $\phi(F) \leq OPT$.

Greedy Cover



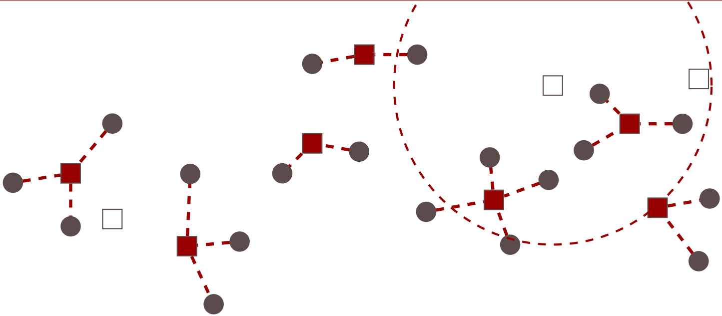
Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

Lemma

If ℓ is a metric, then $\phi(F) \leq OPT$.

Greedy Cover



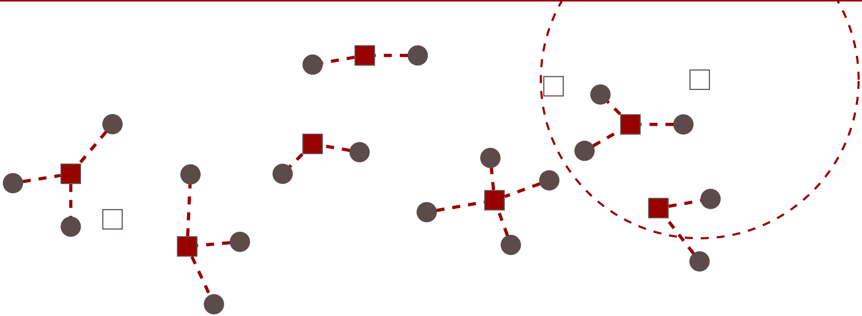
Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

Lemma

If ℓ is a metric, then $\phi(F) \leq OPT$.

Greedy Cover



Algorithm

1. $F = \emptyset$
2. while $\exists v \in \mathcal{C}$ with $\ell(v, F) > 3L$
   - $F_v = \{w \in \mathcal{F} : \ell(v, w) \leq L\}$
   - let $w^* \in F_v$ with $\phi(w^*)$ minimum
   - $F = F \cup \{w^*\}$

Lemma

If ℓ is a metric, then $\phi(F) \leq OPT$.

Length-dependent Costs

- let $\ell = c$ be a metric

Length-dependent Costs

- let $\ell = c$ be a metric

Light Approximate Shortest-path Tree (LAST) [Khuller et al. 1995]

Given a tree T with root r, compute tree T' s.t.

- $c(T'[v, r]) \leq \alpha SP(v)$ ($SP(v) =$ length of shortest r-v-path)
- $c(T') \leq (1 + \frac{2}{\alpha - 1})c(T)$.

Length-dependent Costs

- let $\ell = c$ be a metric

Light Approximate Shortest-path Tree (LAST) [Khuller et al. 1995]

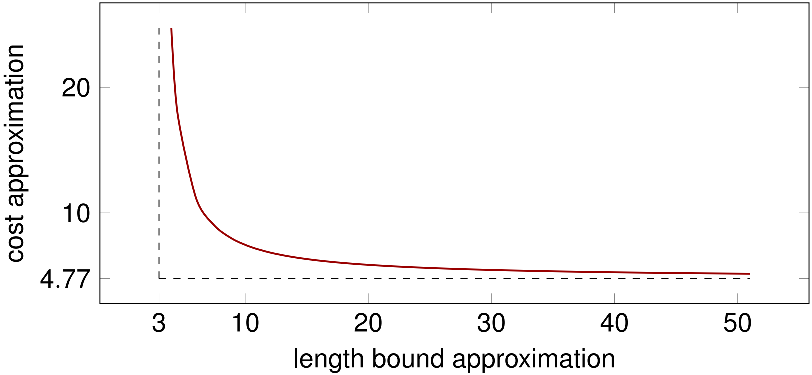
Given a tree T with root r, compute tree T' s.t.

- $c(T'[v,r]) \leq \alpha SP(v)$ ($SP(v) =$ length of shortest r-v-path)
- $c(T') \leq (1 + \frac{2}{\alpha - 1})c(T)$.

Idea

- ignore length bound for SLST and UFL
- compute greedy cover
- apply LAST algorithm

Length-dependent Costs



Approximation Factor

- **length:** $3\alpha \cdot L$
- **cost:** $((1 + \frac{2}{\alpha-1})(\beta_{\mathsf{ST}} + \beta^{\alpha}_{\mathsf{UFL}}) + 1)OPT$

Hop Constraints

- let $\ell \equiv 1$, c be a metric

Approximation for Hop-constrained Trees

- spanning tree: $(1, O(\log n))$ [Althaus et al. 2005]
- Steiner tree: $(1, O(\log n))$ for fixed L [Kortsarz & Peleg 1999]

Hop Constraints

- let $\ell \equiv 1$, c be a metric

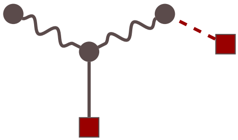
Approximation for Hop-constrained Trees

- spanning tree: $(1, O(\log n))$ [Althaus et al. 2005]
- Steiner tree: $(1, O(\log n))$ for fixed L [Kortsarz & Peleg 1999]

Modified Relieve Procedure

- when relieving subtree rooted at v, connect facility directly to v

! no increase in depth of tree



Hop Constraints

- let $\ell \equiv 1$, c be a metric

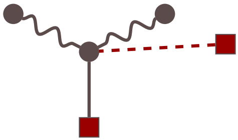
Approximation for Hop-constrained Trees

- spanning tree: $(1, O(\log n))$ [Althaus et al. 2005]
- Steiner tree: $(1, O(\log n))$ for fixed L [Kortsarz & Peleg 1999]

Modified Relieve Procedure

- when relieving subtree rooted at v, connect facility directly to v

! no increase in depth of tree



Hop Constraints

- let $\ell \equiv 1$, c be a metric

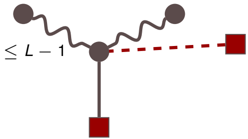
Approximation for Hop-constrained Trees

- spanning tree: $(1, O(\log n))$ [Althaus et al. 2005]
- Steiner tree: $(1, O(\log n))$ for fixed L [Kortsarz & Peleg 1999]

Modified Relieve Procedure

- when relieving subtree rooted at v, connect facility directly to v

! no increase in depth of tree



Hop Constraints

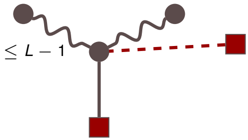
- let $\ell \equiv 1$, c be a metric

Approximation for Hop-constrained Trees

- spanning tree: $(1, O(\log n))$ [Althaus et al. 2005]
- Steiner tree: $(1, O(\log n))$ for fixed L [Kortsarz & Peleg 1999]

Modified Relieve Procedure

- when relieving subtree rooted at v, connect facility directly to v
- ! no increase in depth of tree



- $(1 + \varepsilon,\ O(\log n))$-approximation

Conclusion

- facility location + capacitated trees + length bound
- flexible approximation framework combining several lower bounds

Open questions

? capacitated facilities
? uncertain demands
? $(O(1), O(1))$-approximation for shallow-light Steiner tree

Conclusion

- facility location + capacitated trees + length bound
- flexible approximation framework combining several lower bounds

Open questions

? capacitated facilities

? uncertain demands

? $(O(1), O(1))$-approximation for shallow-light Steiner tree

Thank you!