

Exact and approximation algorithms for designing optical access networks

Ashwin Arulselman
Department of Management Science
University of Strathclyde

Workshop on Clustering and Search techniques for large scale
networks, Oct 23-25, 2015

Outline

Competitive algorithm - Incremental Facility Location

- ▶ Problem definition and motivation
- ▶ Worst case example
- ▶ Algorithm, analysis and results

Joint work with [Olaf Maurer](#) and [Martin Skutella](#)

Branch-cut-and-price algorithm - Buy at Bulk FL

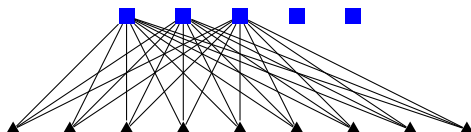
- ▶ Problem definition and MIP model
- ▶ Valid Inequalities
- ▶ Implementation and results

Joint work with [Mohsen Rezapour](#) and [Wolfgang Welz](#)

Problem definition

Input: Given an instance of uncapacitated facility location problem:

- ▶ A set of F facilities
- ▶ A set of D customers
- ▶ Facility opening cost $f : F \rightarrow \mathbb{R}_+$
- ▶ Service cost $c : F \times D \rightarrow \mathbb{R}_+$



Problem definition

Output:

- ▶ A sequence for **opening** facilities
- ▶ A sequence for **serving** customers along with their assignments to an open facility *within the partial sequence*
- ▶ Think of a point in the sequence as an event happening at a point of time

Problem Definition and Motivation

- ▶ Define a partial solution for serving first r customers from a given sequence of facility and customer as SOL_r
- ▶ Find a sequence of facility and customer with

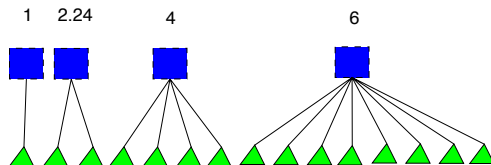
$$\min \max_{r=1 \dots |D|} \frac{SOL_r}{OPT_r}$$

OPT_r is the optimal value for serving any r customers

Why do we care?

- ▶ We have budget restrictions
- ▶ Network planning is deployed in phases

Worst case example



- ▶ The above example has a worst case ratio of 2.24
- ▶ We can extend the above idea to achieve 3 (for around 200 facilities)

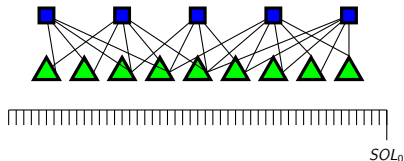
Algorithm

- ▶ We assume we are provided with a base algorithm 'A' (black box)
- ▶ It has a 2-approximation
- ▶ SOL_{ℓ}^A is the solution from algorithm A for serving ℓ customers
- ▶ With slight abuse of notation
$$SOL_{\ell}^A = f(SOL(F)_{\ell}^A) + c(SOL(F, D)_{\ell}^A)$$

The framework 'B' works in two phases

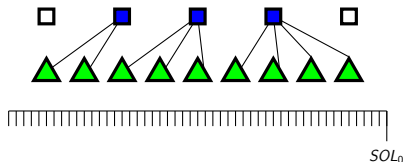
- ▶ Reduction Phase
 - ▶ Construct partial solutions that are competitive and save them
- ▶ Incremental Phase
 - ▶ Glue the saved partial solutions to construct a sequence

Algorithm



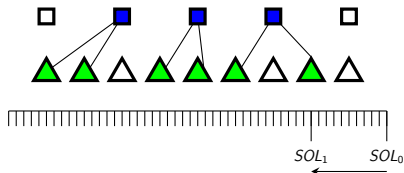
- ▶ **Start:** Approximately serve all customers
($t = 0, SOL_t = SOL_{|D|}^A = SOL_{|D|}^B$)
- ▶ **Reduction Phase:** (Iteration $\ell = |D| - 1$ to 1)
 - ▶ Remove the customer with the highest service cost
 - ▶ Close a facility if it is not serving any customer
 - ▶ Call this solution SOL_ℓ^B
 - ▶ If $2SOL_\ell^A < SOL_\ell^B$
 - ▶ $t = t + 1$
 - ▶ $SOL_t = SOL_\ell^A$
 - ▶ $SOL_\ell^B = SOL_\ell^A$

Algorithm



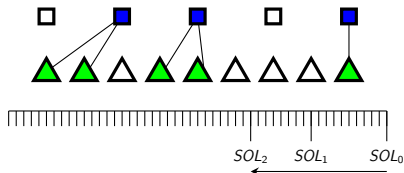
- ▶ **Start:** Approximately serve all customers
($t = 0, SOL_t = SOL_{|D|}^A = SOL_{|D|}^B$)
- ▶ **Reduction Phase:** (Iteration $\ell = |D| - 1$ to 1)
 - ▶ Remove the customer with the highest service cost
 - ▶ Close a facility if it is not serving any customer
 - ▶ Call this solution SOL_ℓ^B
 - ▶ If $2SOL_\ell^A < SOL_\ell^B$
 - ▶ $t = t + 1$
 - ▶ $SOL_t = SOL_\ell^A$
 - ▶ $SOL_\ell^B = SOL_\ell^A$

Algorithm



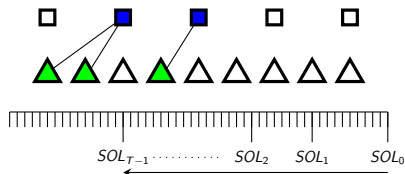
- ▶ **Start:** Approximately serve all customers
($t = 0, SOL_t = SOL_{|D|}^A = SOL_{|D|}^B$)
- ▶ **Reduction Phase:** (Iteration $\ell = |D| - 1$ to 1)
 - ▶ Remove the customer with the highest service cost
 - ▶ Close a facility if it is not serving any customer
 - ▶ Call this solution SOL_ℓ^B
 - ▶ If $2SOL_\ell^A < SOL_\ell^B$
 - ▶ $t = t + 1$
 - ▶ $SOL_t = SOL_\ell^A$
 - ▶ $SOL_\ell^B = SOL_\ell^A$

Algorithm



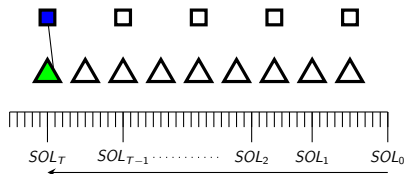
- ▶ **Start:** Approximately serve all customers
($t = 0, SOL_t = SOL_{|D|}^A = SOL_{|D|}^B$)
- ▶ **Reduction Phase:** (Iteration $\ell = |D| - 1$ to 1)
 - ▶ Remove the customer with the highest service cost
 - ▶ Close a facility if it is not serving any customer
 - ▶ Call this solution SOL_ℓ^B
 - ▶ If $2SOL_\ell^A < SOL_\ell^B$
 - ▶ $t = t + 1$
 - ▶ $SOL_t = SOL_\ell^A$
 - ▶ $SOL_\ell^B = SOL_\ell^A$

Algorithm



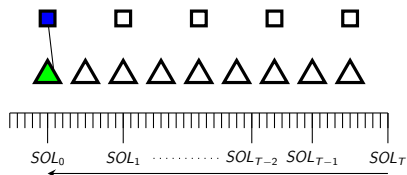
- ▶ **Start:** Approximately serve all customers
($t = 0, SOL_t = SOL_{|D|}^A = SOL_{|D|}^B$)
- ▶ **Reduction Phase:** (Iteration $\ell = |D| - 1$ to 1)
 - ▶ Remove the customer with the highest service cost
 - ▶ Close a facility if it is not serving any customer
 - ▶ Call this solution SOL_ℓ^B
 - ▶ If $2SOL_\ell^A < SOL_\ell^B$
 - ▶ $t = t + 1$
 - ▶ $SOL_t = SOL_\ell^A$
 - ▶ $SOL_\ell^B = SOL_\ell^A$

Algorithm



- ▶ **Start:** Approximately serve all customers
($t = 0$, $SOL_t = SOL_{|D|}^A = SOL_{|D|}^B$)
- ▶ **Reduction Phase:** (Iteration $\ell = |D| - 1$ to 1)
 - ▶ Remove the customer with the highest service cost
 - ▶ Close a facility if it is not serving any customer
 - ▶ Call this solution SOL_ℓ^B
 - ▶ If $2SOL_\ell^A < SOL_\ell^B$
 - ▶ $t = t + 1$
 - ▶ $SOL_t = SOL_\ell^A$
 - ▶ $SOL_\ell^B = SOL_\ell^A$

Algorithm

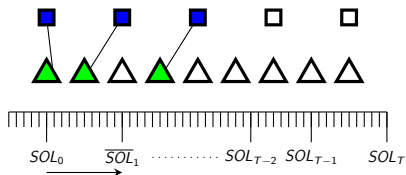


Let $\overline{SOL}_0 = SOL_0$

Incremental phase: (Iteration $k = 0$ to $T - 1$)

- ▶ Let SOL_k have r_k customers
- ▶ SOL_{k+1} has at least $r_{k+1} - r_k$ customers not in \overline{SOL}_k
- ▶ We will pick the cheapest $r_{k+1} - r_k$ customers from this set $SOL_{k+1}^D(r_{k+1} - r_k)$ and the facilities serving them $SOL_{k+1}^F(r_{k+1} - r_k)$ with cost $SOL_{k+1}(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F, R)_{k+1} = \overline{SOL}(F, R)_k \cup SOL_{k+1}^D(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F)_{k+1} = \overline{SOL}(F)_k \cup SOL_{k+1}^F(r_{k+1} - r_k)$

Algorithm

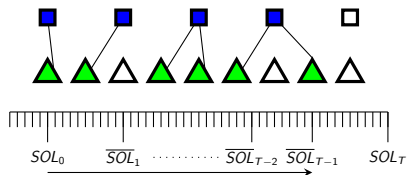


Let $\overline{SOL}_0 = SOL_0$

Incremental phase: (Iteration $k = 0$ to $T - 1$)

- ▶ Let SOL_k have r_k customers
- ▶ SOL_{k+1} has at least $r_{k+1} - r_k$ customers not in \overline{SOL}_k
- ▶ We will pick the cheapest $r_{k+1} - r_k$ customers from this set $SOL_{k+1}^D(r_{k+1} - r_k)$ and the facilities serving them $SOL_{k+1}^F(r_{k+1} - r_k)$ with cost $SOL_{k+1}(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F, R)_{k+1} = \overline{SOL}(F, R)_k \cup SOL_{k+1}^D(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F)_{k+1} = \overline{SOL}(F)_k \cup SOL_{k+1}^F(r_{k+1} - r_k)$

Algorithm

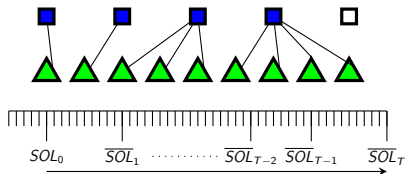


Let $\overline{SOL}_0 = SOL_0$

Incremental phase: (Iteration $k = 0$ to $T - 1$)

- ▶ Let SOL_k have r_k customers
- ▶ SOL_{k+1} has at least $r_{k+1} - r_k$ customers not in \overline{SOL}_k
- ▶ We will pick the cheapest $r_{k+1} - r_k$ customers from this set $SOL_{k+1}^D(r_{k+1} - r_k)$ and the facilities serving them $SOL_{k+1}^F(r_{k+1} - r_k)$ with cost $SOL_{k+1}(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F, R)_{k+1} = \overline{SOL}(F, R)_k \cup SOL_{k+1}^D(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F)_{k+1} = \overline{SOL}(F)_k \cup SOL_{k+1}^F(r_{k+1} - r_k)$

Algorithm

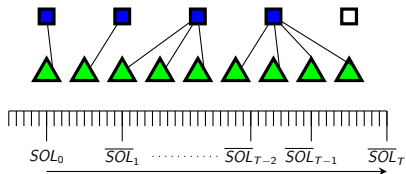


Let $\overline{SOL}_0 = SOL_0$

Incremental phase: (Iteration $k = 0$ to $T - 1$)

- ▶ Let SOL_k have r_k customers
- ▶ SOL_{k+1} has at least $r_{k+1} - r_k$ customers not in \overline{SOL}_k
- ▶ We will pick the cheapest $r_{k+1} - r_k$ customers from this set $SOL_{k+1}^D(r_{k+1} - r_k)$ and the facilities serving them $SOL_{k+1}^F(r_{k+1} - r_k)$ with cost $SOL_{k+1}(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F, R)_{k+1} = \overline{SOL}(F, R)_k \cup SOL_{k+1}^D(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F)_{k+1} = \overline{SOL}(F)_k \cup SOL_{k+1}^F(r_{k+1} - r_k)$

Algorithm



Let $\overline{SOL}_0 = SOL_0$

Incremental phase: (Iteration $k = 0$ to $T - 1$)

- ▶ Let SOL_k have r_k customers
- ▶ SOL_{k+1} has at least $r_{k+1} - r_k$ customers not in \overline{SOL}_k
- ▶ We will pick the cheapest $r_{k+1} - r_k$ customers from this set $SOL_{k+1}^D(r_{k+1} - r_k)$ and the facilities serving them $SOL_{k+1}^F(r_{k+1} - r_k)$ with cost $SOL_{k+1}(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F, R)_{k+1} = \overline{SOL}(F, R)_k \cup SOL_{k+1}^D(r_{k+1} - r_k)$
- ▶ $\overline{SOL}(F)_{k+1} = \overline{SOL}(F)_k \cup SOL_{k+1}^F(r_{k+1} - r_k)$

Proof sketch of Claim

Claim: The algorithm is 8-competitive at each point of the sequence.

Proof sketch of Claim

Claim: The algorithm is 8-competitive at each point of the sequence.

Look at a refinement point k : $2SOL_{r_k}^A < SOL_{r_{k+1}}^B(r_k)$

2* $\text{optimal}(r_k) \leq \text{cost of serving } r_k \text{ customers from a solution obtained from } \text{optimal}(r_{k+1})$

Proof sketch of Claim

Claim: The algorithm is 8-competitive at each point of the sequence.

Look at a refinement point k : $2SOL_{r_k}^A < SOL_{r_{k+1}}^B(r_k)$

$2 \cdot \text{optimal}(r_k) \leq \text{cost of serving } r_k \text{ customers from a solution obtained from optimal}(r_{k+1})$

Let us add $SOL_{k+1}(r_{k+1} - r_k)$ (cost of the $r_{k+1} - r_k$ added in the incremental phase) to both sides

Proof sketch of Claim

Claim: The algorithm is 8-competitive at each point of the sequence.

Look at a refinement point k : $2SOL_{r_k}^A < SOL_{r_{k+1}}^B(r_k)$

$2 \cdot \text{optimal}(r_k) \leq \text{cost of serving } r_k \text{ customers from a solution obtained from optimal}(r_{k+1})$

Let us add $SOL_{k+1}(r_{k+1} - r_k)$ (cost of the $r_{k+1} - r_k$ added in the incremental phase) to both sides

$$\begin{aligned} 2SOL_{r_k}^A + SOL_{k+1}(r_{k+1} - r_k) \\ &< SOL_{r_{k+1}}^B(r_k) + SOL_{k+1}(r_{k+1} - r_k) \\ &\leq 2SOL_{r_{k+1}}^B = 2SOL_{r_{k+1}}^A \end{aligned}$$

Proof sketch of Claim

Claim: The algorithm is 8-competitive at each point of the sequence.

Look at a refinement point k : $2SOL_{r_k}^A < SOL_{r_{k+1}}^B(r_k)$

$2 * \text{optimal}(r_k) \leq \text{cost of serving } r_k \text{ customers from a solution obtained from optimal}(r_{k+1})$

Let us add $SOL_{k+1}(r_{k+1} - r_k)$ (cost of the $r_{k+1} - r_k$ added in the incremental phase) to both sides

$$\begin{aligned} 2SOL_{r_k}^A + SOL_{k+1}(r_{k+1} - r_k) \\ &< SOL_{r_{k+1}}^B(r_k) + SOL_{k+1}(r_{k+1} - r_k) \\ &\leq 2SOL_{r_{k+1}}^B = 2SOL_{r_{k+1}}^A \end{aligned}$$

Remark:

- This is true for all $k = 0$ to $T - 1$

Analysis

$$2SOL_{r_k}^A + SOL_{k+1}(r_{k+1} - r_k) \leq 2SOL_{r_{k+1}}^A$$

Analysis

$$2SOL_{r_k}^A + SOL_{k+1}(r_{k+1} - r_k) \leq 2SOL_{r_{k+1}}^A$$

For any $k = 1$ to T , we can add these terms from $j = 1$ to $k - 1$ to get

Analysis

$$2SOL_{r_k}^A + SOL_{k+1}(r_{k+1} - r_k) \leq 2SOL_{r_{k+1}}^A$$

For any $k = 1$ to T , we can add these terms from $j = 1$ to $k - 1$ to get

$$\begin{aligned} 2SOL(F)_0^B + SOL(R)_0^B + \sum_{j=0}^{k-1} SOL_{j+1}(r_{j+1} - r_j) \\ = 2SOL_{r_k}^A \leq 4OPT_{r_k} \end{aligned}$$

Analysis

$$2SOL_{r_k}^A + SOL_{k+1}(r_{k+1} - r_k) \leq 2SOL_{r_{k+1}}^A$$

For any $k = 1$ to T , we can add these terms from $j = 1$ to $k - 1$ to get

$$\begin{aligned} 2SOL(F)_0^B + SOL(R)_0^B + \sum_{j=0}^{k-1} SOL_{j+1}(r_{j+1} - r_j) \\ = 2SOL_{r_k}^A \leq 4OPT_{r_k} \end{aligned}$$

Analysis

We would lose an additional factor 2 at intermediate points between two refinement point giving an 8-competitive algorithm!

Tighter analysis

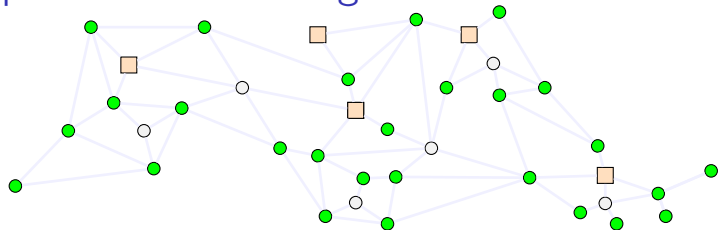
$$2 * \text{Facility cost} + \text{service cost} \leq 8 * \text{Optimal cost}$$

Experiments

Size	#	Max gap (%)	Ave Gap (%)	Time[sec]
200	15	50-60	13-15	250-350
300	15	40-45	11-13	2050-2100

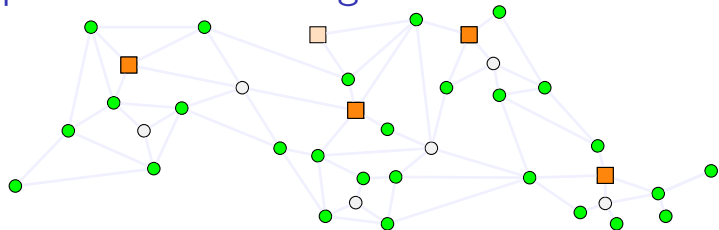
Table: Results of computational experiments from UFLib Library

Typical Network Design



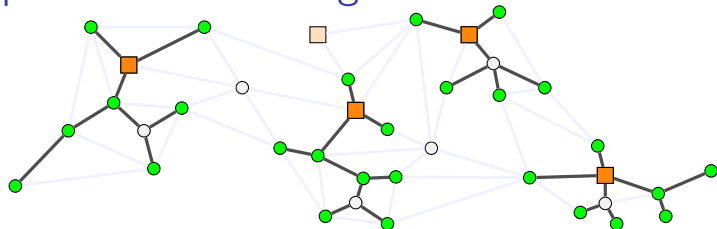
- ▶ Given a set of **demand nodes** in a weighted network
- ▶ Find a minimum cost routing network; and route every client demand to a **an open facility**

Typical Network Design



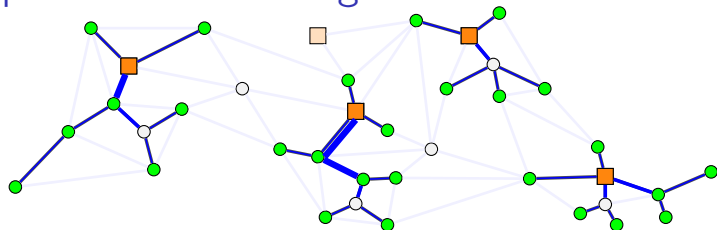
- ▶ Given a set of **demand nodes** in a weighted network
- ▶ Find a minimum cost routing network; and route every client demand to a **an open facility**

Typical Network Design



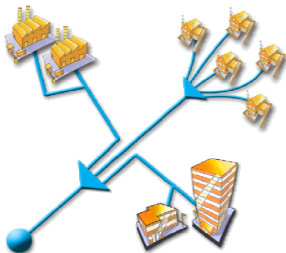
- ▶ Given a set of **demand nodes** in a weighted network
- ▶ Find a minimum cost routing network; and route every client demand to a **an open facility**
- ▶ Cost of routing demand on edge e depends on the total demand (denoted by D_e) routed on that edge

Typical Network Design



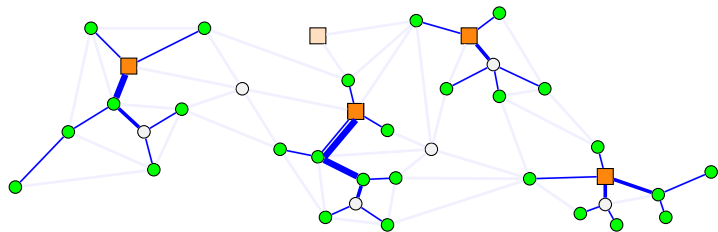
- ▶ Given a set of **demand nodes** in a weighted network
- ▶ Find a minimum cost routing network; and route every client demand to a **an open facility**
- ▶ Cost of routing demand on edge e depends on the total demand (denoted by D_e) routed on that edge
 - ▶ **Steiner Tree**: $\text{cost}_e(D_e) = c_e$, for $D_e > 0$
 - ▶ **Buy-at-Bulk Network Design**: $\text{cost}_e(D_e) = c_e \cdot g(D_e)$, where g is a **concave cost function**
 - ▶ ...

Cable Model



- ▶ In practice costs arise due to discrete capacity cables:
 - ▶ The capacity on a link can be purchased at discrete units: $u_1 < u_2 < \dots < u_K$
costs: $\sigma_1 < \sigma_2 < \dots < \sigma_K$
where $\frac{\sigma_1}{u_1} > \frac{\sigma_2}{u_2} > \dots > \frac{\sigma_K}{u_K}$

Multiple-Sinks (Facilities) Buy-at-Bulk Network Design



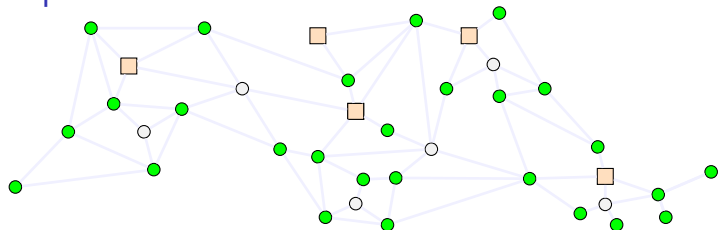
- ▶ Given a set of candidate sinks F (called **facilities**) instead of a single sink
- ▶ We may route demand to any facility, but incur a facility cost
- ▶ Find a trade-off between **facility opening** and **network design** costs

The problem

Input:

- ▶ undirected graph $G = (V, E)$
- ▶ edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- ▶ potential facilities $F \subseteq V$ with opening costs $\mu_i \in \mathbb{Z}_{\geq 0}$, $i \in F$
- ▶ clients $D \subseteq V$ with demands $d_j \in \mathbb{Z}_{>0}$, $j \in D$
- ▶ access cable types K with
 - ▶ capacity $u_k \in \mathbb{Z}_{>0}$, $k \in K$
 - ▶ setup cost (per unit length) $\sigma_k \in \mathbb{Z}_{\geq 0}$, $k \in K$
 $\sigma_1 < \dots < \sigma_K$ and $\frac{\sigma_1}{u_1} > \dots > \frac{\sigma_K}{u_K}$

The problem



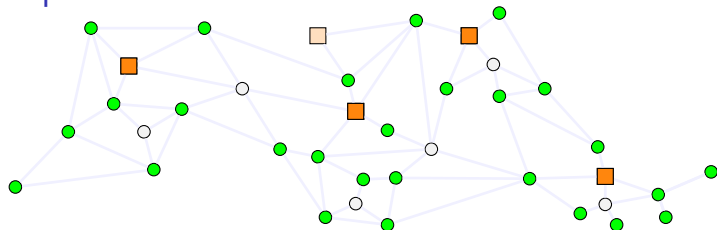
Solution:

- ▶ open facilities $\bar{F} \subseteq F$
- ▶ forest $A^* \subseteq E$ containing one path, for each j , (called P_j) that connects client j to some open facility $i_j \in \bar{F}$
- ▶ cable installation $x : A^* \times K \rightarrow \mathbb{Z}_{\geq 0}$ of sufficient capacity, i.e., $\sum_{j: e \in P_j} d_j \leq \sum_k u_k x_{e,k}$

Goal:

$$\min \sum_{i \in \bar{F}} \mu_i + \sum_{e \in A^*} \sum_{k \in K} \sigma_k c_e x_{e,k}$$

The problem



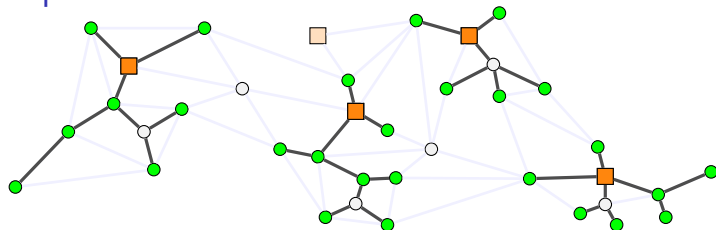
Solution:

- ▶ open facilities $\bar{F} \subseteq F$
- ▶ forest $A^* \subseteq E$ containing one path, for each j , (called P_j) that connects client j to some open facility $i_j \in \bar{F}$
- ▶ cable installation $x : A^* \times K \rightarrow \mathbb{Z}_{\geq 0}$ of sufficient capacity, i.e., $\sum_{j: e \in P_j} d_j \leq \sum_k u_k x_{e,k}$

Goal:

$$\min \sum_{i \in \bar{F}} \mu_i + \sum_{e \in A^*} \sum_{k \in K} \sigma_k c_e x_{e,k}$$

The problem



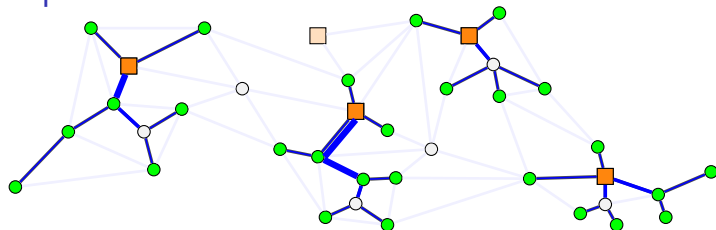
Solution:

- ▶ open facilities $\bar{F} \subseteq F$
- ▶ forest $A^* \subseteq E$ containing one path, for each j , (called P_j) that connects client j to some open facility $i_j \in \bar{F}$
- ▶ cable installation $x : A^* \times K \rightarrow \mathbb{Z}_{\geq 0}$ of sufficient capacity, i.e., $\sum_{j: e \in P_j} d_j \leq \sum_k u_k x_{e,k}$

Goal:

$$\min \sum_{i \in \bar{F}} \mu_i + \sum_{e \in A^*} \sum_{k \in K} \sigma_k c_e x_{e,k}$$

The problem



Solution:

- ▶ open facilities $\bar{F} \subseteq F$
- ▶ forest $A^* \subseteq E$ containing one path, for each j , (called P_j) that connects client j to some open facility $i_j \in \bar{F}$
- ▶ cable installation $x : A^* \times K \rightarrow \mathbb{Z}_{\geq 0}$ of sufficient capacity, i.e., $\sum_{j: e \in P_j} d_j \leq \sum_k u_k x_{e,k}$

Goal:

$$\min \sum_{i \in \bar{F}} \mu_i + \sum_{e \in A^*} \sum_{k \in K} \sigma_k c_e x_{e,k}$$

Compact Formulation

$$\begin{aligned}
 \text{(IP-1)} \quad & \min \sum_{i \in F} \mu_i z_i + \sum_{e \in E} c_e \sum_{n=1}^K \sigma_n x_e^n \\
 & \sum_{e \in \delta^+(j)} f_e^j \geq 1 & \forall j \in D \\
 & \sum_{e \in \delta^+(v)} f_e^j = \sum_{e \in \delta^-(v)} f_e^j & \forall j \in D, v \in V \setminus F, v \neq j \\
 & \sum_{e \in \delta^-(i)} f_e^j - \sum_{e \in \delta^+(i)} f_e^j \leq z_i & \forall j \in D, i \in F \\
 & \sum_{j \in D} d_j (f_{(k,l)}^j + f_{(l,k)}^j) \leq \sum_{n=1}^K u_n x_{kl}^n & \forall kl \in E \\
 & x_e^n, f_e^j, z_i \text{ non-negative integers}
 \end{aligned}$$

Where:

- ▶ z_i indicates if **facility** i is open or not
- ▶ x_e^n indicates if **cable type** n is installed on edge e
- ▶ f_e^j indicates if flow from **client** j uses edge e

Compact Formulation

$$\begin{aligned}
 \text{(IP-1)} \quad & \min \sum_{i \in F} \mu_i z_i + \sum_{e \in E} c_e \sum_{n=1}^K \sigma_n x_e^n \\
 & \sum_{e \in \delta^+(j)} f_e^j \geq 1 && \forall j \in D \\
 & \sum_{e \in \delta^+(v)} f_e^j = \sum_{e \in \delta^-(v)} f_e^j && \forall j \in D, v \in V \setminus F, v \neq j \\
 & \sum_{e \in \delta^-(i)} f_e^j - \sum_{e \in \delta^+(i)} f_e^j \leq z_i && \forall j \in D, i \in F \\
 & \sum_{j \in D} d_j (f_{(k,l)}^j + f_{(l,k)}^j) \leq \sum_{n=1}^K u_n x_{kl}^n && \forall kl \in E \\
 & x_e^n, f_e^j, z_i \text{ non-negative integers}
 \end{aligned}$$

Theorem. The integrality gap of (IP-1) can be arbitrarily large.

Approximate Solution

Modified routing cost:

$$\left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e \leq \left(\sigma_k + D_e \frac{\sigma_k}{u_k} \right) c_e \leq 2 \left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e$$

Approximate Solution

Modified routing cost:

$$\left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e \leq \left(\sigma_k + D_e \frac{\sigma_k}{u_k} \right) c_e \leq 2 \left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e$$

$$(IP-2) \quad \min \sum_{i \in F} \mu_i z_i + \sum_{k=1}^K \sigma_k \sum_{e \in E} c_e x_e^k + \sum_{j \in D} d_j \sum_{k=1}^K \frac{\sigma_k}{u_k} \sum_{e \in \vec{E}} c_e f_{e;k}^j$$

$$\sum_{e \in \delta^+(j)} \sum_{k=1}^K f_{e;k}^j \geq 1 \quad \forall j \in D$$

$$\sum_{e \in \delta^+(v)} \sum_{k=1}^K f_{e;k}^j = \sum_{e \in \delta^-(v)} \sum_{k=1}^K f_{e;k}^j \quad \forall j \in D, v \in V \setminus F, v \neq j$$

$$\sum_{e \in \delta^-(i)} \sum_{k=1}^K f_{e;k}^j - \sum_{e \in \delta^+(i)} \sum_{k=1}^K f_{e;k}^j \leq z_i \quad \forall j \in D, i \in F$$

$$f_{uv;k}^j + f_{vu;k}^j \leq x_e^k \quad \forall j \in D, uv \in E, 1 \leq k \leq K$$

Approximate Solution

Modified routing cost:

$$\left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e \leq \left(\sigma_k + D_e \frac{\sigma_k}{u_k} \right) c_e \leq 2 \left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e$$

$$(IP-2) \min \sum_{i \in F} \mu_i z_i + \sum_{k=1}^K \sigma_k \sum_{e \in E} c_e x_e^k + \sum_{j \in D} d_j \sum_{k=1}^K \frac{\sigma_k}{u_k} \sum_{e \in \vec{E}} c_e f_{e;k}^j$$

$$\sum_{e \in \delta^+(j)} \sum_{k=1}^K f_{e;k}^j \geq 1 \quad \forall j \in D$$

$$\sum_{e \in \delta^+(v)} \sum_{k=1}^K f_{e;k}^j = \sum_{e \in \delta^-(v)} \sum_{k=1}^K f_{e;k}^j \quad \forall j \in D, v \in V \setminus F, v \neq j$$

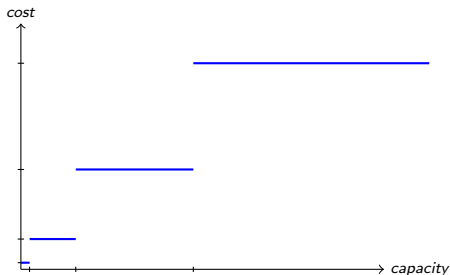
$$\sum_{e \in \delta^-(i)} \sum_{k=1}^K f_{e;k}^j - \sum_{e \in \delta^+(i)} \sum_{k=1}^K f_{e;k}^j \leq z_i \quad \forall j \in D, i \in F$$

$$f_{uv;k}^j + f_{vu;k}^j \leq x_e^k \quad \forall j \in D, uv \in E, 1 \leq k \leq K$$

Theorem (Friggstad, Rezapour, Soto, Salavatipour)

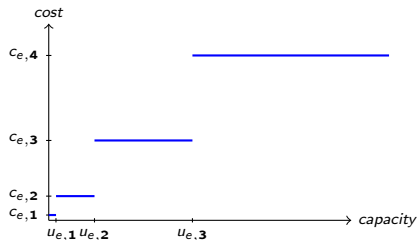
The integrality gap of (IP-2) is at most $O(K)$.

Cable Model



- ▶ $g(x) = \min$ cost set of cables of total capacity at least x
(Integer Minimum Knapsack Problem)
 - ▶ one can compute the optimal combination of cable types for all flow levels on any edge using dynamic programming

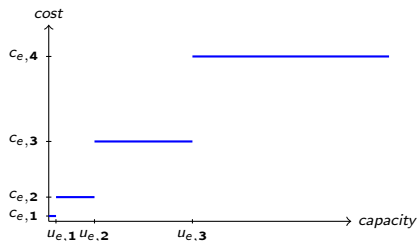
Path based Formulation



- We consider each piece of the step cost function as a module: module i has a cost of $c_{e,i}$ and a capacity of $u_{e,i}$

$$\Rightarrow x_{e,n} \in \{0, 1\}$$

Path based Formulation



- ▶ We consider each piece of the step cost function as a module: module i has a cost of $c_{e,i}$ and a capacity of $u_{e,i}$

$$\Rightarrow x_{e,n} \in \{0, 1\}$$

- ▶ We create a dummy root node r and connect all facilities with the root node.
- ▶ Let $P(j)$ denote the set of all possible paths starting from client j and terminating at node r

$$\Rightarrow y_p \in \{0, 1\}, p \in P(j)$$

Path based Formulation

$$\begin{aligned}
 (\text{IP-3}) \quad & \min \sum_{i \in F} \mu_i z_i + \sum_{e \in E} \sum_{n \in \mathcal{N}_e} c_{e,n} \cdot x_{e,n} \\
 & \sum_{p \in P(j)} y_p = 1, & \forall j \in D \\
 & \sum_{p \in P(j): (i,r) \in p} y_p \leq z_i, & \forall i \in F, \forall j \in D \\
 & \sum_{j \in D} \sum_{\substack{p \in P(j): \\ \{(k,l), (l,k)\} \cap p \neq \emptyset}} d_j y_p \leq \sum_{n \in \mathcal{N}_{kl}} u_{kl,n} x_{kl,n}, & \forall kl \in E \\
 & \sum_{n \in \mathcal{N}_{kl}} x_{kl,n} \leq 1, & \forall kl \in E \\
 & y_p, x_{e,n}, z_i \in \{0, 1\}
 \end{aligned}$$

Theorem

IP-3 is at least as strong as IP-2 in terms of the lower bounds.

Cut Inequalities

- Valid inequalities:

For every client j , and $\bar{S} \subset V$ (containing j ; not r), we have:

$$\sum_{kl: k \in \bar{S}} \sum_{n \in \mathcal{N}_{kl}} x_{kl,n} + \sum_{i \in \bar{S}} z_i \geq 1$$

Cut Inequalities

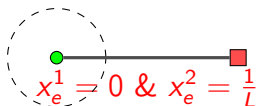
- Valid inequalities:

For every client j , and $\bar{S} \subset V$ (containing j ; not r), we have:

$$\sum_{kl: k \in \bar{S}} \sum_{n \in \mathcal{N}_{kl}} x_{kl,n} + \sum_{i \in \bar{S}} z_i \geq 1$$

- Seperation:

- Given a fractional optimal solution (x^*, y^*, z^*) to IP-3.
- Take the edge capacities to be: $\sum_{n \in \mathcal{N}_{kl}} x_{kl,n}^*$ for all $kl \in E$; and z_i^* for all $i \in V$
- For every client $j \in D$, solve the maximum flow problem with source as j and sink as r . If the flow value is less than 1, then we obtain the violated cut.



Cover Inequalities

- ▶ We define $\theta_{kl} = (D_\theta, M_\theta)$ to be a **cover** if (where $D_\theta \subseteq D$, $M_\theta \subseteq \mathcal{N}_{kl}$, and $U_{kl} = \sum_{n \in \mathcal{N}_{kl}} u_{kl,n}$)

$$\sum_{j \in D_\theta} d_j + \sum_{n \in M_\theta} u_{kl,n} > U_{kl}$$

- ▶ We say that a cover is **minimal** when just removing any item either from D_θ or M_θ results a non-cover
- ▶ If θ_{kl} is a **minimal cover**, then the following **inequalities** are **valid**:

$$\sum_{j \in D_\theta} \sum_{\substack{p \in P(j): \\ \{(k,l), (l,k)\} \cap p \neq \emptyset}} y_p + \sum_{n \in M_\theta} (1 - x_{kl,n}) \leq |M_\theta| + |D_\theta| - 1 \iff$$

$$\sum_{j \in D_\theta} \sum_{\substack{p \in P(j): \\ \{(k,l), (l,k)\} \cap p \neq \emptyset}} y_p \leq \sum_{n \in M_\theta} x_{kl,n} + |D_\theta| - 1$$

Cover Inequalities

Separation:

- ▶ Given a fractional optimal solution (x^*, y^*, z^*) to IP-3.
- ▶ For each $j \in D$, we let

$$w_j^* = \sum_{\substack{p \in P(j): \\ \{(k,l), (l,k)\} \cap p \neq \emptyset}} y_p^*$$

- ▶ A most violated cover inequality is obtained by solving the following **knapsack problem**:

$$\begin{aligned} \min \quad & \gamma = \sum_{n \in F_{kl}} x_{kl,n}^* x_{kl,n} + \sum_{j \in D} (1 - w_j^*) w_j \\ & \sum_{j \in D} d_j w_j + \sum_{n \in F_{kl}} u_{kl,n} x_{kl,n} \geq \sum_{n \in F_{kl}} u_{kl,n} + 1 \\ & x_{kl,n} \in \{0, 1\}, \quad \forall n \in F_{kl} \\ & w_j \in \{0, 1\}, \quad \forall j \in D \end{aligned}$$

Cover Inequalities

Separation:

- ▶ Given a fractional optimal solution (x^*, y^*, z^*) to IP-3.
- ▶ For each $j \in D$, we let

$$w_j^* = \sum_{\substack{p \in P(j): \\ \{(k,l), (l,k)\} \cap p \neq \emptyset}} y_p^*$$

- ▶ A most violated cover inequality is obtained by solving the following **knapsack problem**:

$$\begin{aligned} \min \quad & \gamma = \sum_{n \in F_{kl}} x_{kl,n}^* x_{kl,n} + \sum_{j \in D} (1 - w_j^*) w_j \\ & \sum_{j \in D} d_j w_j + \sum_{n \in F_{kl}} u_{kl,n} x_{kl,n} \geq \sum_{n \in F_{kl}} u_{kl,n} + 1 \\ & x_{kl,n} \in \{0, 1\}, \quad \forall n \in F_{kl} \\ & w_j \in \{0, 1\}, \quad \forall j \in D \end{aligned}$$

- ▶ (x^*, y^*, z^*) **violates** the following **cover inequality** if $\gamma < 1$.

$$\sum_{j \in D'} \sum_{p \in P(j): (k,l) \in p} y_p \leq \sum_{n \in F'_{kl} \cup \{N_{kl} \setminus F_{kl}\}} x_{kl,n} + |D'| - 1$$

Basic Idea of Solution Method

- ▶ Our formulation contains an **exponential** number of variables!

Column Generation:

- ▶ We solve the LP to optimality using simplex with only a **subset** of the variables—**restricted master problem**.
- ▶ We then ask if any variable that has been left out has **negative reduced cost**; if so, that column is added—**pricing problem**
- ▶ The optimal solution might not be integral!

Branch-and-Bound:

- ▶ We use **branching** to handle integrality.

Restricted Master Problem

- ▶ We consider only a subset $P'(j) \subseteq P(j)$ of paths for each j
- ▶ We enrich the restricted master problem by the **routing paths** obtained by a few runs of the following algorithm.

Algorithm GreedyAlgorithm

1. Pick a random permutation of clients in D ;
Let $\Pi = (j_1, j_2, \dots, j_{|D|})$ be the picked permutation.
 2. **For** $i = 1, 2, \dots, |D|$ **do**
 - Greedily route d_{j_i} units of demand from j_i to root r via the cheapest cost **routing path**, using the network constructed by the previous $i - 1$ clients.
-

Restricted Master Problem

- ▶ We consider only a subset $P'(j) \subseteq P(j)$ of paths for each j
- ▶ We enrich the restricted master problem by the **routing paths** obtained by a few runs of the following algorithm.

Algorithm GreedyAlgorithm

1. Pick a random permutation of clients in D ;
Let $\Pi = (j_1, j_2, \dots, j_{|D|})$ be the picked permutation.
 2. **For** $i = 1, 2, \dots, |D|$ **do**
 - Greedily route d_{j_i} units of demand from j_i to root r via the cheapest cost **routing path**, using the network constructed by the previous $i - 1$ clients.
-

Theorem (Charikar & Karagiozova; STOC 2005)

The (inflated) greedy algorithm achieves an approximation ratio of $O(\log^2(|D|))$ for the single-sink non-uniform buy-at-bulk problem (with unit demands).

holds for our problem as well.

Pricing Problem

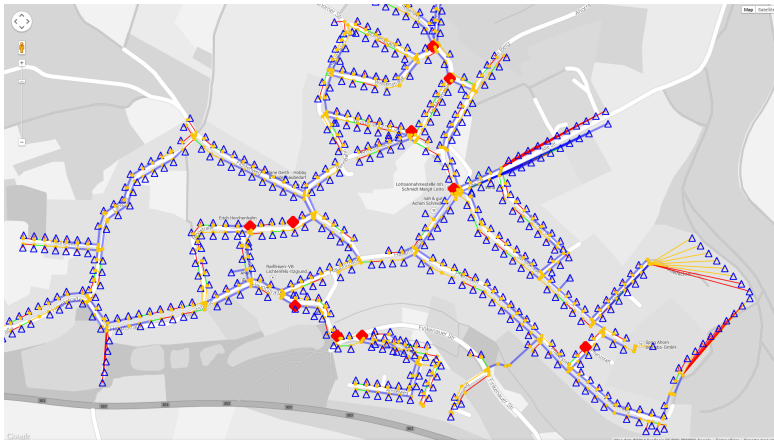
- ▶ We only need to search for some column with **negative reduced cost**

$$\min_{p \in P(j)} - \left(\mu_j + \sum_{\substack{p \in P(j): \\ \{(k,l), (l,k)\} \cap p \neq \emptyset, \\ l \neq r}} d_j \pi_{kl} + \sum_{i \in F} \mathbb{I}_i^p \gamma_i^j \right)$$

- ▶ This corresponds to solving a **shortest path problem**, where we search for routes with a negative reduced cost
 - ▶ we take the weight of an edge kl to be $-d_j \pi_{kl}$, for all $kl \in E$ and weights $-\gamma_i^j$, for all ir edges

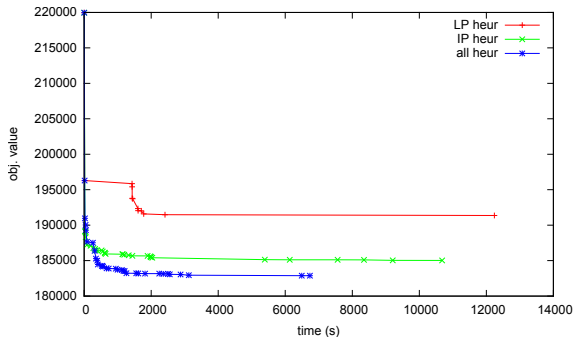
Instances details

- ▶ Each instance corresponds to a region in Germany concerning the potential **client** and **facility** locations.
- ▶ The **street segment** form the **edges**, while the street intersections and traffic circles provide the **nodes**.



Primal heuristics implementation

- ▶ A CPLEX based heuristic
- ▶ A LP based heuristics (similar to the previous one)
- ▶ A hybrid strategy that has parallel implementation



Computational Results

Inst.	V	E	F	D	# vars	# cuts	lp-gap(%)	root-gap(%)	gap(%)
a	1,675	1,722	104	604	12,079	5,089	89.5	19.6	18.2
b	11,544	12,350	890	4,275	43,478	3,759	146.0	53.0	53.0
c	2,271	1,419	498	349	32,081	2,325	82.9	21.5	21.3
d	4,110	4,350	230	1,670	23,418	13,692	151.5	23.9	23.3
e	637	758	101	39	50,739	1,749	69.1	23.0	16.1
f	1,315	1,422	148	238	50,167	5,685	172.7	18.6	15.9
g	3,055	3,177	49	591	2,976	2,134	71.4	13.3	10.7
h	4,227	4,482	319	1,490	31,261	10,865	121.8	20.5	20.5
i	6,750	7,262	531	2,440	33,211	7,165	150.7	32.7	32.7

Table: Results of our algorithm on the real-world instances

- ▶ We report the results after a run time of 36 000 s (ten hours)
- ▶ We are able to solve large real world instances to roughly 20.0%

Summary

Incremental Facility location

- ▶ 8-competitive algorithm
- ▶ Improve LB (3) or UB (8) or both?

BCP for BuyatBulk FL

- ▶ MIP model, Valid inequalities
- ▶ Polyhedral results: Facet defining, separation problem
- ▶ Resilience of the network
- ▶ Incremental strategy

THANK YOU