Approximation of Metrics by Trees
Tree metric approximation

Given a metric $d$ on $V$, there is a randomized polynomial time algorithm that computes a random tree metric $d_{T,\ell}$ on $V$ such that

$$d(u, v) \leq d_{T,\ell}(u, v) \text{ and } \mathbb{E}[d_{T,\ell}(u, v)] \leq O(\log |V|)d(u, v)$$

for all $u, v \in V$. 
Tree metric approximation for
Buy-at-bulk
Network Design
Buy-at-bulk Network Design

Input: graph $G = (V, E)$, lengths $\ell : E \rightarrow \mathbb{R}_+$, terminal pairs $s_i, t_i$ and demands $\Delta_i$ for $i \in [k]$, non-decreasing cost function $f : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ with $f(0) = 0$ and $f(x + y) \leq f(x) + f(y)$

Task: find paths $P_i$ for $i \in [k]$ minimizing $\sum_{e \in E} \ell_e f(\Delta_e)$ with $\Delta_e := \sum_{i \in [k]: e \in P_i} \Delta_i$
Buy-at-bulk Network Design

Input: graph $G = (V, E)$, lengths $\ell : E \rightarrow \mathbb{R}_+$, terminal pairs $s_i, t_i$ and demands $\Delta_i$ for $i \in [k]$, non-decreasing cost function $f : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ with $f(0) = 0$ and $f(x + y) \leq f(x) + f(y)$

Task: find paths $P_i$ for $i \in [k]$ minimizing $\sum_{e \in E} \ell_e f(\Delta_e)$ with $\Delta_e := \sum_{i \in [k]: e \in P_i} \Delta_i$
Algorithm

1. Approximate the metric $d_{G,\ell}$ by a tree metric $d_{T,\ell'}$.

2. For every $i \in [k]$: 
   - Let $T_i = (v_0, \ldots, v_n)$ be the unique $s_i$-$t_i$-path in $T$.
   - For $u, v \in V$ let $P_{uv}$ be a shortest $u$-$v$-path in $G$.
   - Let $P_i$ be a simple path in the concatenation $P_{v_0v_1} \circ \ldots \circ P_{v_{n-1}v_n}$.

3. Return $(P_i)_{i \in [k]}$.
Algorithm

1. Approximate the metric $d_{G, \ell}$ by a tree metric $d_{T, \ell'}$.
2. For every $i \in [k]$:
   - Let $T_i = (v_0, \ldots, v_n)$ be the unique $s_i-t_i$-path in $T$.
   - For $u, v \in V$ let $P_{uv}$ be a shortest $u-v$-path in $G$.
   - Let $P_i$ be a simple path in the concatenation $P_{v_0 v_1} \circ \ldots \circ P_{v_{n-1} v_n}$.
3. Return $(P_i)_{i \in [k]}$. 
Theorem

The algorithm is a randomized $O(\log |V|)$-approximation algorithm for Buy-at-bulk Network Design.

Proof:

\[
\mathbb{E}[\text{ALG}] \leq \mathbb{E} \left[ \sum_{e \in T} \ell'_e f(\Delta'_e) \right] \leftarrow \text{cost of ALG in } T
\]
\[
\leq \mathbb{E} \left[ \sum_{e \in T} \ell'_e f(\bar{\Delta}_e) \right] \leftarrow \text{cost of OPT in } T
\]
\[
\leq O(\log |V|) \text{ OPT} \quad \square
\]
Today you learnt ...

Life is less complicated on a tree (metric).
Combining two Approximation Algorithms: Location Routing
Location + Routing = Location Routing

facility location

vehicle routing
Location + Routing = Location Routing

facility location

vehicle routing

location routing
**Input:** facilities $F$, clients $C$, metric $d$ on $F \cup C$, opening costs $f \in \mathbb{R}_+^F$, demands $\Delta \in \mathbb{R}_+^C$, vehicle capacity $U$  

**Task:** find a set of facilities $F' \subseteq F$ and a collection of tours $\mathcal{T}$ such that  
  
  ▶ every tour contains a facility from $F'$,  
  ▶ every client contained in a tour,  
  ▶ total demand in each tour is at most $U$  

minimizing $\sum_{w \in F'} f_w + \sum_{T \in \mathcal{T}} d(T)$
Input: facilities $F$, clients $C$, metric $d$ on $F \cup C$, opening costs $f \in \mathbb{R}_+^F$, demands $\Delta \in \mathbb{R}_+^C$, vehicle capacity $U$

Task: find a set of facilities $F' \subseteq F$ and a collection of tours $\mathcal{T}$ such that

- every tour contains a facility from $F'$,
- every client contained in a tour,
- total demand in each tour is at most $U$

minimizing $\sum_{w \in F'} f_w + \sum_{T \in \mathcal{T}} d(T)$
Location Routing

Input: facilities $F$, clients $C$, metric $d$ on $F \cup C$, opening costs $f \in \mathbb{R}_+^F$, demands $\Delta \in \mathbb{R}_+^C$, vehicle capacity $U$

Task: find a set of facilities $F' \subseteq F$ and a collection of tours $\mathcal{T}$ such that

- every tour contains a facility from $F'$,
- every client contained in a tour,
- total demand in each tour is at most $U$

minimizing $\sum_{w \in F'} f_w + \sum_{T \in \mathcal{T}} d(T)$
Bound 1: Minimum Spanning Tree

Construct graph $G'$ with edge weights $d'$:

- $r$

- vertices $F \cup C \cup \{r\}$
Construct graph $G'$ with edge weights $d'$:

- vertices $F \cup C \cup \{r\}$
- edge $\{v, w\}$ for each $v, w \in C$
  weight: $d(v, w)$
Construct graph $G'$ with edge weights $d'$:

- vertices $F \cup C \cup \{r\}$
- edge $\{v, w\}$ for each $v, w \in C$ weight: $d(v, w)$
- edge $\{v, w\}$ for each $v \in C, w \in F$ weight: $d(v, w) + f_w$

Lemma
Let $S$ be a minimum spanning tree in $G'$ w.r.t. $d'$. Then $d'(S) \leq \text{OPT}$. 
Bound 1: Minimum Spanning Tree

Construct graph $G'$ with edge weights $d'$:

- vertices $F \cup C \cup \{r\}$
- edge $\{v, w\}$ for each $v, w \in C$
  weight: $d(v, w)$
- edge $\{v, w\}$ for each $v \in C, w \in F$
  weight: $d(v, w) + f_w$
- edge $\{r, w\}$ for each $w \in F$
  weight: 0

$$d(v, w) + f_w$$

$$d(v, w)$$
Bound 1: Minimum Spanning Tree

Construct graph $G'$ with edge weights $d'$:

- Vertices $F \cup C \cup \{r\}$
- Edge $\{v, w\}$ for each $v, w \in C$
  - Weight: $d(v, w)$
- Edge $\{v, w\}$ for each $v \in C, w \in F$
  - Weight: $d(v, w) + f_w$
- Edge $\{r, w\}$ for each $w \in F$
  - Weight: 0

Lemma

Let $S$ be a minimum spanning tree in $G'$ w.r.t. $d'$. Then $d'(S) \leq \text{OPT}$. 
Bound 1: Minimum Spanning Tree

Construct graph $G'$ with edge weights $d'$:

- vertices $F \cup C \cup \{r\}$
- edge $\{v, w\}$ for each $v, w \in C$
  - weight: $d(v, w)$
- edge $\{v, w\}$ for each $v \in C, w \in F$
  - weight: $d(v, w) + f_w$
- edge $\{r, w\}$ for each $w \in F$
  - weight: 0

Lemma

Let $S$ be a minimum spanning tree in $G'$ w.r.t. $d'$. Then $d'(S) \leq \text{OPT}$. 
Bound 1: Minimum Spanning Tree

Construct graph $G'$ with edge weights $d'$:

- vertices $F \cup C \cup \{r\}$
- edge $\{v, w\}$ for each $v, w \in C$
  weight: $d(v, w)$
- edge $\{v, w\}$ for each $v \in C, w \in F$
  weight: $d(v, w) + f_w$
- edge $\{r, w\}$ for each $w \in F$
  weight: 0

Lemma

Let $S$ be a minimum spanning tree in $G'$ w.r.t. $d'$. Then $d'(S) \leq \text{OPT}$. 
Bound 1: Minimum Spanning Tree

Construct graph $G'$ with edge weights $d'$:

- vertices $F \cup C \cup \{r\}$
- edge $\{v, w\}$ for each $v, w \in C$
  - weight: $d(v, w)$
- edge $\{v, w\}$ for each $v \in C, w \in F$
  - weight: $d(v, w) + f_w$
- edge $\{r, w\}$ for each $w \in F$
  - weight: 0

Lemma

Let $S$ be a minimum spanning tree in $G'$ w.r.t. $d'$. Then $d'(S) \leq \text{OPT}$. 
Bound 1: Minimum Spanning Tree

Construct graph $G'$ with edge weights $d'$:

- vertices $F \cup C \cup \{r\}$
- edge $\{v, w\}$ for each $v, w \in C$ weight: $d(v, w)$
- edge $\{v, w\}$ for each $v \in C, w \in F$ weight: $d(v, w) + f_w$
- edge $\{r, w\}$ for each $w \in F$ weight: 0

Lemma

Let $S$ be a minimum spanning tree in $G'$ w.r.t. $d'$. Then $d'(S) \leq OPT$. 
Bound 2: Uncapacitated Facility Location

- construct UFL instance with clients $C$, facilities $F$, opening costs $f$, and connection costs $d''_{vw} = \frac{2\Delta^U_v}{U} d(v, w)$.
Bound 2: Uncapacitated Facility Location

- construct UFL instance with clients \( C \), facilities \( F \), opening costs \( f \), and connection costs

\[
d''_{vw} = \frac{2\Delta v}{U} d(v, w)
\]

Lemma

Let \( F'' \subseteq F \) be an optimal solution to the UFL instance. Then

\[
\sum_{w \in F''} f_w + \sum_{v \in C} d''(v, F) \leq \text{OPT}
\]
construct UFL instance with clients $C$, facilities $F$, opening costs $f$, and connection costs $d''_{vw} = \frac{2\Delta_v}{U} d(v, w)$

**Lemma**

Let $F'' \subseteq F$ be an optimal solution to the UFL instance. Then $\sum_{w \in F''} f_w + \sum_{v \in C} d''(v, F) \leq OPT$.

**Proof**: Let $(F^*, T^*)$ be optimal LR solution. Consider $T \in T^*$:
construct UFL instance with clients $C$, facilities $F$, opening costs $f$, and connection costs $d''_{vw} = \frac{2\Delta_v}{U} d(v, w)$

**Lemma**

Let $F'' \subseteq F$ be an optimal solution to the UFL instance. Then $\sum_{w \in F''} f_w + \sum_{v \in C} d''(v, F) \leq OPT$.

**Proof:** Let $(F^*, T^*)$ be optimal LR solution. Consider $T \in T^*$:
Bound 2: Uncapacitated Facility Location

- construct UFL instance with clients $C$, facilities $F$, opening costs $f$, and connection costs $d''_{vw} = \frac{2\Delta_v}{U} d(v, w)$

**Lemma**

Let $F'' \subseteq F$ be an optimal solution to the UFL instance. Then $\sum_{w \in F''} f_w + \sum_{v \in C} d''(v, F) \leq OPT$.

**Proof:** Let $(F^*, T^*)$ be optimal LR solution. Consider $T \in T^*$:

$$d''_{vT} = \frac{2\Delta_v}{U} d(v, w_T) \leq \frac{\Delta_v}{U} d(T) \quad \forall v \in V(T)$$
Bound 2: Uncapacitated Facility Location

- construct UFL instance with clients \( C \), facilities \( F \), opening costs \( f \), and connection costs \( d''_{vw} = \frac{2\Delta_v}{U} d(v, w) \)

Lemma

Let \( F'' \subseteq F \) be an optimal solution to the UFL instance. Then \( \sum_{w \in F''} f_w + \sum_{v \in C} d''(v, F) \leq \text{OPT} \).

Proof: Let \((F^*, T^*)\) be optimal LR solution. Consider \( T \in T^* \):

\[
d''_{vw_T} = \frac{2\Delta_v}{U} d(v, w_T) \leq \frac{\Delta_v}{U} d(T) \quad \forall v \in V(T)
\]

\[
\sum_{v \in V(T)} d''_{vw_T} \leq \left( \sum_{v \in V(T)} \frac{\Delta_v}{U} \right) d(T) \leq 1
\]

\[
\leq \frac{\Delta_v}{U} d(T)
\]
**Bound 2: Uncapacitated Facility Location**

- construct UFL instance with clients $C$, facilities $F$, opening costs $f$, and connection costs $d''_{\nu w} = \frac{2\Delta_{\nu}}{U} d(\nu, w)$

**Lemma**

Let $F'' \subseteq F$ be an optimal solution to the UFL instance. Then $\sum_{w \in F''} f_w + \sum_{\nu \in C} d''(\nu, F') \leq OPT$.

**Proof:** Let $(F^*, T^*)$ be optimal LR solution. Consider $T \in T^*$:

\[ d''_{\nu w_T} = \frac{2\Delta_{\nu}}{U} d(\nu, w_T) \leq \frac{\Delta_{\nu}}{U} d(T) \quad \forall \nu \in V(T) \]

\[ \sum_{\nu \in V(T)} d''_{\nu w_T} \leq \left( \sum_{\nu \in V(T)} \frac{\Delta_{\nu}}{U} \right) d(T) \leq 1 \]

\[ \sum_{\nu \in C} d''(\nu, F^*) \leq \sum_{T \in T^*} \sum_{\nu \in V(T)} d''(\nu, w_T) \leq \sum_{T \in T^*} d(T) \quad \square \]
Main Algorithm

1. compute MST $S$

Subprocedure: relieve overloaded subtree

1. find node $v$ with $\Delta(S_v) > U$ but $\Delta(S_w) \leq U$ for all children $w$ of $v$

2. partition subtrees of $v$ into groups with demand between $U/2$ and $U$ (except for the one containing $v$)

3. for each group find nearest open facility
**Algorithm**

**Main Algorithm**
1. compute MST $S$
2. compute UFL approximation $F''$

$U = 3, \Delta \equiv 1$
Algorithm

Main Algorithm

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours

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

**Main Algorithm**

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
Algorithm

Main Algorithm
1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   ▶ relieve overloaded subtree using $F''$

$U = 3, \Delta \equiv 1$
Algorithm

Main Algorithm

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$

$U = 3, \Delta = 1$
Algorithm

Main Algorithm
1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

$U = 3, \Delta \equiv 1$
Algorithm

Main Algorithm

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

$U = 3$, $\Delta \equiv 1$
Algorithm

Main Algorithm

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

$U = 3, \Delta \equiv 1$

Subprocedure: relieve overloaded subtree
Algorithm

$U = 3, \Delta = 1$

**Main Algorithm**

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

**Subprocedure: relieve overloaded subtree**

1. find node $v$ with $\Delta(S_v) > U$ but $\Delta(S_w) \leq U$ for all children $w$ of $v$
Algorithm

Main Algorithm

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

Subprocedure: relieve overloaded subtree

1. find node $v$ with $\Delta(S_v) > U$
   but $\Delta(S_w) \leq U$ for all children $w$ of $v$
2. partition subtrees of $v$ into groups with demand between $U/2$ and $U$ (except for the one containing $v$)
Algorithm

Subprocedure: relieve overloaded subtree

1. find node $v$ with $\Delta(S_v) > U$ but $\Delta(S_w) \leq U$ for all children $w$ of $v$
2. partition subtrees of $v$ into groups with demand between $U/2$ and $U$ (except for the one containing $v$)
3. for each group find nearest open facility

Main Algorithm

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut
Algorithm

Main Algorithm
1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

Subprocedure: relieve overloaded subtree
1. find node $v$ with $\Delta(S_v) > U$
   but $\Delta(S_w) \leq U$ for all children $w$ of $v$
2. partition subtrees of $v$ into groups with demand between $U/2$
   and $U$ (except for the one containing $v$)
3. for each group find nearest open facility
Algorithm

Main Algorithm
1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

Subprocedure: relieve overloaded subtree
1. find node $v$ with $\Delta(S_v) > U$
   but $\Delta(S_w) \leq U$ for all children $w$ of $v$
2. partition subtrees of $v$ into groups with demand between $U/2$
   and $U$ (except for the one containing $v$)
3. for each group find nearest open facility

$U = 3$, $\Delta \equiv 1$
Algorithm

Main Algorithm

1. compute MST $S$
2. compute UFL approximation $F''$
3. turn trees into tours
   - relieve overloaded subtree using $F''$
   - double edges, shortcut

Subprocedure: relieve overloaded subtree

1. find node $v$ with $\Delta(S_v) > U$
   but $\Delta(S_w) \leq U$ for all children $w$ of $v$
2. partition subtrees of $v$ into groups with demand between $U/2$
   and $U$ (except for the one containing $v$)
3. for each group find nearest open facility
This semester you learnt ...

many different **techniques** for designing approximation algorithms

- get a feeling which one works in which situation
- adapt them to your optimization problems
- know how to get lower bounds
- be inventive